Plan Your Deployment

Tableau Server on Linux: Start Here ................................................................. 2
Introducing Tableau Server on Linux .............................................................. 3
The New Data Engine in Tableau Server ......................................................... 7
Server Administrator Overview ................................................................. 13
Tableau Services Manager Overview ............................................................ 20
Security ........................................................................................................ 22
Infrastructure Planning .................................................................................. 22

Tableau Server on Linux: Start Here

Tableau Server® on Linux by Tableau Software lets Tableau users share workbooks and
data extracts with other users in your organization.

Use the links below to get started using Tableau Server on Linux.

Plan

<table>
<thead>
<tr>
<th>Server Administrator Overview</th>
<th>Introducing Tableau Server on Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td>Infrastructure Planning</td>
<td>Security</td>
</tr>
<tr>
<td>Tableau Services Manager Overview</td>
<td>Planning a Site</td>
</tr>
</tbody>
</table>

Deploy

<table>
<thead>
<tr>
<th>Install and Configure Tableau Server</th>
<th>Install Drivers on Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jump-start Installation</td>
<td>Install Tableau Server in the AWS Cloud</td>
</tr>
<tr>
<td>Migrate Tableau Server from Windows to Linux</td>
<td>Distributed and High Availability Tableau Server Installations</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>

**Manage Server**

<table>
<thead>
<tr>
<th>tsm Command Line Reference</th>
<th>Extract Refresh Schedules</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add or Update Sites</td>
<td>Add Users to the Server</td>
</tr>
</tbody>
</table>

**Manage Sites**

<table>
<thead>
<tr>
<th>What is a site</th>
<th>Manage Users and Groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage Content Access</td>
<td>Manage Data</td>
</tr>
</tbody>
</table>

**Automate and Extend**

<table>
<thead>
<tr>
<th>Tableau Developer Tools</th>
<th>Tableau Tools and Samples on GitHub</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST API</td>
<td>tsm Command Line Reference</td>
</tr>
<tr>
<td>Web Data Connector (WDC)</td>
<td>Extract API</td>
</tr>
</tbody>
</table>

**Introducing Tableau Server on Linux**

Tableau Server on Linux brings the capabilities of Tableau Server to the Linux platform. This page provides general information about the initial 10.5 release of Tableau Server on Linux.

- Installing
- Upgrading
- What’s New
- General functionality
- Data connectivity
Installing Tableau Server

To get started installing and configuring Tableau Server in a production environment, see Install and Configure Tableau Server. If you want a quick start procedure to install Tableau Server in a non-production environment, see Jump-start Installation.

Upgrading Tableau Server on Linux

Version 10.5 is the first version of Tableau Server available on Linux, so there is no upgrade path. You can migrate your Tableau Server on Windows installation to Linux using the information in Migrate Tableau Server from Windows to Linux.

What's New

Tableau Server on Linux is a new product that includes several new features, including the following:

- Tableau Server includes Hyper, a new in-memory Data Engine technology. For more details, see The New Data Engine in Tableau Server.
- Tableau Services Manager (TSM) is a new feature that you use to configure and administer Tableau Server. To learn more about TSM, see Tableau Services Manager Overview.
- Tableau Server includes hot topology support that allows you to change the number of Backgrounder and VizQL processes running on nodes that already have these processes - without the need to restart Tableau Server. To learn more about these and other Tableau Server processes, see Process Reference.
- Tableau Server now supports Lightweight Directory Access Protocol (LDAP). To learn more about LDAP support in Tableau Server, see Identity Store.

General functionality

Non-administrative functionality in Tableau Server on Linux works as it does for Tableau Server on Windows. The primary areas of likely differences between Server on Windows
and on Linux are supported data sources, and potential display differences due to font differences. For more information, see Migrate Tableau Server from Windows to Linux and Install Drivers on Linux.

**Data connectivity functionality**

Tableau Server on Linux supports a subset of the data connectivity functionality available with Tableau Server on Windows.

For details on which data sources are supported, see the Tableau Server tech specs.

**Fonts on Linux**

Tableau Server uses the fonts installed on the system to render workbooks based on the fonts used when a workbook was created. When a font is not available, Tableau Server will use the closest equivalent based on font families; this is true for both Windows and Linux Servers. On Linux Servers missing fonts may be more obvious because Linux ships with fewer included fonts than Windows and OS/X systems do. This matters because many workbooks are authored in Tableau Desktop on Windows or on Mac.

Tableau Server on Linux ships with the following fonts:

- Arial
- Courier
- Georgia
- Times New Roman
- Verdana
- Trebuchet MS
- Tableau Font

Workbooks which use fonts other than these may appear differently than expected when viewed on Tableau Server on Linux, due to missing fonts. To resolve this issue, install the appropriate fonts onto all nodes in your Tableau Server installation.

For details on a known issue with fonts, see *Support for Asian Character sets* below.
Known issues

The following is a list of the known issues with the initial release of Tableau Server on Linux. To learn how to resolve common issues with Tableau Server, see Troubleshoot Tableau Server on Linux.

*sudoers* file implementation

The current implementation of Tableau Server relies on sudo privileges. This approach will be changed in the next release. Updating the sudoers file conflicts with some system management configuration best practices and security policies. Therefore, the next version of Tableau Server will eliminate the use of the privileged user (tsmagent) and will not add a Tableau-specific user to the sudoers file. Due to this change, upgrading from Tableau Server 10.5 to a newer version will require some additional steps which will be documented and supported by Tableau. See System User and sudo Privileges.

Virtual Container environments

Installation of Tableau Server in virtual container environments such as Docker has not been tested and is not supported. Tableau Server on Linux will not function as expected if installed in these environments.

Hostname

Hostnames that include underscores (_) are not supported by Tableau Server on Linux.

Cannot restore Tableau Server on Windows using Linux backup file

You cannot restore Tableau Server on Windows using a Linux backup file (.tsbak). However, you can restore Tableau Server on Linux using a Windows backup file. For more information about restoring a Windows backup on Tableau Server on Linux, see Migrate Tableau Server from Windows to Linux.
Support for Asian character sets

If you see empty boxes where you expect to see Asian characters in vizzes that are displayed on Tableau Server, then you should install the language-appropriate font packages in your Linux environment.

If you change your user locale after opening a view, the view will not display again

When you change your user locale after opening a view, any subsequent attempt to open the view will fail with an "unexpected error." You can still open views that you have not previously opened.

To work around this issue, sign out of Tableau Server after changing your locale, and then sign back in. All views will display properly.

Restoring Tableau Server from very large restore files can fail on Ubuntu

Restoring Tableau Server from very large restore files (>100 GB) might fail on Ubuntu systems or virtual environments with low I/O bandwidth.

Capacity does not update after activating or deactivating a license key

The Licenses page may not update to show the correct capacity (that is, seats, cores, or users) after you activate or deactivate a license key. To see the correct capacity, restart Tableau Server using the tsm restart command.

The New Data Engine in Tableau Server

Hyper is Tableau’s new in-memory Data Engine technology optimized for fast data ingests and analytical query processing on large or complex data sets. Hyper powers the new Data Engine in Tableau Server 10.5, Tableau Desktop 10.5, Tableau Online, and Tableau Public. The Data Engine in 10.5 is used when creating, refreshing or querying extracts. It is also used for cross-database joins to support federated data sources with multiple connections.
In this article

Performance benefits
Memory and CPU usage
Changes to Server configuration, topology, and extracts

Performance benefits

Starting in 10.5, Hyper technology is integrated with Tableau Data Engine to give you the following key benefits:

- **Faster extract creation**: With Hyper technology, extracts are generated almost as fast as the source system can deliver data, no sorting needed.

- **Support for larger extracts**: Prior to this release, you might have not been able to get all your data into a single extract. With the new Hyper technology, much larger amounts of data can be included in a single extract.

- **Faster analysis of extracts**: In many cases you will see faster querying of data for larger extracts.

Here are some reasons why the new Data Engine performs better on larger or complex extracts and is optimized for faster querying:

- **Hyper technology is designed to consume data faster**: Unlike in previous versions, the new Data Engine does not do any post processing like sorting. With Hyper, post processing steps like sorting are not needed giving the Data Engine the ability to perform better with larger extracts.

- **Hyper technology is memory-optimized**: This means that when needed, all data lives in memory. This results in fast data access times.

- **Hyper technology is CPU optimized**: This means that Data Engine now fully
parallelizes the query execution and utilizes available CPU in such a way that query execution time scales almost linearly with the number of cores in the machine.

- **Hyper is a compiling query engine.** Queries are either interpreted or compiled to the machine code for maximum performance and allowing the Data Engine to get most performance out of modern hardware (CPU, large main-memory capacities).

- **Hyper technology uses advanced query optimizations to make queries faster.** Along with many additional advanced techniques such as, materializing min and max values for each column, mini-indices to optimize search ranges, more granular data block-level dictionaries, advanced logic for join and sub-query performance optimizations, the new Data Engine offers many improvements over the previous Tableau Data Engine in terms of performance and scalability.

**Memory and CPU usage**

The new Data Engine is designed to leverage all available CPU and memory on the machine to provide the fastest response times.

**CPU usage**

Hyper technology leverages the new instruction sets in CPU and is capable of parallelizing and scaling to all the available cores. Hyper technology is designed to scale to many cores efficiently, and also to maximize the use of each single core as much as possible. This means that you can expect to see the CPU being fully used during query processing. Adding more CPU is expected to result in performance improvement.

Modern operating systems such as Microsoft Windows, Apple macOS, and Linux have mechanisms to make sure that even if a CPU is fully used, incoming and other active processes can run simultaneously. In addition, to manage overall resource consumption and to prevent overloading and completely starving other processes running on the machine, the new Data Engine monitors itself to stay within the limits set in the Tableau Server Resource Manager (SRM). Tableau Server Resource Manager monitors the resource consumption and notifies Data Engine to reduce the usage when it exceeds the predefined limit.
Since the new Data Engine is designed to utilize the available CPU, it is normal to see spikes in CPU usage at times. If however, you see high CPU usage (ex: 95%) for extended periods of time (an hour or more), this can mean a couple of things:

- There is a high load of queries. This can happen if a server is under stress due to overload of multiple client requests and the queries are queuing up. If this happens often, it is an indication that more hardware is required to serve the clients. Adding more CPU in this case should help to improve performance.

- There is one long running query. In this case, the Tableau Server resource Manager will stop long running queries based on the timeout settings. This was also true for the Tableau Server versions earlier than 10.5.

Memory usage of the new Data Engine depends on the data required to answer the query. The Data Engine will try to run this in-memory first. A working set memory is allocated to store an intermediate data structure during query processing. In most cases, systems have enough memory to do these types of processing, but if there isn’t enough available memory, the Data Engine shifts to spooling by writing to disk. Spooling, therefore is an indication that more memory may be needed. Memory usage should be monitored and upgraded appropriately to avoid performance issues caused by spooling.

To manage memory resources on the machine, the maximum memory limit for Data Engine is set by Tableau Server Resource Manager (SRM).

**Changes to Server configuration, topology, and extracts**

This section describes changes introduced in Tableau Server 10.5 with the new Data Engine.

**Server configuration and topology**

- In versions 10.4 and earlier, it was possible to configure more than one instance of Data Engine on a node. Beginning in Tableau Server 10.5, a single instance of Data Engine is automatically installed per node where an instance of File Store, Application
Server (VizPortal), VizQLServer, Data Server, or Backgrounder is installed. You can no longer configure multiple instances of Data Engine on a node. The new Data Engine can scale by itself and uses as much CPU and memory as needed, thus removing the need for multiple instances of the Data Engine. For more information on the server processes, see Process Reference.

- The instance of Data Engine installed on the node where File Store is installed is used for querying data for view requests. The instance of Data Engine installed on the node where backgrounder is installed is used for extract creation and refreshes. This is an important consideration when you are doing performance tuning. For more information, see Performance Tuning Examples.

- Data Server, VizQL Server, and the Application Server (VizPortal) all use the local instance of Data Engine to do cross-database joins and create shadow extracts. Shadow extract files are only created when you work with workbooks that are based on non-legacy Excel or text, or statistical files. Tableau creates a shadow extract file in order to load the data more quickly.

- In Tableau Server 10.5 one instance of Data Engine is installed automatically when you install backgrounder. The backgrounder process uses the single instance of Data Engine (hyperd.exe) installed on the same node.

**Scalability with the new Data Engine:**

You can scale up with the new Data Engine: Since cores are fully utilized, adding more cores makes individual queries execute faster which in turn allows for more queries to execute in less time.

Memory usage should be monitored and upgraded appropriately to avoid the performance issues caused by spooling.

For more information on Scalability, see Tableau Server Scalability.
Extracts

Beginning with version 10.5, new extracts use the new .hyper format. However your existing extracts will not automatically be upgraded to the new .hyper format when you upgrade your Tableau Desktop or Tableau Server. The new Data Engine can read both the .tde and .hyper file formats, so you can continue to view workbooks that have .tde extracts. However, when certain extract tasks are performed on the .tde extract, the extract is then upgraded to a .hyper format. An automatic refresh will also upgrade the .tde extract to a .hyper format.

For more information on extract upgrades, see Extract Upgrade to .hyper Format.

Use the Extract API 2.0 to create (.hyper) extracts. Extract API 2.0 requires Tableau 10.5 and later. If you are using Tableau 10.4 and earlier, use the Tableau SDK to create .tde extracts. For more information, see Extract API 2.0.

Note: Tableau SDK is no longer being maintained.

Extract File Size

While there may be some cases where .hyper extracts become bigger after an extract upgrade, there are other cases when .hyper extracts become smaller after an extract upgrade. In general, the overall disk size requirements should be the same for .hyper extracts as it was for .tde extracts.

Version compatibility between Tableau Desktop and Tableau Server

Upgraded extracts are not backward compatible: Once the extracts are upgraded from .tde to .hyper, they cannot be opened or refreshed in Tableau Desktop versions earlier than 10.5. We recommend that Tableau Server and all the Tableau Desktops in your organization be on the same version and be upgraded to 10.5 on the same schedule.

You may not be able to do some tasks when the Tableau Server and Tableau Desktop are on different versions. For example:
• You cannot publish extracted data sources (.hyper) from Tableau Desktop 10.5 to a Tableau Server 10.4 or earlier.

• You cannot download and open an upgraded .hyper extract on a 10.4 Tableau Desktop.

• You cannot use the downgrade and publish feature with a Hyper extract.

For more information and details, see Impact of extract upgrade.

Performance:

As mentioned in the beginning of this topic, in most cases you can expect to see performance improvements when you upgrade to Tableau 10.5 or later. However, in some cases you may see performance issues like slower response times, especially for workbooks with complex calculations. To test your workbooks to see if should expect performance issues, we recommend that you test this before you upgrade. For more information, see Test Workbook Performance. If you have already upgraded your Tableau Server, see Troubleshoot Performance of Workbooks with Calculations.

Installing Tableau Server

If you are installing a new instance of Tableau Server (not performing an upgrade), you do not need to make any special considerations for the new Data Engine. See Install and Configure Tableau Server.

Upgrading to Tableau Server 10.5

Upgrading to Tableau Server 10.5 from beta versions of Tableau Server on Linux is not supported.

Server Administrator Overview

Tableau Server on Linux integrates with a number of components in your IT infrastructure to provide a unique self-service data analytics culture for your users. It’s important that you, as a server administrator, understand how Tableau Server fits into your IT infrastructure.
The topics in this section provide information on planning, deploying, tuning, and managing Tableau Server.

This topic provides a brief overview of how to think about Tableau Server and how it interacts with your existing IT infrastructure.


Architectural overview

Tableau Server is a collection of processes that work together to provide a full self-service analytic platform for your users. The following diagram shows a high-level architectural view of Tableau Server.

Multiple server processes (shown in blue above) work together to provide services at various tiers. The Gateway process is the component that redirects traffic from all Tableau clients to the available server nodes in a cluster.
Data Services is a logical grouping of services that provide data freshness, shared metadata management, governed data sources, and in-memory data. The underlying processes that power Data Services are the Backgrounder, Data Server and Data Engine processes.

Analytics Services, composed of the VizQL and Cache Server processes, provide user-facing visualization and analytics services and caching services.

Sharing and Collaboration, and Content Management Service are powered by the Application Server process. Core Tableau Server functionality such as user login, content management (projects, sites, permissioning, etc.) and administration activities are provided by the Application Server process.

All of the above services use and rely on the Repository process, which contains structured relational data like metadata, permissions, workbooks, data extracts, user info, and other data. The File Store process enables data extract file redundancy across the cluster and ensures extracts are locally available on all cluster nodes. Under heavier loads, extract files are available locally across the cluster for faster processing and rendering.

Tableau’s architecture is flexible, allowing you to run the platform just about anywhere. You can install Tableau Server on-premises, in your private cloud or data center, on Amazon EC2, on Google Cloud Platform, or on MS Azure. Tableau analytics platform can also run atop virtualization platforms. We recommend you follow the best practices for each virtualization platform to ensure the best performance from Tableau Server.

**Tableau and your data**

When you install Tableau Server into your organization, it becomes a core component of the analytics pipeline to the data your users need. It's important to understand how Tableau Server interacts with your business data. Specifically, Tableau Server can store extracts of data in your organization. It can also connect to live data sources. How you choose to provide the data to your Tableau users is informed by a number of variables: data source type, user scenario, performance and access requirements, and infrastructure conditions.
Tableau Server has not been architected as a data warehouse server where static, native data files are housed. In fact, using Tableau Server as a traditional data warehouse is a poor use of your investment. Rather, when it comes to data storage, we recommend hosting optimized data extracts on Tableau Server. While a data extract is often a subset of a larger data source in your organization, you can also create extracts for data sources that are overtaxed during work hours by scheduling the extract refresh for off-hours.

Extracts are also useful for modeling data or to enable highly-performant visualization authoring. For example, to improve visualization authoring and interaction performance you may optimize extracts by filtering the source data to the essential fields for a given department or project.

Tableau Server also provides direct, authorized access to live data sources, allowing users to build and run complex filtered queries against a variety of connected data sources. For this scenario, Tableau requires highly performant network access to the data sources in your organization and to those in the cloud. Tableau Server and the target data sources also need to be properly sized to handle the processing load required by high-volume, complex data operations.

**User access**

Tableau Server is also a web-based collaboration platform, where users connect to share, view, and interact with data visualizations and data sources from a variety of devices. This means that Tableau Server must be accessible to Tableau users within your local protected network. You can also extend access to data visualizations to desktop, mobile, and authenticated web users outside your organization.

Tableau Server integrates with the following user authentication solutions: LDAP/Active Directory, SAML, OpenId, and Kerberos.

**Where should I install Tableau Server in my network?**

Because of the highly-sensitive nature of most data that organizations manage with Tableau Server, and because Tableau Server requires access to internal data stores, Tableau
Server must be run inside a protected network. Authenticated access from the internet is configured to connect to Tableau Server through a reverse proxy or a VPN solution.

Some organizations embed Tableau views in public webpages, or, for internal users, on generic web servers on the internal network.

Tableau Server can be configured to support such scenarios with either authenticated or anonymous access. For authorized access, where users can only view underlying data to which they have permission, you can configure trusted tickets with a generic web server. In this scenario, Tableau Server authorizes access to the underlying data in an embedded view. This scheme enables you to host interactive data visualizations on a web server in a DMZ or outside the protected network.

Anonymous access to embedded Tableau views requires that you enable "guest user" for Tableau Server. Guest user also requires that you license Tableau Server according to the number of cores you are running, rather than a named-user (interactor) model.

Sizing and scalability

Depending on the size and data usage in your organization, you can scale Tableau Server up or out. As you scale your server, you can also selectively allocate resources to meet your data needs and user needs.
When you scale up Tableau Server, you add hardware resources to a single server. For example, you might increase the memory and processing power of the computer running Tableau Server.

When you scale out Tableau Server, you add computers (or nodes). To create a highly available deployment with failover, you need at least three nodes. For example, you might run most CPU-intensive server processes on two nodes and use the third node for the gateway and coordination controller services.

Whether you scale up or scale out, you can selectively allocate resources by configuring the number and type of server processes that run. If your organization has a lot of data and creates a lot of data extracts, you can increase the number of processes that are dedicated to refreshing and storing extracts. Alternatively, if your organization wants to optimize for heavy user loads, you can increase the number of processes dedicated to responding to user requests. Additionally, you can integrate Tableau Server into industry-standard network load balancers to further optimize your server for user requests.

**Tableau Server management model**

Tableau Server has been designed to support a management scheme with two high-level administrators: server administrator and site administrator. In small organizations, these roles may be assumed by the same person or team, but in larger organizations, the roles often diverge.

In this model, server administrators are IT professionals who maintain and deploy heterogeneous server solutions. Essential areas for server administrators may include networking, hardware tuning and maintenance, security and access, and managing users and directory services. The tools and documentation that we deliver with Tableau Server for the server administrator support these core server IT areas.

Site administrator, on the other hand, is an administrative role specific to Tableau Server or Tableau Online deployments. The Tableau site administrator is fundamentally concerned with data content. The site administrator manages users and their access to projects, work-
books, and data sources. To learn about sites and how to plan your deployment for them, see What is a site

Management tools

Tableau Server includes a number of toolsets for managing the system:

- **Tableau Server administrator page**: This is the web-based administrative site that is installed on each Tableau Server instance. Tasks performed on the administrator page are day-to-day tasks for both **server** and **site** administrators. Server-related tasks include creating sites and site administrator accounts, optionally importing users, setting up synchronization with directory services, setting up extract refresh schedules, monitoring server performance and usage, and other global settings.

  Site-related tasks include managing content and assign permissions, running extract refreshes, create groups and projects, monitoring site activity, optionally adding users, and other content-related tasks.

  Permissions required for the Tableau Server administrator page are role based. The roles are generated and managed by Tableau Server.

- **tsm Command Line Reference** - This is the primary interface for server-wide configurations. The configurations made with TSM CLI are rarely revisited after initial configuration. For example: SSL, subscriptions, data caching, service account, SMTP alerting, user authentication, and single-sign on configuration are all performed with TSM CLI.

- **tabcmd**: You can use the tabcmd command-line utility on a Windows computer to create scripts to automate administrative tasks on your Tableau Server site. For example, use tabcmd for creating or deleting users, projects, and groups.

- **REST API**: With the Tableau Server REST API you can manage and change Tableau Server resources programmatically, via HTTP. The API gives you simple access to the functionality behind the data sources, projects, workbooks, site users, and sites on a Tableau server. You can use this access to create your own custom applications or to script interactions with Tableau Server resources.
Security

As an application server connecting to data that may be highly-sensitive, Tableau Server supports and implements a number of industry security standards. Our server admin documentation includes best practices and implementation for user authentication, authorization, data security, and network security.

For more information about security audit compliance, vulnerability reporting, and other security resources, visit http://www.tableau.com/security.

Tableau Services Manager Overview

This section provides an overview of Tableau Services Manager (TSM), which you can use to configure and administer Tableau Server. TSM is new in this version of Tableau Server.

- Functionality
- Components
- Authentication
- Connecting

In the initial version 10.5 release of TSM, you configure and administer Tableau Server using the TSM command line interface.

Functionality

TSM replaces the following tools from previous versions of Tableau Server:

- Tableau Server Configuration utility
- tabadmin command line utility
- Tableau Server Monitor

For example, you use TSM to perform the following tasks:

- Initial configuration of Tableau Server after installation
- Ongoing configuration management, including editing settings and changing the server topology
- Running administrative tasks such as backup, restore, ziplogs, and more
To learn about how tabadmin commands map to TSM command-line commands, see Migrate from Tabadmin to the TSM CLI.

Components

TSM consists of services and clients.

TSM services are administrative services which manage Tableau Server services. TSM services run continuously in your environment after TSM is initialized, even when the rest of Tableau Server is offline.

TSM services include the following:

- TSM Controller
- TSM Agent
- Licensing Service
- Coordination Service (based on Apache Zookeeper™)

You make requests to these services using the TSM command line interface, the client that is supported in the initial version 10.5 release of TSM. You can also make requests to these services using the prerelease (Alpha) Tableau Services Manager API (TSM API), a public REST API.

Authentication

Whether you use the TSM command line interface or the TSM API, you need to authenticate to Tableau Server before you can perform administrative tasks.

You authenticate to TSM with a user that exists on the Tableau Server computer. This user account must use password-based authentication and must be a member of the tsmadmin group that is authorized to access TSM. This user account is distinct from Tableau Server user accounts, including Tableau Server administrators and site administrators.
Connecting

As a security measure, you can only connect to TSM using HTTPS. This is because TSM allows you to perform administrative tasks and to connect to TSM from other computers.

The HTTPS connection relies on a self-signed certificate generated by the Tableau Server installer. Although the security of this HTTPS connection is not reduced by using a self-signed certificate, it does require that the CLI is configured to trust the self-signed certificate. This is done by default as part of the installation process, so you can access TSM from the local computer without additional configuration.

Security

Infrastructure Planning

Confirm Requirements

Before you install Tableau Server, confirm that you meet the following requirements.

- Hardware requirements
- Operating system requirements
- Resource requirements
- Additional requirements
  - Package requirements

Hardware recommendations

The following list describes the hardware recommendations for a single node installation of Tableau Server:

- 8 core, 2.0 GHz or higher processor
- 64-bit processor architecture
• 32 GB memory

• 50 GB disk space available

The core Tableau Server bits must be installed in the /opt directory where there is at least 15 GB of free disk space. You cannot change the installation path for Tableau Server. If you attempt to install Tableau Server on a computer that does not have enough space, the Tableau Server package will install, but you will be unable to continue with setup.

If you plan to make heavy use of extracts then you may need to allocate additional disk space. You can specify a different directory for data (extract) storage during installation. For more information, see Disk Space Requirements, and Install Tableau Server Package.

To see the full list of recommendations and to see the minimum hardware requirements, see Minimum Hardware Requirements and Recommendations for Tableau Server.

For public cloud deployments on Amazon Web Services and Google Cloud Platform, their “vCPU” is actually a CPU hyper-thread, and not a full CPU core. When sizing cloud instances, you will need twice as many vCPU as the Tableau Server CPU core requirements given (4 vCPU for a minimum trial installation, 16 vCPU for a recommended installation).

Operating system requirements

The following distributions of Linux are supported:

• Red Hat Enterprise Linux (RHEL) 7.3 and higher, CentOS 7.3 and higher, Oracle Linux 7.3 and higher.

Version 8.x is not supported.

These are collectively referred to in this documentation as RHEL-like.
• Ubuntu 16.04 LTS only.  

Version 17.04 is not supported.

Previous versions of CentOS and Ubuntu are not supported because Tableau Server requires systemd for process management.

Custom kernels are not supported.

In a multi-node installation of Tableau Server, all of the computer nodes where you are installing Tableau must run Linux and the same distribution of Linux.

Note: You cannot install Tableau Server on a combination of Linux and Windows computers.

Resource requirements

The initial release of Tableau Server on Linux is ready for organizations that are familiar with Linux and have strong infrastructure expertise at integrating Linux applications into their environment. It also helps if your organization has some experience with running Tableau Server in a production environment.

Use the following set of questions to help you assess whether this version of Tableau Server for Linux is a suitable fit for your organization. If it looks like this version won’t be a good fit, circle back for the next version, which will include more automation for installation and configuration.

<table>
<thead>
<tr>
<th>Linux OS knowledge</th>
<th>Do you currently administer other applications on Linux?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shell admin experience</td>
<td>Are you accustomed to using command line shells and editing config files to administer applications?</td>
</tr>
</tbody>
</table>
| Availability of Linux integration | Does your organization's IT infrastructure provide ready access to assistance integrating Linux applications? This can be things like directory serv-
<table>
<thead>
<tr>
<th>ration support</th>
<th>vices, networking, authentication, virtual machine/cloud platform, and operating system.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux distribution</td>
<td>Are you planning or able to use one of the Linux distributions Tableau tests on and supports?</td>
</tr>
<tr>
<td>Datasources</td>
<td>Are all the data connections that you need supported on Linux? (See Install Drivers on Linux.)</td>
</tr>
</tbody>
</table>

**Additional requirements**

Make sure that your environment also meets the following additional requirements:

**Hostname**

The hostname of the computer where you install Tableau Server must be unique in your environment, and once you have installed Tableau Server the hostname cannot change. Additionally, your domain name server (DNS) must be able to resolve the hostname.

**Static IP address**

The computer where you install Tableau Server must have a static IP address.

**Database drivers**

To connect to specific data sources, the computer where you install Tableau Server must have the correct database drivers installed. For more information, see Install Drivers.

**Available Ports**

TSM and Tableau Server each require an available TCP port in order for you to access them. TSM defaults to port 8850, and the Tableau Server Gateway service defaults to port 80 (both ports are configurable). We strongly recommend that you ensure that both port 8850 and 80 are not in use on your Linux system before installing Tableau Server. If those ports are not available, the TSM and gateway ports may be dynamically remapped to
different port numbers, and there is currently no interface for displaying which port they have been remapped to.

**Sudo access**

All installation tasks and administrative tasks for Tableau Server must be run as root. Often this is accomplished using the `sudo` command, but running the commands directly as the root user is also possible.

To install Tableau Server with the root account, you must specify a user account during installation. The account will be used for managing TSM. Specify the account by running the `initialize-tsm` script with the `-a` option. See Help Output for `initialize-tsm` Script.

**Account password**

The user account that you use to install and administer Tableau Server must be able to authenticate with a password. That is, the user must not use another means of authenticating (for example public key authentication).

If the account you are using to install and initialize Tableau Server does not have a password, you can set one using the `passwd` command:

```
sudo passwd $USER
```

**Port access requirements**

If you want to install Tableau Server remotely, for example by means of SSH, ensure that the following ports are open:

- **8850.** The port used for the Tableau Services Manager (TSM) web interface. You can use this interface to configure Tableau Server.
- **80.** The port used for the main Tableau Server web interface.

The Tableau Server installer attempts to open these ports during the installation process, but it can only open these ports for the `firewalld` firewall. If your computer runs another firewall, you must open the ports before you install.

**Virtual Container environments**
Installation of Tableau Server in virtual container environments such as Docker have not been tested and are not supported. Tableau Server on Linux will not function as expected if installed in these environments.

Package requirements

Systemd

Tableau Server requires systemctl to manage services. This package is installed by default on CentOS 7 and Ubuntu 16. If you decide to test Tableau Server on a modified version of these distributions, you can run the following command to confirm that systemctl is installed:

`whereis systemctl`

If systemctl is installed, the installation location is displayed. For example, you might see the following output:

```
systemd: /usr/lib/systemd /etc/systemd /usr/share/systemd /usr/share/man/man1/systemd.1.gz
```

If you have systemctl installed but the Tableau installer is failing requirements checks for systemctl, it’s likely that systemctl is not running. To verify that systemctl is running, run the following command:

`ls /run/systemd`

The output will be a list of files and directories. If systemctl is running, the output will include system. If systemctl is not in the output, then systemctl is not running.

*Continue to the next step: Install and Configure Tableau Server.*

Disk Space Requirements

In general, when estimating the amount of additional disk space to allocate for Tableau Server for day-to-day usage, you must consider whether or not extracts will be published to Tableau Server, and consider the number of workbooks that you expect to publish to
Tableau Server. If you anticipate using extracts, Tableau recommends that you begin with a few hundred gigabytes (GB). If you do not anticipate using extracts, you may only need around 50 GB to fulfill your usage needs.

Looking for Tableau Server on Windows? See [Disk Space Requirements](#).

Here are the factors that affect disk space requirements:

Publishing extracts to Tableau Server

Consider the number of extracts that will be published to Tableau Server and the size of each extract. Test your needs by publishing several extracts to Tableau Server, and then checking the disk space used. You can use this amount of disk space to help you figure out how many extracts will be published to Tableau Server over time as well as how each existing extract will increase in size.

Refreshing extracts

Consider the space needed by the temp directory during an extract refresh. The temp directory, which is where an extract is stored to during a refresh, may require up to three times the final file size of the extract.

While there may be some cases where .hyper extracts become bigger after an extract upgrade, there are other cases when .hyper extracts become smaller after an extract upgrade. In general, the overall disk size requirements should be the same for .hyper extracts as it was for .tde extracts.

Creating many workbooks

If using workbooks, consider the number of workbooks that will be published to Tableau Server. Individual workbooks tend to take up a small amount of disk space. However, if you anticipate thousands of workbooks being published, you may want to allocate additional disk space to accommodate those workbooks.
Logging

To assist with daily management and troubleshooting, Tableau Server creates log files as a part of its normal operations. Depending on the level at which the logging is configured, it can significantly impact the amount of disk space necessary on the Tableau Server computer.

Recommended Baseline Configurations

Determining the topology (number of nodes, number of Tableau Server processes) of your Tableau Server deployment requires you to consider these variables: your environment, sources of data and management to provide self-service data access, workload, and usage. However you may not have enough information about these variables when you deploy Tableau Server for the first time. This topic describes three baseline architectures that can be used as starting points for your Tableau Server installations.

In this article

Hardware Recommendations
Baseline Configurations
Beyond Baseline Configurations

Hardware Recommendations

The hardware recommendations seen below are based on the hardware that the Tableau team uses to test Tableau Server scalability. We suggest that you use these recommendations as starting points for your production deployments. For proof-of-concept deployments, see Minimum Hardware Requirements and Recommendations for Tableau Server.

<table>
<thead>
<tr>
<th>Install Type</th>
<th>Processor</th>
<th>CPU</th>
<th>RAM</th>
<th>Free Disk Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Server</td>
<td>64-bit</td>
<td>8-core (phys-</td>
<td>64 GB</td>
<td>500 GB - 1 TB</td>
</tr>
<tr>
<td>Install Type</td>
<td>Processor</td>
<td>CPU</td>
<td>RAM</td>
<td>Free Disk Space</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------</td>
<td>----------------------</td>
<td>------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Multi-node and enterprise deployments</td>
<td></td>
<td>(8GB/physical Core)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(8GB/physical Core)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(8GB/physical Core)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** For deployments using virtual machines, Tableau recommends dedicated CPU affinity. If you are running Tableau Server in a virtual environment, use your VM host’s best practices for vCPU allocation in relation to the number of physical CPU cores on the VM host. Typically 2 vCPUs = 1 physical core for Tableau Server. For example, for AWS installations, the 4 core minimum recommendation is equivalent of 8 AWS vCPUs. Similarly, follow the best practices provided by your virtual infrastructure provider to make sure Tableau Server has access to the appropriate compute, memory, and data resources. If you are installing Tableau Server in a virtual environment or a cloud-based deployment, see Virtual Machines and Public Cloud Deployments section later in this topic.

Estimating Disk Space

There are several factors that affect disk space requirements, including whether or not you will be publishing extracts and the number of workbooks to Tableau Server. For more information see Disk Space Requirements.

Baseline Configurations

- Single Server Installations
- Two Node Installations (Specialized for extract heavy environments)
- High Availability Installations (HA)
Single Server Installations

Recommendations

We recommend using a single machine to install your Tableau Server for initial deployments with limited usage and are not mission critical. Single Server installations can also be expanded to multi-node installation as your workloads grow.

Here are some instances when a single server installation may not be right for you:

- If your system is considered mission critical and needs to be highly available. High availability is about minimizing the system downtime. It is achieved by eliminating single points of failure, and having a reliable failover mechanism. Tableau Server requires a minimum of a three-node configuration to provide redundancy and eliminate the single points of failure. This is one of the primary reasons to move to a multi-node configuration.

- If you have a lot of active users and a lot of extract refreshes, the two types of loads may be competing for the same resources on the machine. In such a scenario, a single server configuration may not be the right option as you may need additional specialized nodes to isolate the different workloads.

**Note:** Active users represent the interactive, concurrent requests made to Tableau Server, including consuming dashboards on a laptop or mobile device, web authoring, and connecting to and querying Published Data Sources.
Server Configuration

**Stand-alone Tableau Server node**

- Stand-alone single server node with all the processes installed on one machine.
- Below are the number of processes for an 8 core machine:
  - VizQL Server: Set to 2 instances (Number of physical cores divided by 4, up to a maximum of 4).
- Backgrounder, Cache Server, and Data Server: Set to 2 instances.
- All other processes, only one instance of the process is installed, regardless of hardware.

Distributed Installations

Running Tableau Server on more than one machine is called distributed installation, or a cluster. There are various reasons why you might want to have a distributed installation. For example, you may have heavy extract environments which can mean dedicating some hardware resources to Backgrounder process. For systems that have high availability requirements, you need a distributed environment that has at least three nodes.

Two Node Installations (Specialized for extract heavy environments)

Recommendations

Start with a two node configuration when the following conditions apply to you:

- **Extract heavy environment**: Majority of your data sources are extracts. Having just a few, extremely large extracts could put your deployment in this category, as would having very many small extracts.

- **Frequent extract refreshes**: Refreshing an extract is a CPU-intensive task. Deployments where extracts are frequently refreshed (for example, several times a day during business hours) are often helped by more emphasis on the background process, which handles refresh tasks.

**Important**: Two-node configurations do not meet the minimum requirements for high availability. If you need a system that is highly available, see High Availability Installations (HA).
On the initial node, install all the processes except for the backgrounder. Below is the number of instances of the processes for an 8 core machine:

- VizQL Server: Set to 2 instances. (default calculation: Number of physical cores divided by 4, up to a maximum of 4).
- Cache Server, and Data Server: Set to 2 instances.
- Backgrounder: Minimum 2, maximum 4.
All other processes, only one instance of the process is installed, regardless of hardware.

Isolate backgrounder on the additional node. To calculate the minimum number of backgrounder processes to run, divide the computer’s total number of physical cores by 4. To calculate the maximum number, divide the computer’s total physical cores by 2. When you install the backgrounder, Tableau Server automatically installs one instance of the Data Engine.

As you monitor and gather data about the performance and usage, you can fine tune and configure the number of instances for these processes. For example, initially, you can set the number of backgrounders to minimum (total number of cores divided by four), and increase the number of backgrounder processes later if you find that:

- Extract refreshes are taking a long time to complete
- Subscriptions and alerts are not completing on time

For more information on performance tuning, see Performance Tuning Examples topic.

High Availability Installations (HA)

Recommendations

A highly available installation of Tableau Server is a distributed installation that is designed to maximize the availability of Tableau Server. High availability basically means that the system is available with minimal amount of downtime. To build in redundancy for HA related items such as repository, file redundancy, and failover, you need a minimum of three nodes. The tolerance for downtime will vary for each organization and depends on the SLAs you have established in your organization.

High availability is achieved by eliminating single points of failure and detecting failures and setting up a reliable failover system. HA in Tableau Server is mainly achieved by:

- File redundancy with multiple File Store/Data Engine instances.
- Active/Passive Repository across two nodes.
• Creating a backup primary (Windows install).

• Adding an external load balancer to make sure your installation is robust to Gateway failures and make sure that requests only get routed to functioning Gateway processes.

Server Configuration

Three-node configuration:

• To build in redundancy, you need to add additional nodes to host instances of the repository and File Store/Data Engine processes. You can add instances of other processes, including multiple instances of a process on a node.

• To reduce the system’s vulnerability, you can run multiple gateways and additional
instances of some of the server processes. The fewest number of computers required to achieve this configuration is three.

- The repository has also been moved from the initial node to one of the additional nodes, and a second, passive instance has been added to the other new node.

**NOTE:** In certain circumstances you may want to limit the processes running on your initial node. Reasons for doing this include wanting to run as few processes as possible on the node to limit processing requests on the node. You might also remove licensed Tableau Server processes from the node if you have a core-based license and do not want the initial node cores to count against your core use. For more information on licensed processes, see Process Reference.

Virtual Machines and Public Cloud Deployments

In general, the considerations and recommendations described in this topic apply to virtual environment and cloud deployments.

If you are running Tableau Server in a virtual environment, use your VM host’s best practices for vCPU allocation in relation to the number of physical CPU cores on the VM host. Typically 2 vCPUs = 1 physical core for Tableau Server. For example, for AWS installations, the 4 core minimum recommendation is equivalent of 8 AWS vCPUs.

For more information on cloud-based deployments, see:

- Install Tableau Server in the AWS Cloud.

For more information on installing Tableau Server on VMware, see Tableau Server on VMware VSphere.
Beyond Baseline Configurations

Disaster Recovery Considerations

While HA configurations reduce downtime, you may still encounter failures in case of a disaster or hardware failures. In addition to the above considerations, you should evaluate the importance of disaster recovery in your organization and plan for a deployment that helps you meet your disaster recovery goals and objectives.

When planning for disaster recovery (DR) in your Tableau environment, there are two main factors to consider:

- Recovery Time Objective (RTO), a measure of how much downtime your business can accept before a full recovery.
  - Influences how often you restore your backups to an alternative cluster and the amount of infrastructure investment.

- Recovery Point Objective (RPO), a measure of how much data loss your business can tolerate.
  - Influences how often you will need to take backups of your system.
  - For Tableau Server the RPO cannot be shorter than the time it takes to complete a full backup of your server.

The diagram below illustrates how to plan for a range of RTO requirements:
Tableau Server Scalability

These baseline configuration may not be enough as your needs change and grow, and you might need to scale your Tableau Server beyond these configurations. Like other enterprise platforms, Tableau Server scales up by adding processor, memory, and disk to existing node, and scales out by adding more nodes to a cluster. However, scalability and performance are heavily dependent on external systems and user activity. The configuration of your Tableau Server can be different depending on your requirements and variables:

For more information on Tableau Server scalability and the variables affecting scalability, see Tableau Server 10.5 Scalability whitepaper.
Identity Store

Tableau Server requires an identity store to store user and group information. There are two kinds of identity stores: local and external. When you install Tableau Server you must configure either a local identity store or an external identity store. For information about configuration options for the identity store, see identityStore Entity.

Local identity store

When you configure Tableau Server with a local identity store, all user and group information is stored and managed in the Tableau Server repository. In the local identity store scenario, there is no external source for users and groups.

External identity store

When you configure Tableau Server with an external store, all user and group information is stored and managed by an external service. Tableau Server must synchronize with the external identity store so that local copies of the users and groups exist in the Tableau Server repository, but the external identity store is the master source for all user and group data.

Active Directory is an example of an external user store. Tableau Server is optimized to interface with Active Directory. For all other external stores, Tableau Server supports LDAP as a generic way to communicate the identity store. For example, OpenLDAP is one of several LDAP server implementations with a flexible schema. Tableau Server can be configured to query the OpenLDAP server. To do so, the directory administrator must provide information about the schema.

If you have configured the Tableau Server identity store to use LDAP or Active Directory, then all users (including the initial admin account) that you add to Tableau Server must have an account in the directory.
LDAP bind

Clients that wish to query a user store using LDAP must authenticate and establish a session. This is done by binding. There are multiple ways to bind. Simple binding is authenticating with a username and password. For organizations that connect to Tableau Server with simple bind, we recommend configuring an SSL encrypted connection, otherwise the credentials are sent over the wire in plaintext. Another type of binding Tableau Server supports is GSSAPI binding. GSSAPI uses Kerberos to authenticate. In Tableau Server’s case, Tableau Server is the client and the external user store is the LDAP server.

LDAP with GSSAPI (Kerberos) bind

We recommend binding to LDAP directory with GSSAPI. To bind with GSSAPI you will need a keytab file specifically for the Tableau Server service. In the context of Kerberos, GSSAPI bind is all you need during the base installation of Tableau Server. After you install the server, you can then configure Kerberos for user authentication and Kerberos delegation to data sources.

If you are installing into Active Directory, and the computer where you are installing Tableau Server is already joined to the domain, then the computer may already have a configuration file and a keytab file. In this case, the Kerberos files are for Linux operating system functionality and authentication. Strictly speaking, you can use these files for GSSAPI bind, but we don’t recommend using them. Instead, contact your Active Directory administrator and request a keytab specifically for the Tableau Server service.

LDAP over SSL

By default, LDAP with simple bind is not encrypted. If you are configuring LDAP with simple bind, we strongly recommend that you enable LDAP over SSL (LDAPS).

If you already have certificates installed for LDAP on the computer running Tableau Server, then LDAPS should work with minimal configuration during the installation process. Specifically, if you have installed Tableau Server on Redhat Enterprise Linux (RHEL) or Debian systems, and you have valid certificates installed in the default keystore, then you can specify SSL when you configure the identity store.
In the LDAP scenario, on RHEL and Debian systems, Tableau Server trusts the default trusted keystore. It's worth noting that in most cases, each individual service that comprises Tableau Server has its own trusted keystore. However, in the case of the LDAP service, Tableau Server trusts the default trusted keystore. The default password for the Java cacert on RHEL and Debian systems is changeit.

Default trusted keystores are located as follows:

- **RHEL systems**: /etc/pki/ca-trust/extracted/java/cacerts
- **Debian systems**: /etc/ssl/certs/java/cacerts

If you do not have certificates already in place on your computer that are configured for the LDAP server then you must obtain a SSL certificate for the LDAP server and import it into the default system keystore.

Use the "keytool" Java tool to import certificates. In a default installation, this tool is installed with Tableau Server at /opt/tableau/tableau_server/packages/<installer version>/jre/bin/keytool.

The following command (for RHEL-like distributions) imports the certificate:

```
sudo keytool -import -file <path to cert file> -alias <alias of cert> -keystore <path to keystore>
```

System users and groups

Tableau Server on Linux uses two users, and two groups for proper operation.

Users

The initialize-tsm script can automatically create these users and groups locally during installation. See Help Output for initialize-tsm Script. The users and groups can be local or from an LDAP directory service. If you want to use existing user accounts in your LDAP directory, or if you want to create new user accounts in your LDAP directory for Tableau Server then follow these recommendations.
The two users (a privileged user and an unprivileged user) must be configured with the following characteristics:

- No passwords: either the accounts have a random password that nobody knows, or they have an unmatchable password.
- Primary (and only) group membership set to the shared group (see below).

For improved security, we recommend configuring the users with the following characteristics:

- Shell set to `/sbin/nologin`.
- Home directory set to `/`.

The default name for the privileged user is `tsmagent` and for the unprivileged user it is `tableau`.

When locally created, user entries in `/etc/passwd` look like this:

- `tableau:x:993:991:Tableau Server://:/sbin/nologin`
- `tsmagent:x:992:991:Tableau Services Manager Agent://:/sbin/nologin`

Example user entries in `/etc/shadow` are as follows:

- `tsmagent:!!:17423::::::`
- `tableau:!!:17423::::::`

Groups

There is a shared group used by the two users, which must be the primary group for the two users. By default, this group is named `tableau`. As a convenience, any human administrator account can also be added to this group to be able to read the Tableau Server log files (without becoming root).

The final group is used to authorize which users are allowed to use the REST API of the Tableau Services Manager (TSM), which is required to initialize and maintain Tableau Server. Any user in this group will be able to send commands to TSM, so it should be restricted to Tableau Server administrators. By default, this group is named `tsmadmin`. 
If there is an existing LDAP group that matches these criteria, it can be used. Otherwise, a new LDAP group could be created, and all Tableau Server administrators can be added to this new group. Note that at least one administrator must be in this group in order to successfully initialize Tableau Server.

**Authenticating clients**

Basic user authentication in Tableau Server is by username and password sign-in for both local and external user stores. In the local case, user passwords are stored as a hashed password in the repository. In the external case, Tableau Server passes the credentials to the external user store and awaits a response as to whether the credentials are valid. External user stores can also handle other kinds of authentication like Kerberos or SSPI (Active Directory only), but the concept is still the same, Tableau Server delegates the credentials or user to the external store and awaits a response.

You can configure Tableau Server such that username-password sign-in is disabled. In these scenarios other authentication methods, such as trusted authentication, OpenID, or SAML can be used. See Authentication.

**User Management in Active Directory Deployments**

This topic describes important technical details that you should be familiar with if you use Active Directory to authenticate users for Tableau Server. This topic assumes that you are familiar with Active Directory user management and basic Active Directory schema and domain concepts.

**Note:** If you are running Tableau Server on Linux then all external directory communication is configured and managed with a LDAP identity store. In the context of user and group synchronization, Tableau Server configured with LDAP identity store is equivalent to Active Directory. Active Directory synchronization features in Tableau Server function seamlessly with properly configured LDAP directory solutions.
Active Directory user authentication and Tableau Server

Tableau Server stores all user names in the Tableau Server identity store, which is managed by the repository. If Tableau Server is configured to use Active Directory for authentication, you must first import user identities from Active Directory to the identity store. When users sign in to Tableau Server, their credentials are passed to Active Directory, which is responsible for authenticating the user; Tableau Server does not perform this authentication. (By default, NTLM is used for authentication, but you can enable Kerberos or SAML for single sign-on functionality—however, in all these cases, authentication is left to Active Directory.) However, the Tableau user names stored in the identity store are associated with rights and permissions for Tableau Server. Therefore, after authentication is verified, Tableau Server manages user access (authorization) for Tableau resources.

Active Directory user name attributes and Tableau Server

Active Directory uniquely identifies user objects using several attributes. (For details, see User Naming Attributes on the MSDN website.) Tableau Server relies on two Active Directory user naming attributes:

- **sAMAccountName.** This attribute specifies the logon name that was originally designed for use with older versions of Windows. In many organizations, this name is combined with the NetBIOS name for authentication, using a format like `example\jsmith`, where `example` is the NetBIOS name and `jsmith` is the `sAMAccountName` value. Due to the original design in Windows, the `sAMAccountName` value must be less than 20 characters.

  In the Windows Active Directory Users and Computers administrative console, this value is in the field labeled User logon name (pre-Windows 2000) on the Account tab of the user object.

- **userPrincipalName (UPN).** This attribute specifies a user name in the format `jsmith@example.com`, where `jsmith` is the UPN prefix and `@example.com` is the UPN suffix.
In the Windows **Active Directory Users and Computers** administrative console, the UPN is a concatenation of two fields on the **Account** tab of the user object: the **User logon name** field, and the domain drop-down list next to it.

### Adding users from Active Directory

You can **add users individually** from Active Directory, either by typing them in the server environment or by creating a CSV file and importing the users. You can also add Active Directory users by **creating a group via Active Directory** and importing all of the group's users. The result can be different depending on which approach you're using.

**Adding users individually**

In most case, Tableau Server uses the `sAMAccountName` value for the user name. When you import users individually from Active Directory (either by typing in their names or by using a CSV file), Tableau queries Active Directory with the user name that you provide. If a match is found, then that name is imported into Tableau Server and it becomes the name that the user enters in order to sign in to Tableau Server.

The user name that Tableau Server will import into the identity store will be the `sAMAccountName` value unless one of the following is true:

- If the user name that you specify is longer than 20 characters.
- If the user name that you specify contains an `@` character.

If the user name you enter meets either of these conditions, then Tableau will import the UPN prefix of the `userPrincipalName` attribute, which will become the user's Tableau logon user name.

If user names were inadvertently imported using UPN names, you can delete the accounts in Tableau Server and then reimport those accounts using the `sAMAccountName` value for the user name, as shown in **User logon name (pre-Windows 2000)** in the Windows **Active Directory Users and Computers** administrative console.
Adding user groups

If you import an Active Directory user group, Tableau will import all users from the group using the `sAMAccountName`.

Sync behavior when removing users from Active Directory

Users cannot be automatically removed from Tableau Server through an Active Directory sync operation. Users that are disabled, deleted, or removed from groups in Active Directory remain on Tableau Server so that you can audit and reassign the user's content before removing the user's account completely.

However, Tableau Server will act upon user objects differently based how the status of that user object changes in Active Directory. There are two scenarios: deleting/disabling users in Active Directory or removing users from synchronized groups in Active Directory.

When you delete or disable a user in Active Directory and then synchronize that user's group on Tableau Server, the following occurs:

- The user is removed from the Tableau Server group you synchronized.
- The user's role is set to "unlicensed."
- The user will still belong to the All Users group.
- The user is unable to sign in to Tableau Server.

When you remove a user from a group in Active Directory and then synchronize that group on Tableau Server, the following occurs:

- The user is removed from the Tableau Server group you synchronized.
- The users role is retained: it is not set to "unlicensed."
- The user will still belong to the All Users group.
- The user will still have permission to the Tableau Server with access to everything that the All Users group is granted permission to use.

In both instances, to remove a user from Tableau Server, the server administrator must delete the user from the Server Users page in Tableau Server.
Domain nicknames

In Tableau Server, domain nickname is equivalent to the Windows NetBIOS domain name. In a Windows Active Directory forest, a fully qualified domain name (FQDN) can have an arbitrary NetBIOS name. The NetBIOS name is used as the domain identifier when a user logs in to Active Directory.

For example, the FQDN west.na.corp.lan might be configured with a NetBIOS name (nickname) of SEATTLE. The user jsmith in that domain could log on to Windows using either of the following user names:

- west.na.corp.example.com\jsmith
- SEATTLE\jsmith

If you want your users to sign in to Tableau Server with a NetBIOS name instead of the FQDN, then you'll need to verify that the nickname value for each domain where users log in is set. See editdomain for information on how to view and set the nickname value for each domain.

Support for multiple domains

You can add users from a domain that's different from the domain of the Tableau Server computer in these cases:

- Two-way trust has been established between the server’s domain and the users’ domain.

- The server’s domain trusts the users’ domain (one-way trust). See Domain Trust Requirements.

The first time you add a user from the non-server domain, use the fully-qualified domain name with the user name. Any additional users you add from that domain can be added using the domain’s nickname, provided the nickname matches the NetBIOS name.
Duplicate display names

If user display names are not unique across multiple domains, then managing users with the same display name in Tableau can be confusing. Tableau Server will display the same name for two users. For example, consider an organization with two domains, example.lan and example2.lan. If user John Smith exists in both domains, then adding that user to groups and other administrative tasks will be confusing in Tableau Server. In this scenario, consider updating the display name in Active Directory for one of the users to differentiate the accounts.

Sign in to Tableau Server with NetBIOS name

Users can sign in to Tableau Server using the domain nickname (NetBIOS name), for example, SEATTLE\jsmith.

Tableau Server cannot query for NetBIOS name for a given FQDN. As a result, Tableau sets the nickname of a given FQDN according to the first entry in the namespace. For example, given the FQDN west.na.corp.lan, Tableau sets the nickname to west.

Therefore, you might need to update the domain nickname on Tableau Server before users can sign in using the nickname. If you do not update the nickname, users will have to sign in using a fully qualified domain name. For more information, see Users From New Domain Unable to Log In and Do Not Appear in User List in the Tableau Knowledge Base.

Domain Trust Requirements

When you run Tableau Server in an Active Directory environment across multiple domains (either in the same Active Directory forest or in different forests), some Tableau functionality is dependent on the trust relationship between the domains. For example, some administrators manage users in domains that are separate from where they deploy server applications, such as Tableau Server. In other organizations, a Tableau Server deployment might be shared with external partners or with different partners in the organization. Finally, Windows-authenticated data sources, such as SQL Server, MSAS, or Oracle, that Tableau Server connects to may also be in other domains.
If it's feasible, we recommend configuring two-way trust between all domains that interact with Tableau Server. If this is not possible, Tableau Server can be configured to support user authentication where a one-way trust has been configured. In this case, a one-way trust between domains is supported when the domain in which Tableau Server is installed is configured to trust the domain where user accounts reside.

The following illustration shows one-way trust between the domain where Tableau Server is installed and the domain where user accounts reside:

![One-way trust diagram]

In this scenario, Tableau Server is in the dev.local domain, and users from the users.lan Active Directory domain are imported into Tableau Server. A one-way trust is required for this scenario; specifically, the dev.local domain is configured to trust the users.lan domain. Users in the users.lan domain can access Tableau Server in the dev.local with their normal Active Directory credentials. However, you may need to update the domain nickname on Tableau Server before users log on with the nickname. Refer to the Tableau Knowledge Base for more information.

Kerberos single sign-on is supported in this one-way trust scenario.

Review User Management in Active Directory Deployments to understand how multiple domains, domain naming, NetBIOS, and Active Directory user name format influence Tableau user management.
Connecting to live data in one-way trust scenarios

In the one-way trust scenario, users connecting to Tableau Server can connect to live data that’s hosted in the cloud or on any other data source on premises that does not rely on Windows authentication.

Data sources that require Windows-authentication might have additional authentication requirements that complicate the scenario, or that can even prevent Tableau Server users from connecting. This is because Tableau Server uses a service account for authentication with such data sources. If you are running Tableau Server in a different domain than data sources that use Windows authentication, verify that the service account that is used for Tableau Server can access the data source.

Communicating with the Internet

In most enterprises, Tableau Server needs to communicate with the internet. Tableau Server was designed to operate inside a protected internal network. Do not set up Tableau Server directly on the internet or in a DMZ. Instead, communications between your network and the internet should be mediated using proxy servers. If the computer running Tableau Server cannot access the internet directly, then you may need to deploy forward proxy servers to mediate traffic from inside the network to targets on the internet.

For inbound traffic, we recommend running Tableau Server behind reverse proxy servers.

How Tableau communicates with the internet

Tableau Server requires outbound access to the internet for these scenarios:

- Working with maps. Tableau uses map data that is hosted externally.

  Tableau Server needs to connect to maps.tableausoftware.com using port 443. If it cannot make this connection, maps may fail to load. You can test connectivity by accessing the Miami and Havana (blue water) map.

  This is the URL:
Licensing. Tableau products connect to the internet to activate license keys. Unless you activate Tableau software with the Offline Activation Tool, all Tableau products must have continuous access to the internet to validate their licenses.

Tableau Server needs to connection to the following internet locations for licensing purposes:

- licensing.tableau.com:443
- o.ss2.us
- ocsp.rootg2.amazontrust.com
- ocsp.rootca1.amazontrust.com
- ocsp.sca1b.amazontrust.com
- crt.sca1b.amazontrust.com
- crt.rootca1.amazontrust.com
- ocsp.sca0a.amazontrust.com
- crt.sca0a.amazontrust.com
- ocsp.sca1a.amazontrust.com
- crt.sca1a.amazontrust.com
- ocsp.sca2a.amazontrust.com
- crt.sca2a.amazontrust.com
- ocsp.sca3a.amazontrust.com
- crt.sca3a.amazontrust.com
- ocsp.sca4a.amazontrust.com
- crt.sca4a.amazontrust.com

Requests to the above domains may be on port 80 or 443.

If Tableau Server cannot make a connection while attempting to activate its license, you will be prompted to do an offline activation.

- Working with external or cloud-based data.

**Configure Tableau Server for a Forward Proxy Server**

To enable communication from Tableau Server to the internet, deploy Tableau Server behind a forward proxy server. When Tableau Server needs access to the internet, it doesn't send the request directly to the internet. Instead, it sends the request to the forward proxy, which in turn forwards the request. Forward proxies help administrators manage traffic out to the internet for tasks such as load balancing, blocking access to sites, etc.

If you use a forward proxy, you must configure the computers that run Tableau Server inside the network to send traffic to the forward proxy.

Configuring Tableau Server on Linux to work with a forward proxy

Run the following procedure after you have installed Tableau Server.

1. Open Bash and set environment variables `http_proxy` and `https_proxy` to point to your proxy host.

   For example, to set the proxy to example-host for ports 80 and 443, run the following commands:
export http_proxy=http://example-host:80/

export https_proxy=http://example-host:443/

Take care to use http when you specify the URL for the https_proxy environment variable. Do not specify the https protocol for the value of the https_proxy environmental variable.

2. To bypass the proxy server, specify exceptions in the no_proxy variable. Use this variable if your proxy server does not route internal addresses. You should also add exceptions to this proxy configuration to guarantee that all communications within a local Tableau Server cluster (if you have one now or will have one later) do not route to the proxy server.

For example, to specify exceptions for internal traffic and to another node in the cluster, run the following command:

export no_proxy="localhost,127.0.0.1,other.node"

Where other.node is the host name of another node in the Tableau Server cluster.

3. Import the environment variables using systemctl. Run the following commands:

sudo systemctl import-environment http_proxy

sudo systemctl import-environment https_proxy

sudo systemctl import-environment no_proxy

4. Restart the TSM services. Run the following commands:

sudo systemctl restart tabadmincontroller_0

sudo systemctl restart tabsvc_0

sudo systemctl restart appzookeeper_0
5. Run `tsm restart` to finish the configuration.

Air Gapped Environment

Tableau Server can run without internet access. For more information about deploying Tableau Server in organizations without access to the internet, see Install Tableau Server in a Disconnected (Air-Gapped) Environment.

Reverse Proxy

In many enterprises, users also need to access Tableau Server from outside the network (that is, from the internet). For example, in many enterprises, users want to be able to reach Tableau Server from their mobile devices in order to interact with views that are stored on the server. To configure access to Tableau Server from the internet or from mobile devices, you should use a reverse proxy. See Configuring Reverse Proxies for Tableau Server.

Configuring Reverse Proxies for Tableau Server

A reverse proxy is a server that receives requests from external (internet) clients and forwards them to Tableau Server. Why use a reverse proxy? The basic answer is security. A reverse proxy makes Tableau Server available to the internet without having to expose the individual IP address of that particular Tableau Server to the internet. A reverse proxy also acts as an authentication and pass-through device, so that no data is stored where people outside the company can get to it. This requirement can be important for organizations that are subject to various privacy regulations such as PCI, HIPAA, or SOX.

How a reverse proxy works with Tableau Server

The following diagram illustrates the communication path when a client makes a request to Tableau Server that is configured to work with a reverse proxy server.
1. An external client initiates a connection to Tableau Server. The client uses the public URL that's been configured for the reverse proxy server, such as https://tableau.example.com. (The client doesn't know that it's accessing a reverse proxy.)

2. The reverse proxy maps that request in turn to a request to Tableau Server. The reverse proxy can be configured to authenticate the client (using SSL/TLS) as a pre-condition to passing the request to Tableau Server.

3. Tableau Server gets the request and sends its response to the reverse proxy.

4. The reverse proxy sends the content back to the client. As far as the client is concerned, it just had an interaction with Tableau Server, and has no way to know that the communication was mediated by the reverse proxy.

Proxy servers and SSL

For better security, you should configure reverse proxy servers to use SSL for any traffic that's external to your network. This helps to ensure privacy, content integrity, and authentication. Unless you've deployed other security measures to protect traffic between your internet gateway and Tableau Server, we also recommend configuring SSL between the gateway proxy and Tableau Server. You can use internal or self-signed certificates to encrypt traffic between Tableau Servers and other internal computers.

Mobile access

Tableau Server adds an X-header to all HTTP responses for Tableau Mobile sessions. By default, most proxy solutions will preserve X-headers. If your proxy solution does not
preserve X-headers, then you will need to configure your proxy server to preserve the following header to all HTTP responses for Mobile client sessions: **X-Tableau: Tableau Server**.

If you have configured authentication at the proxy server gateway, then your proxy server must respond to Tableau Mobile HTTP requests with a HTTP 302 response. The 302 must include a redirect to the identity provider login page. To view a diagram that describes the 302 authentication sequence, see [Tableau Mobile Authentication Sequence](https://community.tableau.com/docs/redirect) in the Tableau Community.

**Reverse proxy and user authentication**

Tableau Server will always authenticate users. This means that even if you are authenticating inbound connections at the gateway for your organization, Tableau Server will still authenticate the user.

However, not all clients will support user authentication with a reverse proxy:

- For supported web browsers, you can use SAML, OpenID Connect, Kerberos, Trusted Tickets or manual authentication with a reverse proxy. However, we recommend a transparent scenario where user requests are not prompted for authentication at the gateway. This recommendation doesn’t prohibit using SSL for client/server system-level authentication at the gateway proxy, in fact, we strongly recommend SSL system-level authentication.

- The iOS version of the Tableau Mobile supports SAML, Kerberos or manual authentication with a reverse proxy. The same recommendation above applies.

- Tableau Desktop and the Android version of Tableau Mobile do not support authentication with a reverse proxy. For these clients you should use a [VPN solution](https://tableau.com/products/VPN), or configure your reverse proxy to route traffic from these clients directly to Tableau Server for authentication.

If your organization is authenticating with Active Directory:
Active Directory with Enable automatic logon (SSPI) is not supported with a reverse proxy.

Tableau Server must be configured for reverse proxy before configuring Tableau Server for Kerberos. For more information, see Configure Kerberos.

Configure Tableau Server to work with a reverse proxy server

Before you configure Tableau Server, you'll need to collect the following information about the proxy server configuration. To configure Tableau Server, you use the `tsm configuration set` command. The information you need to collect corresponds to options you'll need when you run `tsm`.

Most of the following `tsm` options are also used to configure Tableau Server deployments that operate behind a load balancer. For more information, see Add a Load Balancer.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Corresponding <code>tsm configuration set</code> option</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP address or CNAME</td>
<td>You can either enter an IP address or a CNAME for this option. The public IP address or addresses of the proxy server. The IP address must be in IPv4 format, such as 203.0.113.0, and it must be a static IP. If you are unable to provide a static IP, or if you are using cloud proxies or external load balancers, you can specify the CNAME (Canonical Name) DNS value that clients will use to connect to Tableau Server. This CNAME value must be configured on your reverse proxy solu-</td>
<td><code>gateway.trusted</code></td>
</tr>
<tr>
<td>Item</td>
<td>Description</td>
<td>Corresponding <code>tsm configuration set</code> option</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>FQDN</td>
<td>The fully qualified domain name that people use to reach Tableau Server, such as <code>tableau.example.com</code>. Tableau Server doesn't support a URL for this option, such as <code>example.com/tableau</code>.</td>
<td><code>gateway.public.host</code></td>
</tr>
<tr>
<td>Non-FQDN</td>
<td>Any subdomain names for the proxy server. In the example of <code>tableau-.example.com</code>, the subdomain name is <code>tableau</code>.</td>
<td><code>gateway.trusted_hosts</code></td>
</tr>
<tr>
<td>Aliases</td>
<td>Any public alternative names for the proxy server. In most cases, aliases are designated using CNAME values. An example would be a proxy server <code>bigbox.example.com</code> and CNAME entries of <code>ftp.example.com</code> and <code>www.example.com</code>.</td>
<td><code>gateway.trusted_hosts</code></td>
</tr>
<tr>
<td>Ports</td>
<td>Port numbers for traffic from the client to the reverse proxy server.</td>
<td><code>gateway.public.port</code></td>
</tr>
</tbody>
</table>

If you are using a distributed installation of Tableau Server, then run the following `tsm` commands on the initial node in your cluster.

1. Enter the following command to set the FQDN that clients will use to reach Tableau Server through the proxy server, where `name` is the FQDN:

   ```bash
tsm configuration set -k gateway.public.host -v 'name'
   ```
For example, if Tableau Server is reached by entering `https://tableau.example.com` in the browser, enter this command:

```
tsm configuration set -k gateway.public.host -v 'tableau.example.com'
```

2. Enter the following command to set the address or the CNAME of the proxy server, where `server_address` is the IPv4 address or CNAME value:

```
tsm configuration set -k gateway.trusted -v 'server_ip_address'
```

If your organization uses multiple proxy servers, enter multiple IPv4 addresses, separating them with commas. IP ranges are not supported. To improve start up and initialization of Tableau Server, minimize the number of entries for `gateway.trusted`.

3. Enter the following command to specify alternate names for the proxy server, such as its fully qualified domain name, any not fully qualified domain names, and any aliases. If there's more than one name, separate the names with a comma.

```
tsm configuration set -k gateway.trusted_hosts -v 'name1, name2, name3'
```

For example:

```
tsm configuration set -k gateway.trusted_hosts -v 'proxy1.example.com, proxy1, ftp.example.com, www.example.com'
```

4. If the proxy server is using SSL to communicate with the internet, run the following command, which tells Tableau that the reverse proxy server is using port 443 instead of port 80:

```
tsm configuration set -v gateway.public.port -k '443'
```
**Note**: If the proxy server is using SSL to communicate with Tableau Server, SSL must be configured and enabled on Tableau Server.

5. Enter the following command to commit the configuration change:

```bash
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

**Configure the reverse proxy server to work with Tableau Server**

When a client accesses Tableau Server through a reverse proxy, specific message headers have to be preserved (or added). Specifically, all proxy servers in the message chain must be represented in the `gateway.trusted` and `gateway.trusted_hosts` settings.

The following graphic shows example headers for a single-hop message chain, where the proxy server is communicating directly with Tableau Server:

![Single-hop Message Chain](image)

The following graphic shows example headers for a multiple-hop message chain, where the message traverses two proxy servers before connecting to Tableau Server:

![Multiple-hop Message Chain](image)
The following table describes what these headers are and how they relate to the configuration settings on Tableau Server:

<table>
<thead>
<tr>
<th>Headers</th>
<th>Description</th>
<th>Related Tableau Server settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>REMOTE_ADDR and X-FORWARDED-FOR (XFF)</td>
<td>Tableau Server needs these headers to determine the IP address of origin for requests. X-FORWARDED-FOR header must present IP address chain to Tableau Server in the order the connections have occurred.</td>
<td>The IP address that you set in gateway.trusted must match the IP presented in REMOTE_ADDR. If you sent multiple addresses in gateway.trusted, one of them must match the IP presented in REMOTE_ADDR.</td>
</tr>
<tr>
<td>HOST and X-FORWARDED_HOST (XFH)</td>
<td>These headers are used to generate absolute links to Tableau Server when it replies to the client. X-FORWARDED-HOST header must present host names to Tableau Server in the order the connections have occurred.</td>
<td>The host names that are presented in X-FORWARDED-HOST header must be included in the host names that you specify in gateway.trusted_hosts.</td>
</tr>
<tr>
<td>X-FORWARDED-PROTO (XFP)</td>
<td>This header is required if SSL is enabled for traffic from the client to the proxy, but not for traffic from the</td>
<td>Port configuration on reverse proxy (inbound con-</td>
</tr>
</tbody>
</table>
proxy to Tableau Server.

The X-FORWARDED-PROTO headers are important for scenarios where HTTP or HTTPS is not maintained along each hop of the message route. For example, if the reverse proxy requires SSL for outside requests, but traffic between the reverse proxy and Tableau Server is not configured to use SSL, X-FORWARDED-PROTO headers are required. Some proxy solutions add the X-FORWARDED-PROTO headers automatically, while others do not. Finally, depending on your proxy solution, you might have to configure port forwarding to translate the request from port 443 to port 80.

Connections from client and out-bound connections to Tableau Server) must be specified in the corresponding parameter: gateway.public.port, which is the port clients use to connect to the proxy.

If the proxy server is using SSL to communicate with Tableau Server, SSL must be configured and enabled on Tableau Server.

Validate reverse proxy setup

To validate your reverse proxy setup, publish workbooks and datasources using Tableau Server web authoring or Tableau Desktop. If you are connecting with a web browser to Tableau Server from the internet, verify that you are using a recommended browser. Publish and view workbooks that use existing datasources as well as a datasource that you’ve published. Use the links below to familiarize yourself with connecting with Tableau Server as an end-user.

<table>
<thead>
<tr>
<th>Task</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview of web authoring</td>
<td>Use Tableau on the Web</td>
</tr>
<tr>
<td>Task</td>
<td>Related Topic</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Log in to Tableau Server from Tableau Desktop.</td>
<td>Sign in to Tableau Server or Online</td>
</tr>
<tr>
<td>Log into Tableau Server from a web browser.</td>
<td>Sign in</td>
</tr>
<tr>
<td>Publish a workbook to Tableau Server.</td>
<td>Publish a Workbook</td>
</tr>
<tr>
<td>Publish a data source.</td>
<td>Publish a Data Source</td>
</tr>
<tr>
<td>Open workbook from Tableau Server.</td>
<td>Opening Workbooks from the Server</td>
</tr>
<tr>
<td>Log out Server (with Desktop).</td>
<td>Sign in to Tableau Server or Online</td>
</tr>
<tr>
<td>Download workbook from a web browser.</td>
<td>Download Workbooks</td>
</tr>
<tr>
<td>Check to make sure tabcmd (from a non-server client) works.</td>
<td>tabcmd</td>
</tr>
</tbody>
</table>

**Related topics**

- Tableau Desktop Internet Access Requirements
- Tableau Community: Cannot Create Firewall Rules to Access Mapping Servers
- Add a Load Balancer
Install and Configure

The topics in this section provide information on installing, configuring, and upgrading Tableau Server on Linux.

Looking for Tableau Server on Windows? See Install and Configure Tableau Server

Install and Configure Tableau Server .......................................................... 65
Distributed and High Availability Tableau Server Installations ..................... 101
Post Installation Tasks ............................................................................. 149
Automated Installation of Tableau Server .................................................... 174
Upgrade Tableau Server on Linux ............................................................... 180
Migrate Tableau Server from Windows to Linux ........................................... 183
Uninstall Tableau Server .......................................................................... 202
Install Tableau Server in a Disconnected (Air-Gapped) Environment .......... 205
Install Tableau Server in the AWS Cloud .................................................... 214

Install and Configure Tableau Server

This topic describes the steps to install and configure Tableau Server on Linux. If you are installing a distributed deployment (cluster), use the steps in this topic to install the initial node, then, to install additional nodes, see Distributed and High Availability Tableau Server Installations.

If you want a quick start procedure to install Tableau Server in a non-production environment, see Jump-start Installation.

You configure Tableau Server from the command line by activating a license, registering Tableau Server, and configuring various settings including authentication. For more information on using the tsm command line interface, see tsm Command Line Reference.

Before you begin

To install Tableau Server you must have a computer that satisfies the hardware requirements. You will get an informational message if your computer meets the minimum requirements but does not satisfy the recommended minimum requirements. In this case, your computer hardware can handle a trial installation of Tableau but is not adequate for a production environment. For more information, see Confirm Requirements.

Configure Tableau Server

This topics in this section describes how to install and configure Tableau Server from the command line using tsm and tabcmd. By default, when you run tsm commands on the command line, you are authenticated with the username you are logged in as. If you want to authenticate to TSM as a different user, use the -u flag. The username you specify must be able to authenticate with a password.

Installation steps

The following steps describe how to install Tableau Server on a single computer. Use the steps to install Tableau Server in a single server deployment. Use the steps to install the initial node in a multi-node Tableau Server deployment. Run the steps sequentially.

1. Install Tableau Server Package
2. Activate and Register Tableau Server
3. Configure Initial Node Settings

4. Finalize Installation

Confirm Requirements

Before you install Tableau Server, confirm that you meet the following requirements.

- Hardware requirements
- Operating system requirements
- Resource requirements
- Additional requirements
  - Package requirements

Hardware recommendations

The following list describes the hardware recommendations for a single node installation of Tableau Server:

- 8 core, 2.0 GHz or higher processor
- 64-bit processor architecture
- 32 GB memory
- 50 GB disk space available

The core Tableau Server bits must be installed in the `/opt` directory where there is at least 15 GB of free disk space. You cannot change the installation path for Tableau Server. If you attempt to install Tableau Server on a computer that does not have enough space, the Tableau Server package will install, but you will be unable to continue with setup.

If you plan to make heavy use of extracts then you may need to allocate additional disk space. You can specify a different directory for data (extract) storage during installation. For more information, see Disk Space Requirements, and Install Tableau Server Package.
To see the full list of recommendations and to see the minimum hardware requirements, see Minimum Hardware Requirements and Recommendations for Tableau Server.

For public cloud deployments on Amazon Web Services and Google Cloud Platform, their “vCPU” is actually a CPU hyper-thread, and not a full CPU core. When sizing cloud instances, you will need twice as many vCPU as the Tableau Server CPU core requirements given (4 vCPU for a minimum trial installation, 16 vCPU for a recommended installation).

Operating system requirements

The following distributions of Linux are supported:

- Red Hat Enterprise Linux (RHEL) 7.3 and higher, CentOS 7.3 and higher, Oracle Linux 7.3 and higher.
  
  Version 8.x is not supported.
  
  These are collectively referred to in this documentation as RHEL-like.

- Ubuntu 16.04 LTS only.
  
  Version 17.04 is not supported.

Previous versions of CentOS and Ubuntu are not supported because Tableau Server requires systemd for process management.

Custom kernels are not supported.

In a multi-node installation of Tableau Server, all of the computer nodes where you are installing Tableau must run Linux and the same distribution of Linux.

**Note:** You cannot install Tableau Server on a combination of Linux and Windows computers.
Resource requirements

The initial release of Tableau Server on Linux is ready for organizations that are familiar with Linux and have strong infrastructure expertise at integrating Linux applications into their environment. It also helps if your organization has some experience with running Tableau Server in a production environment.

Use the following set of questions to help you assess whether this version of Tableau Server for Linux is a suitable fit for your organization. If it looks like this version won’t be a good fit, circle back for the next version, which will include more automation for installation and configuration.

<table>
<thead>
<tr>
<th>Linux OS knowledge</th>
<th>Do you currently administer other applications on Linux?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shell admin experience</td>
<td>Are you accustomed to using command line shells and editing config files to administer applications?</td>
</tr>
<tr>
<td>Availability of Linux intgration support</td>
<td>Does your organization’s IT infrastructure provide ready access to assistance integrating Linux applications? This can be things like directory services, networking, authentication, virtual machine/cloud platform, and operating system.</td>
</tr>
<tr>
<td>Linux distribution</td>
<td>Are you planning or able to use one of the Linux distributions Tableau tests on and supports?</td>
</tr>
<tr>
<td>Datasources</td>
<td>Are all the data connections that you need supported on Linux? (See Install Drivers on Linux.)</td>
</tr>
</tbody>
</table>

Additional requirements

Make sure that your environment also meets the following additional requirements:

Hostname
The hostname of the computer where you install Tableau Server must be unique in your environment, and once you have installed Tableau Server the hostname cannot change. Additionally, your domain name server (DNS) must be able to resolve the hostname.

**Static IP address**

The computer where you install Tableau Server must have a static IP address.

**Database drivers**

To connect to specific data sources, the computer where you install Tableau Server must have the correct database drivers installed. For more information, see Install Drivers.

**Available Ports**

TSM and Tableau Server each require an available TCP port in order for you to access them. TSM defaults to port 8850, and the Tableau Server Gateway service defaults to port 80 (both ports are configurable). We strongly recommend that you ensure that both port 8850 and 80 are not in use on your Linux system before installing Tableau Server. If those ports are not available, the TSM and gateway ports may be dynamically remapped to different port numbers, and there is currently no interface for displaying which port they have been remapped to.

**Sudo access**

All installation tasks and administrative tasks for Tableau Server must be run as root. Often this is accomplished using the sudo command, but running the commands directly as the root user is also possible.

To install Tableau Server with the root account, you must specify a user account during installation. The account will be used for managing TSM. Specify the account by running the initialize-tsm script with the –a option. See Help Output for initialize-tsm Script.

**Account password**
The user account that you use to install and administer Tableau Server must be able to authenticate with a password. That is, the user must not use another means of authenticating (for example public key authentication).

If the account you are using to install and initialize Tableau Server does not have a password, you can set one using the `passwd` command:

```
sudo passwd $USER
```

**Port access requirements**

If you want to install Tableau Server remotely, for example by means of SSH, ensure that the following ports are open:

- **8850.** The port used for the Tableau Services Manager (TSM) web interface. You can use this interface to configure Tableau Server.
- **80.** The port used for the main Tableau Server web interface.

The Tableau Server installer attempts to open these ports during the installation process, but it can only open these ports for the `firewalld` firewall. If your computer runs another firewall, you must open the ports before you install.

**Virtual Container environments**

Installation of Tableau Server in virtual container environments such as Docker have not been tested and are not supported. Tableau Server on Linux will not function as expected if installed in these environments.

**Package requirements**

**Systemd**

Tableau Server requires `systemd` to manage services. This package is installed by default on CentOS 7 and Ubuntu 16. If you decide to test Tableau Server on a modified version of these distributions, you can run the following command to confirm that `systemd` is installed:

```
whereis systemd
```
If `systemd` is installed, the installation location is displayed. For example, you might see the following output:

```
systemd: /usr/lib/systemd /etc/systemd /usr/share/systemd
        /usr/share/man/man1/systemd.1.gz
```

If you have systemd installed but the Tableau installer is failing requirements checks for systemd, it's likely that systemd is not running. To verify that systemd is running, run the following command:

```
ls /run/systemd
```

The output will be a list of files and directories. If systemd is running, the output will include `system`. If `system` is not in the output, then systemd is not running.

*Continue to the next step: Install and Configure Tableau Server.*

Minimum Hardware Requirements and Recommendations for Tableau Server

The following minimum hardware requirements and recommendations apply to all computers running Tableau Server, including physical hardware and virtual machines (VMs):

- **Minimum requirements** are the minimum hardware your computer must have in order to install Tableau Server. If your computer does not meet these requirements, the Setup program will not install Tableau Server. These requirements are appropriate for testing and prototyping.

- **Minimum recommendations** are higher than minimum requirements, and represent the minimum hardware configuration you should use for a production installation of Tableau Server. If your computer meets the minimum requirements but does not meet these recommendations, the setup program will warn you but you can continue the installation.
In addition, Tableau Server should not be installed on a physical computer or on a VM instance that is also running resource-intensive applications such as databases or application servers.

**Note:** If you install Tableau Server on a computer that meets the minimum requirements but does not have at least 8 cores and 16 GB of system memory, the default number of all processes installed is reduced to one of each process by design. For more information about processes, see Server Process Limits.

Looking for Tableau Server on Windows? See [Minimum Hardware Requirements and Recommendations for Tableau Server](#).

### Minimum Hardware Requirements

The computer on which you are installing or upgrading Tableau Server must meet the minimum hardware requirements. If the setup program determines that your computer does not meet the following requirements, you will not be able to install Tableau Server. For more information on how the Setup program determines hardware, see "Determining Computer Hardware," below.

These minimum requirements are appropriate for a computer that you use for prototyping and testing of Tableau Server. They apply to single-node installations and to each computer in a distributed installation.

<table>
<thead>
<tr>
<th>Minimum Hardware Requirements</th>
<th>CPU</th>
<th>RAM</th>
<th>Free Disk Space</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2-core</td>
<td>8 GB</td>
<td>15 GB</td>
</tr>
</tbody>
</table>

For the requirements:
• Free disk space is calculated after the Tableau Server Setup program is unzipped. The setup program uses about 1 GB of space. You may need to allocate additional disk space depending on the various factors like whether or not you will be using extracts. For more information, see Disk Space Requirements.

• Core count is based on "physical" cores. Physical cores can represent actual server hardware or cores on a virtual machine (VM). Hyper-threading is ignored for the purposes of counting cores.

**Note:** For Tableau Server 10.5, you need a minimum of 2 physical cores. If you are installing on an Amazon EC2 instance, this means 4 vCPUs. For more information, see Amazon EC2 Instances.

**Minimum Hardware Recommendations**

For production use, the computer on which you install or upgrade Tableau Server should meet or exceed the minimum hardware recommendations. These recommendations are general. Actual system needs for Tableau Server installations can vary based on many factors, including number of users and the number and size of extracts. If the setup program determines that your computer does not meet the following recommendations, you will get a warning, but you can continue with the setup process.

<table>
<thead>
<tr>
<th>Install Type</th>
<th>Processor</th>
<th>CPU</th>
<th>RAM</th>
<th>Free Disk Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single node</td>
<td>64-bit</td>
<td>8-core, 2.0 GHz or higher</td>
<td>32 GB</td>
<td>50 GB</td>
</tr>
<tr>
<td>Multi-node and enterprise deployments</td>
<td>Contact Tableau for technical guidance. Nodes must meet or exceed the minimum hardware recommendations, except nodes running backgrounder, where 4 cores may be acceptable.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Install Tableau Server Package

This topic guides you through the process of installing the Tableau Server package and starting Tableau Services Manager (TSM).

- Install Tableau Server
  - Install the Tableau Server package
  - Start TSM

Prerequisites

Before proceeding with the procedures in this topic, Confirm Requirements.

Install Tableau Server

Install Tableau Server with your distribution's package manager, then run a script to start TSM. The script is included with the installed package.

For security reasons, you cannot use the root user as the installation user or the installation might fail. Use sudo instead.

Important: The hostname of the server must not change after you start TSM. For example, this might happen if you use the cloud-init package to initialize a virtual machine, and you install Tableau Server on that virtual machine. For more information, see the Introducing Tableau Server on Linux.

Install the Tableau Server package

Tableau Server must be installed in the /opt directory. However, you can specify a different directory for data (extracts and extract metadata) storage. The next section, Start TSM, describes how to specify a non-default data directory.

1. Log on as a user with sudo access to the computer where you want to install Tableau Server.

2. Download the .rpm or .deb installer package from the Tableau Server 10.5
3. Navigate to the directory where you copied the .rpm or .deb package.

4. Use the package manager to install the Tableau Server package.
   - On RHEL-like distributions, including CentOS, run the following command:
     ```
     sudo yum install tableau-server-<version>.x86_64.rpm
     ```
     **Note:** When you use `yum` to install Tableau Server, all dependent packages are automatically downloaded and installed. This is the preferred method for installing Tableau. If your organization does not allow you to use `yum` and you must install using `rpm -i`, you must also install all dependent packages separately.

   - On Ubuntu, run the following command:
     ```
     sudo apt-get update
     sudo apt-get -y install gdebi-core
     sudo gdebi -n tableau-server-<version>_amd64.deb
     ```

5. If you are installing a multinode deployment (cluster) of Tableau Server and you are running a local firewall on each node, then you must manually configure a licensing daemon port before you start TSM. See Set Licensing Daemon Port.

Start TSM

1. Navigate to the scripts directory:
   ```
   cd /opt/tableau/tableau_server/packages/scripts.<version>/
   ```

2. Run the following script to start TSM:
   ```
   sudo ./initialize-tsm --accepteula
   ```
The --accepteula flag accepts the Tableau Server End User License Agreement (EULA). The EULA is available in the following location:

/opt/tableau/tableau_server-
packages/docs.<version>/Commercial_EULA.txt

Other options you might want to use include:

- The -a flag to specify a user to be added to the tsmadmin and tableau groups instead of the user running this script.

- The --unprivileged-user flag to set a different service account. By default a new user called tableau will be created. This account is an unprivileged service account under which most Tableau process run. We recommend creating a different user only in the case where a tableau user account already exists on the computer.

- The -d flag to change the location of the "data directory" where Tableau Server stores extracts, information about extracts, and more.

By default, Tableau Server uses the following location for the directory:

/var/opt/tableau/tableau_server

Notes:
- You cannot specify a data directory location on a Network File System (NFS) volume.
- You cannot specify a data directory location with a path that includes a period ("."). If there is a period in the path, initialization will fail.

- The --debug flag for troubleshooting. This option displays each command in the script as it is run and can make it easier to troubleshoot issues. Use of this option results in extensive output to the screen.
**Note**: If you want to manually manage port assignment for TSM and Tableau Server processes, you may need to use one or more port-related switches with initialize-tsm. For more information, see Controlling port remapping with initialize-tsm.

To view all options and help output for `initialize-tsm` script, run the following command:

```
sudo ./initialize-tsm -h
```

3. Log off and log on again to the terminal before you configure Tableau Server.

When you log on again, you create a new session in which group membership changes have taken effect. The new session also has access to the environment variables added by the `initialize-tsm` script.

Alternatively, you can run the following command to update your path for the current session (but not to update your group membership):

```
source /etc/profile.d/tableau_server.sh
```

**Next step**

- Activate and Register Tableau Server

**Activate and Register Tableau Server**

This topic describes how to activate and register Tableau Server.

**Prerequisite**

Before proceeding with the procedures in this topic, complete the following prerequisites as outlined in Install and Configure Tableau Server:

- Install Tableau Server Package
Log in to Tableau Services Manager

Before you can proceed you must log in to Tableau Services Manager (TSM). The account that you use to configure the rest of the installation must be a member of the tsmadmin group that was created during initialization.

To log in to TSM, run the following command:

```
tsm login -u <username>
```

What if I can't log in?

If you get an authentication error, verify that the user account is in the tsmadmin group. To view the user accounts in the tsmadmin group, run the following command:

```
grep tsmadmin /etc/group
```

If the user account is not in the group, run the following command to add the user to the tsmadmin group:

```
sudo usermod -G tsmadmin -a <username>
```

After you have added the user to the tsmadmin group, run the tsm login command.

Activate Tableau Server

To activate Tableau Server for production use, you must have a license key.

To activate a license key, run the following command:

```
tsm licenses activate -k <KEY>
```

To activate a two-week trial, run the following command:

```
tsm licenses activate -t
```

If you are unable to activate Tableau, for example, if you get an error like this:

```
License Server not available
```
see Activate Tableau Offline.

Register Tableau Server

After activation, you need to register Tableau Server. To do this, create a registration file and then pass it as an option with the `tsm register` command.

1. Generate a template that you can edit by running the following command:

   ```bash
   tsm register --template > /path/to/registration_file.json
   ```

2. Edit the template file to create your completed registration file.

   Here is an example of a registration file in the required format:

   ```json
   {
     "zip": "03079",
     "country": "USA",
     "city": "Salem",
     "last_name": "Smith",
     "industry": "Software",
     "eula": "yes",
     "title": "Software Applications Engineer",
     "phone": "5556875309",
     "company": "Example",
     "state": "NH",
     "department": "Engineering",
     "first_name": "Jason",
     "email": "jsmith@example.com"
   }
   ```

3. After saving changes to the file, pass it with the `--file` option to register Tableau Server:

   ```bash
   tsm register --file /path/to/registration_file.json
   ```
Next step

- If you are running a local firewall then you must configure the firewall before continuing, see Configure Local Firewall
- If you are not running a local firewall, then continue to Configure Initial Node Settings.

Activate Tableau Offline

Tableau Server requires at least one product key that both activates the server and specifies the number of license levels you can assign to users. You can access your product keys from the Tableau Customer Account Center.

If you are not able to activate Tableau Server directly, for example, if your computer is not connected to the internet, or has a firewall that restricts access outside your intranet, you can follow the steps below to activate Tableau offline.

Offline activation of Tableau Server involves several steps. If you have never activated a product key before on your computer, you need to run through the steps twice, once to initialize licensing on the computer, and a second time to activate the key. If you have had a license on the computer in the past you only need to run through the steps once.

Follow these steps to activate Tableau Server offline:

1. Generate an offline activation request file.
2. Upload the request file to Tableau and download the resulting activation file.
3. Activate or initialize your license using the activation file.

If this is the first time a Tableau product key has been activated on the computer, repeat the steps a second time.

The procedures below describe each of these steps in more detail.

Step 1 - Generate an offline activation request file

1. Run the following command:

   `tsm licenses get-offline-activation-file -k <product-key> -o <target-directory>`
The target directory must exist.

2. Copy the offline activation file (offline.tlq) from the target directory to a computer that has internet access.

Step 2 - Upload the offline activation request to Tableau

1. Go to the Tableau Product Activations page.

2. Complete the instructions to choose and upload your offline.tlq file.
   
   This creates an activation file, activation.tlf.

3. Download the resulting activation file from Tableau.

Step 3 - Initialize or activate your license

1. Copy the activation file (activation.tlf) to a location accessible from your Tableau Server computer.

2. Run the following command:
   
   tsm licenses activate -f <path-and-activation-file>

First time license activation

If this is the first Tableau license activated on the computer, you will see this message:

Your license has been initialized. To complete the activation, we need one more exchange. Generate and send to Tableau a second activation request file.

and you need to repeat the steps above to activate your license.

If you are not sure whether you need to repeat the steps, you can run this command:

   tsm licenses list

If you see a message like this one you need to repeat the steps above:

No licenses are currently activated.
Subsequent license activation

If this is not the first Tableau license activated on the computer, or if you have completed the steps above two times, you should see a success message:

*Activation successful.*

Tableau Server is activated. If you need additional assistance, contact Tableau Technical Support.

**Configure Local Firewall**

This topic explains how to configure the firewall on the computer running Tableau Server. A local firewall should be enabled on the operating system to protect Tableau Server in single and multi-node deployments. In a distributed (multi-node) installation of Tableau Server, communication between nodes does not use secure communication. Therefore, you should enable firewalls on the computers that host Tableau Server.

We recommend that you configure the firewall so that only two ports are accessible to external traffic: the gateway port and the tabadmincontroller port. By default, these are ports 80 and 8850 respectively. Additionally, if you are running in a distributed deployment, then you will need to open the port range, 27000-27009, so licensing can communicate across nodes.

The gateway port is used for HTTP connection to Tableau Server. We recommend that you use SSL for the gateway port. If you will be using SSL, then the port must be 443 because Tableau Server does not support other ports for SSL. The procedures below describe how to configure the firewall for the gateway port. Configure the Tableau Server gateway (Configure Initial Node Settings) to match the port you set here.

The examples below describe how to configure the firewall on single and multi-node deployments of Tableau Server running on RHEL/CentOS distributions. The examples use *Firewalld*, which is the default firewall on CentOS.
Single-node configuration

1. Open a bash shell and run the following TSM command to retrieve the port number for the `tabadmincontroller` port:

   `tsm topology list-ports`

   Make a note of the `tabadmincontroller` port. By default, this port is 8850.

2. Start `firewalld`:

   `sudo systemctl start firewalld`

3. Verify that the default zone is a high-security zone, such as `public`. If it is not, we recommend changing it to a high-security zone.

   `sudo firewall-cmd --get-default-zone`

   `sudo firewall-cmd --set-default-zone=public`

4. Add ports for the gateway port and the `tabadmincontroller` port. In the example below we use the default ports (80 and 8850).

   `sudo firewall-cmd --permanent --add-port=80/tcp`

   `sudo firewall-cmd --permanent --add-port=8850/tcp`

5. Reload the firewall and verify the settings.

   `sudo firewall-cmd --reload`

   `sudo firewall-cmd --list-all`

Multi-node cluster configuration

In addition to enabling ports, configuring the firewall on a multi-node cluster requires additional steps to ensure that nodes can communicate with each other.
Before you begin

You must manually configure the licensing daemon port on each node. See Set Licensing Daemon Port.

You will need the IP address for each node in the cluster. The example here uses \(<node1IP>\) as a placeholder for the initial node IP address, and \(<node2IP>\) and \(<node3IP>\) as placeholders for the IP addresses of two additional nodes.

Step 1: Configure initial node.

1. Open a bash shell and run the following TSM command to retrieve the port number for the `tabadmincontroller` port:

   ```bash
tsm topology list-ports
   ```

   Make a note of the `tabadmincontroller` port. By default, this port is 8850.

2. Run the following commands to determine the range of port numbers that TSM may dynamically select. You will specify this range later in this procedure. Make a note of the port range.

   ```bash
tsm configuration get -k ports.range.min
   tsm configuration get -k ports.range.max
   ```

   A typical range is 8000 to 9000.

3. Start `firewalld`:

   ```bash
   sudo systemctl start firewalld
   ```

4. Verify that the default zone is a high-security zone, such as `public`. If it is not, we recommend changing it to a high-security zone.

   ```bash
   firewall-cmd --get-default-zone
   sudo firewall-cmd --set-default-zone=public
   ```
5. Add ports for the gateway port and the tabadmincontroller port. In the example below we use the default ports (80 and 8850). You must also add a port range (27000-27010) to enable licensing communication between nodes.

```
sudo firewall-cmd --permanent --add-port=80/tcp
sudo firewall-cmd --permanent --add-port=8850/tcp
sudo firewall-cmd --permanent --add-port=27000-27010/tcp
```

6. Configure the firewall to allow all traffic from the other nodes in the cluster. For the ports option, specify the range you noted in Step 2. Run the command for each of the additional nodes in your cluster. For example:

```
sudo firewall-cmd --permanent --add-rich-rule='rule family=ipv4 source address=<node2IP>/32 port port=8000-9000 protocol=tcp accept'
sudo firewall-cmd --permanent --add-rich-rule='rule family=ipv4 source address=<node3IP>/32 port port=8000-9000 protocol=tcp accept'
```

7. Reload the firewall and verify the settings.

```
sudo firewall-cmd --reload
firewall-cmd --list-all
```

**Step 2: Configure additional nodes**

Each node in the cluster must be able to communicate with the initial node and with the other nodes.

Run this procedure on each additional node in the cluster. In this example, the node at IP address, `<node2IP>`, communicates with the initial node at `<node1IP>` and a third node at `<node3IP>`.
1. Start firewalld:

   ```
   sudo systemctl start firewalld
   ```

2. Verify that the default zone is a high-security zone, such as `public`. If it is not, we recommend changing it to a high-security zone.

   ```
   firewall-cmd --get-default-zone
   ```

   ```
   sudo firewall-cmd --set-default-zone=public
   ```

3. Configure the firewall to allow `gateway` and `tabadmincontroller` access from the other nodes in the cluster. For example:

   ```
   sudo firewall-cmd --permanent --add-rich-rule='rule family=ipv4 source address=<node1IP>/32 port port=80 protocol=tcp accept'
   ```

   ```
   sudo firewall-cmd --permanent --add-rich-rule='rule family=ipv4 source address=<node1IP>/32 port port=8000-9000 protocol=tcp accept'
   ```

   ```
   sudo firewall-cmd --permanent --add-rich-rule='rule family=ipv4 source address=<node3IP>/32 port port=80 protocol=tcp accept'
   ```

   ```
   sudo firewall-cmd --permanent --add-rich-rule='rule family=ipv4 source address=<node3IP>/32 port port=8000-9000 protocol=tcp accept'
   ```

   In this example, since the `tabadmincontroller` port (8850) is included in the port range, it is not explicitly specified in a command.

4. Reload the firewall and verify the settings.

   ```
   sudo firewall-cmd --reload
   ```
Configure Initial Node Settings

This topic describes how to configure initial node settings:

- Configure identity store settings
- Configure gateway settings (optional)
- Configure caching options (optional)
- Configure sample workbook installation (optional)

Prerequisites

Before proceeding with the procedures in this topic, complete the following prerequisites as outlined in Install and Configure Tableau Server:

1. Install Tableau Server Package
2. Activate and Register Tableau Server

If you are running a local firewall then you must configure the firewall before you configure the initial node settings, see Configure Local Firewall

Configure identity store settings

You must configure the identity store settings for the Tableau Server computer.

Use the json template in the identityStore Entity to create a json file. After you have filled in the options with the appropriate values, you can then pass the json file with this command:

```bash
tsm settings import -f /path/to/file.json.
```

Configure gateway settings (optional)

Depending on your network requirements, you may need to configure the gateway settings for the Tableau Server computer. For example, if you are enabling SSL or configuring access to Tableau Server with a reverse proxy, you may need to configure gateway settings. See gatewaySettings Entity for more information.
Use the json template in the gatewaySettings Entity to create a json file. After you have filled in the options with the appropriate values, you can then pass the json file with this command:

```
tsm settings import -f /path/to/file.json.
```

**Configure caching options (optional)**

By default, Tableau Server will cache and reuse data for as long as possible. You can change this behavior by configuring the caching option. Use this command:

```
tsm configuration set -k vizqlserver.data_refresh -v <value>
```

Where `<value>` is one of these options:

- `nil` (empty string). This is the default and configures Tableau Server to cache and reuse data for as long as possible.
- `<n>"<n>"` specifies the maximum number of minutes data should be cached.
- `0` (zero). A value of zero indicates that the cache should be refreshed each time a page is reloaded.

You can change this configuration after installing Tableau Server by using the `tsm configuration set` command above, and then applying changes with the `tsm pending-changes apply` command.

**Configure sample workbook installation (optional)**

By default, Tableau Server will install sample workbooks in the Default site when you initialize the server.

If you do not want to install sample workbooks during installation, run the following command:

```
tsm configuration set -k install.component.samples -v false
```

You can publish samples after installation by using the `publishsamples tabcmd` command.

**Next step**

- Finalize Installation
Configuration File Example

This article provides an example of a complete JSON configuration file, with gatewaySettings and identityStore entities specified. In addition, a configuration key sets the gateway timeout to 900 seconds.

Your configuration file will look different depending on the options you need to set.

You might set multiple .json configuration files during installation. To set the values for each file in Tableau Server, you run the following command, once for each configuration file:

```
tsm settings import -f /path/to/file.json
```

After you set the configuration files, run tsm pending-changes apply to apply the changes from all of the .json files you’ve set.

```json
{
    "configEntities": {
        "gatewaySettings": {
            "_type": "gatewaySettingsType",
            "port": 80,
            "firewallOpeningEnabled": true,
            "sslRedirectEnabled": true,
            "publicHost": "localhost",
            "publicPort": 80
        },
        "identityStore": {
            "_type": "identityStoreType",
            "type": "local",
            "domain": "example.lan",
            "nickname": "EXAMPLE"
        }
    },
    "configKeys": {
        "gateway.timeout": "900"
    }
}
```
Entities vs keys

As shown in the example above, there are two classes of configuration parameters: configEntities and configKeys.

configEntities

Certain types of configuration are done through entity sets that map to specific scenarios, such as the identity store and gateway configurations. When you pass a set of configEntities with the `tsm settings import -f /path/to/file.json` command, TSM validates the configuration. If values passed are invalid, TSM will provide an error. This enables you to make changes during the configuration process, rather than experience a configuration failure at initialization or run time.

Entities can be set only by including a configEntities block in a .json file.

configKeys

Entities cover only a small portion of the configuration values that can be set. Hundreds of keys correspond to parameters stored in .yml files. Tableau Server uses these parameters to store all of the configuration information for all services.

You can set individual keys with the `tsm configuration` command. But during deployment, setting them along with other configuration scenarios in JSON files, as shown above, is more convenient.

Unlike configEntities, configKeys are not validated.

**Note:** We do not recommend setting parameters that are not documented in `tsm configuration set Options`. 
Finalize Installation

This topic describes the final steps of Tableau Server on Linux installation:

- Apply pending configuration changes
- Verify LDAP configuration (Optional)
- Initialize and start Tableau Server
- Create the initial admin user
- Install PostgreSQL driver (optional)

Prerequisites

Before proceeding with the procedures in this topic, complete the following prerequisites as outlined in Install and Configure Tableau Server:

1. Install Tableau Server Package
2. Activate and Register Tableau Server
3. Configure Initial Node Settings

Apply pending configuration changes

Now that you’ve created and set initial configuration, you must apply them. When you apply configuration changes, tsms will verify the settings you’ve set before committing them.

To apply configuration changes to Tableau Server, run this command:

```
  tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Once this command has completed, TSM processes are running, and Tableau Server is configured but is not running.
Verify LDAP configuration (Optional)

If your identity store uses Active Directory or OpenLDAP, then we recommend verifying LDAP connectivity before proceeding.

To do so, run the following commands before you initialize the server:

```
tsm user-identity-store verify-user-mappings -v <user name>
tsm user-identity-store verify-group-mappings -v <group name>
```

User and group names must be valid names that exist in the LDAP server that you are connecting to. If your LDAP connection is set up correctly, then the user or group attributes will be returned to the shell. If your connection is not set up correctly, then an error will be returned.

Initialize and start Tableau Server

Initialize Tableau Server.

- To initialize and start Tableau Server, use the `--start-server` option:

  ```
tsm initialize --start-server --request-timeout 1800
  
  This saves time by starting the server running after initialization.
  ```

- If you are going to reconfigure Tableau Server after initialization, leave the `--start-server` option off:

  ```
tsm initialize --request-timeout 1800
  
  This stops the server after initialization.
  ```

Start Tableau Server. If you did not use the `--start-server` option during initialization and are finished configuring Tableau Server, use this command to start the server:

```
tsm start --request-timeout 900
```
Note: If you experience timeouts when installing or configuring Tableau Server, you may need to specify a longer timeout. For more information, see Install fails due to timeouts.

Create the initial admin user

Before you can sign in to the Server, you need to create the initial user who will be a Tableau Server administrator. The server must be running when you create the initial admin user.

If you have configured the Tableau Server identity store to use LDAP or Active Directory, then the initial admin user that you specify must be an account in the directory. The initial admin user is generally distinct from the user account on the Tableau Server computer that you use to run tsm. However, these accounts can be the same if you have configured the Tableau Server identity store to use LDAP or Active Directory and the initial admin user is a member of the tsmadmin group on the Tableau Server computer.

To create the initial user, run the following tabcmd command, enclosing the values in single quotes:

```
tabcmd initialuser --server 'localhost:80' --username '<new-admin-username>' --password '<new-admin-password>'
```

This might take several minutes. The command displays a success message when it finishes.

Install PostgreSQL driver (optional)

Tableau Server includes a PostgreSQL database, referred to as the Tableau Server repository. The repository contains workbook and user metadata, data extract files, and configuration data. As an administrator, you will often need to access the repository with a feature called administrative views. Administrative views are powerful monitoring tools that can help you optimize Tableau Server and better understand how your users are interacting with Tableau content. In addition to the pre-built administration views that are included with Tableau Server, you can create your own custom views of the repository.
To access the repository, you must install the PostgreSQL driver.

1. Download PostgreSQL drivers from the Driver Download page.

2. Install the drivers on your Tableau Server computer:

   - On CentOS and RHEL, download the .rpm file and then run the following command:
     
     ```
     sudo yum install tableau-postgresql-odbc-9.5.3-1.x86_64.rpm
     ```

   - On Ubuntu, download the .deb file and then run the following command:
     
     ```
     sudo dpkg -i tableau-postgresql-odbc_9.5.3_amd64.deb
     ```

3. To validate that the drivers installed, navigate to the Administrative Views in Tableau Server.

Next steps

- Install Tableau Server on Additional Nodes
- Post Installation Tasks
- Sign in to Tableau Server Admin Pages

Initial Node Installation Defaults

By default, the Tableau Server installer configures the number of process instances that Tableau Server runs based on the hardware detected by the installer. The default configuration applies to single-server installations and to the initial server of a multi-node installation.

You can calculate the default configuration based on the following rules for each process, where the number of cores refers to the number of logical CPU cores:

<table>
<thead>
<tr>
<th>Process Name</th>
<th>Number of Processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>VizQL Server</td>
<td>Equal to the number of cores divided by four, up to a maximum of four process instances.</td>
</tr>
<tr>
<td>Process Name</td>
<td>Number of Processes</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------------------------------</td>
</tr>
<tr>
<td>Backgrounder</td>
<td>Set to two unless the number of cores is fewer than eight.</td>
</tr>
<tr>
<td>Cache Server</td>
<td>Set to two unless the number of cores is fewer than eight.</td>
</tr>
<tr>
<td>Data Server</td>
<td>Set to two unless the number of cores is fewer than eight.</td>
</tr>
</tbody>
</table>

For all other process types, the number of process instances is set to one, regardless of the hardware.

Here's an example default configuration for a computer with 16-cores:

<table>
<thead>
<tr>
<th>Process Name</th>
<th>Number of Processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>VizQL Server</td>
<td>4</td>
</tr>
<tr>
<td>Application Server</td>
<td>1</td>
</tr>
<tr>
<td>Backgrounder</td>
<td>2</td>
</tr>
<tr>
<td>Cache Server</td>
<td>2</td>
</tr>
<tr>
<td>Data Server</td>
<td>2</td>
</tr>
<tr>
<td>File Store</td>
<td>1</td>
</tr>
<tr>
<td>Data Engine</td>
<td>1</td>
</tr>
</tbody>
</table>

**Jump-start Installation**

This topic provides all of the steps required to perform a basic, quick-start installation of Tableau Server. The purpose of the configuration described here is to provide the quickest, simplest path to a Tableau Server installation on a computer running the CentOS 7 or Ubuntu distribution of Linux. Use this procedure as practice, to try out the Tableau Server installation, management, and user experience before your actual server deployment.
**Important**: Do not use this procedure as a stand-alone resource for deploying Tableau Server into a production environment.

The organization of this topic maps to the broad steps presented in Install and Configure Tableau Server.

The procedures in this topic will install an instance of Tableau Server for Linux with the following characteristics:

- Operating system: CentOS 7 or Ubuntu
- Identity store: local authentication
- Gateway port: 80
- Tableau Server administrator account: admin

**Before you begin**

Review the topic, Confirm Requirements. The procedure here assumes that you have installed Linux on conforming hardware and according to the environmental requirements specified in that topic.

Note that the computer you install on must meet the minimum hardware requirements specified in Minimum Hardware Requirements and Recommendations for Tableau Server. The setup program will not install Tableau Server onto systems that do not meet the minimum hardware requirements.

**Step 1: Install Tableau Server package and start Tableau Services Manager**

Install Tableau Server with your distribution’s package manager, then run a script to initialize Tableau Services Manager (TSM). Tableau Services Manager is a the management toolset used to install, configure, and manage Tableau services.

The initialize script is included with the installed package. For more details, see Install Tableau Server Package.
1. Log on as a user with sudo access to the computer where you want to install Tableau Server.

2. Navigate to the directory where you copied the Tableau Server installation package.

3. Use the package manager to install the Tableau Server package.

   - On RHEL-like distributions, including CentOS, run the following command:

     ```
     sudo yum install tableau-server-<version>.x86_64.rpm
     ```

     **Note:** When you use `yum` to install Tableau Server, all dependent packages are automatically downloaded and installed. This is the preferred method for installing Tableau. If your organization does not allow you to use `yum` and you must install using `rpm -i`, you must also install all dependent packages separately.

   - On Ubuntu, run the following command:

     ```
     sudo apt-get update
     sudo apt-get -y install gdebi-core
     sudo gdebi -n tableau-server-<version>_amd64.deb
     ```

4. Navigate to the `scripts` directory:

   ```
   cd /opt/tableau/tableau_server/packages/scripts.<version>/
   ```

5. Run the following script to start TSM:

   ```
   sudo ./initialize-tsm --accepteula
   ```

6. After initialization is complete, close the terminal session:

   ```
   exit or logout
   ```
Step 2: Activate and register Tableau Server

Before you can configure Tableau Server you must activate a license and register. For more details, see Activate and Register Tableau Server.

1. Log on as a user with sudo access and log in to TSM. Run the following command:
   
   tsm login -u <username>

2. Activate the Tableau Server license. Run the following command:

   tsm licenses activate -k <license_key>

3. Register Tableau Server. Generate a template that you can edit by running the following command:

   tsm register --template > /path/to/registration_file.json

4. Open a text editing program, fill in the registration file, save it, then pass it with the following command:

   tsm register --file /path/to/registration_file.json

Step 3: Configure local identity store

You must configure the identity store settings. This procedure simplifies installation setting identity store to local authentication. Workbook examples are installed by default. For more details on customizing these defaults, see Configure Initial Node Settings.

- Pass the configuration file with the following command:

   tsm settings import -f /opt/tableau/tableau_server-/packages/scripts.<version>/config.json

Step 4: Finalize installation

The final steps of installation are to apply changes, initialize and start TSM, and then to create the administration account. More details about these steps are at Finalize Installation.
1. Apply the configurations you made in the previous steps. Run the following command:

   \texttt{tsm pending-changes apply}

2. Initialize and start Tableau Server. Run the following command:

   \texttt{tsm initialize --start-server --request-timeout 1800}

3. Create the Tableau Server administrator account. Run the following command:

   \texttt{tabcmd initialuser --server 'localhost:80' --username 'admin' --password '<password>'}

   Where ' <password> ' is a strong password. Enclose the password and other arguments in single quotes.

Use the admin account that you created to access the Tableau Server admin web pages. See Sign in to Tableau Server Admin Pages.

\textbf{Step 5: Validate your installation (optional)}

A good way to validate that Tableau Server is installed and running properly is to look at the built-in administrative views.

1. Download PostgreSQL drivers from the \textbf{Driver Download page}.

2. Install the drivers on your Tableau Server computer:
   
   - On CentOS and RHEL, download the .rpm file and then run the following command:

     \texttt{sudo yum install tableau-postgresql-odbc-9.5.3-1.x86_64.rpm}

   - On Ubuntu, download the .deb file and then run the following command:

     \texttt{sudo dpkg -i tableau-postgresql-odbc_9.5.3_amd64.deb}
3. To validate that the drivers installed, navigate to the Administrative Views in Tableau Server.

Distributed and High Availability Tableau Server Installations

The most basic way to run Tableau Server is to install a single node. With this type of installation you have a fully functional Tableau Server, with all Tableau Services Manager (TSM) and Tableau Server processes running on that single node, but this may not be the optimal way to use Tableau Server. You can decide how to install Tableau based on your organization’s needs, and your resources.

- **Single-node installation**—This type of installation is reasonable for testing, running trials, and for environments that can handle occasional downtime and system availability due to lack of redundancy. All server processes are running on a single machine. There is less redundancy and fewer safeguards in the event of a problem with one of the server processes. You also need to make sure the computer you install Tableau Server on has adequate resources to handle the processes and the demands of users and data.

- **Distributed installation**—This type of installation is also called a multi-node installation and requires multiple computers so you can install and run server processes on those distributed nodes. Spreading the server processes out over multiple nodes can extend the reliability and efficiency of Tableau Server by providing redundancy and additional computing power. With the right configuration, a distributed installation can also provide you with automatic repository failover. For more information on failover, see Repository Failover.

- **Highly available (HA) installation**—An HA installation of Tableau Server is a special type of multi-node installation with a minimum of three nodes and multiple instances of key processes (the Repository, File Store/Data Engine (Hyper), and Coordination Service) on different computers. With an HA installation, there is built-in
redundancy of those key processes, including multiple File Stores, and automatic Repository failover. The goal is to minimize system downtime by eliminating single points of failure, and enabling detection of failures with failover where possible.

Prerequisite

These instructions assume that your cluster meets the Distributed Requirements.

**Note:** You cannot install Tableau Server on a combination of Linux and Windows computers.

Creating a distributed Tableau Server installation

These are the general steps you follow to create a distributed installation of Tableau Server:

1. Begin by installing Tableau Server on your initial node.
   
   For details, see Install and Configure Tableau Server.

2. Generate a node configuration (bootstrap) file on the initial node.

   For details, see Generate the node bootstrap file.

3. Install Tableau Server on an additional node using the node bootstrap file.

   For more information, see Install an additional node.

4. Configure your additional node with the processes you want to run on it.

   For more information, see Configure the additional node.

5. Repeat Steps 3 and 4 for any additional nodes you want to install.

6. Deploy a new Coordination Service ensemble.

   For more information, see Deploy a Coordination Service Ensemble.
Creating a highly available (HA) Tableau Server installation

A high availability Tableau Server installation is a special type of distributed installation, designed to accommodate failure in key server components without loss of complete server functionality. To create an HA installation, follow the same steps you take to create a distributed deployment but include additional steps to make the deployment highly available. These additional steps include adding at least two additional nodes (for a minimum of three nodes in the cluster), adding a second instance of the repository, and second instances of the data engine/file store, adding additional gateway processes, and deploying a Coordination Service ensemble. You can also add a load balancer to distribute requests among the gateways.

At a high level, these are the steps you follow to create a highly available installation of Tableau Server:

1. Begin by installing Tableau Server on your initial node.
   For details, see Install and Configure Tableau Server.

2. Generate a node configuration (bootstrap) file on the initial node.
   For details, see Generate the node bootstrap file.

3. Install Tableau Server on at least two additional nodes using the node bootstrap file.
   For more information, see Install an additional node.

4. Configure each additional node with the processes you want to run on it. These must include a second copy of the Tableau Server repository, and a second copy of the data engine and file store, as well as additional instances of the gateway.
   For more information, see Configure the additional node.

5. Deploy a Coordination Service ensemble.
   For more information, see Deploy a Coordination Service Ensemble.
6. (Optional) Configure a load balancer.

For more information, see Add a Load Balancer.

For details on how to create a three-node HA installation, see Example: Install and Configure a Three-Node HA Cluster.

Recover from an initial node failure

With a Tableau Server installation, the initial node includes two services that are only installed on that node, the License service, and the TSM Controller. If there is a problem with the initial node, Tableau Server may not continue to function, even when configured for high availability. To recover from a situation where the initial node fails, you can move the TSM Controller and the License service to one of your already configured nodes. This allows you to recover from the failure while using resources you already have in the cluster. You do not have to configure a standby initial node in case the initial node fails.

For details on how to recover from a failure on the initial node, see Recover from an Initial Node Failure.

Configure Coordination Service ensemble on additional nodes

Configuring a Coordination Service on multiple nodes provides additional duplication of processes and so reduces the possibility of server downtime due to an issue with one of the Coordination Service nodes. For details on how to deploy a Coordination Service ensemble on your cluster, see Deploy a Coordination Service Ensemble.

Distributed Requirements

Before you start to configure a distributed Tableau Server installation, read the following and make sure you meet the requirements.
Hardware

All the computers you use in your cluster must meet the requirements described in Confirm Requirements, but the computers do not need to be identical.

Additional hardware guidelines for high availability

For a Tableau Server installation to support failover and high availability, it must meet these additional guidelines:

- **Failover—three computers**: To configure a cluster that provides failover support for the file store/data engine (Hyper) and repository processes, you need at least three computers or VMs: an initial node and two additional nodes.

- **Multiple gateways—three computers and a load balancer**: Adding multiple Gateway processes to your Tableau Server installation and using a load balancer to automatically distribute requests to those gateways enhances the reliability of Tableau further. To configure a cluster that provides failover support and multiple gateways, you need to add a load balancer to front your three-node cluster.

Software

- All nodes in a distributed installation must be running the same version of Tableau Server.

- All nodes must be running the same distribution of Linux.

Installation location

When you start Tableau Services Manager (TSM) you can specify a non-default location for the data directory. This location must match on all nodes in your Tableau Server installation.
Networking and Ports

- **Ports**: As with any distributed system, the computers or VMs you use need to be able to communicate with one another.

- **Same subnet**: Do not install a distributed system across multiple subnets. Each node in the cluster must be installed on the same subnet.

- **Static IP addresses**: Any computer running Tableau Server, whether it's a single server installation or part of a cluster, must have a static IP address.

- **Discoverable**: Each node in the cluster must be discoverable from other node computers using DNS or a local host file.

- **Time zone and time**: Each node in the cluster must be in the same timezone, with their system clocks synchronized. Many Linux distributions have clock synchronization enabled. You should use whatever synchronization tools recommended by your distro. Two examples of synchronization tools are chronyd for RHEL-like distros, and ntpd for other distros.

Best Practices

- **Backup**: It's a best practice to create a backup prior to making significant system changes. See Back Up Tableau Server Data for steps.

SSL

If you are planning to configure SSL for a highly available Tableau Server cluster with multiple gateways and a load balancer (learn more), make sure that the SSL certificate you use was issued for the load balancer's host name. See Configure SSL for External HTTP Traffic to and from Tableau Server (Linux) for other details.
Install Tableau Server on Additional Nodes

After you install Tableau Server on one computer (or node), the server is functional and ready for use, but it has no redundancy. If there is a problem with a process or a problem with the computer, Tableau Server may be unavailable. In addition, all processes are running on a single computer, so there can be contention for resources on that computer.

You can extend your Tableau Server installation by adding Tableau to additional nodes, creating a distributed installation. This article describes the general steps for installing Tableau Server on additional nodes and assumes you have already installed Tableau on an initial node. For more information on installing Tableau on the initial node, see Install Tableau Server Package.

**Note:** If you are installing Tableau Server on multiple nodes, you should install and configure one node at a time. This makes it easier to troubleshoot any issues you might run into.

The general steps for installing Tableau Server on an additional node are:

- Confirm prerequisites
- Generate the node bootstrap file
- Install an additional node
- Configure an additional node
- Install drivers

**Confirm prerequisites**

Before you install Tableau Server on any node, ensure that each node meets the system requirements. For more information, see Confirm Requirements.

**Generate the node bootstrap file**

To install Tableau Server on additional computers you use the same installer you did for the initial node, along with a "bootstrap" file you generate on the initial node.
**Important**: The bootstrap file contains a copy of the master keystore file used for encrypting the configuration secrets. We strongly recommend that you take additional measures to secure the bootstrap file using mechanisms as described in Securing secrets for import and export operations.

1. On the first node, open a terminal and type this command to generate the bootstrap file:

   ```
   tsm topology nodes get-bootstrap-file --file <bootstrap>.json
   ```

2. Copy the original installer you used on the first computer along with the bootstrap.json file you generated and put them in a location accessible from the new computer you are adding Tableau Server to. This could be a mounted network share, or directly on the new computer.

**Install an additional node**

1. Open a terminal, and navigate to the directory where you copied the installer you used on the initial node.

2. Use the package manager to install the Tableau Server package.

   ```
   sudo yum install tableau-server-<version>.x86_64.rpm
   ```

**Note**: When you use `yum` to install Tableau Server, all dependent packages are automatically downloaded and installed. This is the preferred method for installing Tableau. If your organization does not allow you to use `yum` and you must install using `rpm -i`, you must also install all dependent packages separately.
On Ubuntu, run the following command:

```
sudo apt-get update
dsudo apt-get -y install gdebi-core
dsudo gdebi -n tableau-server-<version>_amd64.deb
```

3. Navigate to the `scripts` directory:

```
cd /opt/tableau/tableau_server/packages/scripts.<version>/
```

4. Initiate communication between the initial node and the new node:

```
sudo ./initialize-tsm -b /path/to/<bootstrap>.json -u <admin-user-on-first-node> --accepteula
```

- Use the `-b` flag to provide the path to the bootstrap file that you copied to the computer. If you have encrypted the bootstrap file, then you must pipe the file as described in Securing secrets for import and export operations.

- Use the `-u` flag to enter the user name of the administrative user on the initial node. This is the name of an administrative user on the computer, not the Tableau Server administrator. You will be prompted for the user password.

**Important:** You must enter the credentials of the same user that you used during the installation process on the primary node.

- The `--accepteula` flag accepts the Tableau Server End User License Agreement (EULA). The EULA is available in the following location:

```
/opt/tableau/tableau_server-/packages/docs.<version>/EULA.rtf
```
Configure the additional node

**Note:** This basic example illustrates how to set the topology on a node. For a more detailed, working multi-node example, see Example: Install and Configure a Three-Node HA Cluster.

On the initial (original) node, set the topology for the newly added node. The topology specifies which processes should run on the node, and how many instances of each process should run. The topology for the node will depend on your environment and organizational needs. The below are just examples of setting the topology.

1. Get the node-id for the new node:

   ```
   tsm topology list-nodes -v
   ```

   The `-v` option lists the nodes and the processes they are currently running. You can identify the newly added node because it will not have any processes on it.

2. Specify the individual processes that should run on this node.

   Do this with the following command:

   ```
   tsm topology set-process -n <nodeID> -pr <processname> -c <n>
   ```

   You must add an instance of the Cluster Controller to each node.

   For example, to add the Cluster Controller, two instances of the Backgrounder, and a Gateway to node2:

   ```
   tsm topology set-process -n node2 -pr clustercontroller -c 1
   tsm topology set-process -n node2 -pr backgrounder -c 2
   tsm topology set-process -n node2 -pr gateway -c 1
   ```
The specific processes and process counts you set will depend on your organizational environment and needs. Some processes are added automatically when you add another process. For more information, see Configure Nodes and Process Reference.

3. Apply the node configuration. The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

```
 tsm pending-changes apply
```

A warning about deploying a Coordination Service ensemble displays if you have deployed a three- or five-node cluster. If this is the only warning, you can safely override it using the `--ignore-warnings` option to apply the configuration changes in spite of the warning.

```
 tsm pending-changes apply --ignore-warnings
```

**Configure firewall rules (optional)**

If you are running a local firewall, then you need to configure firewall rules for all the nodes in the cluster. For more information, see Configure Local Firewall.

**Install drivers**

You need to install drivers so that Tableau Server can connect to data and run extracts. Install these drivers on nodes that are running any of the following processes:

- VizQL Server (vizqlserver)
- Application Server (vizportal)
- Data Server (dataserver)
- Backgrounder (backgrounder)
Drivers and administrative views

If you want to use the built-in administrative views in Tableau Server, you also need to install the PostgreSQL driver on any nodes running any of the above processes.

For more information, see Install Drivers.

Add a Load Balancer

You can enhance the reliability of Tableau Server by running gateways on multiple nodes, and configuring a load balancer to distribute requests across the gateways. Unlike the repository process, which can be active or passive, all gateway processes are active. If one gateway in a cluster becomes unavailable, the load balancer stops sending requests to it. The load balancer algorithm you choose determines how the gateways will route client requests.

**Note:** If you will be using Kerberos authentication, you need to configure Tableau Server for your load balancer before you configure Tableau Server for Kerberos. For more information, see Configure Kerberos.

Guidelines

Note the following as you configure your load balancer to work with Tableau Server:

- **Tested load balancers:** Tableau Server clusters with multiple gateways have been tested with Apache and F5 load balancers.
  
  If you are using an Apache load balancer and creating custom administrative views, you need to connect directly to the Tableau Server repository. You cannot connect through the load balancer.

- **Tableau Server URL:** When a load balancer is in front of a Tableau Server cluster, the URL that's accessed by Tableau Server users belongs to the load balancer, not the initial Tableau Server node.

- **Trusted host settings:** The computer running the load balancer must be identified to
Tableau Server as a trusted host. See the procedure below for how to configure Tableau Server.

- **Proxy server configurations:** The settings used to identify a load balancer to Tableau Server are the same ones that are used to identify a proxy server. If your Tableau Server cluster requires both a proxy server and a load balancer, both must use a single external URL defined in `gateway.public.host` and all proxy servers and load balancers must be specified in `gateway.trusted` and `gateway.trusted_hosts`. For more information, see Configuring Reverse Proxies for Tableau Server.

**Configure Tableau Server to work with a load balancer**

Use the following key-value pairs to configure Tableau Server for a load balancer. You can set the values individually with the `tsm configuration set` command.

Or, use the configuration file template example at the end of this topic to create a `configKey json` file. After you have filled in the options with the appropriate values, pass the json file and apply settings with `tsm` commands.

**Load balancer configuration key-value pairs**

The key-value pairs described in this section can be set with the `tsm configuration set` command. After you have set all values, run the following command:

```
tsm pending-changes apply
```

**gateway.public.host**

**Value:** name

The value, `name` is the URL that will be used to reach Tableau Server through the load balancer.

For example, if Tableau Server is reached by entering `tableau.example.com` in a browser address bar, enter this command:
tsm configuration set -k gateway.public.host -v "tableau.example.com"

gateway.public.port

Value: port_number

By default, Tableau assumes that the load balancer is listening on port 80 for external communications. Use this key to specify a different port.

For example, if your load balancer is configured for SSL and listening on port 443, enter the following command:

tsm configuration set -k gateway.public.port -v "443"

gateway.trusted

Value: server

Specify the load balancer(s) IPv4 address or computer name.

The value for server can be a comma-separated list, for example:

tsm configuration set -k gateway.trusted -v "10.32.139.45, 10.32.139.46, 10.32.139.47"

or

tsm configuration set -k gateway.trusted -v "proxy1, proxy2, proxy3"

gateway.trusted_hosts

Value:

Specify alternate names for the load balancer(s), such as its fully-qualified domain name, any non-fully-qualified domain names, and any aliases. These are the names a user might type in a browser. Separate each name with a comma:
tsm configuration set -k gateway.trusted_hosts -v "name1, name2, name3"

For example:

tsm configuration set -k gateway.trusted_hosts -v "lb.example.com, lb, ftp.example.com, www.example.com"

Configuration template example

You can configure all of the settings in one operation by customizing the following configuration template example and creating a configKey json file.

Refer to the key-value pair descriptions above to customize the values for your organization.

```json
{
  "configKeys": {
    "gateway.public.host": "tableau.example.com",
    "gateway.public.port": "443",
    "gateway.trusted": "proxy1, proxy2, proxy3",
    "gateway.trusted_hosts": "lb.example.com, lb, ftp.example.com, www.example.com"
  }
}
```

When you are finished, save the json file.

Pass the json file with tsm to set the configuration, and then apply settings by running the following commands:

```bash
ts m settings import -f /path/to/file.json

ts m pending-changes apply
```

For more information about using configKey schemes, see Configuration File Example.
Example: Install and Configure a Three-Node HA Cluster

A multi-node high availability installation of Tableau Server can help to maximize the efficiency and availability of Tableau Server. When configuring a multi-node Tableau Server deployment, the steps you perform are designed to build in redundancy, helping to reduce your potential downtime. In addition to simply improving efficiency by moving or adding server processes to additional nodes, you can create a highly available (HA) installation of Tableau Server by satisfying these requirements:

- Add additional nodes for a total of at least three nodes
- Deploy a Coordination Service ensemble on at least three nodes
- Add a second instance of the File Store on one of the additional nodes (Data Engine will be installed automatically, if it is not already on the node)
- Add a second instance of the Repository (pgsql) on one of the additional nodes

A Tableau Server installation that includes these additions will have built-in redundancy and can support failover in the event of a problem with the repository. This example shows how to do this, and more.

A Single Server System

After installing Tableau Server on an initial node, you have a system that is running everything it needs to function. It has at least one instance of all server processes and is the most basic configuration of Tableau Server. It has no redundancy. The server topology looks like this (some TSM-specific processes are not shown):
A Three-Node System

To build in redundancy, you need to add additional nodes to host instances of the repository and the data engine and file store processes. You can add instances of other processes, including multiple instances of a process on a node. In addition, to reduce the system’s vulnerability, you can run multiple gateways and additional instances of some of the server processes. The fewest number of computers required to achieve this configuration is three.
In the diagram below, the file store process has been moved off the initial node and onto both additional nodes. The repository has also been moved from the initial node to one of the additional nodes, and a second, passive instance has been added to the other new node. Finally, the stateless server processes (shown in blue) have been added to the additional nodes to provide redundancy.

Configuration steps

This procedure describes how to configure a three-node HA Tableau Server cluster with two repository instances and two file store/data engine instances on the additional nodes as pictured above.

Before you begin

Before you install Tableau Server on any additional nodes, ensure that each additional node meets the distributed requirements. See Distributed Requirements for details.
Step 1: Install the initial node

See Install and Configure Tableau Server.

Step 2: Generate a bootstrap file for the additional nodes

To install Tableau Server on additional computers you use the same installer you did for the initial node, along with a "bootstrap" file you generate on the initial node.

**Important**: The bootstrap file contains a copy of the master keystore file used for encrypting the configuration secrets. We strongly recommend that you take additional measures to secure the bootstrap file using mechanisms as described in Securing secrets for import and export operations.

1. On the first node, open a terminal and type this command to generate the bootstrap file:

   ```
tsm topology nodes get-bootstrap-file --file <bootstrap>.json
   
   ```

2. Copy the original installer you used on the first computer along with the bootstrap.json file you generated and put them in a location accessible from the new computer you are adding Tableau Server to. This could be a mounted network share, or directly on the new computer.

Step 3: Install an additional node

1. On the first additional node, open a terminal session, and navigate to the directory where you copied the installer you used on the initial node.

2. Use the package manager to install the Tableau Server package.
   
   - On RHEL-like distributions, including CentOS, run the following command:
     ```
sudo yum install tableau-server-<version>.x86_64.rpm
     ```
Note: When you use `yum` to install Tableau Server, all dependent packages are automatically downloaded and installed. This is the preferred method for installing Tableau. If your organization does not allow you to use `yum` and you must install using `rpm -i`, you must also install all dependent packages separately.

- On Ubuntu, run the following command:
  ```
  sudo apt-get update
  sudo apt-get -y install gdebi-core
  sudo gdebi -n tableau-server-<version>_amd64.deb
  ```

3. Navigate to the `scripts` directory:
   ```
   cd /opt/tableau/tableau_server/packages/scripts.<version>/
   ```

4. Initiate communication between the initial node and the new node:
   ```
   sudo ./initialize-tsm -b /path/to/<bootstrap>.json -u <admin-user-on-first-node> --accepteula
   ```

   - Use the `-b` flag to provide the path to the bootstrap file that you copied to the computer. If you have encrypted the bootstrap file, then you must pipe the file as described in Securing secrets for import and export operations.

   - Use the `-u` flag to enter the user name of the administrative user on the initial node. This is the name of an administrative user on the computer, not the Tableau Server administrator. You will be prompted for the user password.

**Important:** You must enter the credentials of the same user that you used during the installation process on the primary node.
Step 4: Install a second additional node

Install Tableau Server on the second additional node:

1. On the second additional node, open a terminal session, and navigate to the directory where you copied the installer you used on the initial node.

2. Use the package manager to install the Tableau Server package.
   - On RHEL-like distributions, including CentOS, run the following command:
     
     ```bash
     sudo yum install tableau-server-<version>.x86_64.rpm
     ```
   - On Ubuntu, run the following command:
     
     ```bash
     sudo gdebi -i tableau-server-<version>_amd64.deb
     ```

3. Navigate to the `scripts` directory:

   ```bash
   cd /opt/tableau/tableau_server/packages/scripts.<version>/
   ```

4. Run the following command to initiate communication between the new node and the first node:

   ```bash
   sudo ./initialize- tsm -b /<path>/<to>/bootstrap>.json -u
   <admin-user-on-first-node> --accepteula
   ```

   - Use the `-b` flag to provide the path to the node bootstrap file that you copied to the computer. If you have encrypted the node configuration file, then you must pipe the file as described in Securing secrets for import and export operations.
   - Use the `-u` flag to enter the user name of the administrative user on the initial
node. This is the name of an administrative user on the computer, not the Tableau Server administrator. You will be prompted for the user password.

**Important:** You must enter the credentials of the same user that you used during the installation process on the primary node.

- The `--accepteula` flag accepts the Tableau Server End User License Agreement (EULA). The EULA is available in the following location.

```
/opt/tableau/tableau_server-
/packages/docs.<version>/EULA.rtf
```

**Step 5: Add a process to the additional nodes**

1. On the initial node, configure a cluster controller instance on each additional node:

   ```
   tsm topology set-process -n <nodeID_second-node> -pr
custercontroller -c 1
   
   tsm topology set-process -n <nodeID_third-node> -pr clustercontroller -c 1
   ```

2. Apply the node configuration changes:

   ```
   tsm pending-changes apply --ignore-warnings
   ```
   
   A warning about deploying a Coordination Service ensemble displays because you have deployed a three-node cluster. Use the `--ignore-warnings` option to apply the configuration changes in spite of the warning. You will deploy a new Coordination Service ensemble in the next step.

**Step 6: Deploy a Coordination Service ensemble**

If you install a total of three or more nodes, you should also deploy a Coordination Service ensemble. If you do not, you will get a warning message every time you make changes to the
server configuration or topology. You can ignore this message, but as a best practice you should deploy a multi-node Coordination Service ensemble.

When you install Tableau Server, a single instance of the Coordination Service is installed on the initial node. TSM and Tableau Server depend on the Coordination Service to function properly, so to provide redundancy and ensure availability on multi-node installations, configure additional instances of the Coordination Service by deploying a Coordination Service ensemble. Coordination Service ensembles are installed with one, three, or five instances of the Coordination Service. In a three-node installation of Tableau Server, the recommended number of Coordination Service instances is three, one on each node.

1. On the initial node, stop the server:

   tsm stop

2. Get the node IDs for each node in the cluster:

   tsm topology list-nodes -v

3. Deploy a three-node Coordination Service ensemble:

   tsm topology deploy-coordination-service -n
   <nodeID1,nodeID2,nodeID3>

4. Wait for several minutes and verify that the Administration Agent and Administration Controller are running:

   tsm status -v

5. Remove the original single-node Coordination Service ensemble:

   tsm topology cleanup-coordination-service

For details on how to deploy a new Coordination Service ensemble, see Deploy a new Coordination Service ensemble.
Step 7: Configure processes for the second node

1. On the initial node, set the topology for the second node. The topology specifies which processes should run on the node, and how many instances of each process should run. For details about configuring nodes, see Configure Nodes. Data engine will be added automatically when you add other processes. For details on when data engine is added, see Process Reference.

   a. Get the node-id for the new node:  
      ```bash
tsm topology list-nodes -v
      ```

   b. Specify individual processes that will run on the new node:
      ```bash
tsm topology set-process -n <node-id> -pr gateway -c 1
      tsm topology set-process -n <node-id> -pr vizqlserver -c 2
      tsm topology set-process -n <node-id> -pr vizportal -c 2
      tsm topology set-process -n <node-id> -pr backgrounder -c 2
      tsm topology set-process -n <node-id> -pr cacheserver -c 2
      tsm topology set-process -n <node-id> -pr searchserver -c 1
      tsm topology set-process -n <node-id> -pr dataserver -c 2
      tsm topology set-process -n <node-id> -pr filestore -c 1
      tsm topology set-process -n <node-id> -pr pgsql -c 1
      ```
2. Apply the node configuration changes. You will be prompted with a message that Tableau Server will restart.

`tsm pending-changes apply`

**Step 8: Configure processes for third node**

On the initial node, set the topology for the new node. The topology specifies which processes should run on the node, and how many instances of each process should run. Data engine will be added automatically when you add other processes. For details on when data engine is added, see Process Reference.

1. Get the node-id for the new node:

   `tsm topology list-nodes -v`

2. Specify individual processes that will run on the new node:

   `tsm topology set-process -n <node-id> -pr gateway -c 1`
   `tsm topology set-process -n <node-id> -pr vizqlserver -c 2`
   `tsm topology set-process -n <node-id> -pr vizportal -c 2`
   `tsm topology set-process -n <node-id> -pr backgrounder -c 2`
   `tsm topology set-process -n <node-id> -pr cacheserver -c 2`
   `tsm topology set-process -n <node-id> -pr searchserver -c 1`
   `tsm topology set-process -n <node-id> -pr dataserver -c 2`
   `tsm topology set-process -n <node-id> -pr filestore -c 1`

You will add a repository to this node, but before you can add do this you need to remove the repository from the initial node.

3. Apply the node configuration. You will be prompted with a message that Tableau
Server will restart.

tsm pending-changes apply

4. Start the server:

tsm start

Step 9: Remove the Repository and File Store from initial node

You are limited to maximum of two instances of the repository, so you need to remove it from the initial node before you can add an instance to the third node.

1. Get the node-id for the initial node:

   tsm topology list-nodes -v.

2. Verify the repository on the second node has a status of "passive":

   tsm status -v.

   Once the new instance of the repository is fully synchronized and shows as "passive" you can remove the repository from the initial node by setting the process count to 0 (zero):

3. Remove the repository on the first node.

   tsm topology set-process -n <nodeID> -pr pgsql -c 0

4. Decommission and remove the file store.

   Once the additional instances of file store are installed and synchronized:

   a. Decommission the original file store on the initial node.

      tsm topology filestore decommission -n <nodeID> --override
b. When the decommission command completes, remove the file store from the node by applying the pending changes. You will be prompted with a message that Tableau Server will restart.

    tsm pending-changes apply

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Step 10: Configure the Repository on the third node

Add an instance of the repository to the third node. You need to do this after adding removing the repository from the initial node because you can configure a maximum of two instances of the repository in a cluster, and only if you have three or more nodes in the cluster.

1. On the initial node, get the node-id for the third node:

    tsm topology list-nodes -v

2. Specify the repository:

    tsm topology set-process -n <node-id> -pr pgsql -c 1

3. Apply the node configuration changes. You will be prompted with a message that Tableau Server will restart.

    tsm pending-changes apply

Step 11: Configure firewall rules (optional)

If you are running a local firewall, then you need to configure firewall rules for all the nodes in the cluster. For more information, see Configure Local Firewall.
Step 12: Remove processes from initial node (optional)

In certain circumstances you may want to limit the processes running on your initial node. Reasons for doing this include wanting to run as few processes as possible on the node to limit processing requests on the node. You might also remove processes from the node if you have a core-based license and do not want the initial node cores to count against your core use. In this case you would remove all licensed processes from the node. For more information, see Licensed processes.

In this example, we are removing all the stateless server processes, leaving only the Gateway, Search & Browse, Licensing, Cluster Controller, and Coordination Service processes (along with some required TSM services).

1. Remove the unwanted processes by setting the process count to 0 (zero):

   tsm topology set-process -n <node-id> -pr vizportal -c 0
   tsm topology set-process -n <node-id> -pr vizqlserver -c 0
   tsm topology set-process -n <node-id> -pr backgrounder -c 0
   tsm topology set-process -n <node-id> -pr cacheserver -c 0
   tsm topology set-process -n <node-id> -pr dataserver -c 0

2. Apply the node configuration changes. The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

   tsm pending-changes apply
Add a Load Balancer

At this point, all three nodes have gateways, which are used to route requests to available server processes. All gateways are active, but to further reduce the potential for downtime in the cluster, you can configure a load balancer. For more information, see Add a Load Balancer.

Deploy a Coordination Service Ensemble

The Coordination Service is built on Apache ZooKeeper, an open-source project, and coordinates activities on the server, guaranteeing a quorum in the event of a failure, and serving as the source of "truth" regarding the server topology, configuration, and state. The service is installed automatically on the initial Tableau Server node, but no additional instances are installed as you add other nodes. Because the successful functioning of Tableau Server depends on a properly functioning Coordination Service, we recommend that for server installations of three or more nodes, you add additional instances of the Coordination Service by deploying a new Coordination Service ensemble. This provides redundancy and improved availability in the event that one instance of the Coordination Service has problems.

Hardware requirements

The hardware you use for Tableau Server can have an effect on how well the Coordination Service runs. In particular:

- **Memory.** The Coordination Service maintains state information in memory. By design, the memory footprint is small, and is typically not a factor in overall server performance.

- **Disk speed.** Because the service stores state information on disk, it benefits from fast disk speed on the individual node computers.

- **Connection speed** between nodes. The service communicates continuously
between cluster nodes; a fast connection speeds between nodes helps with efficient synchronization.

Because of the Coordination Service is I/O intensive, if you are running Tableau Server on computers that meet or just exceed the minimum hardware requirements, you may want to configuration a Coordination Service ensemble that puts the service on nodes that are not being used for other server processes. This reduces the chance of delays due to I/O contention between server processes. For information on how to deploy an ensemble on dedicated nodes, see Coordination Service-only nodes below.

The Coordination Service Quorum

To ensure that the Coordination Service can work properly, the service requires a *quorum*—a minimum number of instances of the service. This means that the number of nodes you have in your installation impacts how many instances of the Coordination Service are running.

Number of Coordination Service instances to use

The maximum number of Coordination Service instances you can have in an ensemble on Tableau Server depends on how many Tableau Server nodes you have in your deployment. Configure a Coordination Service ensemble based on these guidelines:

<table>
<thead>
<tr>
<th>Total number of server nodes</th>
<th>Recommended number of Coordination Service nodes in ensemble (must be 1, 3, or 5)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-2 nodes</td>
<td>1 node</td>
<td>This is the default and requires no changes unless you want to move the Coordination Service off your initial node and onto your additional node.</td>
</tr>
<tr>
<td>3-4 nodes</td>
<td>3 nodes</td>
<td></td>
</tr>
<tr>
<td>5 or</td>
<td>5 nodes</td>
<td>Five is the maximum number of Coordination Service nodes.</td>
</tr>
<tr>
<td>Total number of server nodes</td>
<td>Recommended number of Coordination Service nodes in ensemble (must be 1, 3, or 5)</td>
<td>Notes</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>more nodes</td>
<td>Service instances you can install.</td>
<td></td>
</tr>
</tbody>
</table>

**Deploy a new Coordination Service ensemble**

The following steps illustrate how to deploy a new Coordination Service ensemble on an existing three-node Tableau Server cluster and clean up the old one.

1. On the initial node, open a terminal session and type this command to stop Tableau Server:

   `tsm stop`

2. Get the node IDs for each node in the cluster:

   `tsm topology list-nodes -v`

3. Use the `tsm topology deploy-coordination-service` command to add a new Coordination Service ensemble by adding the Coordination Service to specified nodes. You must specify the node(s) that the Coordination Service should be added to. The command also switches Tableau Server to use the new ensemble.

   For example, deploy the Coordination Service to all three nodes of a three-node cluster:

   `tsm topology deploy-coordination-service -n <nodeID1,nodeID2,nodeID3>`

4. After deploying the new Coordination Service ensemble, verify that the new
Coordination Service ensemble is running properly:

```bash
tsm status -v
```

You should see the initial node with two instances of the Coordination Service. The Coordination Service may show with a status of "unavailable" while the service is synchronizing between nodes on the cluster. If you see this status after deploying a new ensemble, wait several minutes before checking again. You should also see a status of "running" for the Administration Controller and Administration Agent.

**Note:** If there is a problem with an instance of the Coordination Service (if it shows as **stopped** for example), you can toggle back to your previous Coordination Service ensemble using the `tsm topology toggle-coordination-service` command. You can toggle back to the previous ensemble only if you have not run the `cleanup-coordination-service` command. Tableau Server cannot be running when you use this command.

5. Remove the old ensemble. This step is **required**. You cannot run Tableau Server with multiple Coordination Service ensembles configured.

```bash
tsm topology cleanup-coordination-service
```

Tableau Server cannot be running when you use this command.

6. Start Tableau Server:

```bash
tsm start
```

**Deploy an ensemble on Coordination Service-only nodes**

One way to accommodate the high I/O impact of the Coordination Service is to deploy an ensemble on nodes that only run the Coordination Service and the Cluster Controller. The following steps illustrate how to deploy a Coordination Service ensemble on an existing multi-node Tableau Server cluster.
Note: For a core-based Tableau Server license, Coordination Service-only nodes do not count against the total count of licensed cores.

1. Add additional nodes to your cluster and configure them to run only the Coordination Service.
   a. Add three additional nodes.
      See Install Tableau Server on Additional Nodes.
   b. From the initial node of the cluster, configure the new nodes to run the Cluster Controller:

      ```bash
      tsm topology set-process -pr clustercontroller -n <nodeID> -c 1
      ```
   c. Apply the configuration changes. The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

      ```bash
      tsm pending-changes apply
      ```

      A warning about deploying a Coordination Service ensemble displays because you have deployed a six-node cluster. If this is the only warning, you can safely override it using the `--ignore-warnings` option to apply the configuration changes in spite of the warning.

      ```bash
      tsm pending-changes apply --ignore-warnings
      ```
d. Confirm that all nodes are up and running:

```
tsm status -v
```

2. On the initial node of a multi-node cluster, open a terminal session and type this command to stop Tableau Server:

```
tsm stop
```

3. Get the node IDs for each node in the cluster:

```
tsm topology list-nodes -v
```

4. Use the `tsm topology deploy-coordination-service` command to add a new Coordination Service ensemble by adding the Coordination Service to specified nodes. You must specify the node(s) that the Coordination Service should be added to. The command also switches Tableau Server to use the new ensemble.

   For example, deploy the Coordination Service to three nodes of a six-node cluster:

```
tsm topology deploy-coordination-service -n <node4,node5,node6>
```

5. After deploying the new Coordination Service ensemble, verify that the new Coordination Service ensemble is running properly:

```
tsm status -v
```

You should see the initial node with one instance of the Coordination Service, plus the fourth, fifth, and sixth nodes each running one instance of the Coordination Service. The Coordination Service may show with a status of "unavailable" while the service is synchronizing between nodes on the cluster. If you see this status after deploying a new ensemble, wait several minutes before checking again.
Note: If there is a problem with an instance of the Coordination Service (if it shows as stopped for example), you can toggle back to your previous Coordination Service ensemble using the tsm topology toggle-coordination-service command. You can only toggle back to the previous ensemble if you have not run the cleanup-coordination-service command. Tableau Server cannot be running when you use this command.

6. Remove the old ensemble. This step is required. You cannot run Tableau Server with multiple Coordination Service ensembles configured.

   tsm topology cleanup-coordination-service

   Tableau Server cannot be running when you use this command.

7. Start Tableau Server:

   tsm start

Repository Failover

In a Tableau Server installation, the repository (pgsql) database is one of the key required processes. The Tableau Server repository stores information about Tableau Server users, groups and group assignments, permissions, projects, data sources, and extract metadata and refresh information. Because it is critical to the server functioning, Tableau Server has a built-in automatic "failover" for the repository when server is installed in a distributed environment that meets certain requirements.

Automatic repository failover

Automatic repository failover means that if there is a problem with the active Tableau Server repository, the server will automatically switch to using the passive repository. This does not happen immediately, to protect against momentary issues with the repository that don’t justify a switch, but if the repository is unavailable for more than five minutes, failover occurs.
For automatic repository failover to work, your Tableau Server installation needs:

- A minimum of three nodes
- Two instances of the repository installed

Optional but highly recommended:

- A multi-node Coordination Service ensemble deployed

With these conditions satisfied, repository failover will occur if the active repository becomes unavailable, either due to a problem with the process, or a problem with the node the process is running on. If the original repository becomes available again (if, for example, the node is restarted and all processes come up properly), it is made the passive repository, available for failover if necessary.

Manual repository failover

There may be reasons you want to shift back to the original repository after failover occurs. One reason would be if that instance of the repository is installed on a computer with more resources. To do this, use the tsm topology repository-failover command to manually switch back to the original repository. For more information, see tsm topology failover.repository.

Recover from an Initial Node Failure

The initial node of a multi-node Tableau Server on Linux deployment includes several processes that by default only run on that node. This means that if there is a problem with the initial node, you need to take some action to get those processes running on one of your other Tableau Server nodes. There is no need to have a backup node ready to add to your cluster. You can add these processes to one of your other existing nodes.

If your initial node fails for reasons that are recoverable in a relatively short amount of time (for example, a hardware failure you can correct), you should first attempt to bring the node
back up without using the procedure below. If you have a highly available installation of Tableau, and processes are available on other, still-functioning nodes, your users can continue to use Tableau Server while you correct simple issues on the initial node. The main impacts of a temporarily unavailable initial node is that you cannot manage the server, including making configuration changes, and stopping and starting Tableau, and, because the licensing service is on the initial node, there is a risk of processes becoming unlicensed.

**Note:** The steps in this article require server downtime and can be disruptive, and should only be used in the event of a catastrophic failure of the initial node. If you are unable to get your initial node running again, use the following steps to move key TSM processes to another node in your cluster.

General requirements

- As part of the process for setting up a multi-node Tableau Server installation you should have deployed a Coordination Service ensemble. The process below assumes there was a Coordination Ensemble deployed before there was a problem with the initial node. For more information about deploying a Coordination Service ensemble, see Deploy a Coordination Service Ensemble.

Move the TSM Controller and License Service to another node

If there is a problem with the initial node, the TSM Controller and the Licensing Service need to be started on another node. Follow these steps to get the Controller and Licensing Service working on another node.

1. On a node that is still working, run the Controller recovery script. At a terminal prompt on a working node, type the following command:

   ```bash
   sudo /opt/tableau/tableau_server-/packages/scripts.<version>/move-tsm-controller -n <nodeID>
   ```
where "nodeID" is the ID for the node you want the TSM Controller to run on. For example:

```
sudo /opt/tableau/tableau_server-
/packages/scripts.10400.17.0802.1319/move-tsm-controller -n node2
```

2. Verify the Controller is running on the node:

```
tsm status -v
```

3. Stop Tableau Server:

```
tsm stop
```

4. Add the License Service to the node:

```
tsm topology set-process -pr licenseservice -n <nodeID> -c 1
```

5. Remove the old License Service from the original node, where nodeID is the initial node that has failed:

```
tsm topology set-process -pr licenseservice -n <nodeID> -c 0
```

6. (Optional) You can also add other processes that had been running on the initial node but are not running on this node. For instance, to add an cache server:

```
tsm topology set-process -pr cacheserver -n node2 -c 1
```

7. Apply the changes:

```
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the
server is stopped, but in that case there is no restart. You can suppress the prompt using the \(-r\) option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

8. Restart the TSM Controller:

```
sudo systemctl restart tabadmincontroller_0.service
```

**Note:** This step may take a few minutes. If you apply pending changes before the controller has restarted, TSM will not be able to connect to the controller. You can verify that the controller is running by using the `tsm status -v` command. Tableau Server Administration Controller should be listed as "is running".

9. Apply pending changes (there may not appear to be any, but this step is required):

```
tsm pending-changes apply
```

10. Activate the Tableau Server license on the new Controller node:

```
tsm licenses activate -k <product-key>
```

11. Verify the license is properly activated:

```
tsm licenses list
```

12. If the initial node was running the Coordination Service, you need to deploy a new Coordination Service ensemble that does not include that node. If you have a three node cluster and the initial node was running the Coordination Service, you must deploy a new, single-instance Coordination Service ensemble on a different node. In this example, a single instance of the Coordination Service is being deployed to the second node:

```
tsm topology deploy-coordination-service -n <nodeID2>
```
Then clean up the old ensemble:

```bash
tsm topology cleanup-coordination-service
```

For more information about deploying Coordination Service ensembles, see Deploy a Coordination Service Ensemble.

13. If the initial node was running a File Store instance, you need to remove that instance:

```bash
tsm topology filestore decommission --delete-filestore
```

14. Apply pending changes, using the `--ignore-warnings` flag if the new Coordination Service ensemble you deployed in Step 12 above is a single node ensemble:

```bash
tsm pending-changes apply --ignore-warnings
```

15. Remove the initial node, where `nodeID` is the initial node that has failed:

```bash
tsm topology remove-nodes -n nodeID
```

16. Apply pending changes, using the `--ignore-warnings` flag if the new Coordination Service ensemble you deployed in Step 12 above is a single node ensemble:

```bash
tsm pending-changes apply --ignore-warnings
```

17. Start Tableau Server:

```bash
tsm start
```

At this point your server should be able to run. The next step is to attempt to recover the original initial node.
Configure Nodes

Use TSM CLI commands to configure the topology of a node. The initial node is configured with a default that includes all the processes used by Tableau Services Manager (TSM) and Tableau Server. When you add additional nodes you need to specify which processes will run on those nodes, and how many instances of those processes will run. You may also want to change the topology of the initial node, either adding instances of existing processes, or moving some of those processes to your additional nodes (this is common when setting up a distributed installation of Tableau Server).

- Adding processes to a node
- Changing the number of processes on a node
- Removing all instances of a process from a node
- Moving all instances of a process from one node to another node

To configure nodes, run commands from the initial node and use the node ID to specify which node you are configuring. To determine the node ID, use the tsm topology list-nodes command. Use the tsm topology set-process command to add, update or remove a process on a node. You need to specify the node you are configuring, the process you are adding, updating, or removing, and the number of instances of the process. After setting the topology for a node you need to apply the changes to Tableau Server.

Apply changes using the tsm pending-changes apply command. After the changes are applied, Tableau Server is returned to the state it was in before the command was run. This means that if it was running, it will be restarted, and if it was stopped it will remain stopped after pending changes have been applied. In most cases, if Tableau Server is running when you apply pending changes, the server is stopped so that changes can be applied, and then restarted. The exception is if you are changing the number of instances of Backgrounder, or VizQL Server on an existing node. With changes to those processes on an existing node, Tableau Server does not have to be stopped if it is running.
You need the node ID for a node in order to configure the node. To determine the node ID, use this command:

```
tsm topology list-nodes -v
```

**Note:** Examples here show some process names. For a complete list, see Process Reference.

### Adding processes to a node

Use the `tsm topology set-process` command to add a process to a node. You need to specify the node you are configuring, the process you are adding, and the number of instances of the process.

For example, this command adds two instances of backgrounder to node1:

```
tsm topology set-process -n node1 -pr backgrounder -c 2
```

### Changing the number of processes on a node

Change the number of processes on a node by specifying an already configured process and providing a new value for the number of instances.

For example, on a node (node1) that is already running backgrounder, this command changes the number of instances to four:

```
tsm topology set-process -n node1 -pr backgrounder -c 4
```

### Removing all instances of a process from a node

Remove a process from a node by specifying a count of 0 instances for that process on the node.

For example, this command removes the backgrounder process from node1:

```
tsm topology set-process -n node1 -pr backgrounder -c 0
```
Moving all instances of a process from one node to another node

In most cases you move a process from one node to another by setting the process instance count on the first node to zero, and setting the count to a non-zero value on the second node.

For example, these commands remove Backgrounder from node0 and add two instances of it to node1:

```
    tsm topology set-process -n node0 -pr backgrounder -c 0
    tsm topology set-process -n node1 -pr backgrounder -c 2
```

Two processes require additional steps before you can set their instance count to zero: File Store, and the Repository (pgsql). You must have at least one instance of each of the processes in your Tableau Server installation, and you must add the second instance and allow it to synchronize with the first before you remove the process on the original node. For more information see Move the Repository Process and Move the File Store Process.

Remove a Node

If your Tableau Server installation includes a node you no longer need, you can remove it to simplify your installation, and to free up the hardware resources on that node.

**Note:** To remove a node from a cluster it must have been configured with a process at some point in the past. If you are removing a node on which you’ve not configured any processes, then you must add a process on it, run the `tsm pending-changes apply` command, and then remove the node. For example, you might add one instance of Cluster Controller to the node:

```
    tsm topology set-process -n <nodeID> -pr clustercontroller -c 1.
```

You cannot remove a node if:

- It is running the only instance of the Repository. You need to move the Repository to another node before removing the original node. See Move the Repository Process.
- It is running the only instance of the File Store. You need to move the File Store to another node before removing the original node. See Move the File Store Process.
- It is running an instance of the Coordination Service. You need to redeploy a new Coordination Service ensemble before removing the node. See Deploy a Coordination Service Ensemble.

Use the tsm topology remove-nodes command to remove nodes from a cluster. After running the remove-nodes command, apply pending changes to complete the removal.

The following example shows how to remove node2 from an existing cluster:

```bash
tsm topology remove-nodes --node-names "node2"

```

```bash
tsm pending-changes apply
```

### Move the Repository Process

If you want to move the only instance of your repository, or are deleting a node from your Tableau Server cluster and that node is hosting the only instance of the repository, you must add a second instance of the repository, or move the existing instance to another node before deleting the node. There must always be at least one active instance of the repository, so you cannot remove an instance if it is the only instance. You cannot add a second instance of the repository unless you have at least three nodes. The steps below assume you have at least three nodes in your Tableau Server installation.

**Note:** If you are also moving the file store, you can move the repository at the same time. See Move the File Store Process.

Before making a change to the repository, create a full backup of Tableau Server. For more information, see tsm maintenance backup.
Add a new instance of the repository.

1. Add the repository (pgsql) to another node:

   `tsm topology set-process -n <nodeID> -pr pgsql -c 1`

2. Apply the changes. The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

   `tsm pending-changes apply`

3. Wait for the new repository on the second node to synchronize with the repository on the first node.

   `tsm status -v`

   Wait until the new repository status shows as "passive".

Remove an instance of the repository.

Once the new instance of the repository is fully synchronized and shows as "passive" you can remove the original instance:

1. Remove the repository from the first node by setting the process count to 0 (zero):

   `tsm topology set-process -n <nodeID> -pr pgsql -c 0`

2. Apply the change. The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

   `tsm pending-changes apply`
Move the File Store Process

You cannot remove an instance of file store if it is the only instance on the server. You cannot run Tableau Server without at least one instance of file store. This means if you need to move the file store, or if you are deleting a server node that is hosting the only instance of the file store, you must first move it to another node.

Moving the file store is a two-part process:

- Adding a second instance of file store (if there is not an existing second instance).
- Decommissioning and removing an instance of file store.

Adding a second instance of file store

1. Create a full backup of Tableau Server. For more information, see Back Up Tableau Server Data.

2. Add the file store to a second node.

   ```
   tsm topology set-process -n <nodeID> -pr filestore -c 1
   ```

   The file store is automatically added. Data engine is also added if it is not already on the node.

   Apply the configuration changes:

   ```
   tsm pending-changes apply
   ```

   The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

3. Check the status of the new file store instance.
tsm status -v

Wait for the new file store to synchronize with the file store on the first node. When synchronization is complete the new file store has a status of "running" instead of "synchronizing".

Decommissioning and removing an instance of file store

Once you have a second instance of file store installed and synchronized you can decommission and remove the original instance. You must decommission the original instance before you remove it. Doing this guarantees that any unique files on the file store node are duplicated to another file store node.

1. Decommission the original file store:

   tsm topology filestore decommission -n <nodeID> --override

2. When the decommission command completes, remove the file store from the node by applying the pending configuration changes. The file store is automatically removed. Data engine is also removed unless an instance of one of these processes is installed on the node: VizQL Server, Application Server (Vizportal), Data Server, or Backgrounder.

   tsm pending-changes apply

   The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the -r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.
Update Configuration for a New IP Address

You can and should use hostnames (computer names) of your server computers when configuring Tableau Server. Doing this makes it relatively easy to update the server configuration if an IP address changes, and using hostnames is the only way Tableau Server supports IPv6 networks.

If you use IP addresses instead of hostnames when initially configuring Tableau Server, the process to update the configuration when an IP address changes is significantly more complicated. Tableau Server uses IP addresses for communications between processes and nodes. As a result, if an IP address changes, Tableau Server will not function properly until you update the server configuration. This might happen if a VM is down for a significant amount of time (longer than the DNS record time to live) and the network environment uses DHCP.

Updating a Tableau Server configuration that was originally configured using hostnames

To update the configuration if the IP address changes on the initial node:

- On each additional node, restart Tableau Server.

To update the configuration if the IP address changes on an additional node:

1. On the initial Tableau Server node, open a terminal session.

2. Stop Tableau Server by typing this command:

   tsm stop

   **Note:** If the IP address on the initial node has changed, the `tsm stop` command will not work because the additional nodes will not accept connections from the new IP address. If you cannot stop Tableau Server, restart the additional nodes and then try again to stop Tableau Server.
3. Update the configuration:

    tsm pending-changes apply

    This will apply the configuration change and restart Tableau Server.

Post Installation Tasks

Security Hardening Checklist

The following list provides recommendations for improving the security ("hardening") of your Tableau Server installation.

1. Update to the current version

We recommend that you always run the latest version of Tableau Server. Additionally, Tableau periodically publishes maintenance releases of Tableau Server that include fixes for known security vulnerabilities. (Information regarding known security vulnerabilities can be found on the Security Bulletins page.) We recommend that you review maintenance release notifications to determine whether you should install them.

To get the latest version or maintenance release of Tableau Server, visit the Customer Portal page.

2. Configure SSL/TLS with a valid, trusted certificate

Secure Sockets Layer (SSL/TLS) is essential for helping to protect the security of communications with Tableau Server. Configure Tableau Server with a valid, trusted certificate (not a self-signed certificate) so that Tableau Desktop, mobile devices, and web clients can connect to the server over a secured connection. For more information, see SSL.
3. Disable older versions of TLS

Tableau Server uses TLS to authenticate and encrypt many connections between components and with external clients. External clients, such as browsers, Tableau Desktop, Tableau Mobile connect to Tableau using TLS over HTTPS. Transport layer security (TLS) is an improved version of SSL. In fact, older versions of SSL (SSL v2 and SSL v3) are no longer considered to be adequately secure communication standards. As a result, Tableau Server does not allow external clients to use SSL v2 or SSL v3 protocols to connect. We recommend that you only allow external clients to connect to Tableau Server with TLS v1.2.

Specially, we recommend that you disable TLS v1 and TLS v1.1 on Tableau Server. However, before you disable a specific version of TLS, verify that the browsers that your users connect to Tableau Server with support TLS v1.2. In some cases, you may need to preserve support for TLSv1.1.

The following tsm command enables TLS v1.2 (using the "all" parameter) and disables SSL v2, SSL v3, TLS v1, and TLS v1.1 (by prepending the minus [-] character to a given protocol).

```shell
tsm configuration set -k ssl.protocols -v 'all -SSLv2 -SSLv3 -TLSv1 -TLSv1.1'
```

```shell
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

4. Disable Triple-DES cipher suite

The Triple-DES cipher suite is no longer considered adequate to encrypt sessions on the internet. Specifically, running Triple-DES ciphers leaves the Tableau Server vulnerable to
information disclosure and denial of service attacks. You can learn more at the National Vulnerability Database webpage for CVE-2016-2183.

Triple-DES is enabled by default on the version of OpenSSL that is running on Tableau Server. However, other deprecated cipher suites (MD5 and RC4) are disabled. To add Triple-DES to the list of disabled ciphers, run the following commands. (The !aNULL parameter forces clients to use a legitimate cipher.)

```sh
tsm configuration set -k ssl.ciphersuite -v
'HIGH:MEDIUM:!aNULL:!MD5:!RC4:!3DES'
```

5. Configure SSL encryption for internal traffic

Configure Tableau Server to use SSL to encrypt all traffic between the Postgres repository and other server components. By default, SSL is disabled for communications between server components and the repository. We recommend enabling internal SSL for all instances of Tableau Server, even single-server installations. Enabling internal SSL is especially important for multi-node deployments. See Configure SSL for Internal Postgres Communication

6. Enable firewall protection

Tableau Server was designed to operate inside a protected internal network.

**Important:** Do not run Tableau Server, or any components of Tableau Server on the internet or in a DMZ. Tableau Server must be run within the corporate network protected by an internet firewall. We recommend configuring a reverse proxy solution for internet clients that need to connect to Tableau Server. See Configuring Reverse Proxies for Tableau Server.

A local firewall should be enabled on the operating system to protect Tableau Server in single and multi-node deployments. In a distributed (multi-node) installation of Tableau
Server, communication between nodes does not use secure communication. Therefore, you should enable firewalls on the computers that host Tableau Server. See Configure Local Firewall.

To prevent a passive attacker from observing communications between nodes, configure a segregated virtual LAN or other network layer security solution.

7. Restrict access to the server computer and to important directories

Tableau Server configuration files and log files can contain information that is valuable to an attacker. Therefore, restrict physical access to the machine that is running Tableau Server. In addition, make sure that only authorized and trusted users have access to the Tableau Server files in the `/var/opt/tableau/tableau_server/` directory.

8. Generate fresh secrets and tokens

Any Tableau Server service that communicates with repository or the cache server must first authenticate with a secret token. The secret token is generated during Tableau Server setup. The encryption key that internal SSL uses to encrypt traffic to Postgres repository is also generated at during setup.

We recommend that after you install Tableau Server, you generate new encryption keys for your deployment.

These security assets can be regenerated with the `tsm security regenerate-internal-tokens` command.

Run the following commands:

```
tsm security regenerate-internal-tokens

tsm pending-changes apply
```

9. Disable services that you're not using

To minimize the attack surface of the Tableau Server, disable any connection points that are not needed.
REST API

The REST API interface is enabled by default. If no applications will make REST API calls to your installation of Tableau Server 9.3 (or later), disable it by using the following commands:

```
tsm configuration set -k api.server.enabled -v false
tsm pending-changes apply
```

You can disable REST API only on versions of Tableau Server 9.3 and later.

JMX Service

JMX is disabled by default. If it’s enabled but you’re not using it, you should disable it by using the following:

```
tsm configuration set -k service.jmx_enabled -v false
tsm pending-changes apply
```

10. Verify session lifetime configuration

By default, Tableau Server does not have an absolute session timeout. This means that client sessions can remain open indefinitely if the Tableau Server inactivity timeout is not exceeded. (The default inactivity timeout is 240 minutes.)

If your security policy requires it, you can set an absolute session timeout. Before you do that, you must enable session lifetime timeout. Run the following commands.

```
tsm configuration set -k wgserver.session.apply_lifetime_limit -v true
tsm configuration set -k wgserver.session.lifetime_limit -v 'value', where value is the number of minutes. The default is 1440, which is 24 hours.
tsm configuration set -k wgserver.session.idle_limit -v 'value', where value is the number of minutes. The default is 240.
tsm pending-changes apply
```
11. Enable HTTP Strict Transport Security for web browser clients

HTTP Strict Transport Security (HSTS) is a policy configured on web application services, such as Tableau Server. When a conforming browser encounters a web application running HSTS, then all communications with the service must be over a secured (HTTPS) connection. HSTS is supported by major browsers.

For more information about how HSTS works and the browsers that support it, see The Open Web Application Security Project web page, HTTP Strict Transport Security Cheat Sheet.

To enable HSTS, run the following commands on Tableau Server:

```bash
 tsm configuration set -k gateway.http.hsts -v true
```

By default, HSTS policy is set for one year (31536000 seconds). This time period specifies the amount of time in which the browser will access the server over HTTPS. You should consider setting a short max-age during initial roll-out of HSTS. To change this time period, run

```bash
 tsm configuration set -k gateway.http.hsts_options -v max-age-e=<seconds>
```

For example, to set HSTS policy time period to 30 days, enter

```bash
 tsm configuration set -k gateway.http.hsts_options -v max-age=2592000
```

```bash
 tsm pending-changes apply
```

12. Disable Guest access

Core-based licenses of Tableau Server include a Guest user option, which allows any user in your organization to see and interact with Tableau views embedded in web pages.

Guest user access is enabled by default on Tableau Servers deployed with core-based licensing.

Guest access allows users to see embedded views. The Guest user cannot browse the Tableau Server interface or see server interface elements in the view, such as user name, account settings, comments, and so on.
If your organization has deployed Tableau Server with core licensing and Guest access is not required, then disable Guest access.

You can disable Guest access at the server or site level.

You must be a server administrator to disable the Guest account at either the server or the site level.

**To disable Guest access at the server level:**

1. In the site menu, click **Manage All Sites** and then click **Settings > General**.
2. For **Guest Access**, clear the **Enable Guest account** check box.
3. Click **Save**.

**To disable Guest access for a site:**

1. In the site menu, select a site.
2. Click **Settings**, and on the Settings page, clear the **Enable Guest account** check box.

For more information, see Guest User.

**Change List**

<table>
<thead>
<tr>
<th>Date</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 2017</td>
<td>Ported and updated for Linux platform.</td>
</tr>
</tbody>
</table>

**Configure SMTP Setup**

Tableau Server can email server administrators about system failures, and email server users about subscribed views and data-driven alerts. First, however, you need to configure the SMTP server that Tableau Server uses to send email. After configuring SMTP and com-
Completing the steps to configure alerts (Configure Alerts), a restart of server will trigger an email alert. This confirms that you have set up alerts correctly.

1. For the initial configuration of SMTP, we recommend that you use the configuration file template below to create a json file.

   ```json
   {
   "configKeys": {
   "svcmonitor.notification.smtp.server": "SMTP server url",
   "svcmonitor.notification.smtp.send_account": "SMTP user name",
   "svcmonitor.notification.smtp.port": 25,
   "svcmonitor.notification.smtp.password": "SMTP server password",
   "svcmonitor.notification.smtp.ssl_enabled": false,
   "svcmonitor.notification.smtp.from_address": "From email address",
   "svcmonitor.notification.smtp.target_addresses": "To email address1, address2",
   "svcmonitor.notification.smtp.canonical_url": "Tableau Server URL"
   }
   }
   ```

2. Run the `tsm settings import -f /path/to/file.json` to pass the json file with the appropriate values to Tableau Services Manager to configure Tableau Server for SMTP. Tableau Services Manager will validate the entity values.

3. Run the `tsm pending-changes apply` command to apply the changes. See `tsm pending-changes apply`.

- 156 -
Configuration file reference

This table lists all of the options that can be used to configure SMTP.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc-mon-itor.notification.smtp.server</td>
<td>Address of SMTP server.</td>
</tr>
<tr>
<td>Example:</td>
<td>&quot;svc-mon-itor.notification.smtp.server&quot;: &quot;mail.example.com&quot;</td>
</tr>
<tr>
<td>svc-mon-itor.notification.smtp.send_account</td>
<td>User name for SMTP account.</td>
</tr>
<tr>
<td>svc-mon-itor.notification.smtp.port</td>
<td>Port number for SMTP server. The default is 25.</td>
</tr>
<tr>
<td>svc-mon-itor.notification.smtp.password</td>
<td>Password for SMTP server account.</td>
</tr>
<tr>
<td>Example:</td>
<td>&quot;svc-mon-itor.notification.smtp.password&quot;: &quot;password&quot;</td>
</tr>
<tr>
<td>svc-mon-itor.notification.smtp.ssl_enabled</td>
<td>Specifies whether the connection to the SMTP server is encrypted. The default is false.</td>
</tr>
</tbody>
</table>

**Note:** This should be left as false. Encryption should not be used for SMTP.
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>SMTP connections are not supported for alerts or subscriptions.</strong></td>
</tr>
<tr>
<td><strong>svc-monitor.notification.smtp.from_address</strong></td>
<td>Email address that will send an alert if there’s a system failure. The email address must have valid syntax (for example, <a href="mailto:ITalerts@bigco.com">ITalerts@bigco.com</a> or noreply@mycompany), but it does not have to be an actual email account on Tableau Server. (Some SMTP servers may require an actual email account, however.)</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You can override the system-wide email address on a per-site basis. For more information, see What is a site.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
</tr>
<tr>
<td></td>
<td>&quot;svc-monitor.notification.smtp.from_address&quot;: &quot;<a href="mailto:donot-reply@example.com">donot-reply@example.com</a>&quot;</td>
</tr>
<tr>
<td><strong>svc-monitor.notification.smtp.target_addresses</strong></td>
<td>Email address to receive alerts. If email alerts are enabled, you need to include at least one address. Separate multiple addresses with commas.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
</tr>
<tr>
<td></td>
<td>&quot;svc-monitor.notification.smtp.target_addresses&quot;: &quot;<a href="mailto:donot-reply@example.com">donot-reply@example.com</a>&quot;</td>
</tr>
</tbody>
</table>
You can configure the following server alerts:

- Email alerts for system failures
- Disk space usage:
  - Recording usage history
  - Email alerts when space crosses or remains below pre-configured thresholds

**Note:** You need to configure SMTP before you can configure subscriptions or alerts. For more information, see Configure SMTP Setup.

For information about configuring user subscription notifications, see Set Up a Server for Subscriptions.
Alerts for system failures

Tableau Server can send email alerts to server administrators when there is a system failure.

When you configure alerts, Tableau Server sends an email to the recipients listed in the `svc-monitor.notification.smtp.target_addresses` option of the SMTP configuration.

Tableau Server sends an email when the data engine, repository, or gateway server processes stop or restart, or when the initial Tableau Server node stops or restarts. If you are running a single-server installation (all processes on the same machine), health alerts are only sent when Tableau Server is up. No "down" alerts are sent. If you are running a distributed installation that’s configured for failover, a DOWN alert means that the active repository or a data engine instance has failed and the subsequent UP alert means that the passive instance (repository) or second instance (data engine) of that process has taken over.

To configure email alerts for system failures run the following commands:

```
 tsm configuration set -k svcmonitor.notification.smtp.enabled -v true
 tsmpending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Disk space monitoring and usage

When you configure Tableau Server to record disk space usage, information about free disk space is saved in the repository and you can view the usage history using the Administrative
To configure Tableau Server to record disk space usage, run the following commands:

tsm configuration set -k storage.monitoring.record_history_enabled -v true

```
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

You can configure Tableau Server to send email alerts when disk space usage on any node crosses a threshold, or remains below the threshold.

To enable email alerts for low disk space, run the following commands:

```
tsm configuration set -k storage.monitoring.email_enabled -v true
```

```
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

**Setting thresholds**

There are two thresholds you must set. Thresholds are expressed in percentage of disk space remaining.
If free disk space on any node in your Tableau Server cluster drops below this percentage, Tableau Server sends alerts.

You can set a warning threshold and a critical threshold. The critical threshold must be less than the warning threshold. The following commands show how to set a warning threshold of 15% and a critical threshold of 8%:

```
tsm configuration set -k storage_monitoring.warning_percent -v 15
```

```
tsm configuration set -k storage_monitoring.critical_percent -v 8
```

```
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Setting email interval

Set the `storage_monitoring.email_interval_min` key to specify how often, in minutes, warning and critical alerts should be sent. The default value is 60 minutes.

To set the email interval to 20 minutes, run the following commands:

```
tsm configuration set -k storage_monitoring.email_interval_min -v 20
```

```
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r`
option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

Configuring alerts with a json file

To make all alerts settings with a single configuration, you can pass a json file with the following command:

```bash
tsm settings import -f </path/to/file.json>
```

Copy and edit the following template to create a file for your alert configuration.

```json
{
    "configKeys": {
        "svcmonitor.notification.smtp.enabled": true,
        "storage.monitoring.record_history_enabled": true,
        "storage.monitoring.email_enabled": true,
        "storage.monitoring.warning_percent": 20,
        "storage.monitoring.critical_percent": 15,
        "storage.monitoring.email_interval_min": 60
    }
}
```

Install Drivers on Linux

Tableau connectors require a driver to talk to the database. Before you can connect to data sources from Tableau Server, you must install drivers for the data sources you want to connect to. You can find information about supported data sources for Tableau Server on Linux on the Tableau Server tech specs page. You can find driver links and installation instructions for all the supported connectors on the Driver Download page.

**Important:** You must install the PostgreSQL driver if you want to use the built-in administrative views.
Install drivers in a cluster

You need to install the drivers for your data sources on the initial node in a Tableau Server cluster. If you install Tableau Server on multiple nodes, you must also install drivers on any node that runs any of the following processes:

- Application Server (Vizportal)
- Backgrounder
- Data Server
- VizQL Server

Files and Permissions in TSM

This topic covers the permissions requirements needed for Tableau Services Manager (TSM) to access and use files. This information is intended for server administrators. This topic does not address permissions used for managing users and content within Tableau Server (permissions for content and users). For information related to these permissions, see Content Permissions and Ownership.

During installation of TSM and Tableau Server, a privileged user (tsmagent) and an unprivileged user (tableau) are created and added to a server authorized group (tableau). These enable the work done by TSM and Tableau Server processes. For detailed information about these users, see System User and sudo Privileges.

Permissions requirements for TSM apply to both files, and to the directories in which the files are placed. When TSM creates and manages files, the files get put into specific default locations with the necessary permissions and you don’t need to worry about setting permissions. When you create, copy, or move files yourself, or when you put files into non-default locations, you need to be aware of permission requirements so that TSM can properly access the files. Common cases (For information about using non-default locations, see tsm File Paths.)

General rules for permissions and TSM are:
• Files—If the *tableau* group has access to a file (if it is the group owner and has read access to the file), the users in the group have access to the file. An alternate approach is to give "other" read access.

• Directories—If the *tableau* group has read and execute access to the directory that contains a file, and any parent directories of that directory, the users in the group have access to the file.

Situations that may require you to adjust permissions include server backup files and site import archives that you copy from a different computer or to a non-default location, customization files such as logos or images, and security certificates such as SSL certificates.

For example, if you migrate from Tableau Server on Windows to Tableau Server on Linux, you use a backup created in Windows to restore data to your Linux server. Because this backup file isn't created by TSM, it may not have the correct permissions for the restore process to access it. You need to make sure the backup file and the directory structure you copy it into have the proper permissions. Similarly, if you are copying files like certificates to additional nodes in a cluster, you need to make sure the files and the directories you copy them into have the permissions the *tableau* user needs in order to access them.

**Setting permissions for individual files**

If you are using a file you copy to one of the default locations created by TSM, you need to make sure the ownership and permissions on the file allow TSM access by giving the *tableau* user read access. You can do this in one of two ways:

• You can give the *tableau* user read access by giving the *tableau* group (in a default installation) read and execute access to a file using the *chgrp* and *chmod* commands. For example:

```
chgrp tableau <backup>.tsbak
chmod g+rx <backup>.tsbak
```
• Alternately, you can give world read and execute access to the file:

```bash
chmod o+rx <backup>.tsbak
```

**Setting permissions for directories**

In addition to setting the proper permissions on the files themselves, TSM also needs permissions for the directory that contains the file, as well as any parent directories. If you are using a non-default location for files that TSM will access, you will need to make sure permissions for the parent directory or directories that contain the file allow read and execute access.

You can address this issue in a couple of ways:

• Change group ownership of the directory to the `tableau` group, and add group read and execute permission to the directory. Doing this makes files in the directory more available to the `tableau` user.

```bash
chgrp tableau <directory-name>
chmod g+rx <directory-name>
```

• Alternatively, you can add world read and execute permission to the directory. This makes files in the directory more available to all users on the system. This approach may require additional steps to ensure security of other files in the directory. For example, you may want to make sure other files in the directory are not world readable so other users cannot read them.

```bash
chmod o+rx <directory-name>
```

**Hint:** You can use `namei -mo` command to list an entire permissions tree. This can make it easier to see what directories need to have permissions adjusted to allow access by the `tableau` group. You can find more information on the internet.
Copying Files in a Distributed Deployment

If you are running Tableau Server for Linux in a distributed deployment, then you may need to copy files manually across the nodes as part of your initial configuration.

The following optional scenarios require manual file copy to other nodes:

- **OpenID Connect:** If you are not hosting a publicly available metadata configuration file, then you must host the file on each Tableau Server that services client authentication requests.

- **Mutual SSL:** The certificate authority public certificate must be on each Tableau Server that services client authentication requests.

- **SAML:** Multiple files (certificate, key, idp metadata) must be distributed to each Tableau Server that services client authentication requests.

- **Kerberos:** Keytab files must be distributed to each Tableau Server

- **Portal customization:** Images for Tableau Server portal customization.

This topic provides relevant information about distributing files for the scenarios listed above.

**Note:** Copying the server secrets keystore file is not covered in this topic. This operation must be done using tsm settings along with an encryption process. For more information, see Manage Server Secrets.

**Before you begin**

In some cases, the files you need to copy may contain sensitive information or security tokens that are not encrypted. We recommend using a secure process to copy files in those cases. There are a number of methods you can use to transfer files securely in a Linux environment. Use a process that is compatible with your security policy and particular Linux implementation.
By default, permissions for the directories that are discussed in this topic will inherit the correct directory permissions so that Tableau processes can access and read the files. For more information, see Files and Permissions in TSM.

OpenID Connect

Clients that authenticate with OpenID Connect (OIDC) require access to the configuration file for the IdP your organization is using. Most public IdPs host a copy of the configuration file on a public website. However, if you are using a private IdP or your IdP does not host a public configuration file, then you must host the file on Tableau Server.

<table>
<thead>
<tr>
<th>File naming</th>
<th>The name of the file is specified by the administrator during the initial node configuration. See Configure OpenID Connect.</th>
</tr>
</thead>
<tbody>
<tr>
<td>File location</td>
<td>The location for the file is specified by the administrator during the initial node configuration. See Configure OpenID Connect.</td>
</tr>
<tr>
<td>Requires secure trans-fer process?</td>
<td>No. Information shared in the OIDC IdP configuration file is intended to be public.</td>
</tr>
<tr>
<td>Process requirements</td>
<td>If you are using OIDC authentication then you must install the configuration file on nodes in your cluster that are running the Application Server (vizportal.exe) process.</td>
</tr>
</tbody>
</table>

Mutual SSL

The certificate and key files must be on each Tableau Server that services client authentication requests.

| File naming | Certificate file and key file: See Configure SSL for External HTTP Traffic to and from Tableau Server (Linux). These files are named according to the process that generates them. |
### Default file location

The location for the files are specified by the administrator during the initial node configuration.

### Requires secure transfer process?

Yes. The key file should be moved securely.

### Process requirements

If you are using SSL mutual authentication then you must install the configuration file on nodes in your cluster that are running the Application Server (vizportal.exe) process.

---

**SAML**

There are multiple files that need to be distributed to other nodes in a cluster.

<table>
<thead>
<tr>
<th>File naming</th>
<th>The following files need to be distributed:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Certificate file and key file: See SAML Requirements These files are named according to the process that generates them.</td>
</tr>
<tr>
<td></td>
<td>• IdP config file: default name is <code>samlmetadata.xml</code>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default file location</th>
<th>The location for the files are specified by the administrator during the initial node configuration.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The documentation example suggests <code>/var/opt/tableau/tableau_server/data/saml</code>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Requires secure transfer process?</th>
<th>Yes. The key file should be moved securely.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Process requirements</th>
<th>If you are using SAML authentication then you must install the configuration file on nodes in your cluster that are running the Application Server (vizportal.exe) process.</th>
</tr>
</thead>
</table>

### Kerberos

If you are using LDAP and authenticating users with Kerberos (using GSAPI bind), then the keytab file for the Tableau Server service must be moved to other nodes in the cluster.
File naming

The name of the file is specified by the administrator during the initial node configuration.

Default file location

The location for the files are specified by the administrator during the initial node configuration.

Requires secure transfer process?

Yes.

Process requirements

The Kerberos keytab file for Tableau Server services is used for user authentication and may also be used for data source access. Therefore, you should copy the keytab file to all nodes of the cluster, as the keytab file may be used by most processes.

Portal customization

You can customize the Tableau Server portal with your own branding and images. See tsm customize.

If you are running the Gateway process on more than one node, then you will need to distribute the image file to each of those nodes.

File naming

The name of the file is specified by the administrator during the initial node configuration.

Default file location

/var/opt/tableau/tableau_server-/data/tabsvc/httpd/images/custom

Before copying the logo file, you must create the images and custom directories in the path first.

Requires secure transfer process?

No.

Process requirements

If you are customizing your portal, then you must install the image file on nodes in your cluster that are running the Gateway process.
Server Crash Reporter

The Tableau Server administrator can enable an option to allow logs and related files to be sent to Tableau when the server has an issue that results in a crash. These files are used by Tableau to identify and address issues that cause crashes. By default this option is disabled, and it should only be enabled in organizations that are not subject to regulations related to data privacy.

**Important:** Do not enable crash reporting if your data is subject to privacy regulations.

If Tableau Server has a problem that results in a crash, log files and dump files are generated. If the crash data upload feature is enabled, these files are automatically gathered and zipped into an encrypted package that is sent in the background, at the scheduled time. The encrypted package is sent in small pieces to limit impact to network performance. Only one crash report is packaged and uploaded at a time (a new crash report is not packaged until the previous package has been uploaded) and is sent in a "first in, first out" order. You can schedule the sending for a low-use window to further reduce any impact to your users.

The encrypted package is made up crash dump files and logs that include the following:

- Crash/core dump files
- Error log files related to the crash
- Manifest files related to the crash

The files can contain data that includes:

- Machine-specific information (for example: hardware, operating system, domain).
- A snapshot of the contents of memory at the time of the crash, including application activity details like information about data connections, actions taken by the user in Tableau, and data being worked on in Tableau.
- Tableau information including customer-identifiable information.
See Also:

- Configure Server Crash Reporter

Configure Server Crash Reporter

Server crash reporting is disabled by default. This topic provides a reference of all the keys that you can set to enable and configure crash reporting on Tableau Server. Crash reports are encrypted and sent to Tableau. See Server Crash Reporter for more information.

Use the configuration file template below to create a json file. After you have filled in the options with the appropriate values, pass the json file and apply settings with the following commands:

```bash
$ tsm settings import -f /path/to/file.json
$ tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Because the configuration file is using `configKey` class, the values that you pass are not validated by TSM as they are when you use `configEntities` class. You can verify and set individual options by using the `tsm configuration` commands.

Crash reporter settings

The crash reporter settings in the template below specify a range of options for configuring Tableau Server to send crash reports to Tableau.

**Important:** Do not enable crash reporting if your data is subject to privacy regulations.
Configuration template

Use this template to configure the gateway settings.

For more explanation about configuration files, entities, and keys see Configuration File Example.

```
{
  "configKeys": {
    "servercrashupload.enabled": "true",
    "servercrashupload.scheduled_time": "1:00:00 UTC",
    "servercrashupload.proxy_server_host": "",
    "servercrashupload.proxy_server_port": "",
    "servercrashupload.proxy_server_username": "",
    "servercrashupload.proxy_server_password": "",
    "servercrashupload.preserve_upload_packages": "false",
    "servercrashupload.delete_completed_dumps": "false"
  }
}
```

Configuration file reference

This table includes keys that you can set to configure crash reporting.

servercrashupload.enabled

**Default:** false.

**Set to true to enable crash reporting.**

servercrashupload.scheduled_time

**Default:** 1:00:00 UTC

Specifies the scheduled time that crash uploads will begin. Enter time of day in 24 hour format.

servercrashupload.proxy_server_host
If your organization uses a proxy server to communicate with the internet, specify the host name.

servercrashupload.proxy_server_port

If your organization uses a proxy server to communicate with the internet, specify the port number.

servercrashupload.proxy_server_username

If your proxy server requires authentication, specify the user name with this key.

servercrashupload.proxy_server_password

If your proxy server requires authentication, specify the password with this key.

servercrashupload.preserve_upload_packages

**Default:** false.

To save all packages that are created for a crash reporting, set this key to true. By default, packages are saved to `/var/opt/tableau/tableau_server-/data/tabsvc/clustercontroller/tabcrashreporter`.

servercrashupload.delete_completed_dumps

**Default:** false.

To delete all dumps after they are sent, set this key to true.

**Automated Installation of Tableau Server**

Tableau provides an automated-installer script to automate an install of Tableau Server. The script is community supported. You can download the script and use it as written, or modify it for your specific needs.
Benefits of using the automated installer

- With a single command, you can install, configure and get to a working instance of Tableau Server.
- The command can be run without user input making it suitable for automation.
- The configuration can be set once and used for all your installations, making this a repeatable process.

When not to use the automated installer:

- If you are installing for the first time, it is recommended that you test the install manually before automating the process. Any issues that block installation are easier to resolve interactively, and once you have resolved these issues, you can use the automated installer.
- If you are testing or trying new configuration parameters such as authentication methods, it is recommended that you run the install manually first. TSM validates configuration entities and rejects configuration parameters that are not valid. Once you have the correct parameters identified, then the automated installer can be used.
- If you are unable to or do not want to enter passwords into the secrets file, using the automated installer might not be an option for you.

Before you begin

Review the Confirm Requirements topic to make sure you have installed Linux on a machine that meets the operating system requirements and the minimum hardware requirements for Tableau Server.

Note: If you are doing production installation of Tableau Server, review the minimum hardware recommendations. The recommendations represent the minimum hardware configuration you should use for a production installation of Tableau Server.
To do an automated install, you have to use the automated installer package which uses the Tableau Server install package as an input. We recommend that you download both these packages before you begin as follows:

1. Download both the automated installer package and the Tableau Server installer package.

   1. Download the automated installer package from GitHub for the distribution you are using. The automated installer packages can be found in the packages sub directory.

   **Note:** The version of automated installer package you use must match the version of the Tableau Server installer package. For example, use 10.5.0 version of the automated installer package with 10.5.0 version of the Tableau Server installer package.

2. Download the Tableau Server installer package from the Tableau Server 10.5 Downloads and Release Notes page. The one you choose depends on which Linux distribution you are using. For example, for RHEL like systems, tableau-server-<version>.x86_64.rpm.

3. Download the config.json, reg_templ.json, and the secrets templates.

2. Copy the packages and templates to a location on or accessible from the computer where you are going to install Tableau Server.

**How to use the automated installer**

First a brief explanation of what the automated installer does:

The automated installer installs the Tableau Server installer package, creates the directories, sets the right permissions required to run Tableau Server, and starts the Tableau Services Manager (TSM) setup. Once the TSM setup is completed, the automated installer then runs tsm commands to install, configure, and start Tableau Server. During installation,
by default, it will activate a trial license. If you have an actual product key you can provide the key on the command line or activate the key after you run the script. Most of the command line options in the automated installer are the same as the options used by tsm initialize command.

To run the automated installer without user input you must provide the following required command line options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-s &lt;secrets-file&gt;</td>
<td>The name of the secrets file. The secrets file should have the user names and passwords for TSM administrator and the Tableau Server administrator accounts.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Providing the password in the secrets file is optional. However, if passwords are not found in the secrets file, you will be prompted to provide them during installation.</td>
</tr>
<tr>
<td></td>
<td>The automated installer package includes the template for the secrets file.</td>
</tr>
<tr>
<td>-f &lt;config-file&gt;</td>
<td>The name of the configuration JSON file. The automated installer package includes the template for the configuration file.</td>
</tr>
<tr>
<td>-r &lt;registration-file&gt;</td>
<td>The name of the registration file. The automated installer package includes the template for the registration file.</td>
</tr>
<tr>
<td>--accepteula</td>
<td>Indicates that you have accepted the End User License Agreement.</td>
</tr>
<tr>
<td>&lt;package-file&gt;</td>
<td>The rpm or deb Tableau Server installer.</td>
</tr>
</tbody>
</table>

**Use the -h option to see a full list of all the command line options.**

Use the following steps to do an unattended install:
Step 1: Install the automated installer

1. Log onto the computer as a user with sudo access.

2. Use the package manager to install the script package:

   • ON RHEL-like distributions, including CentOS, run the following command:

     sudo yum install /path/to/tableau-server-automated-installer-<version>.noarch.rpm

   • On Ubuntu, run the following command:

     sudo apt-get update
     sudo apt-get -y install gdebi-core
     sudo gdebi -n /path/to/tableau-server-automated-installer-<version>.deb

   The automated installer package download includes templates for the configuration file (config.json), registration file (reg_templ.json) and the secrets (secrets) file that you can use to modify for your requirements as described in the next step. The installer script, and the templates for the initial node configuration, Tableau Server registration, and secrets file are installed to:

     /opt/tableau/tableau_server_automated_installer/automated-installer.<version>

Step 2: Create files to provide additional configuration information required to run the automated install

Since the automated installer is meant to run without user interaction, you must provide the following additional information:
1. Run the following command to copy the templates, `config.json`, `reg_templ.json`, and `secrets`, to a another directory like your home directory. We don't recommend that you edit the template files directly:

   ```bash
cp /opt/tableau/tableau_server_automated_installer-
   /automated-installer.<version>/{config.json,reg_tem-
   pl.json,secrets} ~
   ```

2. Edit the configuration template, `config.json` to provide the initial node configuration settings. You must provide identity store settings for the Tableau Server Computer. Depending on your network requirements, you may need to also provide the gateway settings. Caching option is set to cache and reuse data for as long as possible. Sample workbooks are installed by default. The template includes the minimum required information, so the template is a starting point. For more information on configuration settings, see Configure Initial Node Settings.

3. Edit the registration file `reg_templ.json` to provide the information needed to register Tableau Server. For more information, see Activate and Register Tableau Server.

4. Edit the secrets file using the template `secrets` with the user name and password for the TSM administrator and Tableau Server administrator accounts.

   - The TSM administrator account should be the same user as the sudo admin running the script. If you do not want to specify the password in the secrets file, you can leave it blank, and you will be prompted to provide the password during install.

   - The Tableau Server administrator account is the initial account that is created by the installer and is used to administer Tableau Server.

Step 3: Run the automated install

1. Log onto the computer as a user with sudo access.

   Run the following command to run the script on RHEL-like distributions:
sudo /opt/tableau/tableau_server_automated_installer-automated-installer.<version>/automated-installer -s /path/to/secrets -f /path/to/config.json -r /path/to/regtempl.json --accepteula /path/to/tableau-server-<version>_.x86_64.rpm

or this command for Ubuntu:

sudo /opt/tableau/tableau_server_automated_installer-automated-installer.<version>/automated-installer -s /path/to/secrets -f /path/to/config.json -r /path/to/regtempl.json --accepteula /path/to/tableau-server-<version>_amd64.deb

**Important:** You must specify `--accepteula` key to acknowledge and accept the end user license agreement (EULA) in the command that you use the run the script. The EULA is available in the following location: `/opt/tableau/tableau_server/packages/docs.<version>/EULA.rtf`.

**Note:** If you are adding this machine as an additional node to an existing cluster, you must specify the `-b bootstrap` flag and the node configuration file from the initial server. For more information on how to generate the node configuration file, see Generate the node bootstrap file.

---

**Upgrade Tableau Server on Linux**

Tableau Server version 10.5 is the first release of Tableau that runs on Linux. You can upgrade from version 10.5 to version 10.5.x. Upgrades from beta versions of Tableau
Server on Linux are not supported. If you are a Tableau Server on Windows user and want to switch to Tableau Server on Linux, see Migrate Tableau Server from Windows to Linux.


As a best practice you should *always* make a full backup before upgrading any version of Tableau Server. Save this backup file to a location that is not part of your server installation. Creating and saving a backup preserves your pre-upgrade data and configuration and gives you a rollback option in the event of issues during the upgrade. For more information, see Back Up Tableau Server Data.

When upgrading Tableau Server on Linux, you do not need to uninstall the previous version. After upgrading you can uninstall the package for the previous version if you choose to do so. To understand the difference between removing and uninstalling Tableau Server on Linux, see Uninstall Tableau Server.

Upgrading Tableau Server requires a stop and start the server as part of the upgrade process. During this stop/restart Tableau Server is unavailable.

Upgrading will also apply any pending changes. If you have pending changes you do not want applied, you can discard those changes using the `tsm pending-changes discard` command. To see what changes might be pending, use the `tsm pending-changes list` command.

### Upgrading to 10.5.x

Follow these steps on each node in your cluster to upgrade Tableau Server.

1. On each node in your cluster:
   a. Copy the Tableau Server 10.5.x .rpm or .deb package to location accessible from the computer you are upgrading.
If you are upgrading a distributed deployment of Tableau Server, then copy the .rpm or .deb package to each node in the cluster or to a location accessible from each node.

b. Log on as a user with sudo access to the computer you are upgrading.

c. Navigate to the directory where you copied the .rpm or .deb Tableau Server package that you received as part of the Tableau Server pre-release program.

d. Use the package manager to install the Tableau Server package.
   - On RHEL-like distributions, including CentOS, run the following command:
     
     ```
     sudo yum install tableau-server-<version>.x86_64.rpm
     ```
   
   - On Ubuntu, run the following commands:
     
     ```
     sudo gdebi -n tableau-server-<version>_amd64.deb
     ```

2. After you have installed the new package on every node in your cluster, stop Tableau Server:

   ```
   tsm stop
   ```

3. With Tableau Server stopped, run the following command on any one node. Do not run this command on multiple nodes:

   ```
   sudo /opt/tableau/tableau_server-/packages/scripts.<version>/upgrade-tsm --accepteula
   ```

   where `<version>` is the new version you are upgrading to.

4. After the upgrade is completed, exit the terminal session on the initial node and log in again. This ensures that your session will be using the updated TSM version.
5. Start Tableau Server:

```
$ tsm start
```

Migrate Tableau Server from Windows to Linux

Version 10.5 is the first version of Tableau Server available on Linux, so there is no upgrade path from earlier versions of Tableau Server on Linux. Existing customers running Tableau Server on Windows can migrate to Linux by taking a backup of their existing Tableau installation and restoring it to a fresh installation on Linux. This topic describes the steps necessary to do this migration. You cannot upgrade from a beta version of Tableau Server on Linux to the officially released version.

The basic steps to migrate from Tableau Server on Windows to Tableau Server on Linux include:

1. **Step 1: Plan your migration**—Plan for your migration, including gathering all the information you’ll need to be successful. During this step you should familiarize yourself with potential differences between Tableau Server on Windows and Tableau Server on Linux.

2. **Step 2: Create a backup**—Create a backup of Tableau Server on Windows.

3. **Step 3: Install Tableau Server on Linux and restore the Windows backup**—Install a fresh instance of Tableau Server on Linux in a test environment so you can test out the migration, then restore your Windows backup. The restore of your Windows backup will restore the Tableau content (users, projects, sites, workbooks and data sources), but will not restore customizations, so you will need to spend some time configuring Tableau Server on Linux to match the expectations in your organization. If you have a multi-node installation you will need to add nodes and configure them separately.

4. **Step 4: Test Tableau Server on Linux**—Try Tableau Server on Linux to make sure
content is there as you expect, and users are able to perform all the actions they do on Windows. Look specifically at any changes identified as potential differences between Tableau Server on Windows and Tableau Server on Linux. Include key stakeholders in the testing both to leverage their knowledge and to help communicate the upcoming changes.

5. Step 5: Install Tableau Server on Linux in your production environment and restore the Windows backup—Once you’re satisfied that Tableau Server on Linux gives you the functionality you need, install Tableau in your production Linux environment and restore the Windows backup.

Step 1: Plan your migration

A successful migration from Windows to Linux requires some preparation beforehand. You will need to satisfy the following requirements:

- **Identity store**: You can only restore from a backup that has the same type of identity store as the running server. For example, a backup from a server using local authentication can be restored to a Tableau Server initialized with local authentication, but a backup from a server using Active Directory authentication cannot be restored to a server initialized with local authentication.

- **Server administrator**: You must have password for at least one of the user accounts with Server Administrator privileges from the Tableau Server source (Windows) deployment. You must use the same user as the Server Administrator in the migrated deployment to Linux.

- **Clean installation of Tableau Server for Linux**: When you install Tableau Server on Linux later in this topic, be sure to use the same identity store type as you are using on Windows, and do not create users or content. When you restore the Windows backup file to the Linux deployment of Tableau Server, all user data and content will be replaced from the Windows backup file.

- **Differences between Windows and Linux**: Review the differences between
Tableau Server on Windows and on Linux so you are aware of them and can investigate in your test environment.

• If you are migrating from a version of Server on Windows prior to 10.5, approach the migration like an upgrade and familiarize yourself with any changes between your existing version and 10.5 by reading What's Changed in the Server on Windows help.

• Any custom fonts you use may need to be installed on your Linux computer, and may render differently there than on Windows.

• Connection options for Linux are a subset of those available for Tableau Server on Windows. Review the connection types that are available and make sure the ones you need are supported.

• TSM replaces tabadmin. Understand the TSM commands and how they compare to older tabadmin commands. For a list of tabadmin commands and their TSM equivalents, see Migrate from Tabadmin to the TSM CLI.

• Release notes for Tableau Server on Linux: Review information about the release, including any known issues. See Introducing Tableau Server on Linux for more information.

Step 2: Create a backup

1. Log on to the computer running Tableau Server on Windows.

2. Open a command prompt as an administrator.

3. Navigate to the bin directory. For example:

   ```
   cd "C:\Program Files\Tableau\Tableau Server\10.4\bin"
   ```

4. Run the following command:

   ```
   tabadmin backup tabserver -v -d
   ```
Include the \texttt{-v} flag to verify the integrity of the backup. Include the \texttt{-d} flag to include the date in the file name. In the example, \texttt{tabserver} will be used as the base file name of the resulting backup file. The date of the backup will be appended to the file name, for example, \texttt{tabserver-2017-12-20.tsbak}.

For more information, see \textit{Back Up Tableau Server Data} in the Tableau Server on Windows help.

**Step 3: Install Tableau Server on Linux and restore the Windows backup**

In a test environment, install Tableau Server on Linux:

- Install Tableau Server for Linux according to the procedure, Install and Configure Tableau Server. Use the same identity store as on your Windows deployment, and do not create users or content.

Restore the Tableau Server on Windows backup:

1. Copy the Windows backup file to the computer running Tableau Server on Linux. By default the restore process will look for the file in this location:

   \texttt{/var/opt/tableau/tableau_server/data/tabsvc/files/backups/}

   You can change the location. For more information, see \texttt{tsm File Paths}.

2. Run the following command to stop Tableau Server:

   \texttt{tsm stop}

3. Run the following command to restore from the backup file:

   \texttt{tsm maintenance restore -f <filename.tsbak>}

   \textit{Where} \texttt{<filename.tsbak>} \textit{is the name of your backup file, for example, tabserver-2017-10-20.tsbak.}

   For more information, see \texttt{tsm maintenance restore}
4. Run the following command to start Tableau Server:

```
  tsm start
```

After restoring your Tableau content, you may need to configure Tableau Server. For example, if you are migrating from a multi-node installation, you will need to add and configure the additional nodes. You will also need to configure any customizations you made on Windows, including copying over image or logo files if applicable.

**Step 4: Test Tableau Server on Linux**

Once you’ve installed Tableau Server on Linux and restored your Windows backup, you can test the new version of Tableau Server. Because the restore process only restores content to Tableau, you may need to update configuration, topology (adding additional nodes for example), and any customizations you have on your Windows installation.

You’ll want to be familiar with the known differences between Windows and Linux, as well as any potential problem areas. Test basic functionality, along with any special aspects of server that your organization relies on. For example, there may be key data sources that your organization uses with Tableau. Test these to make sure you’re seeing what you expect.

These are some areas of testing to consider:

- **User access.** Confirm that Tableau Server users, including administrators, can sign in. Test your normal user sign in process. Have some of your users participate in the testing to make sure they are able to sign in as expected, and that they can get to the same content that they have access to in your production environment.

- **Viewing built-in administrative views.** With this release of Tableau Server on Linux, you must install PostgreSQL drivers manually, and the administrative views depend on these. Confirm that you have installed the drivers necessary by accessing the built-in administrative views. For more information, see Administrative Views.

- **Data source availability.** Tableau Server on Linux supports a subset of the data
sources on Tableau Server on Windows. You need to confirm that the data sources used by your organization are supported on Linux, and install any drivers required. For details on which data sources are supported, see the Tableau Server tech specs. For information about installing drivers, see Install Drivers on Linux.

- **Access to file-based data sources on shared drives.** Data sources such as Excel files on network drives will require special actions in order to be accessible from Linux. You or your IT department will need to mount the drives and update any workbooks using these data sources.

- **Viewing published workbooks.** Have users who are familiar with the content try to view published workbooks to make sure they appear as expected. Test views embedded in web pages (for example, in SharePoint pages). Be especially aware of how fonts may differ between Windows and Linux, and some custom fonts may need to be added to your Linux computer, or replaced with other fonts if they are not available on Linux. Dashboard layouts may appear different as well, due to differences in fonts.

- **Server processes.** Sign in to Tableau Server as a server administrator, and then open the Server Status page to confirm that all services and processes are running as expected. Be aware of intentional changes due to changes in version 10.5. For more information, see The New Data Engine in Tableau Server.

- **Publishing workbooks and data sources.** Have users publish workbooks and data sources from Tableau Desktop to make sure this goes as you expect. You may need to install drivers to support the data source connections you are using. See Install Drivers on Linux.

- **Subscriptions and extract refreshes.** Manually run some extract refreshes to confirm that they complete successfully. Run some key scheduled extract refreshes to confirm that they complete as expected.

- **Permissions.** Confirm that permissions are still set as expected for users and content.
- Command-line utilities and APIs. If applicable, test the command line utilities (tsm and tabcmd) and programmatic access via APIs. With Tableau Server on Linux, tabadmin is replaced by TSM. For more information, see Migrate from Tabadmin to the TSM CLI.

Step 5: Install Tableau Server on Linux in your production environment and restore the Windows backup

When you have completed testing and have identified those areas that require additional changes on your part, or communication to your users, you are ready to install Tableau Server on Linux in your production environment and restore the Windows backup. To do this, follow the same steps described above.

**Important:** You cannot take a backup from Tableau Server on Linux and restore that backup to Tableau Server on Windows. This means that once you migrate to Linux, you cannot easily switch back to Tableau Server on Windows. Keep this in mind as you test your migration, and consider keeping your installation on Windows until you are certain that you no longer need it.

Migrate from Tabadmin to the TSM CLI

The Tableau Services Manager (TSM) command-line interface (CLI) replaces the tabadmin CLI in Tableau Server on Linux. This page maps tabadmin commands to TSM CLI commands to help you to migrate to the TSM CLI. To learn more about the TSM CLI, see tsm Command Line Reference.

Tabadmin commands with a corresponding TSM CLI command

The following table shows which tabadmin commands correspond to commands available in the TSM CLI.

<p>| Command | Tabadmin Command(s) | Comparable TSM CLI Command |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activate a license</td>
<td><code>tabadmin activate --activate</code></td>
<td>tsm licenses activate</td>
</tr>
<tr>
<td>Deactivate licenses</td>
<td><code>tabadmin activate --return</code></td>
<td>tsm licenses deactivate</td>
</tr>
<tr>
<td>Activate a trial license</td>
<td><code>tabadmin activate --trial</code></td>
<td>tsm licenses activate --trial</td>
</tr>
<tr>
<td>Specify whether Tableau Server starts at system start-up time</td>
<td><code>tabadmin auto-start</code></td>
<td>tsm configuration set -k service.init.state</td>
</tr>
<tr>
<td>Create a backup of the data managed by Tableau Server</td>
<td><code>tabadmin backup</code></td>
<td>tsm maintenance backup</td>
</tr>
<tr>
<td>Clean up temporary files and old log files</td>
<td><code>tabadmin cleanup</code></td>
<td>tsm maintenance cleanup</td>
</tr>
<tr>
<td>Update the server configuration with any changes you've made</td>
<td><code>tabadmin configure</code></td>
<td>tsm pending-changes apply</td>
</tr>
<tr>
<td>Customize</td>
<td><code>tabadmin customize</code></td>
<td>tsm customize</td>
</tr>
</tbody>
</table>

Note: Added in version 10.5.1
<p>| the server name and logos | tomize |
| Enable access to the repository | <code>tabadmin dbpass</code> | <code>tsm data-access repository-access enable</code> |
| Disable access to the repository | <code>tabadmin dbpass --disable</code> | <code>tsm data-access repository-access disable</code> |
| Set a file store instance to read-only mode | <code>tabadmin decommission</code> | <code>tsm topology filestore decommission</code> |
| Delete one or more Web Data Connectors (WDCs) from Tableau Server | <code>tabadmin delete_webdataconnector</code> | <code>tsm configuration set -k web-dataconnector.whitelist.fixed</code> |
|  |  | To learn more, see Web Data Connectors in Tableau Server on Linux. |
| Add a Web Data Connector (WDC) to Tableau Server | <code>tabadmin import_webdataconnector</code> and <code>tabadmin whitelist_webdataconnector</code> | <code>tsm configuration set -k web-dataconnector.whitelist.fixed</code> |
|  |  | TSM does not support importing WDCs, instead it lets you add WDCs to a safe list (or &quot;whitelist&quot;). To learn more, see Web Data Connectors in Tableau Server on Linux. |</p>
<table>
<thead>
<tr>
<th>Operation Description</th>
<th>Command</th>
<th>Command Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>List Web Data Connectors (WDCs) used by Tableau Server</td>
<td><code>tabadmin list_web-dataconnectors</code></td>
<td><code>tsm configuration get -k web-dataconnector.whitelist.fixed</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>To learn more, see Web Data Connectors in Tableau Server on Linux.</td>
</tr>
<tr>
<td>Export a site from Tableau Server</td>
<td><code>tabadmin export-site</code></td>
<td><code>tsm sites export</code></td>
</tr>
<tr>
<td>Initiate a repository failover</td>
<td><code>tabadmin failoverrepository</code></td>
<td><code>tsm topology failover-repository</code></td>
</tr>
<tr>
<td>Get a configuration option</td>
<td><code>tabadmin get</code></td>
<td><code>tsm configuration get</code></td>
</tr>
<tr>
<td>Get the OpenID redirect URL</td>
<td><code>tabadmin get_openid_redirect_url</code></td>
<td><code>tsm authentication openid get-redirect-url</code></td>
</tr>
<tr>
<td>Import site .csv files into Tableau Server</td>
<td><code>tabadmin import-site</code></td>
<td><code>tsm sites import</code></td>
</tr>
<tr>
<td>Import a site into Tableau Server using .csv files</td>
<td><code>tabadmin import-site_verified</code></td>
<td><code>tsm sites import-verified</code></td>
</tr>
<tr>
<td>Display license</td>
<td><code>tabadmin licenses</code></td>
<td><code>tsm licenses list</code></td>
</tr>
<tr>
<td>Task Description</td>
<td>Command 1</td>
<td>Command 2</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>--------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>Information for Tableau Server</td>
<td>tabadmin recom-mission</td>
<td>tsm topology filestore recom-mission</td>
</tr>
<tr>
<td>Move a file store from read-only mode to an active read/write state</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regenerate internal security tokens</td>
<td>tabadmin regenerate_internal_tokens</td>
<td>tsm security regenerate-internal-tokens</td>
</tr>
<tr>
<td>Register Tableau Server</td>
<td>tabadmin register</td>
<td>tsm register</td>
</tr>
<tr>
<td>Rebuild the search index for Tableau Server</td>
<td>tabadmin reindex</td>
<td>tsm maintenance reindex-search</td>
</tr>
<tr>
<td>Stop and restart all Tableau Server processes</td>
<td>tabadmin restart</td>
<td>tsm restart</td>
</tr>
<tr>
<td>Restore from a</td>
<td>tabadmin restore</td>
<td>tsm maintenance restore</td>
</tr>
<tr>
<td>Tableau Server backup file</td>
<td>[tabadmin set]</td>
<td>[tsm configuration set]</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Set a configuration option</td>
<td>[tabadmin sitestate]</td>
<td>[tsm sites unlock]</td>
</tr>
<tr>
<td>Activate or suspend a site</td>
<td>[tabadmin start]</td>
<td>[tsm start]</td>
</tr>
<tr>
<td>Start all Tableau Server processes</td>
<td>[tabadmin status]</td>
<td>[tsm status]</td>
</tr>
<tr>
<td>Get the status of Tableau Server and server processes</td>
<td>[tabadmin stop]</td>
<td>[tsm stop]</td>
</tr>
<tr>
<td>Stop all Tableau Server processes</td>
<td>[tabadmin ziplogs]</td>
<td>[tsm maintenance ziplogs]</td>
</tr>
<tr>
<td>Create an archive (.zip) file with Tableau Server log files</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tabadmin commands with no corresponding TSM CLI command

The following table lists the tabadmin commands for which a comparable TSM CLI command is not available.

<table>
<thead>
<tr>
<th>Command Description</th>
<th>Tabadmin Command</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add or remove a user from the system administrator group</td>
<td>tabadmin administrator</td>
<td>You can use the Tableau Server REST API Add User to Group and Remove User from Group methods to add or remove a user from the system administrator group.</td>
</tr>
<tr>
<td>Create a new key to encrypt sensitive information stored in the repository</td>
<td>tabadmin assetkeys</td>
<td></td>
</tr>
<tr>
<td>Clear the server cache</td>
<td>tabadmin clearcache</td>
<td></td>
</tr>
<tr>
<td>Identify a second server node for backup</td>
<td>tabadmin fail-overprimary</td>
<td>TSM does not have primary nodes, so a TSM equivalent to this command is not needed.</td>
</tr>
<tr>
<td>Manage credentials for delegated data access on Tableau Server</td>
<td>tabadmin manage_global_credentials</td>
<td>We recommend that you use Kerberos delegation to Apache Impala for global credential management. To learn more, see Kerberos and Enable Kerberos Delegation for Hive/Impala in the Tableau Community.</td>
</tr>
<tr>
<td>Reset the password for a</td>
<td>tabadmin passwd</td>
<td>If your server uses local authentication, you can use the Tableau authentication methods to reset the password.</td>
</tr>
<tr>
<td>Tableau Server account</td>
<td>Server REST API Update User method to reset the password for a user account.</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Reset the Tableau Server administrator account</td>
<td>tabadmin reset</td>
<td></td>
</tr>
<tr>
<td>Reset binding between Tableau Server user ID and Open ID Connect identity provider</td>
<td>tabadmin reset_openid_sub</td>
<td></td>
</tr>
<tr>
<td>Determine whether your environment meets the minimum requirements to run Tableau Server</td>
<td>tabadmin validate</td>
<td></td>
</tr>
<tr>
<td>Verify that a backup of the Tableau Server repository will restore successfully</td>
<td>tabadmin verify_database</td>
<td></td>
</tr>
<tr>
<td>The tsm maintenance backup command automatically verifies that a backup will restore correctly unless you use the --skip-verification parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prepare VizQL processes for fast load times after a Tableau Server restart</td>
<td>tabadmin warmup</td>
<td></td>
</tr>
<tr>
<td>The tabadmin warmup command is no longer necessary, as Tableau Server is now optimized to automatically provide fast load</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Test Workbook Performance

In most cases, you can expect to see similar or better performance after upgrading your Tableau Server from versions 10.4 or earlier to 10.5 or later. However, for workbooks that use complex calculations, you may see performance issues after you upgrade. This article describes how to test performance and troubleshooting these performance issues before you upgrade your production Tableau Server from versions 10.4 and earlier to 10.5 or later.

If you have already upgraded your production Tableau Server to 10.5 or later, and you are seeing performance issues for workbooks using extracts and have calculations, see Troubleshoot Performance of Workbooks with Calculations.

Since Linux was first released in 10.5, this only applies when you are migrating from Tableau Server versions 10.4 or earlier on Windows to Linux 10.5 or later.

To make sure your workbook performance is same or better after your upgrade to Tableau 10.5 or later, we strongly recommend that you set up a test environment and do performance testing before you upgrade your production Tableau Server.

Use the following sequence of steps for testing workbook performance:

1. Setup your test environment
2. Capture performance metrics and analyze
3. Troubleshoot performance issues
4. Capture the conclusions and results
Setup your test environment

1. Create a test environment that mirrors your production environment as closely as possible.

   Follow the steps described in Migrate Tableau Server from Windows to Linux.

2. Disable any automated or scheduled extract refresh schedules.

Capture performance metrics and analyze

1. Run Tabjolt or any other tool of your choice to capture performance metrics on your current version of Tableau Server to set a baseline. Use workbooks that have calculations and use extracts as the data source for testing. For more information about how install and use Tabjolt, see the Tabjolt Installation Guide.

2. Perform a full backup to capture the metrics from the first step.

3. Upgrade your test environment to 10.5.3 or later. We are recommending 10.5.3 as the minimum version since the settings you need to enable in order to resolve performance issues later on are only available in 10.5.3 or later versions of Tableau.

4. **Checkpoint:** Run Tabjolt to compare workbook performance before and after upgrade.

   If you see performance degradation after upgrade at this step, it may be due to other Tableau Server issues and might require further investigations. For more information, see Record and Analyze Workbook Performance.

   If you see no difference in performance before and after upgrade, continue to the next step.

5. Select a subset of workbooks and run a full refresh of the extracts. Doing a full refresh will upgrade the extracts from .tde to .hyper.

6. **Checkpoint:** Run the tool again to compare the performance of your workbooks...
between step 4 and step 6 which is essentially before and after doing extract
refreshes. Specifically, check for workbooks with slower response times or failure to
load.

Troubleshoot performance issues

If you find that you are unable to test due to performance issues, or you find that VizQL
server is consuming all or most of the available memory on the machine while viewing them,
follow steps 1 through 3.

For workbooks that load successfully, but have slower response after upgrade, follow steps
4 and 5.

1. Enable the following setting that checks the number of query nodes required for
   Tableau to generate the workbook and displays an error when the set limit is
   exceeded. This helps prevent VizQL process from consuming the memory resources
   on the machine that might result in critical Server issues:

   Use TSM CLI to run the following commands:

   tsm configuration set -k native_api.node_limit_checker_pre_rewrite_disable -v false

   tsm pending-changes apply

   **Note:** This setting is only available in Tableau Server 10.5.3 or later. While
   enabling this setting improves the reliability and stability of Tableau Server, you
   may see errors on a very small percentage of workbooks with when you enable
   this setting. For example, the Tableau team saw errors on 0.01% of the work-
   books with this setting enabled on Tableau Online.

2. Test the workbooks by viewing or interacting with them. You should see an error mes-
   sage informing you about exceeding the node limits on workbooks that have
performance issues - The error message might say something like: "**Logical Query tree has 2348182 nodes, maximum number allowed is 1000000....**"

3. Use Tableau Desktop to troubleshoot your workbook performance. Run this command to enable the node limit check in your Tableau Desktop:

   ```
   tableau.exe -DNodeLimitCheckerPreRewriteDisable=false
   ```

   **Note:** This setting is only available in Tableau Desktop 10.5.3 or later.

   In Tableau Desktop, download and open the workbooks that displayed the error message about high node limits in step 2. Try optimizing the performance of workbooks using methods described in the [Best Practices for Creating Calculations in Tableau](#) topic. If successful, then publish them to Tableau Server. If you are still experiencing performance issues, continue to the next step.

4. For workbooks that load successfully, but the overall response times is slower after upgrade, try using the **Compute Calculations Now** option. If you have a large number of workbooks that fall under this category, consider enabling the server wide setting to retain materialized calculations for all extracts when they are upgraded from .tde to .hyper:

   Use TSM CLI to run the following commands:

   ```
   tsm configuration set -k native_api.preserve_calculations_on_hyper_refresh_conversion -v true
   tsm pending-changes apply
   ```

   **Considerations for turning on this setting:**

   Turn this setting on only if majority of your workbooks require calculations to be materialized. There are resource and time costs associated with this setting, as described below:
• When this setting is set to true, the extract file size will increase, affecting the overall disk space.

• Extracts with materialized calculations take a longer time to refresh than extracts without materialized calculations.

This setting only affects extracts that are not yet upgraded to .hyper. Extracts already upgraded to .hyper before turning on this setting will not have the calculations materialized. This setting will also not affect workbooks and extracts published from Tableau Desktop, or any new extract refreshes published to Tableau Server that do not use the **Compute Calculations Now** option.

**Note:** This setting is only available in Tableau Server 10.5.3 or later.

5. Enable the extract refresh schedules and monitor the performance of your workbooks to make sure that your workbooks are performing as expected.

**Capture the conclusions and results from your testing**

Make a note of the settings and steps that helped resolve the performance issues to use them to prevent performance degradation when you upgrade your Production Tableau Server.

For example, if during your testing, you found that you needed to enable the `tabadmin set native_api.preserve_calculations_on_hyper_refresh_conversion true` setting, remember to disable your extract refreshes just before you upgrade your production Tableau Server.

**After upgrade, enable the** `tabadmin set native_api.preserve_calculations_on_hyper_refresh_conversion true` **setting, and re-enable your extract refreshes once you have confirmed that you are not experiencing slow response times after upgrade.**
Uninstall Tableau Server

There are two main "uninstall" scenarios that Tableau Server on Linux supports:

- Uninstall a previous Tableau Server package: *After you upgrade* to a new version of Tableau Server and put it into production, you can uninstall any previous package(s) to free up disk space. You do not need to uninstall as part of the upgrade process.

- Completely remove Tableau Server: Completely remove Tableau Server from the computer. If you want to completely remove Tableau Server, we recommend you do not uninstall the package first because uninstalling the package uninstalls the script that is used to completely remove Tableau Server and all its files. See Remove Tableau Server from Your Computer.

Uninstall a Tableau Server package

Use this procedure for the first two scenarios above: to free up disk space after you have upgraded to a newer version of Tableau Server, or to uninstall all packages of Tableau Server on a computer.

Uninstalling the current, running version of Tableau Server does not remove all files. When you uninstall the Tableau Server package for the current instance of Tableau Server the following operations run:

- All files under /opt/tableau/tableau_server are removed. These files are the unmodified installation files.
- Tableau Server services are stopped and disabled
- Service files for all Tableau Server services are persisted

To reinstall after uninstalling the running instance of Tableau Server, reinstall the same package, run `initialize-tsm`, then `update-tsm`.

1. Look at the environment.bash file to confirm which version of Tableau Server is currently in use. At a command prompt, type:
grep TABLEAU_SERVER_DATA_DIR_VERSION /etc/-
  opt/tableau/tableau_server/environment.bash

2. Determine which versions of the Tableau Server package are installed on your computer.

   • On RHEL-like distributions, including CentOS, run the following command:
     
yum list installed tableau-server"*"

   • On Ubuntu, run the following command:
     
     apt list --installed tableau-server"*"

3. Remove the Tableau Server package with your package manager.

   • On RHEL-like distributions, including CentOS, run the following command:
     
sudo yum remove tableau-server-<version>.x86_64

   • On Ubuntu, run the following commands:
     
sudo apt-get purge tableau-server-<version>_amd64

Remove Tableau Server from Your Computer

**Warning:** The steps below *completely remove* Tableau Server on Linux, and delete users and groups created by `initialize-tsm`, all related data, configuration information, and logs. This includes any files in `/tmp` or `/var/tmp` that are owned by users configured in `/etc/opt/tableau/tableau_server/environment.bash` as privileged and unprivileged users (by default, `tsmagent` and `tableau`). Tableau Server licenses are also deactivated, unless you omit the `-l` option when running the command shown below.

If you want to uninstall a particular Tableau Server package to free up disk space (after upgrading, for example), see Uninstall Tableau Server.
As part of the regular installation of Tableau Server, a script is installed that provides you a way to completely remove Tableau and all associated files from your computer. This is something you would only do if you did not care about your Tableau data, configuration, or log files. The obliterate script will not remove any drivers you installed separately, even those you installed to use with Tableau Server.

The tableau-server-obliterate script is intended for when you want to completely remove Tableau Server from your computer. You might want to do this for a couple of different reasons:

- You no longer want Tableau Server installed on the computer. Use the tableau-server-obliterate script to remove Tableau Server completely.

If you have already uninstalled Tableau Server and now you want to remove it, run the copy of the tableau-server-obliterate script located in the /var/tmp directory. If the script is not in that directory, reinstall the Tableau Server package and then run the tableau-server-obliterate script.

- Troubleshooting Tableau Server installation problems—If you run into issues installing Tableau, you may need to use the tableau-server-obliterate script to completely remove Tableau Server from your computer before reinstalling. Doing this will clean up any older settings or states (such as the /etc/opt/tableau/tableau_server/environment.bash file) and allow you to reinstall on a "clean" computer. If you are doing this, you can leave off the -l option to preserve licensing information on the computer. In this case, you will not need to activate your license when you reinstall Tableau Server.

**Note:** The script will remove all log files, so if you run into an issue that you want to troubleshoot, or if Tableau Support requests log files for a support case, you should zip up the log files before running the script. For details on gathering log files, see Work with Log Files.
Running the tableau-server-obliterate script

To completely remove Tableau Server from a computer:

1. Deactivate any active product keys.

   The -l option for the obliterate script removes all licensing files from the computer. It first attempts to deactivate any active licenses, but will remove all licensing information whether or not the deactivation was successful. We recommend you run the tsm licenses deactivate command before running the obliterate script so that the script does not remove licenses that are still active.

   
   tsm licenses deactivate -k <product_key>

2. Run the tableau-server-obliterate script:

   sudo /opt/tableau/tableau_server-
   /packages/scripts.<version>/tableau-server-obliterate -y -y
   -y -l

   If you have a multi-node (distributed) installation of Tableau Server, run the tableau-server-obliterate script on each node in the cluster. You do not need to deactivate licenses on any additional nodes.

Install Tableau Server in a Disconnected (Air-Gapped) Environment

You can install Tableau Server in a disconnected environment that has no outside network access of any kind. Such disconnected environments, commonly referred to as air-gapped, are used when high security is needed to prevent data breaches or to guard against hacking. Air-gapped environments have no internet access, no outside network access, no outside wireless access, etc. The only means of getting software and data into or out of an air-gapped environment is by using removable media such as USB sticks or writeable optical CDs or DVDs.
Installing Tableau Server in an air-gapped environment is an advanced task for IT administrators who are familiar with the security considerations, best practices, and pitfalls of installing software in air-gapped environments.

The following Tableau Server features will be unavailable or will have reduced functionality in an air-gapped environment:

- **Maps** – Tableau Server uses externally hosted map data. Maps are unavailable in an air-gapped environment unless you also install a map server in your air-gapped environment. You can use the following workarounds.

- **Licensing** – Tableau Server needs to connect to the internet in order to activate license keys. However, you can manually activate the license keys.

- **External data** – Any data located outside your air-gapped environment is unavailable.

**Prerequisites**

In order to install Tableau Server in an air-gapped environment, you’ll need the following:

- Trusted computer with limited access to the internet that you can use to download the installation packages and resources required by Tableau Server. A trusted computer has been scanned and cleared of any viruses and malware.

- Trusted removable media that you can use to transfer the downloaded software to your air-gapped environment. Trusted removable media is removable media that is new and previously unused and comes from a reputable or known source. Trusted removable media has been scanned and verified that it does not contain any viruses or malware.

- Air-gapped environment with computers and storage that meet the requirements for installing Tableau Server.
Installing Tableau Server on an Air-Gapped Computer Running Linux

The easiest way to install Tableau Server on a computer in an air-gapped environment is to do so before the computer is placed into the air-gapped environment. If that’s not possible you’ll need to download the required packages to a trusted computer outside the air gap:

1. On a trusted computer outside the air gap with internet access, download the Tableau Server installation package.

2. Extract the list of dependent packages:

   **On Ubuntu:**

   ```bash
dpkg --field <debfile> Depends (where <debfile> is the name of the .deb package you downloaded from Tableau).
   
   Example command:

   ```bash
dpkg --field tableau-server-linux-1.deb Depends
   ```

   **Example output:**

   ```
   ca-certificates, fontconfig, net-tools, bash-completion, ca-certificates-java, freeglut3, libegl1-mesa, libfreetype6, libgssapi-krb5-2, libxcomposite1, libxrender1, libxslt1.1, lsb-core
   ```

   **On RHEL and RHEL-like Linux distributions:**

   ```bash
   yum -q deplist <RPM file> (where <RPM file> is the .rpm package you downloaded from Tableau).
   
   Example command:
   ```
yum -q deplist tableau-server-linux_1.rpm

Example output:

package:  tableau-server-10400.17.0703.1600.x86_64  10400-17.0703.1600
dependency:  /bin/sh
provider:  bash.x86_64  4.2.46-21.el7_3
dependency:  bash-completion
provider:  bash-completion.noarch  1:2.1-6.el7
dependency:  ca-certificates
provider:  ca-certificates.noarch  2017.2.14-70.1.el7_3
dependency:  fontconfig
provider:  fontconfig.x86_64  2.10.95-10.el7
provider:  fontconfig.i686  2.10.95-10.el7
dependency:  freeglut
provider:  freeglut.x86_64  2.8.1-3.el7
provider:  freeglut.i686  2.8.1-3.el7
dependency:  freetype
provider:  freetype.x86_64  2.4.11-12.el7
provider:  freetype.i686  2.4.11-12.el7
dependency:  krb5-libs
provider:  krb5-libs.x86_64  1.14.1-27.el7_3
provider:  krb5-libs.i686  1.14.1-27.el7_3
dependency:  libXcomposite
provider:  libXcomposite.x86_64  0.4.4-4.1.el7
provider:  libXcomposite.i686  0.4.4-4.1.el7
dependency:  libXrender
provider:  libXrender.x86_64  0.9.8-2.1.el7
provider:  libXrender.i686  0.9.8-2.1.el7
dependency:  libxslt
provider:  libxslt.x86_64  1.1.28-5.el7
provider: libxslt.i686 1.1.28-5.el7
dependency: mesa-libEGL
provider: mesa-libEGL.x86_64 11.2.2-2.20160614.el7
provider: mesa-libEGL.i686 11.2.2-2.20160614.el7
dependency: net-tools
provider: net-tools.x86_64 2.0-0.17.20131004git.el7
dependency: redhat-lsb-core
provider: redhat-lsb-core.x86_64 4.1-27.el7.centos.1
provider: redhat-lsb-core.i686 4.1-27.el7.centos.1

3. Download each of the dependent packages:

   **On Ubuntu:**

   apt-get download <package1> <package2>…

   **On RHEL and RHEL-like Linux distributions:**

   yumdownloader <package1> <package2>…

4. Transfer the packages to your removable media.

5. On your air-gapped computer, insert the removable media containing the Tableau Server installation package and dependent packages, and then run the installer.

6. After installation is complete, you can activate the Tableau Server license keys. For more information, see Activating Tableau Server in an Air-Gapped Environment.

**Activating Tableau Server in an Air-Gapped Environment**

Because an air-gapped computer is not connected to the internet, you’ll need to perform the Tableau Server activation process manually.
Step 1. Transcribe data from your air-gapped system into an activation request template.

1. On your Tableau Server in the air-gapped environment, use TSM to obtain the offline activation file. At a command prompt:

   tsm licenses get-offline-activation-file -k <product-key> -o <target-directory>

   The <target-directory> must exist.

2. Copy the offline activation file (offline.tlq) from the target directory to a trusted computer that has internet access.

3. On a trusted computer that is connected to the internet and is capable of sending files via email, right-click and choose Save link as to download the server_firstpass_linux.tlq file. The file is a simple XML file.

4. Open both offline.tlq and server_firstpass_linux.tlq in a text editor and turn on line numbering.

5. Update the following XML elements in server_firstpass_linux.tlq with the values for the same elements in offline.tlq on the disconnected computer. All the Machine / Hash values in the .tlq files are Hex values. The only valid characters are 0 - 9 and A - F. Use all caps for any letters.

   Line 2 - <EntitlementId>

   Line 5 - <ClientVersion>

   Line 5 - <MachineIdentifier> (Optional: If this XML element is present, copy the value. If it's missing you can skip it.)

   Line 11 - <Value>
6. Upload the edited template (offline.tlq) to the Tableau Offline Activation website.

7. The website should say The activation was successful. Please click here to download your activation file.

Download the activation.tlf file and transfer it to your Tableau Server.

Step 2. Upload the offline activation request to Tableau

1. Go to the Tableau Offline Activation website.

2. Complete the instructions and upload your offline.tlq file.

   This creates an activation file, activation.tlf.

3. Download the resulting activation file from Tableau.

Step 3. Initialize or activate your license

1. Copy the activation file (activation.tlf) to a location accessible from your Tableau Server computer.

2. Run the following command:

   
   ```
   tsm licenses activate -f <path-and-activation-file>
   ```

First time license activation

If this is the first Tableau license activated on the computer, you will see this message:
Your license has been initialized. To complete the activation, we need one more exchange. Generate and send to Tableau a second activation request file.

You need to follow the steps below to activate your license. If you are not sure whether you need to repeat the steps, you can run this command:

tsm licenses list

If you see a message like this one you need to repeat the steps above:

No licenses are currently activated.

Subsequent license activation

If this is not the first Tableau license activated on the computer, or if you have completed the steps above two times, you should see the message "Activation successful."

Note: You may have to perform offline activation twice to completely activate your air-gapped Tableau Server. The first offline activation sets up Trusted Storage on the computer if Tableau has never been licensed on it before. The second offline activation propagates the licensing information into Trusted Storage. If you’re upgrading an existing install of Tableau Server, right-click and choose Save link as to download the server_secondpass_linux.tlq file. It will match all the places to put the values from the generated .tlq file.

Update the following XML elements in offline_second_pass.tlq with the values for the same elements in offline.tlq on the disconnected computer. All the Machine / Hash values in the .tlq files are Hex values. The only valid characters are 0 - 9 and A - F. Use all caps for any letters.

Line 2 - <EntitlementId>
Upload the edited template (offline.tlq) to the Tableau Offline Activation website.

The website should say The activation was successful. Please click here to download your activation file.

Download the activation.tlf file and transfer it to your server.

Displaying Maps in an Air-Gapped Environment

In an air-gapped environment, maps in Tableau Server will be unavailable due to the lack of internet access in the air-gapped environment. You can use the following workarounds to display maps in an offline environment:

- **Option 1** – Use the offline background maps
  
  To use the offline background map select **Map > Background maps > Offline**. For more information, see **Select Background Maps**.

  **Note**: Due to size limitations with the offline map, zooming in too deep may result in errors as the images for the more close-up maps are not stored offline.

- **Option 2** – Use a map as a background image
Locate the desired map online and save it as an image file, or copy a map from another location. Set this map image as a static background image in Tableau Desktop. For more information, see Use Background Images in Your Views.

**Note:** Zooming in and out will not work with background images.

- **Option 3** – Use a local GeoServer

For more information, see Use Web Map Service (WMS) Servers.

## Install Tableau Server in the AWS Cloud

Even if you don’t have your own infrastructure and server hardware, you can deploy an enterprise-level Tableau Server installation in the cloud. Building a cloud-based solution has many benefits over an on-premises installation. For example, the overall total cost of ownership for building a Tableau Server solution in the cloud is normally much less than a similar on-premises solution because you don’t have to buy all of the expensive hardware. In addition, the cloud can provide better uptime, reliability, and fault-tolerance, especially if you deploy your solution across different regions and Availability Zones.


You can build and scale Tableau Server in the Amazon Web Services (AWS) cloud using Amazon Elastic Compute Cloud (Amazon EC2) instances. Running Tableau Server on AWS is an excellent choice if you want the flexibility of scaling up and scaling out without having to purchase and maintain an expensive fleet of servers. Tableau Server can take advantage of many native AWS services. For example, you can use the following services in AWS with Tableau Server:

- **Amazon CloudWatch**: Monitors the components of your AWS-based solution.

  **Amazon Route53**: DNS web service for connecting user requests to your AWS
infrastructure.

- **AWS Certificate Manager**: Helps you deploy SSL and TLS certificates for use with AWS.

- **AWS CloudFormation**: Helps you to create and manage related AWS resources in a controlled manner using pre-defined templates.

- **Elastic Load Balancing**: You can use Elastic Load Balancing to spread Tableau requests across multiple Tableau gateways.

In addition, you can store the data you use with Tableau Server using any of the following AWS features. Tableau provides native data **connectors** that enable you to connect to data in the following AWS data sources:

- **Amazon Athena**: Interactive query service that you can use to analyze data in Amazon S3 using SQL.

- **Amazon Aurora**: High-performance MySQL- or PostgreSQL-compatible database.

- **Amazon EMR**: Managed Hadoop framework for processing large amounts of data.

- **Amazon Redshift**: Data warehousing solution for storing and mining large amounts of data.

- **Amazon Relational Database Service (Amazon RDS)**: Relational database solution supports **Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle**, and **Microsoft SQL Server**.

- **Amazon Simple Storage Service (Amazon S3)**: Cloud-based storage solution.

**Tableau Server on AWS deployment options**

You can self-deploy Tableau Server on an Amazon EC2 instance that you provision, deploy Tableau Server using the AWS CloudFormation templates in the Tableau Server on AWS Quick Start, or deploy Tableau Server on AWS using an AWS Marketplace Amazon Machine Image (AMI). The following table shows the differences between each option.
<table>
<thead>
<tr>
<th>Tableau Server</th>
<th>AWS Self-Deployment</th>
<th>AWS Quick Start</th>
<th>AWS Marketplace AMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Production ready</td>
<td>✅</td>
<td>✅</td>
<td>✗</td>
</tr>
<tr>
<td>Upgradable</td>
<td>✅</td>
<td>✅</td>
<td>✗</td>
</tr>
<tr>
<td>Install on Windows</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
<tr>
<td>Install on Linux</td>
<td>✅</td>
<td>✅</td>
<td>✗</td>
</tr>
<tr>
<td>Scale-up</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
<tr>
<td>Scale-out (add additional nodes)</td>
<td>✅</td>
<td>✅</td>
<td>✗</td>
</tr>
<tr>
<td>Active Directory support</td>
<td>✅</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>14-day trial license</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
<tr>
<td>BYOL license</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
<tr>
<td>Subscription license</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
</tbody>
</table>

Each of the deployment options are described in more detail below:

- **Self-Deploy to an Amazon EC2 Instance** – Provides the most flexibility and options to customize Tableau Server to your environment. If you want to quickly launch a system use the Tableau Server on AWS Quick Start or an AWS Marketplace AMI instead.

  Self-deployment of Tableau Server to an Amazon EC2 instance is recommended for development, test, and production environments that start simple but may need to scale-up or scale-out later.
For more information about self-deployment, see Self-Deploy a Single Tableau Server on AWS. For more information about deploying in a distributed environment, see Self-Deploy Tableau Server on AWS in a Distributed Environment.

- **Tableau Server on AWS Quick Start** – Installs Tableau Server on a standalone Amazon EC2 instance or on a cluster of standalone Amazon EC2 instances using an AWS CloudFormation template. This deployment strategy allows you to go live quickly and limit your overall costs.

  The Tableau Server on AWS Quick Start is recommended for development, test, and production environments that start simple but may need to scale-up or scale-out later. You can use your own license (BYOL) or a free 14-day trial version.

  For more information about installing Tableau Server on Linux using the Quick Start, see Tableau Server on Linux in the AWS Cloud.

- **Tableau Server AWS Marketplace AMIs** –Installs Tableau Server on Windows with an AWS Marketplace AMI. You can use a 14-day free trial or after Tableau Server is up-and-running on the Amazon EC2 instance, you can apply the Tableau Core or Interactor license you’ve purchased to the server. The Tableau Server AWS Marketplace AMIs are recommended for quick testing and proof-of-concept environments only. For more information about AWS Marketplace AMIs, see the AWS Marketplace. The Tableau Server AWS Marketplace AMIs have the following limitations:

  - Choose BYOL, 10, 25, 50 or 100 user AMIs running Microsoft Windows.
  - Tableau Server license included in the hourly fee paid to AWS. If you’re using the Bring Your Own License (BYOL) AMI, the license is sold separately.
  - Single machine, scale-up (vertical scaling) only.
  - No Active Directory. Local authentication only. If you need Active Directory authentication, see Self-Deploy to an Amazon EC2 Instance.
  - No in-place upgrades. You may not upgrade this software to a newer version of
Tableau Server. Instead, you must backup Tableau Server and restore your users and content to a newer AMI running Tableau Server.

- AMIs may be deployed to a specific set of optimal Amazon EC2 instance types and sizes.

To install an AWS Marketplace AMI


2. In the search box enter Tableau.

3. Select the Tableau Server product you want to use. When prompted, sign in to your AWS account.

4. Review the product details, and then click Continue when you are ready to deploy Tableau Server on AWS.

5. On the AMI details page, click the 1-Click Launch tab.

   You can configure all the settings for your instance on this tab. This option allows you to launch only one Amazon EC2 instance at a time.

   **Note:** If you prefer to use the Amazon EC2 console to launch your instance, click the Manual Launch tab from the AMI details page. This option allows you to launch multiple Amazon EC2 instances.

6. In the Region section, select the AWS region where you want to launch your Tableau Server instance.

7. In the EC2 Instance Type section, choose your preferred instance type. See the previous section for recommendations.

8. In the VPC Settings section, choose the Amazon Virtual Private Cloud
(Amazon VPC) that your instance will be launched in.

To ensure that your VPC is secure, see Security in the *Amazon Virtual Private Cloud User Guide* at the AWS website.

9. In the **Security Group** section, review the proposed security group settings and either choose from your existing security groups or accept the new proposed security group.

10. In the **Key Pair** section, choose a key pair to associate with the instance. This key is used to connect to your instance over Secure Shell (SSH) or to decrypt your Windows administrator password.

11. Click **Accept Terms & Launch with 1-Click** to create your new instance.

**What You Need Before You Begin**

Before deploying Tableau Server on AWS, you must have the following:

- An AWS account.
- An Amazon EC2 key pair.
- A Tableau Server trial license or product key.

  - For a standalone deployment, you can use a 14-day trial version of Tableau Server, which doesn’t require the use of a product key. After 14 days, you’ll need to purchase a license, or use a subscription license.

  - For a cluster-based deployment, you need a user-based server license (which covers all authorized users of Tableau Server), a core-based server license (with a minimum of 16 cores), or a subscription license. To obtain a product key, contact sales.

- (Optional) A domain managed by Amazon Route 53.
- (Optional) An SSL certificate managed by AWS Certificate Manager in the region
where you are deploying Tableau Server.

- Storage on Amazon EC2 instance (a general purpose SSD (gp2) is recommended)
- (Optional) An Elastic IP address if you aren’t using ELB or Amazon Route53.

Best Practices for Installing Tableau Server on AWS

The following best practices make installing Tableau Server in the cloud a better experience.

- Read the Tableau Server Plan Your Deployment section.
- Read What You Need Before You Begin.
- Read the Tableau Server Security Hardening Checklist.
- If you’re new to the cloud, use the AWS Free Tier to get familiar with working in the cloud before you deploy Tableau Server on AWS.
- Read about and understand network topology on AWS and how to architect it.

Keeping Costs Down

AWS offers cloud-based services on a pay-as-you-go basis. Costs are determined by the services you run and amount of time you use them. Different combinations of instance types and sizes have different costs. For more information services pricing, see AWS Pricing at the AWS website. You can estimate your total monthly costs using the AWS Simple Monthly Calculator at the AWS website. You can also compare on-premises vs the cloud using the AWS Total Cost of Ownership (TCO) Calculators at the AWS website.

To help monitor and control usage costs on an ongoing basis, you can use Amazon CloudWatch to alert you when your monthly AWS costs reach your predefined spending threshold. For more information, see Monitor Your Estimated Charges Using Cloudwatch in the Amazon CloudWatch User’s Guide at the AWS website.
Tableau Server on AWS Topology

When deploying Tableau Server on AWS, you may choose the level and depth of integration between Tableau Desktop and Tableau Server on your Amazon EC2 instance(s). You can use Tableau Desktop and its ability to extract data from your data sources to act as a bridge between your data and Tableau Server. Based on your needs, or if you already have a lot of data in the AWS cloud, you might instead choose to leverage the full range of AWS services in conjunction with Tableau Server. The following diagram shows data source integration with Tableau Desktop and Tableau Server.

In the following diagram, all of your data is hosted on AWS. You can analyze both structured and unstructured data managed in a secure, scalable manner. You may leverage data that resides on AWS only, unmanaged data that lives outside of AWS, or a combination of both. This flexibility greatly increases the ability of your organization to support moves to the cloud from on-premise data, since both types of data is supported equally.
Selecting an AWS Instance Type and Size

Choosing the right instance for your workload is an important factor for a successful Tableau Server deployment. You can choose from a wide range of Amazon EC2 instance types. For a complete list of all available instance types and sizes, see Amazon EC2 Instance Types at the AWS website.

At minimum, a 64-bit Tableau Server requires a 4-core CPU (the equivalent of 8 AWS vCPUs) and 8 GB RAM. However, a total of 8 CPU cores (16 AWS vCPUs) and 64GB RAM are strongly recommended for a single production Amazon EC2 instance.

An AWS vCPU is a single hyperthread of a two-thread Intel Xeon core for M4, M3, C4, C3, and R3 instances. A simple way to think about this is that an AWS vCPU is equal to half a physical core. Therefore, when choosing an Amazon EC2 instance size, you should double number of cores you have purchased or wish to deploy with. Example: You have purchased an 8 core license for Tableau Server (or need to support enough active users where 8 cores are warranted). You should choose an Amazon EC2 instance type with 16 vCPUs.
Typical instance types and sizes for development, test, and production environments

- m4.4xlarge
- r4.4xlarge
- r4.8xlarge

Recommended specifications for a single production instance

<table>
<thead>
<tr>
<th>Component/Resource</th>
<th>Amazon AWS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>16+ vCPU</td>
</tr>
<tr>
<td>Operating System</td>
<td>Linux:</td>
</tr>
<tr>
<td></td>
<td>- Redhat Enterprise Linux (RHEL) 7.3 or higher (not 8.x)</td>
</tr>
<tr>
<td></td>
<td>- CentOS 7 or higher (not 8.x)</td>
</tr>
<tr>
<td></td>
<td>- Oracle Linux 7 or higher (not 8.x)</td>
</tr>
<tr>
<td></td>
<td>- Ubuntu 16.04 LTS</td>
</tr>
<tr>
<td>Memory</td>
<td>64+ GB RAM (4GB RAM per vCPU)</td>
</tr>
<tr>
<td>Storage</td>
<td>Two volumes:</td>
</tr>
<tr>
<td></td>
<td>30-50 GiB volume for the operating system</td>
</tr>
<tr>
<td></td>
<td>100 GiB or larger volume for Tableau Server</td>
</tr>
<tr>
<td>Storage type</td>
<td>EBS recommended (SSD (gp2) or Provisioned IOPS)</td>
</tr>
<tr>
<td>Disk latency</td>
<td>Less than or equal to 20ms as measured by the <code>iostat</code> command in Linux.</td>
</tr>
</tbody>
</table>
Self-Deploy a Single Tableau Server on AWS

For the most flexibility in security, scaling, and capacity, you can perform a self-deployment of Tableau Server on AWS. Building a cloud-based solution has many benefits over an on-premises installation. For example, the overall total cost of ownership for building a Tableau Server solution in the cloud is normally much less than a similar on-premises solution because you don’t have to buy all of the expensive hardware. In addition, the cloud can provide better uptime, reliability, and fault-tolerance, especially if you deploy your solution across different regions and Availability Zones.

When you deploy Tableau Server in a cluster in AWS, we recommend that you use Amazon Elastic Block Store (EBS) volumes with Amazon EC2. EBS provides persistent block-level storage volumes for use with Amazon EC2 instances. You should attach a separate EBS General Purpose (SSD) volume with a volume size of at least 100 GiB to allow room for installing Tableau Server. This volume size supports a baseline performance of 300 input/output operations per second (IOPS) with the ability to burst to 3,000 IOPS. For more information, see General Purpose SSD (gp2) Volumes in the Amazon EC2 User Guide for Windows Instances at the AWS website.

For heavy workloads where the storage subsystem must provide an absolute minimum number of IOPS for performance, you can either create a larger EBS General Purpose (SSD) volume to get up to 10,000 IOPS with the ratio of 3 IOPS per GiB, or use an EBS Provisioned IOPS (SSD) volume with the IOPS you need. In either case, we recommend that you use EBS-optimized instances if you plan to run heavy extracts and do a lot of data processing. For more information about Tableau Server performance, see Tableau Server Performance Overview.

Perform the following tasks to install and configure Tableau Server in the AWS cloud.

Step 1: Create a Virtual Private Cloud (VPC)
Step 2: Configure networking and security
Step 3: Launch an Amazon EC2 instance
Step 1: Create a Virtual Private Cloud (VPC)

Your first step is to create a Virtual Private Cloud (VPC).

1. Sign in to AWS and then navigate to the Amazon VPC console (https://console.aws.amazon.com/vpc/)

2. Use the region selector to choose the location where your other AWS resources are located. Often, this is the region closest to you.

3. Under Resources, click Start VPC Wizard, on the Step 1: Select a VPC Configuration screen, click VPC with a Single Public Subnet, and then click Select.

4. On the Step 2: VPC with a Single Public Subnet screen, give your VPC a name, and then click Create VPC. You can leave all other settings at their default values.

Step 2: Configure networking and security

To allow inbound traffic to reach your VPC, you should limit traffic to three standard ports (HTTP, HTTPS, and RDP). For more information, see Recommended Network ACL Rules for your VPC in the Amazon Virtual Private Cloud User Guide at the AWS website.

1. Navigate to the Amazon EC2 console (https://console.aws.amazon.com/ec2/).

2. Use the region selector to choose the location where you created your VPC.

3. In the navigation pane, click Security Groups, and then click Create Security
4. On the Create Security Group screen, in the Security group name field, enter a name for your security group.

5. In the Description field, enter a description for the security group.

6. In the VPC field, select your VPC from the list.

7. Click the Inbound tab, click Add Rule, and then in the Type list, select HTTP (80), and in the Source column, choose My IP for each rule. This limits inbound traffic to your computer. To specify an IP address range instead, choose Custom, and then enter the range in CIDR notation.

For more information, see Adding Rules to a Security Group in the Amazon EC2 User Guide for Linux Instances at the AWS website.

**Note:** This port is required to receive incoming unencrypted web traffic.

8. Click Add Rule, in the Type list, select HTTPS (443), and in the Source column, choose My IP for each rule. This limits inbound traffic to your computer. To specify an IP address range instead, choose Custom, and then enter the range in CIDR notation.

For more information, see Adding Rules to a Security Group in the Amazon EC2 User Guide for Linux Instances at the AWS website.

**Note:** This port is required to receive incoming encrypted web traffic.

9. Click Add Rule, in the Type list, select RDP (3389), and in the Source column, choose My IP for each rule. This limits inbound traffic to your computer. To specify an IP address range instead, choose Custom, and then enter the range in CIDR
For more information, see Adding Rules to a Security Group in the Amazon EC2 User Guide for Linux Instances at the AWS website.

Note: This port is required to accept Remote Desktop Protocol (RDP) requests so that you can connect to the EC2 instance.

10. Click Create.

Step 3: Launch an Amazon EC2 instance

After you create your VPC, you can launch an Amazon EC2 instance into it.

For more information about how to launch and connect to a Linux instance, see Getting Started with Amazon EC2 Linux Instances in the Amazon EC2 User Guide for Linux Instances at the AWS website.

1. Navigate to the Amazon EC2 console (https://console.aws.amazon.com/ec2/).

2. Use the region selector to choose the location where you created your VPC.

3. Under Create Instance, click Launch Instance.

4. Select an Amazon Machine Image (AMI) that meets the system requirements for Tableau Server.

5. On the Step 2: Choose Instance Type screen, select the instance size you want (for example, m4.2xlarge).
Note: For Tableau Server v10 on a 64-bit virtual machine, you need a minimum of 4 physical cores. On AWS, this means 8 vCPUs. For more information, see Amazon EC2 Instance Types at the AWS website.

6. Click **Next: Configure Instance Details**.

7. On the **Step 3: Configure Instance Details** screen, in the **Network** list, select your VPC.

8. Click through the steps in the wizard until you get to **Step 6: Configure Security Group**.

9. On the **Step 6: Configure Security Group** screen, click **Select an existing security group**, and then select the security group that you created earlier.

10. Click **Review and Launch**, review your configuration, and then click **Launch**.

11. When you are prompted, create a new key pair, download it as a .pem file, and keep it in a safe place. You need the key pair in order to create a password that you can use to log into the Amazon EC2 instance.

12. When you have finished creating and downloading your key pair, click **Launch Instances**.

13. Click **View Instances** and locate your instance in the list.

14. Copy the instance ID. You’ll need this in the next step.

**Step 4: Create an elastic IP address for the VPC**

After you launch your Amazon EC2 instance, you can create and allocate a static public IP address to your VPC.
1. Navigate to the Amazon VPC console (https://console.aws.amazon.com/vpc/).

2. Use the region selector to choose the location where you created your VPC.

3. In the navigation pane, click Elastic IPs.

4. Click Allocate new address, and then click Allocate.

5. In the New address request succeeded dialog box, click the Elastic IP address.

6. On the Action menu, click Associate address.

7. In Resource type, select Instance.

8. In the Instance drop-down list box, select your instance, and then click Associate.

9. Copy the new private IP address. You’ll need this later.

**Step 5: Log in to Amazon EC2**

To run Tableau Server on AWS, you install it on the Amazon EC2 instance or instances that you just configured. To begin, you log into the Amazon EC2 instance where you want to install Tableau Server.

Use Secure Shell (SSH) from a computer running Linux to connect to your Amazon EC2 instance running Linux. If you’re connecting from a computer running Microsoft Windows, you’ll need to install an SSH client, such as PuTTY, to connect to your Amazon EC2 instance running Linux. For more information, see Connecting to Your Linux Instance Using SSH in the Amazon EC2 User Guide for Linux Instances at the AWS website.

**Step 6: Install Tableau Server**

Now that you’re logged in to your EC2 instance, you can install Tableau Server. The installation steps on an EC2 instance are the same as they are for any other computer. For
instructions, see Install and Configure.

After you install Tableau Server and create a Tableau Server administrator account and sign in, you can add users.

If you want to run a Tableau Server cluster, install servers (as workers) on additional Amazon EC2 instances. For more information, see Self-Deploy a Distributed Tableau Server.

Self-Deploy Tableau Server on AWS in a Distributed Environment

If you want to run Tableau Server in a highly available (HA) distributed environment, also known as a cluster, you need to launch three or more Amazon EC2 instances of the same type and capacity to your Amazon Virtual Private Cloud (VPC) and configure them as worker nodes.

The following scenario assumes that you have three Amazon EC2 instances with Tableau Server installed on each instance. One instance is configured as the primary node, and the other two instances are configured as worker nodes.

The following steps show how to install and deploy Tableau Server on a cluster of three Amazon EC2 instances in a highly available and scalable configuration.

Step 1: Create a Virtual Private Cloud (VPC)

(Optional) Step 2: Create an AWS Directory Service for the VPC

Step 3: Deploy Three Amazon EC2 Instances

Step 4: Install and Configure Tableau Server

Step 5: Create a Load Balancer for the Tableau Server Cluster

Step 1: Create a Virtual Private Cloud (VPC)
The following steps assume that you have an Amazon VPC with at least six subnets (three public and three private) in different Availability Zones, as shown below. Be sure to use static IP addresses when setting up your IP addresses. For information about how to create a VPC with public and private subnets, see Scenario 2: VPC with Public and Private Subnets in the Amazon Virtual Private Cloud User Guide at the AWS website.

Launch one EC2 instance into each of the three subnets in your VPC for use as Remote Desktop gateways.

For more information about how to launch and connect to a Linux instance, see Getting Started with Amazon EC2 Linux Instances in the Amazon EC2 User Guide for Linux Instances at the AWS website.

(Optional) Step 2: Create an AWS Directory Service for the
VPC

Follow the steps in the Create a Microsoft AD Directory section, in the AWS Directory Service Administration Guide at the AWS website, to create a fully-managed Samba-based directory in AWS. When you create a directory with Microsoft AD, the AWS Directory Service creates two directory servers and DNS servers. The directory servers are created in different subnets in your Amazon VPC for redundancy, so that your directory remains accessible even if a failure occurs.

Use the two private subnets available within your Amazon VPC to create the Microsoft AD so that you can run Tableau Server across Availability Zones.

Step 3: Deploy Three Amazon EC2 Instances

Deploy three Amazon EC2 instances across three Availability Zones, as shown in the following figure. You’ll use the node you installed previously as the initial server and these two new nodes as additional servers. All of the instances should be of the same type and capacity.
After you have launched the EC2 instances, connect to them from one of the Remote Desktop Gateway (RDGW) instances by using the credentials that you decrypted for the local administrator account.

**Step 4: Install and Configure Tableau Server**

You’ll install Tableau Server on the Amazon EC2 instances you launched in Step 3: Deploy Three Amazon EC2 Instances and configure the instances as an initial server and the rest as additional servers. For more information about installing and configuring Tableau Server
on an initial server and additional servers, see Install Tableau Server on Additional Nodes.

Step 5: Create a Load Balancer for the Tableau Server Cluster

Follow the steps outlined in Getting Started with Elastic Load Balancing in the Elastic Load Balancing User Guide at the AWS website to launch a load balancer within your VPC.

1. In Step 1: Select a Load Balancer Type, if you want the load balancer to be publicly accessible, select the two public subnets. Otherwise, select the Create an internal load balancer check box and choose the two private subnets.

   If you choose to expose the load balancer with a public endpoint, make sure that you configure Elastic Load Balancing with SSL, as explained in Create a Classic Load Balancer with an HTTPS Listener in the Elastic Load Balancing Classic Load Balancer Guide at the AWS website.

2. In Step 2: Configure Your Load Balancer and Listener of the instructions, in the Elastic Load Balancing User Guide at the AWS website, ensure that your security group is configured to allow access on port 80 or 443 only, with the source limited to hosts or ranges of hosts that will access Tableau Server.

3. In Step 4: Configure Your Target Group, you can specify the ping path as /.

4. In Step 5: Register Targets with Your Target Group, select the Tableau Server instances and ensure that Enable CrossZone Load Balancing is selected so that the load balancer can load-balance the traffic across the instances in multiple Availability Zones.

   Update Tableau Server to use the load balancer. For more information, see Add a Load Balancer in the Tableau Server Help.

Alternatively, you can configure Tableau Server to work with a load balancer by performing
the following steps.

1. Create a subnet for your load balancer that has a CIDR block with a /27 bitmask, for 32 IP addresses.

   ![Note: The IP addresses provided by Elastic Load Balancing are dynamic, and Tableau Server needs a list of static IP addresses for this configuration. To make this work, we’re creating a subnet with the smallest possible CIDR range so that the IP addresses the load balancer has are limited to a finite set.]

   For the next steps we’ll use the tsm command line interface, which is installed with Tableau Server by default. You can use tsm to perform administrative tasks from the command line on Tableau Server. For a general overview, see tsm Command Line Reference in the Tableau Server Help.

2. In the Tableau Server bin directory, enter the following command, where name is the URL that will be used to reach Tableau Server through the load balancer:

   ```
   tsm configuration set gateway.public.host "name"
   ```

3. Enter the following command, where server1, server2, and so on are the IP addresses for the given CIDR range of subnets for Elastic Load Balancing:

   ```
   tsm configuration set gateway.trusted "server-1,server2,..,server30"
   ```

4. Apply the changes:

   ```
   tsm apply-pending-changes
   ```

   The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt
using the \texttt{-r} option, but this does not change the restart behavior. For more inform-
ation, see \texttt{tsm pending-changes} apply.

5. Start the server so the changes can take effect.

\texttt{tsm start}

\section*{Securing Tableau Server on AWS}

Whether you deploy Tableau Server on premises or in the cloud, it's important to take steps to make your deployment secure. For information about making Tableau Server more secure, see \texttt{Security}.

In addition to the security features built in to Tableau Server, AWS provides other features that you can use to help secure your Tableau Server environment, such as:

- \textbf{Amazon VPC} adds another layer of network security to your environment by creating private subnets.

- \textbf{Security Groups} determine which inbound and outbound traffic can connect to your network. Limit inbound to your IP addresses in your \textit{Classless Inter-Domain Routing (CIDR)} block. Do not use 0000\textbackslash 0, which is unsecure because it allows all traffic to access your server.

- \textbf{AWS Identity and Access Management (IAM)} allows specific control over user access to features within AWS.

- \textbf{AWS Direct Connect} allows a dedicated network connection from a corporate net-

  work to AWS using industry-standard 802.1Q VLANs through an AWS Direct Con-

  nect partner. For more information, see \textit{Requesting Cross Connects at AWS Direct Connect Locations} in the \textit{AWS Direct Connect User Guide} at the AWS website.

- \textbf{Amazon EBS Encryption} offers a simple and performant way to encrypt data at rest inside your disk volumes and data-in-transit between EC2 instances and EBS stor-

  age.
You can implement enterprise application security in AWS and Tableau Server to enable a single report or dashboard to securely serve the needs of a broad and diverse user base, including both internal and external users. Enterprise application security has three main components:

- Network
- Client Access
- Data

**Network**

Network security for Tableau Server in AWS relies on the use of Amazon VPC security groups with SSL for securing internal and external communications. For more information, see [Security Groups for Your VPC](https://docs.aws.amazon.com/isg/latest/security-groups.html) in the Amazon Virtual Private Cloud User Guide at the AWS website.

**Amazon VPC**

An Amazon VPC is a distinct, isolated network within the cloud; network traffic within each Amazon VPC is isolated from all other Amazon VPCs. Using an Amazon VPC allows you to create your own network subnets and divide application layers into network subnets for a greater level of control. We recommend that you install and run Tableau Server in a separate subnet within your Amazon VPC so that you can configure the network for access to Tableau Server and other data sets. The following figure shows a typical installation of a single-node Tableau Server in an Amazon VPC.
Security Groups

Security groups enable you to define what types of network traffic can access Tableau Server. Amazon EC2 security groups act as a firewall that governs network traffic into and out of Amazon EC2 instances. You can define and assign security groups that are appropriate for your Amazon EC2 instances. By default, Amazon EC2 instances are launched with security groups that allow no inbound traffic. Before you can access your EC2 instance, you need to make changes to allow the appropriate inbound traffic.

Here are the minimum requirements for connections to Tableau Server on an EC2 instance:

- Connection via RDP (port 3389) using a Remote Desktop client to access and manage the instance and services.
- Standard web traffic via HTTP (port 80) and HTTPS (port 443), to view content hosted on, and to publish to Tableau Server.
- Communication between Tableau Server components on different instances (if any) should be allowed.

Based on these requirements, you should enable only three standard ports for inbound traffic to your EC2 instance: HTTP 80, HTTPS 443, and RDP 3389. You should also limit remote access (port 3389) from a few hosts, and also limit HTTP and HTTPS traffic to hosts within your corporate network or to a trusted set of clients.
Client Access

By default, Tableau Server uses standard HTTP requests and responses. Tableau Server can be configured for HTTPS (SSL) with customer-supplied security certificates. When Tableau Server is configured for SSL, all content and communications between clients are encrypted and use the HTTPS protocol. When you configure Tableau Server for SSL, the browser and SSL library on the server negotiate a common encryption level. Tableau Server uses OpenSSL as the server-side SSL library, and is pre-configured to use currently accepted standards. Each web browser that accesses Tableau Server via SSL uses the standard SSL implementation provided by that browser. For more information about how Tableau Server uses SSL, see SSL. Tableau Server will listen for SSL traffic only on port 443. You may not configure custom ports for SSL/TLS.

If you’re using Elastic Load Balancing (ELB), ELB can also perform SSL termination on your behalf. Allowing ELB to handle encryption/decryption of web traffic is an easy way to secure the client’s connection with Tableau Server without needing to manually configure SSL on Tableau Server itself. For more information, see AWS Elastic Load Balancing: Support for SSL Termination at the AWS website.

AWS Directory Service

Optional. The AWS Directory Service is a managed service that allows you to connect your AWS resources to an existing on-premises directory such as Microsoft Active Directory (with AD Connector), or to set up a new, stand-alone directory in the AWS cloud (with Simple AD). Connecting to an on-premises directory is easy, and after this connection is established, all users can access AWS resources and applications with their existing corporate credentials.

Using the AWS Directory Service, you can choose to use Active Directory-based authentication instead of local authentication, which creates users and assigns passwords using Tableau Server’s built-in user management system. To set up Active Directory-based authentication, in the configuration step after installing Tableau Server, you must choose Active Directory. It is not possible to switch between Active Directory and local authentication later.
Data

Tableau Server uses native drivers (relying on a generic ODBC adapter when native drivers are not available) to connect to databases whenever possible, for processing result sets, for refreshing extracts, and for all other communications with the database. You can configure the driver to communicate on non-standard ports or use transport encryption, but this type of configuration is transparent to Tableau Server. However, since the Tableau Server-to-database communication is typically behind a firewall, you may choose not to encrypt this communication.

Connecting to Data Stores in AWS

You can launch AWS resources, such as Amazon Relational Database Service (Amazon RDS), Amazon Elastic MapReduce (Amazon EMR), or Amazon Redshift, into an Amazon VPC. By placing the Tableau Server into the same Amazon VPC as your data stores, you can ensure that your traffic never leaves the Amazon VPC.

You can use subnets with security groups to launch your resources into different layers but allow them to communicate securely within an Amazon VPC, as illustrated in the following diagram.
Connecting to Data Stores Outside of AWS

You can optionally connect your Amazon VPC to your own corporate data center by using an IPsec hardware VPN connection, thus making the AWS cloud an extension of your data center. A VPN connection consists of a virtual private gateway attached to your Amazon VPC and a customer gateway located in your data center. You might choose to use AWS Direct Connect, which is a network service that provides an alternative to using the Internet to utilize AWS cloud services. AWS Direct Connect lets you establish a dedicated network connection by using industry-standard 802.1Q VLANs through an AWS Direct Connect partner. For more information, see Requesting Cross Connects at AWS Direct Connect Locations in the AWS Direct Connect User Guide at the AWS website.

You can use the same connection to access public resources (such as objects stored in Amazon Simple Storage Service (Amazon S3) using public IP address space) and private resources (such as Amazon EC2 instances running within an Amazon VPC using a private IP space), while maintaining network separation between the public and private environments.

Encrypting Data at Rest

Amazon EBS encryption offers a transparent and simple way to encrypt volumes which may contain personally identifiable information (PII). EBS encryption encrypts both data at rest inside the volume and data in transit between the volume and the instance using AES-256. This feature has little-to-no impact on Tableau Server performance. Therefore, we recommend that you take advantage of this service regardless of whether your systems store PII.

Optimizing Performance of Tableau Server on AWS

Optimizing the performance of Tableau Server when it is installed on an Amazon EC2 instance in the AWS cloud adds another dimension to tuning your Tableau Server solution. This section discusses tuning Tableau Server for the cloud. For general performance tuning information, see the Tableau Server Performance Overview. For information about tools you can use to help optimize performance, see Performance Resources.
It is important to keep in mind that every deployment of Tableau Server on AWS is different because everyone’s workload is unique. Your company’s employees are different, leverage different data, ask different kinds of questions, and have different business needs than other companies. For this reason, we recommend testing your Tableau Server workloads on a variety of different Amazon EC2 instance types before putting them into production. The demands of your workload will likely be influenced by the following factors:

- Heavy, light, or moderate Tableau Data Extract use
- The proportion of people who view vs. interact with visualizations and dashboards
- Tableau Data Extract refreshes which occur during or after working hours
- The number of concurrent individuals during any given period
- View and dashboard complexity
- Size of the community who leverage Tableau Web Authoring

Performance Best Practices

The following performance best practices may be useful for you to reference as you deploy Tableau on AWS:

- Always run at least 8 cores per Amazon EC2 Instance

  Even with a relatively low number of users, EC2 Instances with fewer than 16 vCPUs (the equivalent of 8 cores), do not consistently perform well. For example, one r4.4xlarge instance, which has 16vCPUs, is able to handle more users, with a lower response time and error rate, than two r4.2xlarge instances, which have 8vCPUs each. This pattern is consistent as you scale up, with four 16vCPU instances and two 32vCPU instances dramatically outperforming eight 8vCPU instances.

- The workload greatly determines the results
The robustness of your workload goes a long way in determining how it performs on different EC2 instance types. Using a different set of dashboards, for example, you will see noticeable differences in performance from the same underlying virtual machine instances. Attempting to compare performance using any other workload than your own is not very useful.

- More CPU is better

CPU is often the main bottleneck for Tableau Server performance. In general, when you want to do more with Tableau, we recommend you add more and better CPU.

- Make sure your Amazon EC2 instance has enough RAM

When we ran the same workloads on instances with less CPU but more RAM, we experienced higher transactions per second (TPS), lower response time, and lower error rate. EC2 instances with too little RAM can cancel out the benefit of high-end CPU. In production, run with at least 30 GB RAM but shoot for 8 GB of RAM per core. While it is important to choose an instance with a lot of CPU, running Tableau Server on instances starved for RAM is going to lead to low performance—no matter how much CPU you have.

- You want to use SSD-based volumes, but may not need Provisioned IOPS

Tableau Server encompasses a number of processes and components, including an industrial strength database (PostgreSQL) that stores the system’s metadata. Tableau Server needs a reasonable level of disk throughput in order to perform well, and we advise using Amazon Elastic Block Store (EBS) SSD-based volumes only. Magnetic disks lack the throughput needed to handle the database’s requests effectively. In our test, we ran both general purpose SSD (gp2) and EBS-provisioned IOPS volumes, with two EBS disks being used for most tests. Most EBS volumes had 1500 provisioned IOPS. After running the tests again with General Purpose SSDs, our results were nearly identical with moderately demanding workloads. While there are certainly cases where Provisioned IOPS will make a noticeable difference in the
performance of your Tableau Server workloads on AWS, you shouldn’t assume that you need Provisioned IOPS by default. The best way to find out, of course, is to test your Tableau Server workloads yourself.

Scaling Tableau Server on AWS

Tableau Server is designed to scale up with more CPU cores and memory, and scale out when you add servers. This architecture allows you to maximize the use of compute resources while giving you the ability to scale massively.

To build in redundancy, you need to add additional servers, which host copies of the repository, file store, and other processes. In this scenario, you should isolate the initial server in its own node, and it should ideally run as few of the server processes as possible.

**Note:** Although you can split an 8-core license onto two 4-core machines, we recommend that you scale in increments of 8-core machines only.

Load Balancing

On AWS, Elastic Load Balancing (ELB) automatically distributes incoming application traffic across multiple Amazon EC2 instances in the cloud. It enables you to achieve greater levels of fault tolerance in your applications, and seamlessly provides the required amount of load balancing capacity needed to distribute application traffic.

You can use ELB to distribute requests across multiple gateways in a Tableau Server cluster. In the figure below, all three nodes have gateways, which are used to route requests to available server processes. Unlike the repository process, there aren’t any passive or standby gateway processes—all gateways are active. When you add a load balancer to a Tableau Server cluster, the URL that’s accessed by Tableau Server users belongs to the load balancer, not to the initial Tableau Server.
High Availability

Now that you have provided redundancy for the file store, repository, and gateway by adding additional nodes, you can additionally build redundancy for the initial Tableau Server. You can do this by creating a backup of the initial Tableau Server. Although the backup must be licensed during installation, it does not count as one of the three environments allowable under the Tableau end user license agreement (EULA).
To configure for high availability, you need to run a failover cluster plus an additional computer as the backup for your primary Tableau Server. If you configure for high availability, the initial Tableau Server and the backup may be running few or no Tableau Server processes. For more information, see High Availability in the Tableau Server Help.

Troubleshooting Tableau Server on AWS

Follow the suggestions in this topic to resolve common issues with Tableau Server when installed on an Amazon EC2 instance in the AWS cloud.

- **TSM CLI does not work from additional nodes**

  The TSM cli is designed so you can run tsm commands from any node in a server cluster by including the `-s` option to specify the name or IP address of the initial node. The following symptom might occur when running TSM commands from multiple nodes in AWS:

  - When running a tsm command with the `-s` option, a message can display:

    Unable to verify the server's HTTPS certificate.

  To run TSM at the command line from any node other than the initial node, you need to use the `-s` switch and the private IP address of the initial node.

- **Tableau Server does not use all CPU cores**

  One or more of the following symptoms might occur when working with Tableau Server core-based licensing:

  - When installing Tableau Server, the installer might show fewer cores than you would expect to see for the computer running Tableau Server.

  - On the Licenses page on Tableau Server, the number of cores listed under Licenses in Use is less than you would expect to see.
• When you run `tsm licenses list`, Tableau Server might report fewer cores in use that you would expect to see.

For more information, see Not All Cores Recognized by Tableau Server.

• There is high I/O latency

Changing the disk driver performance setting to Better Performance may give you better results.
Manage Sites

What is a site ................................................................. 248

Planning a Site .................................................................. 251

Manage Users and Groups ................................................. 254

Dashboard-based Custom Portals ....................................... 305

Data Connection Authentication ........................................ 309

Manage Content Access .................................................... 330

What is a site

You might be used to using the term site to mean “a collection of connected computers,” or perhaps as the short form of “website.” In Tableau-speak, we use site to mean a collection of users, groups, and content (workbooks, data sources) that’s walled off from any other groups and content on the same instance of Tableau Server. Another way to say this is that Tableau Server supports multi-tenancy by allowing server administrators to create sites on the server for multiple sets of users and content.

All server content is published, accessed, and managed on a per-site basis. Each site has its own URL and its own set of users (although each server user can be added to multiple sites). Each site’s content (projects, workbooks, and data sources) is completely segregated from content on other sites.

Note: This article pertains to configuring sites on Tableau Server deployments. For Tableau Online, see Site Administrator Role and Tasks.
In this article

Site administrator tasks

Steps for setting up your site

See also

Site administrator tasks

Where the Server Administrator site role gives a user unrestricted access to the entire Tableau Server deployment, the Site Administrator site role gives a user unrestricted access at the site level.

Although a server administrator can work at both the server and site levels, we make a distinction between the two sets of tasks. The site administrator is typically in charge of creating and maintaining the framework that enables Tableau users in the organization to publish, share, manage, and connect to data sources and workbooks. Site administrator tasks include any of the following:

- Creating project hierarchies to organize the site’s data sources and workbooks. This can include delegating project-level management to project leaders.
- Creating groups and assigning permissions that allow users to access only the content they need.
- Adding and removing users, assigning their site roles. This is allowed by default on a site; however, a server administrator can restrict this access to the server level only.
- Managing the site’s extract and subscription schedules.
- Monitoring site activity.
For more information about the distinction between server administrator and site administrator, see Administrator-level access to sites, in the Manage Server section.

Steps for setting up your site

The table below shows a loose sequence of steps for setting up a site, along with links to topics where you can get more information. You can complete the steps in any order that makes sense for you.

However, before you perform the steps to configure the site, we recommend spending some time with the articles in this section, learning about site authentication, site roles, projects, and permissions. Ideally you would document a plan for your projects, groups, and overall permissions strategy. Then set up a few projects and add a preliminary set of users, to test the plan and resolve issues before you add the remaining users. You can change many site settings after your users are working with the site, but try to go in with the intention of minimizing post-production changes.

<table>
<thead>
<tr>
<th>Plan</th>
<th>To supplement the recommendations above this table, get an overview of how the site components work together in Planning a Site.</th>
</tr>
</thead>
</table>
| Configure access | Work with the server administrator to determine how users sign in to the site, and configure the site appropriately.  
For example, if the server is configured for single sign-on using SAML, you might configure SAML authentication at the site level as well. |
| Create projects and the permissions structure | Projects help you organize content, delegate project-level content management, and manage permissions effectively. To get started, see Use Projects to Manage Content Access. |
### Add users

Determine the users who can sign in to the site. See Add Users to a Site.

### Get your data to Tableau Server

After you create your projects and permissions structure, designate approved users for publishing and managing vetted data sources to the appropriate projects on the site.

Once content is published to the site, you can maintain connection information (credentials, access tokens) and refresh schedules. For more information, see Refresh Data on a Schedule.

### Analyze site usage and performance

Monitor usage, performance, and other metrics. See Administrative Views.

### See also

If you are a server administrator on your Tableau Server deployment, you can learn more about sites, when to use them (vs. projects), and more in Sites Overview, in the Manage Server section.

### Planning a Site

Before you add users and content to a site, we recommend that you plan the following aspects of the site.

- Projects
- Users and groups
- Site roles and permissions
- Extract refresh schedules
- Steps for setting up your site

The subsequent sections go over these site components, assuming that you are familiar with

**Note:** This article and section apply only to Tableau Server deployments that you maintain, whether on-premises or in the cloud. If you use Tableau Online, see Manage Content Access.

**Projects**

You can create projects on a site, which act as containers in which you can organize related data sources and workbooks. For example, you might set up a project to contain all of the certified data sources and workbooks your organization uses for mission-critical decisions. Or you might set up projects by department.

Projects are also useful for managing permissions. Once you know how your users need to access content, it’s usually easier to create projects based on those the type of content, and maintain permissions at the project level.

Every site has a default project named **Default**. When you create projects, the new projects get their initial set of permissions from the default project. In effect, the default project is a template for new projects. As we explain in related articles, for most environments, we recommend that you use the Default project only as a permissions template, and not as a container for published content.

For more information, see Use Projects to Manage Content Access.

**Users and groups**

Any user who will publish data sources and workbooks to a site must be able to sign in to it. If the user already has an account on the server, you’ll need to add that user to the appropriate site. You can add a user to more than one site as well. If the user doesn’t already exist, you
need to create a user account. Either way, make a list of the users who will need to be able to sign in to each site.

Note: The server license might restrict how many users you can have. Tableau Server licenses are based on either cores or users. If the server has a user-based license, there’s an upper limit to how many users can have active accounts on the server. Check with the server administrator to make sure that you’ll be able to have an account for all your users.

In general, we recommend that you create groups on the server and then add users to the groups. This helps to make permissions much easier to manage. You can assign permissions on groups, to give those permissions to all users in the group. (See the next section.)

A typical strategy is to create groups for users who use content in similar ways. For example, you might create a group named SalesWBPublishers for all the users in the Sales department who publish workbooks, and a separate group named SalesDSPublishers for people in Sales who publish data sources. Each of these sets of users needs its own set of capabilities, so it makes sense to have a group for each for these needs.

Site roles and permissions

Each user has a site role that determines the maximum permissions that they can have on the site. As part of your site planning, you need to decide each user’s site role. A user with a site role that’s too restrictive might not be able to do the work they need. By the same token, a security best practice is to limit users’ capabilities to only those that they need to do their work. This is referred to as following the principle of least privilege.

You or a site administrator you delegate this task to must also determine the permissions a user needs to work with content. Each content asset (workbook, data source, project) supports a set of capabilities. For example, you can View or Add Comments to a workbook. Before a user can perform tasks on a workbook, their permissions must allow those
capabilities. A recommended practice is to sketch out a mapping of permissions to users outside of Tableau before you try to set this up on the server.

As we just noted, site roles act as an upper limit on permissions. Permissions determine what a user can do within the context of the site role. A user whose site role is **Interactor** can never publish to the site, regardless of the permissions you grant them. A user whose site role is **Publisher** can publish a workbook to the site, but only if that user has permission to save and view workbooks.

**Extract refresh schedules**

If users publish data sources or workbooks that include extracts, you usually want to make sure that the extracts are refreshed, so that they contain the latest data. Users can manually refresh an extract, but this isn’t always a good idea if the extract is large, and the refresh takes a long time. Instead, you can set up schedules for when an extract should be refreshed. Another planning task for a site administrator, therefore, is to think about when extracts should be refreshed and to work out schedules.

**Manage Users and Groups**

**Add Users to a Site**

Everyone who needs to access Tableau Server—whether to browse, publish, edit content or administer the site—must be added as a user. Administrators can add users to sites in the following ways:

- By adding a local user account or a user account from Active Directory, as described in this topic. You can also add users by importing an Active Directory group. For details, see Create Groups via Active Directory.

- By importing a CSV file that contains user information. For details, see Import Users and CSV Import File Guidelines.
**Note:** If you are running Tableau Server on Linux then all external directory communication is configured and managed with a LDAP identity store. In the context of user and group synchronization, Tableau Server configured with LDAP identity store is equivalent to Active Directory. Active Directory synchronization features in Tableau Server function seamlessly with properly configured LDAP directory solutions.

In a single-site environment, administrators can add users to a site on the Users page. In a multi-site environment, you will use the Site Users page. Server administrators must give site administrators the ability to add users to sites. This setting can be enabled or disabled by the server administrator.

**Note:** Users can be added to sites, or to the server. To add users to the server, see Add Users to the Server. The options available for adding users depends on the authentication method that you select when you first configure Tableau Server. If you are using local authentication, you cannot add Active Directory users. If you are using Active Directory, you cannot add local users.

If you add a user without assigning site membership and role, the user is assigned the Unlicensed role and won’t use a server license (user-based licensing only). The user will exist in Server Users, but will not be a member of any site until you add that user to the site.

On the Users (single-site) or Site Users (multi-site) page you can see the users on the site you’re currently signed into. You can add users to (or remove them from) the current site only. If a user belongs to more than one site, you can remove that user from the current site.

**Note:** When a site administrator removes a user from a site (and the user only belongs to that one site), the user will be automatically deleted from the server if that user doesn’t own any content.
This screenshot is from a single-site environment. In a multi-site environment, this would be the Site Users page.

To add local users to a site

1. In a site, click **Users**, click **Add Users**, and then click **New User**.

   ![Add Users to this Site](image)

   **Note**: This screenshot is from a multi-site environment. In a single-site environment, this would be the Users page.

2. Enter a user name. If the server is configured for local authentication, using an email
address for the user name is the best way to avoid user name collisions (for example, jsmith@example.com instead of jsmith).

Also enter information in the following fields:

- **Display Name**—Type a display name for the user (e.g., John Smith).
- **Password**—Type a password for the user.
- **Confirm password**—Retype the password.
- **Email**—This is optional and can be added at a later time in the user profile settings.

3. Select a site role. For details on site roles, see Set Users’ Site Roles.

4. Click **Add User**.

**Note for multi-site servers:** A site administrator can edit an existing local user account only if the administrator has control over all of the sites the user is a member of.
For example, if User1 is a member of sites A and B, an administrator of site B only cannot edit User1’s full name or reset the password.

To add Active Directory users to a site

Before adding users to a site, be sure to review User Management in Active Directory Deployments to understand how multiple domains, domain naming, NetBIOS, and Active Directory user name format influence Tableau user management.

1. In a site, click Users, and then click Add Users, and then click Active Directory User.

2. Enter one or more user names (separated by semicolons). If you are adding a user that is from the same Active Directory domain that the server is running on, you can type the AD user name without the domain. The server’s domain will be assumed.
Note: Do not enter the user’s full name in this field; it can cause errors during the importing process.

3. Select a site role. For details on site roles, see Set Users’ Site Roles.

4. Click Import Users.

Set Users’ Site Roles

When you add users to a site on Tableau Server, you must apply a site role to them. The site role determines which users or groups can publish, interact with, or only view published content, or who can manage the site’s users and administer the site itself.

Note: Tableau Server site roles do not correspond to user licenses that you purchase from Tableau (if you are using user-based licensing instead of core-based server licensing). Those licenses allow a certain number of users on the server.
In this article

Who can publish content

Capabilities the site role determines

General capabilities allowed with each site role

Change a user’s site role

Site roles and Active Directory import and synchronization

Who can publish content

The following site roles allow publishing content:

- Server Administrator (Tableau Server only)
- Site Administrator
- Publisher

Users with a site role of Interactor, Viewer, and Unlicensed cannot publish content to the server.

Capabilities the site role determines

The site role is set at the user level (that is, not on content), to cap what a user can do with content site-wide. For example, a person whose site role is Interactor can never publish content from Tableau Desktop, even if a project’s permission rule assigns the Publisher permissions role to a group the user is a member of.

Site roles also determine who can configure the site or server itself. For example only a Site Administrator or Server Administrator can manage users.
General capabilities allowed with each site role

- **Server Administrator** (Tableau Server only): The server administrator has full access to all server and site functionality, all content on the server, and all users.

- **Site Administrator**: Site administrators can manage groups, projects, workbooks, and data sources (including connection information) for the site.

On Tableau Server, server administrators can determine whether to not allow site administrators to manage users and assign site roles and site membership. By default on Tableau Server, and always on Tableau Online, site administrators are allowed these capabilities.

Site administrators also have unrestricted access to the site’s content. A user can be a site administrator on multiple sites.

- **Publisher**: Publishers can sign in, browse the server, and interact with the published views. They also can connect to Tableau Server from Tableau Desktop to publish (upload) and download workbooks and data sources. Publishers cannot manage users.

- **Interactor**: Interactors can sign in, browse the server, and interact with the published views. They are not allowed to publish to the server.

- **Viewer**: Viewers can sign in and see published views on the server but cannot use any interaction features like filtering and sorting. Users with this site role can be allowed only to view and add and view comments.

- **Unlicensed**: Unlicensed users cannot sign in to the server. Users are assigned the Unlicensed role in the following circumstances:
  
  - You import users from a CSV file.
  
  - The number of available licenses is reached at the time you add or import users.
• You remove a user who owns content on the site. The user will still own the content but not be able to do anything with it.

See also How Permissions are Evaluated.

Change a user’s site role

By default both server and site administrators can manage user settings. On Tableau Server, server administrators can configure a site to disallow site administrators the ability to manage users.

1. Sign in to the site, and go to the **Users** area.

2. Select the users, and then select **Actions > Site Role**.

3. Select the new site role, and then click **Change Site Role**.
Site Role

Choose a site role for user “Andrew Smith”.

Publisher
Server Administrator
Site Administrator
Publisher
Interactor
Viewer
Unlicensed

Change Site Role

You can hover the pointer over the information icon to display a matrix that shows the maximum level of general capabilities each site role allows.

<table>
<thead>
<tr>
<th>Site role</th>
<th>Web Access</th>
<th>Interact/Edit</th>
<th>Publish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Publisher</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Interactor</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Viewer</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unlicensed</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Server administrators can manage Server and manage/create Sites
Site administrators can manage a Site

Site roles and Active Directory import and synchronization

When you import Active Directory users to a site, you can specify the site role. If a user is not yet a member of any site on the server, the user is added to the site with the assigned role.
When you synchronize Active Directory groups, the site role is applied through the **Minimum Site Role** setting on the **Groups - Details** page.

**Note:** If you are running Tableau Server on Linux then all external directory communication is configured and managed with a LDAP identity store. In the context of user and group synchronization, Tableau Server configured with LDAP identity store is equivalent to Active Directory. Active Directory synchronization features in Tableau Server function seamlessly with properly configured LDAP directory solutions.

If a user already exists in a Tableau Server site, the site role assigned during the import or sync process will be applied if it gives the user more access in a site. Importing or synchronizing users and groups will promote a user’s site role, but not demote a user’s site role.

If a user already has the ability to publish, that ability is maintained.

To guarantee a user maintains a site role with equal or greater capabilities in server after an import, the following matrix shows the rules applied for site roles on import.

**Note:** The **Import Site Role** row abbreviated headers indicate the site role specified for import. The **Current Site Role** column headers represent the current user site role. The table values represent the abbreviated resulting site role.

- Site Administrator: SA
- Publisher: P
- Interactor: I
- Unlicensed: U

<table>
<thead>
<tr>
<th>Import Site Role</th>
<th>SA</th>
<th>P</th>
<th>I</th>
<th>V</th>
<th>U</th>
</tr>
</thead>
<tbody>
<tr>
<td>Site Administrator</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
</tr>
</tbody>
</table>
### Assign Site Membership

You can set a site so that both server and site administrators can add or remove users on that site and change their site roles. At the All Sites level, server administrators can manage the sites existing users can sign in to (site membership), and their site role on each.

1. In the site menu, click **Manage All Sites**, and then click **Users**.

2. Select one or more users, and then select **Actions > Site Membership**.

<table>
<thead>
<tr>
<th>Import Site Role</th>
<th>Current Site Role</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SA</td>
</tr>
<tr>
<td>(SA)</td>
<td></td>
</tr>
<tr>
<td>Publisher</td>
<td>SA</td>
</tr>
<tr>
<td>(P)</td>
<td></td>
</tr>
<tr>
<td>Interactor</td>
<td>SA</td>
</tr>
<tr>
<td>(I)</td>
<td></td>
</tr>
<tr>
<td>Viewer</td>
<td>SA</td>
</tr>
<tr>
<td>(V)</td>
<td></td>
</tr>
<tr>
<td>Unlicensed</td>
<td>SA</td>
</tr>
<tr>
<td>(U)</td>
<td></td>
</tr>
</tbody>
</table>
3. Select one or more sites, and a role for each site, and then click **Save**.
Guest User

Core-based licenses of Tableau Server include a Guest user option, which you can use to allow people who don’t have an account on the server to see and interact with Tableau views embedded in web pages.

Guest user access is enabled by default when Tableau Server is installed with a core-based license. If you do not intend to use Guest user access, then you should disable it.

Guest access allows users only to see embedded views. The Guest user cannot browse the Tableau Server interface or see server interface elements in the view, such as user name, account settings, comments, and so on.

**Note:** Enabling the Guest user for a site can increase the number of potential simultaneous viewers beyond the user list you might be expecting. The administrative view Status > Traffic to Views can help you gauge the activity.

Guest user permissions

A Guest user can have the following permissions:

- **Projects, Workbooks, and Views**: View, Export Image, Summary Data, View Comments, Filter, Full Data, Web Edit, Download (to save a local copy)

- **Data Sources**: View and Download

When a Guest user is included in a group that has a permission rule set on a content item, Guest user permissions do not affect the permission-levels of other users in that group.

Enable or Disable Guest access

You must be a server administrator to change Guest account settings at either the server or the site level. Guest access is enabled by default in core-based licensing installation.
1. In the site menu, click **Manage All Sites** and then click **Settings > General**.

2. For **Guest Access**, select or clear **Enable Guest account** to toggle Guest user access.

3. Click **Save**.

This enables the Guest user on all sites. You can then go to the same setting for a specific site. To disallow Guest access for a site:

1. In the site menu, select a site.

2. Click **Settings**, and on the Settings page, clear the **Enable Guest account** check box.

If the Guest account is enabled on some or all sites, and you turn off Guest access at the server level, it is turned off for all sites as well.

**Additional Guest account characteristics**

The Guest user is unique in the following additional ways:

- As a single user account, it represents all unauthenticated users accessing embedded Tableau views.

- You cannot edit it or select it as the owner of a content resource.

- You can add it as a member of groups on a site.

- When enabled, it is a member of the All Users group.

- You cannot delete the account; however, you can turn off access to it by clearing the check box described in the steps above.

- If the Guest user needs to access a workbook with an extract connection, the Guest must also have the **View** capability on the published data source.

  The Guest user is not allowed to connect to published data sources.
• It is not allowed to save custom views.

• Guest cannot be used in a user filter.

**Import Users**

To automate the process of adding users to a site, you can create a CSV file that contains user information, and then import the file. Site administrators can import users to a particular site; server administrators (Tableau Server only) can import users at the server level, to add them to multiple sites.

**Note:** This topic contains the steps for importing, assuming that you have already created the CSV file. If you have not created the file yet, see CSV Import File Guidelines for a list of file format requirements and import options.

**Add users from a CSV file**

The following steps describe how to add users to a site or to the server. The images reflect adding users at the site level.
1. Do one of the following:

- To add users at the site level, select **Users**, and then **Add Users**.

![Add Users to this Site](image)

- To add users at the server level on a **single-site** server, select **Users**, and then **Add Users**.

- To add users at the server level on a **multi-site** server, open the list of sites, and select **Manage All Sites**. Select **Users**, and then **Add Users**.

2. Click **Import From File**, click **Browse** and navigate to the file, and then click **Import Users**.

![Import users from file](image)

Tableau displays the results of the import process (names in this image are blurred).
3. Click **Done**.

**How users’ site roles are assigned or maintained**

When you import at the site level or on a single-site server, you can include site-role assignments in the .csv file. If a user already exists in the Tableau Server site, the site role assigned during the import process will be applied only if it gives the user more access. Importing users and groups will promote a user’s site role but not demote it.

For a multi-site server, when you import users in the Server Users page, you create server users with no site affiliation. Because these users do not belong to a site, they cannot have a site role. The only site role a user can have at the server level is **Unlicensed** or **Server Administrator**.

When you assign site membership to a user, you can specify the user’s site role for each site. For information, see **Assign Site Membership**.

**Importing at the server level in multi-site environments**

If the server is running multiple sites and you are a server administrator, you can import a CSV file from two locations. Where existing user accounts are concerned, each location has
different capabilities.

- The **Server Users** page appears in a multi-site environment. Only server administrators can access this page.

![Server Users page](image)

You can import the CSV file from here if you want to update existing user accounts in addition to adding new ones. For example, if you import a file that has a new password for each existing user, their passwords will be reset.

- The **Site Users** page.

![Site Users page](image)

Server administrators can add new user accounts with CSV imports. If the CSV file includes existing users, the **Password** and **Display Name** fields must either match the existing or be left blank. If new passwords or full names are used, the import will fail.
Importing to a single-site environment

Server and site administrators on a single-site server perform CSV user imports from the Users page in a site.

Multi-site versus single-site import

Users can belong to more than one site on the same server, but they must use the same credentials for each site. This becomes important when you’re adding users to a site and those users might already be members of a different site. If you try to import a user who already exists, and if the user’s credentials in the CSV file don't match the existing credentials, the import fails for that user.

Note: The issue of credentials mismatch during import doesn't apply if the server is configured to use Active Directory for authentication. In that case, the CSV file should never contain a password, because user passwords are managed by Active Directory.

If you’re importing users to a site and think that the users might already exist on the server, you can try leaving the Password column in the CSV file blank. When you import the users, if a user who is defined in the CSV already exists in another site, the user is added to the site where you’re importing. However, if the user doesn’t already exist on the server, the user is created, and the CSV import window alerts you that the new user doesn’t have a password.
You can then use the server administrator pages to assign a password to any user who doesn’t have one.

**CSV Import File Guidelines**

You can automate adding users by creating a comma-separated values (CSV) file with user information and then importing the file. You can include attributes in the CSV file, such as site role and the ability to publish, to apply to the users at the same time you import them.

To import users, you can use the server or site administration pages or the `tabcmd utility`. For details, see Import Users or `createsiteusers filename.csv`.

You can import users at the site or server level. If you import users to a site, site roles are applied to the user. If you specify site roles, but importing users would exceed your license limits, users are imported as Unlicensed. If you import users to the server (not into a specific site), the user isn’t assigned to a site, and site roles in the CSV file, such as Publisher and Interactor, are ignored, and the Unlicensed site role is assigned during the import.

**CSV file format requirements**

When you create the CSV file for importing users, make sure that the file meets the following formatting requirements:

- The file does not include column headings. Tableau Server assumes that every line in the file represents a user.

- The file is in UTF-8 format, and includes the byte-order mark (BOM).

- Character encodings such as BIG-5 have been converted to UTF-8. You can do this by opening the file in a text editor and using the **Save As** command.

- If a name includes the "@" character other than as a domain separator, you need to refer to the symbol using the hex format: `\0x40`

  For example, `user@fremont@myco.com` should be `user-\0x40fremont@myco.com`
Required Columns in the CSV File

The following values are required for each user:

- User name

- Password: If Tableau Server is configured to use Active Directory authentication, there must be a Password column, but the column itself should be empty. If the server is using local authentication, you must provide passwords for new users.

Additional import file options

The CSV file can contain the following fields, in the order shown here:

- User name. The user name. If the server is configured to use Active Directory, this value must match a user defined in Active Directory. If the user name is not unique across domains, you must include the domain as part of the user name (for example, example\Adam or adam@example). This is the only required field.

- Password. A password for the user. If the server is configured to use Active Directory, this value is not used.

- Display name. The display name is part of the information that's used to identify a user on the server. If the user's display name is already in use, Tableau Server updates the existing user information with the settings in the CSV file. If the server is configured using Active Directory, this value is not used.

- Site role. The site role determines the maximum permissions a user can have on content published to the site. Site roles include Site Administrator, Publisher, Interactor, or Viewer, and a couple of others for publishing only. For more information, see Set Users' Site Roles.
Note: Tableau versions earlier than 9.0 use license level instead of site roles. For more information, go to the Tableau Help page and select the documentation specific to your version.

- Administrator level (System, Site, or None). This setting determines whether the user is imported as an administrator. If you are using the site administration pages, you can set the administrator role to System only if you are importing while managing the server. If you are using the server administration pages to import users while you are managing a site, and if the administrator role for a user in the CSV file is set to System, Tableau Server imports the user as a site administrator.

- Publisher permissions (yes/true/1 or no/false/0). This setting determines whether the user has publisher permissions. If you are using the site administration pages, the publisher setting is used only if you are importing into an individual site. If you are importing users while managing a server, this value isn’t used.

- Email address. The email address is part of the information that’s used to identify a user on the server. If the email address is already in use, Tableau Server updates the existing user information with the settings in the CSV file.

The order of the columns is significant. The first column is treated as the user name, the second as the password, the third as display name, etc., regardless of the content in the columns.

Settings and site roles

The license level, administrator, and publisher settings for a user are used during the import process to set a user’s site role. The following table shows how the settings are converted to site roles.

<table>
<thead>
<tr>
<th>CSV settings</th>
<th>Site role</th>
</tr>
</thead>
<tbody>
<tr>
<td>License level=(any)</td>
<td>System (server) administrator. This setting is valid only if you are importing users while managing the</td>
</tr>
<tr>
<td>Administrator=System</td>
<td></td>
</tr>
<tr>
<td>CSV settings</td>
<td>Site role</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Publisher=true</strong></td>
<td>server.</td>
</tr>
<tr>
<td><strong>License level=(any)</strong></td>
<td>Site administrator. This setting is valid only if you are importing users while managing a specific site.</td>
</tr>
<tr>
<td><strong>Administrator=Site</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Publisher=true</strong></td>
<td></td>
</tr>
<tr>
<td><strong>License level=Interactor</strong></td>
<td>Publisher</td>
</tr>
<tr>
<td><strong>Administrator=None</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Publisher=true</strong></td>
<td></td>
</tr>
<tr>
<td><strong>License level=Interactor</strong></td>
<td>Interactor</td>
</tr>
<tr>
<td><strong>Administrator=None</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Publisher=false</strong></td>
<td></td>
</tr>
<tr>
<td><strong>License level=Viewer</strong></td>
<td>Viewer</td>
</tr>
<tr>
<td><strong>Administrator=None</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Publisher=false</strong></td>
<td></td>
</tr>
<tr>
<td><strong>License level=Unlicensed</strong></td>
<td>Unlicensed</td>
</tr>
<tr>
<td><strong>Administrator=None</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Publisher=false</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Notes**

- If you are importing users while managing the server, you can create users with only two site roles: system (server) administrator and Unlicensed. All other settings are site specific. In that case, if the administrator level for a user in the CSV file is not System, the user's site role is set to Unlicensed.

- If you have a user-based server installation, and if adding users would exceed the number of users allowed by your license, the users are added as unlicensed users.

**Example**

The following example shows a CSV file that contains information for several users.
henryw, passw0rd, Henry Wilson, Interactor, None, yes, henryw@example.com
freds, pa$$word, Fred Suzuki, Viewer, None, no, freds@example.com
alanw, p@ssword, Alan Wang, Interactor, Site, yes, alanw@example.com
michellek, mypassword, Michelle Kim, Interactor, System, yes, michellek@example.com

If you import this file while managing a site, four users are added to that site. The Administrator mode for user Michelle is set to System. However, because you are importing the users into a site, Tableau Server sets user Michelle to be a site administrator, not a system administrator. Three of the users are allowed to publish.

If you import this file while managing the server, four users are added to the server, but they are not added to any site. The site roles in the CSV file (Interactor and Viewer) must be associated with site users, so the site role for the users who are not administrators is set to Unlicensed.

**Set the User Authentication Type for SAML**

On a site that has been configured for site-specific SAML, administrators can specify users’ authentication type. For example, if Tableau Server was configured for site-specific SAML and server-wide SAML, administrators can specify which users authenticate with site-specific SAML and which users authenticate with server-wide SAML.

You can assign authentication type at the time you add users to Tableau Server, as well as any time afterward.

1. When you’re signed in to the Tableau Server site, select **Users**.

2. On the **Site Users** page, select the check boxes next to the users you want to assign an authentication type.

3. On the **Actions** menu, select **Authentication**.
4. In the Authentication dialog box, select **Site SAML** or **Server Default**.

**Notes**

- Users that authenticate with site-specific SAML can only belong to one site. If a user needs to belong to multiple sites, set their authentication type to the server default. Depending on how site-specific SAML was configured by the server administrator, the server default is either local authentication or server-wide SAML.

- If you change users’ authentication to site-specific SAML, the next time they sign in, they will be directed to your identity provider’s site to provide their credentials.

**View, Edit, and Delete Users**

**View and edit site users**

Sign in to a site as an administrator, and then click **Users**. In this page you can set group membership, set site role, or remove the user from the site.
Click a user name to see the content they own.

Click **Settings** in a user page to view their account settings. The user **Settings** page is available when the user is a member only of sites that the site administrator also controls, and site administrators are allowed to manage users in the site settings.
If Tableau Server is running multiple sites, **Server Users** lists all users on the server system, and **Site Users** displays all users for the current site.

If the server is configured to use the internal user management system (Local Authentication), you can edit the **Display Name, Email**, and **Password** for users after they have been added. If you are making many changes, you may find it easier to import the changes from a CSV file. For details, see Import Users and CSV Import File Guidelines.

**For multi-site servers:** Site administrators can edit an existing user’s account as long as the user is a member only of sites that the site administrator also controls, and site administrators are allowed to manage users in the site settings. For example, if User Joe is a member of Site A and Site B and the site administrator is only an administrator of Site B, the site administrator cannot edit Joe’s Full Name or reset his password.

**View and edit server users**

Sign into Tableau Server as a server administrator. On the site menu, click **Manage All**
Sites, and then click Users. In this page you can set site membership or delete the user from the server.

![Server Users](image)

Click a user name to view account settings. The user Settings page is available when the user is a member only of sites that the site administrator also controls, and site administrators are allowed to manage users in the site settings.

![Andrew Smith](image)
Search for users

To search for a specific user, in the Search box on the left, type all or part of the user’s name, and then press Enter.

The search operation checks the display name and user name attributes.

You can use the asterisk (*) character as a search wildcard. For example, searching for John* will return all user names that start with John.

Remove users from a site

You can remove a user from a site only if the user does not own any content (projects, workbooks, views, or data sources). If you attempt to remove a user who owns content, the user site role will be set to Unlicensed, but not removed.

Note: When a site administrator removes a user from a site (and the user only belongs to that one site), the user will be automatically deleted from the server if that user doesn't own any content.

1. In a site, click Users. Select one or more users to delete, and then select Actions > Remove.
Delete users from the server

You can delete a user from Tableau Server only if the user does not own any content (projects, workbooks, views, or data sources). If you attempt to delete a user who owns content, the user site role will be set to Unlicensed, but the user will not be deleted.

If a user is a member of multiple sites, and owns content in one or more of those sites, the user will be removed from the sites in which they don’t own content. The user will remain a member in sites where they do own content, but demoted to the Unlicensed site role.

1. In the site menu, click Manage All Sites, and then click Users. In a single-site environment, click Users.

   Select one or more users to delete, and then click Actions > Delete.

2. Click Remove in the confirmation dialog.
2. Click **Delete** in the confirmation dialog box.

**Change passwords for users of a single site**

To change the password for a user with membership to a single site, sign in to Tableau Server as a site administrator or a server administrator.

1. Ensure that the correct site is selected in the menu.
2. Click **Users**.
3. Click the display name of a user.
4. Click **Settings**.
5. Click the **Change Password** link, edit the password, and then click **Save Password**.

Change passwords for users of multiple sites

To change the password of a user with membership to multiple sites, sign in to Tableau Server as a server administrator.

1. In the site menu, click **Manage All Sites**.

2. Click **Users**.

3. Click the display name of a user.
4. Click the **Change Password** link, edit the password, and then click **Save Password**.

![Tableau Server Users](image)

**Groups**

**Add Users to a Group**

You can organize Tableau Server users into groups to make it easier to manage multiple users. You can create groups on the server or import groups from Active Directory.

**Note**: If you are running Tableau Server on Linux then all external directory communication is configured and managed with a LDAP identity store. In the context of user and group synchronization, Tableau Server configured with LDAP identity store is equivalent to Active Directory. Active Directory synchronization features in Tableau Server function seamlessly with properly configured LDAP directory solutions.

To keep Active Directory group membership up-to-date:
• Site administrators can synchronize selected groups on demand in a site. For more information, see *Synchronize Active Directory Groups on a Site*.

• Server administrators can synchronize all Active Directory groups on the server based on a schedule or on-demand. For more information, see *Synchronize All Active Directory Groups on the Server*.

To add a user to a group, the group must already exist.

**Add users to a group (Users page)**

1. In a site, click **Users**.

2. Select the users you want to add to a group, and then click **Actions > Group Membership**.

3. Select the groups and then click **Save**.
Add users to a group (Groups page)

1. In a site, click **Groups**, and then click the name of the group.

2. In the group’s page, click **Add Users**.

3. Select the users to be added, and then click **Add Users**.

Create a Local Group

Local groups are created using the Tableau Server internal user management system. After you create a group you can add and remove users.
1. In a site, click **Groups**, and then click **New Group**.

2. Type a name for the group and click **Create**.

Create Groups via Active Directory

You can import Active Directory groups to create matching groups on Tableau Server, as well as a user for each member of an Active Directory group that is not already on the server.

**Note:** If you are running Tableau Server on Linux then all external directory communication is configured and managed with a LDAP identity store. In the context of user and group synchronization, Tableau Server configured with LDAP identity store is equivalent to Active Directory. Active Directory synchronization features in Tableau Server function seamlessly with properly configured LDAP directory solutions.

Each user is assigned a site role as part of the import process. If any of the users to be imported exist in Tableau Server, the site role assigned during the import process is applied only if it gives the user more access to the server. Importing users does not demote site roles.
Before importing groups, review User Management in Active Directory Deployments to understand how multiple domains, domain naming, NetBIOS, and Active Directory user name format influence Tableau user management.

1. In a site, click **Groups**, and then click **Add Groups**

2. Type the name of the Active Directory group you want to import, and then select the group name in the resulting list.

3. Select the site role for the users.
4. Click **Import**.

**Note:** You cannot change the name of groups imported from Active Directory. The group name can only be changed in Active Directory.

**Synchronize Active Directory Groups in a Site**

At any time, you can synchronize an Active Directory group with Tableau Server to ensure new users in Active Directory are also added in Tableau Server. You can synchronize individual groups or multiple groups at once.

**Note:** If you are running Tableau Server on Linux then all external directory communication is configured and managed with a LDAP identity store. In the context of user
and group synchronization, Tableau Server configured with LDAP identity store is equivalent to Active Directory. Active Directory synchronization features in Tableau Server function seamlessly with properly configured LDAP directory solutions.

1. In a site, click **Groups**.

   On the Groups page, select one or more groups.

2. Click **Actions > Synchronize**.

   ![Groups page screenshot](image)

   Set the minimum site role for users in an Active Directory group

   In the **Groups - Details** page, administrators can set the minimum site role for group users to apply during synchronization.

   This setting does not run synchronization; it sets the minimum site role to applied to the group every time synchronization runs. When you synchronize Active Directory groups, new users are added to the site with the minimum site role. If a user already exists, the minimum site role will be applied if it gives the user more access in a site. If you don’t set a minimum site role, new users are added as **Unlicensed** by default.
Note: A user’s site role can be promoted but never demoted based on the minimum site role setting. If a user already has the ability to publish, that ability will always be maintained. For more information on minimum site role, see Site roles and Active Directory import and synchronization.

1. In a site, click Groups.

2. On the Groups page, select a group.

   Click Actions > Minimum Site Role.
3. Select the minimum site role, and then click **Change Site Role**.

What happens when users are removed in the source Active Directory?

Users cannot be automatically removed from the Tableau Server through an Active Directory sync operation. Users that are disabled, deleted, or removed from groups in Active Directory remain on Tableau Server so that administrators can audit and reassign the user's content before removing the user's account completely. For more information, see Sync behavior when removing users from Active Directory.

What happens when a user name changes in the source Active Directory

By default, Tableau Server will not synchronize changes to the user display name after the initial synchronization when the corresponding account is created in Tableau Server. For example, if the user name jsmith is used for the display name John Smith, changing the display name in Active Directory to Joe Smith will not synchronize to the corresponding jsmith user in Tableau Server.

To change this behavior run the following tsm commands:

```shell
tsm configuration set -k vizportal.adsync.update_system_user -v true
```
The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the -r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

What happens when an Active Directory group is removed from Tableau Server?

Many Tableau administrators use Active Directory groups to import and create users. After the users are imported into Tableau Server, administrators will then delete the group in Tableau Server. Deleting a group does not delete the users in it.

Synchronize All Active Directory Groups on the Server

As a server administrator, you can synchronize all Active Directory groups on a regular schedule or on-demand on the General tab of the Settings page for the server.

Note: If you are running Tableau Server on Linux then all external directory communication is configured and managed with a LDAP identity store. In the context of user and group synchronization, Tableau Server configured with LDAP identity store is equivalent to Active Directory. Active Directory synchronization features in Tableau Server function seamlessly with properly configured LDAP directory solutions.
The Last synchronized time indicates the time that synchronization most recently began.

Synchronize Active Directory groups on a schedule

1. **Single-site**: Click Settings > General.

   **Multisite**: In the site menu, click Manage All Sites and then click Settings > General.

2. Scroll down the page to Active Directory Synchronization, and then select Synchronize Active Directory groups on a regular schedule.
3. Select the frequency and time of synchronization.

4. Click **Save**.

**Synchronize all Active Directory groups on demand**

At any time, you can synchronize Active Directory groups with Tableau Server to ensure that new users and changes in Active Directory are reflected in all Active Directory groups on Tableau Server.

1. **Single-site**: Click **Settings > General**.

   **Multisite**: In the site menu, click **Manage All Sites**, and then click **Settings > General**.
2. Under **Active Directory Synchronization**, click **Synchronize All Groups**.

View synchronization activity

You can view the results of synchronization jobs in the **Background Tasks for Non Extracts** administrative view. **Queue Active Directory Groups Sync** is the task that queues and indicates the number of **Sync Active Directory Group** tasks to be run.

1. **Single-site**: Click **Status**.

**Multisite**: In the site menu, click **Manage All Sites** and then click **Status**.

2. Click the **Background Tasks for Non Extracts** link.

3. Set the **Task** filter to include **Queue Active Directory Groups Sync** and **Sync Active Directory Group**.

You can quickly navigate to this administrative view by clicking the **View synchronization activity** link in the **Settings** page for the server.

Set the minimum site role for users in an Active Directory group

In the **Groups - Details** page, you can set the minimum site role for group users to be applied during Active Directory synchronization.
This setting does not run synchronization; instead, it sets the minimum site role to applied to the group every time synchronization runs. The result is that when you synchronize Active Directory groups, new users are added to the site with the minimum site role. If a user already exists, the minimum site role is applied if it gives the user more access in a site. If you don’t set a minimum site role, new users are added as Unlicensed by default.

**Note:** A user’s site role can be promoted but never demoted based on the minimum site role setting. If a user already has the ability to publish, that ability will always be maintained. For more information on minimum site role, see Site roles and Active Directory import and synchronization.

1. In a site, click **Groups**.

2. On the Groups page, select a group.

   Click **Actions > Minimum Site Role**.

3. Select the minimum site role, and then click **Change Site Role**.

![Minimum Site Role](image)
What happens when users are removed in the source Active Directory?

Users cannot be automatically removed from the Tableau Server through an Active Directory sync operation. Users that are disabled, deleted, or removed from groups in Active Directory remain on Tableau Server so that administrators can audit and reassign the user’s content before removing the user’s account completely. For more information, see Sync behavior when removing users from Active Directory.

Quick Start: Synchronize All Active Directory Groups on a Schedule

After you import Active Directory groups in Tableau Server, you can make sure they stay synchronized in Tableau Server by setting up a schedule. You can also synchronize all Active Directory groups on the server on-demand, at any time. The minimum site role setting for the group is applied when users are synchronized.

**Note:** If you are running Tableau Server on Linux then all external directory communication is configured and managed with a LDAP identity store. In the context of user and group synchronization, Tableau Server configured with LDAP identity store is equivalent to Active Directory. Active Directory synchronization features in Tableau Server function seamlessly with properly configured LDAP directory solutions.

1. Set a minimum site role for synchronization

   In a site, click **Groups**. Select a group, and then click **Actions > Minimum Site Role**. Select the minimum site role, and then click **Change Site Role**. Server and site administrators can set the minimum site role for group users to be applied during Active Directory synchronization. If you don't set a minimum site role, new users are added as **Unlicensed**.
Synchronizing can promote a user's site role, but will never demote a user's site role.

2 Set the schedule

Server administrators can enable synchronization for all Active Directory groups on the General tab of the Settings page for the server. Enable synchronization, select the frequency settings, and then click Save.
All Active Directory groups on the server are synchronized according to the same schedule.

3 Run synchronization on-demand (optional)

On the General tab of the Settings page, click Synchronize All Groups to synchronize all Active Directory groups on Tableau Server immediately. Click this button at any time to ensure new users and changes are reflected in all Active Directory groups on the server.
Click **Synchronize All Groups** to synchronize all Active Directory groups on the server outside of a schedule.

4 View the status of synchronization tasks

Server and site administrators can view the results of Active Directory synchronization jobs in the **Background Tasks for Non Extracts** administrative view. On the server or in a site, click **Status**. Under **Analysis**, click **Background Tasks for Non Extracts** and filter on the **Queue Active Directory Groups Sync** and **Sync Active Directory Group** tasks.

![Background Tasks for Non Extracts](image)

**Queue Active Directory Groups Sync** queues the **Sync Active Directory Group** tasks to be run.

**Delete Groups**

You can delete any group from the server (with the exception of the All Users group). When you delete a group, the users are removed from the group but they are not deleted from the server.
1. In a site, click **Groups**.

2. On the Groups page, select one or more groups to delete.

3. Select **Actions > Delete**.

![Image of TableauGroups page](image)

**Dashboard-based Custom Portals**

**Note:** This overview was inspired by the work of Tableau Zen Master Mark Jackson. For more details about the process, check out Mark's blog.

The standard Tableau Online or Tableau Server interface works great for many organizations, but if you want to create a company-branded experience and don't have API developer skills, consider creating a custom portal based on a Tableau dashboard. A custom portal lets you organize content around specific departments or job roles, and you can even incorporate training that helps people interact with data more fully. As the volume of Tableau content grows, a custom portal guides your users directly to the data they need, while provid-
ing ready access to related views and Tableau Server search so they can easily explore further.

Sketch out a portal design

Begin outside of Tableau, on paper or in a wireframing application. Consider the structure of your organization and the number of Tableau views that apply to each group of users. Do you simply need one level of navigation that points immediately to content? Or might you need to start with a main page that provides navigational links for separate groups of users or report types, followed by a second level with related dashboards and views?

<table>
<thead>
<tr>
<th>Medical Staff</th>
<th>Illness trend report</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Patient demographics</td>
</tr>
<tr>
<td>Facilities</td>
<td>Hospital occupancy</td>
</tr>
<tr>
<td></td>
<td>Real estate transactions</td>
</tr>
</tbody>
</table>

After you get buy-in on your design from key stakeholders and data consumers, you’re ready to move on to the next phase.

Gather images for logos and navigation elements

Start thinking about images while refining your initial mockup, and then gather them from libraries of clip-art or approved brand graphics, or create them from scratch in an application like Photoshop or SnagIt. PowerPoint can also be a good source if you plan to create navigation thumbnails of common chart types.
Lay out text, images, and selected sheets on a dashboard

Create a workbook with a dashboard for the portal, using a tiled layout for more predictable positioning and scaling of elements on different devices. Then start adding Text and Image objects, as well as any sheets for data views you want to highlight right up front in your portal. To refine spacing between these dashboard elements, insert Blank objects and adjust their size. For more information, see Create a Dashboard.

**Tip:** For dynamic elements that show popular or recently created content, include sheets based on custom administrative views.

Link dashboard elements to content

If you have a second level of navigation in your portal, use filter actions to point to a secondary dashboard from the main one. To create links that directly open data views, right-click Image objects and choose Set URL. (In our example below, each colored arrow and accompanying text is an image that links to a view URL.) You can even point to empty views with preloaded data sources, encouraging users to create new Tableau content in the web-authoring workspace.

**Tip:** To add hyperlinks to text objects, include the full URL (for example, http://www.tableau.com).
### Publish, test, and refine your portal

Publish the workbook to your server and distribute the dashboard URL to your users. Portal design, like data analysis itself, is a cyclical process. Now that your portal is out in the wild, start gathering user feedback so you can continuously improve the experience.
Tip: As a finishing touch, hide the Tableau toolbar to give your portal a custom feel. After the question mark at the end of the dashboard URL, add `:embed-d=y&;toolbar=n`

Data Connection Authentication

The topics in this section describe how to configure authenticated connections to various data sources.

More information

- Tableau Server on Linux - Connecting to a Windows Shared Directory (Tableau Community)
- Setting an Oracle Connection to Use TNSNames.ora or LDAP.ora (Tableau Support)

Enable Kerberos Service Account Access

You can configure Tableau Server to use a service account to access a database on behalf of users on Tableau Desktop. In this scenario, Tableau Server connects to databases with a service account (also referred to as a "RunAs service account") in an Active Directory environment. Workbooks and data sources that use this service account are published as "Authentication: Server RunAs account" in Tableau Desktop.

Enabling Kerberos service account access is a requirement for Kerberos delegation. See Enable Kerberos Delegation.

Requirements

Kerberos service account access requires Active Directory.
• The Tableau Server information store must be configured to use LDAP - Active Directory.
• The Linux computer where Tableau Server is installed must be joined to Active Directory domain.
• MIT Kerberos is not supported.
• RunAs service account: This is an Active Directory user account that is used as the security principal to connect to the database on behalf of Tableau Server. The RunAs service account must have read access to the target database. The RunAs service account must be a user account in the Windows Active Directory domain. If your users are in a different Active Directory domain than Tableau Server and the data source, then domain trust must be configured. See Domain Trust Requirements.

Configuration process

This section provides an example of the process to enable Kerberos service account access. The scenario also includes example names to help describe the relationships between the configuration elements.

1. Create a domain user account to act as the RunAs service account. This account must have read access to the target database.

   In the example here, the RunAs service account is User principal named tab-srv@example.com.

2. Create a keytab file for the RunAs service account.

   For example, the following commands create a keytab (tabsrv-runas.keytab) using the ktutil tool:

   ```
   sudo ktutil
   ktutil: addent -password -p tabsrv@example.com -k 2 -e <encryption scheme>
   ```

   Encryption schemes for this command include RC4-HMAC, aes128-cts-hmac-shal-96, and aes256-cts-hmac-shal-96. Consult your IT team for the correct encryption scheme for your environment and data source.
ktutil: wkt tabsrv-runas.keytab

Tableau Server will use the RunAs service account and the associated keytab to authenticate and make a direct connection to the database.

3. Copy the keytab into the Tableau Server data directory and set proper ownership and permissions.

   mkdir /var/opt/tableau/tableau_server/keytab
   sudo cp -p tabsrv-runas.keytab /var/opt/tableau/tableau_server/keytab
   sudo chown $USER /var/opt/tableau/tableau_server-/keytab/tabsrv-runas.keytab
   chgrp tableau /var/opt/tableau/tableau_server-/keytab/tabsrv-runas.keytab
   chmod g+r /var/opt/tableau/tableau_server/keytab/tabsrv-runas.keytab

4. Run the following TSM commands to enable RunAs access, set the RunAs service account, and associate the keytab file with the service account:

   tsm configuration set -k features.RunAsAuthLinux -v true
   tsm configuration set -k native_api.datasource_runas_principal -v tabsrv@example.com
   tsm configuration set -k native_api.datasource_runas_keytab_path -v /var/opt/tableau/tableau_server-/keytab/tabsrv-runas.keytab

5. Run the following TSM command apply the changes to Tableau Server:

   tsm pending-changes apply

The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt
using the –r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

Enable Kerberos Delegation

Kerberos delegation enables Tableau Server to use the Kerberos credentials of the viewer of a workbook or view to execute a query on behalf of the viewer. This is useful in the following situations:

- You need to know who is accessing the data (the viewer’s name will appear in the access logs for the data source).
- Your data source has row-level security, where different users have access to different rows.

Requirements

To enable Kerberos delegation, you must first configure Kerberos service account access. See Enable Kerberos Service Account Access.

Kerberos delegation requires Active Directory.

- The Tableau Server information store must be configured to use LDAP - Active Directory.
- The Linux computer where Tableau Server is installed must be joined to Active Directory domain.
- MIT Kerberos is not supported.
- Delegation service account: Grant delegation rights for this account to the target database Service Principal Names (SPNs). This account is delegated authority to access resources on behalf of the initiating source user. This account must be a user account in the Windows Active Directory domain. If your users are in a different Active Directory domain than Tableau Server and the data source, then domain trust must be configured. See Domain Trust Requirements.
Configuration process

This section provides an example of the process to enable Kerberos delegation. The scenario also includes example names to help describe the relationships between the configuration elements.

1. Enable Kerberos Service Account Access

2. Tableau Server will need a Kerberos service ticket to delegate on behalf of the user that is initiating the call to the database. You must create a domain account that will be used to delegate to the given database. In this example, the user configured as the delegation account is `delegate_user_account`.

If you are operating in Active Directory, then the delegation account must be configured with Active Directory User and Computers:

   - Open the Properties page for the delegated user, click the Delegation tab and select Trust this user for delegation to specified services only and Use any authentication protocol.

3. Set service principal names (SPN) in your directory to associate the Tableau Server as a Service principal with the delegation account. The following example shows how to set the SPNs and how to create a single keytab file (kerberos.keytab) for both delegation and SSO functionality.

   The following commands set the SPN and create a keytab file. Running these commands require administrative privilege on the Kerberos key distribution center (KDC).

   ```bash
   setspn -s HTTP/tableau-server EXAMPLE\delegate_user_account
   setspn -s HTTP/tableau-server.example.com EXAMPLE\delegate_user_account
   ktpass /out kerberos.keytab /mapuser delegate_user_account@EXAMPLE.COM /princ HTTP/tableau-server.example.com@EXAMPLE.COM /pass * /crypto ALL
   ```

4. Run the following TSM commands to enable Kerberos delegation, set the delegation
service account, and associate the keytab file with the service account:

```bash
tsm configuration set -k features.LinuxMITKerberos -v true
 tsm configuration set -k native_api.datasource_impersonation_runas_principal -v HTTP/tableau-server.example.com@EXAMPLE.COM
 tsm configuration set -k native_api.datasource_impersonation_runas_keytab_path -v /var/opt/tableau/tableau_server/keytab/kerberos.keytab
```

5. Run the following TSM command apply the changes to Tableau Server:

```bash
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

6. (Optional) If you are operating in an environment with multiple domains then you will need to configure krb5.conf to map principal names to local user names for each Kerberos realm. See Kerberos delegation multi-domain configuration.

7. Enable delegation for data connections:

- **SQL Server**—See [Enabling Kerberos Delegation for SQL Server](https://community.tableau.com) in the Tableau Community.

- **MSAS**—See [Enabling Kerberos Delegation for MSAS](https://community.tableau.com) in the Tableau Community.

- **PostgreSQL**—See [Enabling Kerberos Delegation for PostgreSQL](https://community.tableau.com) in the Tableau Community.
• **Teradata**—See [Enabling Kerberos Delegation for Teradata](#) in the Tableau Community.

**OAuth Connections**

For Google BigQuery, Google Analytics, Salesforce, OneDrive, Dropbox, and QuickBooks Online, an alternative to storing your sensitive database credentials with Tableau Server is to create connections using the **OAuth 2.0** standard.

From Tableau, when you sign in to data with a provider that uses OAuth, you are redirected to the provider’s sign-in page. After you provide your credentials and authorize Tableau to access your data, the data provider sends Tableau an **access token** that uniquely identifies requests from Tableau. For more information, see Overview of the OAuth process below.

Using OAuth connections provides the following benefits:

- **Security**: Your database credentials are never known to or stored in Tableau Server, and the access token can be used only by Tableau.

- **Convenience**: Instead of having to embed your data source ID and password in multiple places, you can use the token provided for a particular data provider for all published workbooks and data sources that access that data provider.

  In addition, for live connections to Google BigQuery data, each workbook viewer can have a unique access token that identifies the user, rather than sharing a single user name and password credential.

**Overview of the OAuth process**

The following steps describe a workflow in the Tableau environment that calls the OAuth process.

1. You take an action that requires access to a cloud data source.

   For example, you open a workbook that’s published to Tableau Server.
2. Tableau directs you to the cloud data provider’s sign-in page. The information that is sent to the data provider identifies Tableau as the requesting site.

3. When you sign in to the data, the provider prompts you to confirm your authorization for Tableau Server to access the data.

4. Upon your confirmation, the data provider sends an access token back to Tableau Server.

5. Tableau Server presents your workbook and data to you.

The following workflows can use the OAuth process:

- Creating a workbook and connecting to the data source from Tableau Desktop or from Tableau Server.

- Publishing a data source from Tableau Desktop.

- Signing in to Tableau Server from an approved client, such as Tableau Mobile or Tableau Desktop.
Access tokens for data connections

You can embed credentials based on access tokens with data connections, to enable direct access after the initial authentication process. An access token is valid until a Tableau Server user deletes it, or the data provider revokes it.

It is possible to exceed the number of access tokens your data source provider allows. If that’s the case, when a user creates a new token, the data provider uses length of time since last access to decide which token to invalidate to make room for the new one.

Access tokens for authentication from approved clients

By default, Tableau Server sites allow users to access their sites directly from approved Tableau clients, after users provide their credentials the first time they sign in. This type of authentication also uses OAuth access tokens to store the users' credentials securely.

For more information, see Disable Automatic Client Authentication

Configure the Server for OAuth Support

Instead of individual usernames and passwords, OAuth works through limited-purpose access tokens. Before you can obtain access tokens needed to create an OAuth connection in Tableau, you need to configure your server so that the data provider sending the token can recognize Tableau Server as a trusted destination. The following section describes how to prepare for setting up OAuth regardless of your data provider. The topics listed below it contain the steps for configuring your server for specific data providers.

Preparing for Configuring OAuth Support

Before you begin the configuration steps specific to your data provider, complete the following prerequisites:

- Obtain the fully qualified domain name of each Tableau Server node that will host views that connect to this data source. For example:

  https://sales.your_domain.com
If you use Salesforce.com, you will need to provide an https address.

- Make sure at least one of your data-provider accounts is enabled for API access.

  For **Google** data types, you need access to the developers console on the Google Cloud Platform.

  For **Salesforce.com**, you need access to the Force.com platform.

  For **QuickBooks Online**, you need access to the Intuit platform.

**Configure Settings for Your Data Provider**

When you complete the OAuth-preparation steps, you can configure the appropriate settings with your data provider.

- Set up OAuth for Google
- Set up OAuth for Salesforce.com
- Set up OAuth for QuickBooks Online

**Set up OAuth for Google**

This topic describes how to set up your Google BigQuery, Google Sheets, and Google Analytics data sources for OAuth. Complete these steps for each Tableau Server instance.

*Note* Before you complete these steps, make sure you have completed the prerequisites described in Preparing for Configuring OAuth Support.

Set up OAuth by following these two procedures:

- Get required information from Google and enable API access.
- Use the information you obtained to configure your server.
Obtain a Client ID and Enable Google APIs

**Note** These steps reflect the settings in the Google Cloud Platform console at the time of this writing. For more information, see [Using OAuth 2.0 for Web Server Applications](#) in the Google Developers Console Help.

1. Sign in to [Google Cloud Platform](#), and then click **Go to my console**.

2. On the drop-down menu next to the Google Cloud Platform title, select **Create project**.

3. In the new project form that appears, complete the following:
   - Give the project a meaningful name that reflects the Tableau Server instance for which you’ll use this project.
   - Determine whether you want to change the project ID.

   **Note** After you create the project, you will not be able to change the project ID. For information, click the question mark icons.

4. Open the new project, and navigate to **APIs Manager > Credentials**.

5. Click the **OAuth consent screen** tab and then enter a meaningful name for Product
name shown to users.

6. Click the **Credentials** tab, then click the **Create credentials** drop-down list, and then select **OAuth client ID**. Complete the following:

   - Select **Web Application**.

   - For **Authorized JavaScript Origins**, enter the local computer name of your Tableau Server.

   - For **Authorized Redirect URI**, replace the existing text with the Internet address for your server, and add the following text to the end of it: `auth/add_oauth_token`. For example:

     ```
     https://your_server_url.com/auth/add_oauth_token
     ```

7. Copy the **Authorized Redirect URI**, and paste it in a location that you can access from your Tableau Server computer.

8. Click **Create**.

9. Copy the following values that Google returns, and paste them in a location that you can access from your Tableau Server computer:

   - Client ID

   - Client secret

10. In **APIs Manager > Dashboards**, verify that **BigQuery API, Google Drive API** (to enable Google Sheets) or **Analytics API** is enabled. To enable APIs, click **ENABLE API** at the top of the page.

Configure Tableau Server for Google OAuth

Using the information you obtained by completing the steps in Obtain a Client ID and Enable Google APIs, configure your Tableau Server:
- On the Tableau Server computer, open the bash shell and run the following commands:

```bash
tsm configuration set -k oauth.google.client_id -v <your_client_ID>

tsm configuration set -k oauth.google.client_secret -v <your_client_secret>

tsm configuration set -k oauth.google.redirect_uri -v <your_authorized_redirect_URI>

tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Managing access tokens

After you configure the server for OAuth, you can allow users to manage their own access tokens in their profile settings, or you can manage the tokens centrally. For more information, see Allow Saved Access Tokens.

Set up OAuth for Salesforce.com

This topic describes how to set up your Salesforce.com data sources for OAuth. Complete these steps for each Tableau Server instance.

**Note:** Before you complete these steps, make sure you have completed the prerequisites described in Preparing for Configuring OAuth Support.

Set up OAuth by following these two procedures:
• Create a Connected App in Salesforce
• Use the information you obtained to configure your server.

Create a Connected Salesforce App

1. Sign in to your Salesforce.com developer account, click your user name in the upper-right, and then select Setup.

2. In the left navigation column, under App Setup, select Create > Apps.

3. In the Connected Apps section, click New.

4. Complete the Basic Information, and in the API section, select Enable OAuth Settings.

5. In the new OAuth settings that appear, for Callback URL, type the fully qualified domain name of your server, using the https protocol, and append the following text to
the URL: auth/add_oauth_token.

For example:

https://www.your_server.com/auth/add_oauth_token

6. Move the following items from Available OAuth Scopes to Selected OAuth Scopes:

   - Access and manage your data (api)
   - Access your basic information (id)
   - Perform requests on your behalf at any time (refresh_token)

7. Click Save.

After you save the app, Salesforce populates the API section with the following IDs that you will use to configure Tableau Server:

   - Consumer Key
   - Consumer Secret
   - Callback URL

Configure Tableau Server for Salesforce.com OAuth

1. On the Tableau Server computer, open the bash shell and run the following commands:

   tsm configuration set -k oauth.salesforce.client_id -v
   <your_client_ID>
2. (Optional) To change the default login server, type the following command:
   
   tsm configuration set -k oauth.salesforce.server_base_url -v <URL>

3. Enter the following command to apply changes:
   
   tsm pending-changes apply

   The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the -r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

Managing access tokens

After you configure the server for OAuth, you can allow users to manage their own access tokens in their profile settings, or you can manage the tokens centrally. For more information, see Allow Saved Access Tokens.

Set up OAuth for QuickBooks Online

This topic describes how to set up your QuickBooks Online data sources for OAuth authentication. Complete these steps for each Tableau Server instance.

Note: Before you complete these steps, make sure you have completed the pre-requisites described in Preparing for Configuring OAuth Support.

Setting up OAuth for QuickBooks Online consists of two tasks:
- Create a Connected App on the Intuit developer platform.
- Use the information you get as part of the Connected App to configure your server.

Create a Connected Intuit App

1. Sign in to your Intuit developer account, and then click **My Apps**.
2. In the **Just start coding** section, click **Select APIs**.
3. Select **Accounting** and click **Create App**.
4. In the **Get your app ready for submission** section, click the link to get your production keys.

**Important:** You must use production keys rather than development keys.

5. Copy the app token, OAuth consumer key, and OAuth consumer secret.

Configure Tableau Server for QuickBooks Online

- On the Tableau Server computer, open the bash shell and run the following commands:

  ```
  tsm configuration set -k oauth.quickbooks.oauth_callback_uri -v http://YOUR-SERVER/auth/add_oauth_token
  tsm configuration set -k oauth.quickbooks.consumer_key -v <your_consumer_key>
  tsm configuration set -k oauth.quickbooks.consumer_secret -v <your_consumer_secret>
  tsm pending-changes apply
  ```

  The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the
server is stopped, but in that case there is no restart. You can suppress the prompt using the –r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

Managing access tokens

If you run an extract refresh job for your QuickBooks Online data source, Tableau Server attempts to renew access tokens for you. To help ensure that your access tokens do not expire, run your extract refresh jobs more than once a month. Otherwise, the access tokens from QuickBooks Online expire and your extract refresh jobs fail. If your access tokens do expire, you can edit your saved credentials from the Settings page.

The saved credentials can be managed centrally or by your users. For more information, see Allow Saved Access Tokens.

Allow Saved Access Tokens

After you configure Tableau Server for OAuth, you can decide to allow users to manage their own OAuth credentials, or you want to manage them centrally. If you want users to manage their own, you need to enable user profile settings from the server.

**Note:** If you have not yet configured your server to enable OAuth data connections, see the related topics listed below.

1. Sign in to Tableau Server as a server administrator.

2. **Single-site:** Click Settings > General.

   **Multisite:** In the site menu, click Manage All Sites and then click Settings > General.

3. In the Saved Credentials section, select the following:
• **Allow users to save passwords for data sources** (allows users to save their individual credentials with data sources).

• **Allow users to save OAuth access tokens for data sources**

4. Click **Save**.

After you select these check boxes, users will see a **Manage Credentials** section in their profile settings, where they can add access tokens for OAuth data connections.

Managing credentials centrally

Server administrators alternatively can manage OAuth credentials centrally. This can work well, for example, if multiple users work from the same data, and you have a dedicated user account for your data provider.

To manage credentials centrally, you do the following:
• Clear the check boxes described in the preceding procedure.

• Edit connection information as data sources are published.

When you edit the connection, you embed credentials that use an OAuth access token instead of an individual’s user name and password.

When the settings for saving passwords and access tokens are not enabled, the Manage Credentials section is excluded from users’ profile settings.

See also

Set up OAuth for Google

Set up OAuth for Salesforce.com

Set up OAuth for QuickBooks Online

**Configure SAP HANA SAML delegation**

You can configure Tableau Server to use SAML delegation to provide a single sign-on (SSO) experience for SAP HANA. This scenario is not dependent on SAML authentication to Tableau Server. You do not need to use SAML sign on with Tableau Server in order to use HANA SAML delegation. You can sign in to Tableau Server using whatever method you choose.

With SAML delegation for SAP HANA, Tableau Server functions as an identity provider (IdP).

**Before you begin**

Configuring SAML delegation with SAP HANA requires configuration on both Tableau Server and on SAP HANA. This topic provides configuration information about configuring Tableau Server. Before you configure Tableau Server, you must complete the following:

• Acquire a SAML certificate and key file for Tableau Server. The certificate file must be a PEM-encoded x509 certificate with the file extension .crt or .cert. This file is used by
Tableau Server, and must also be installed on HANA. The private key must be a DER-encoded private key file that is not password protected, and that has the file extension .der. This file is only used by Tableau Server.

- Install the certificate in HANA.
- Install the latest version of SAP HANA driver (minimum version is 1.00.9) on Tableau Server.
- Configure network encryption from Tableau Server to SAP HANA (recommended).

For more information about generating the certificate/key pair, encrypting the SAML connection, and configuring SAP HANA, see How to Configure SAP HANA for SAML SSO with Tableau Server in the Tableau Community.

Configure Tableau Server SAML for SAP HANA

The following procedure describes how to configure SAML for SAP HANA on Tableau Server using tsm data-access set-saml-delegation. You can also configure SAML for SAP HANA using the sapHanaSettings Entity.

If you are running Tableau Server in a distributed deployment, then you may need to copy the certificate and key files to other nodes in the cluster. See Copying Files in a Distributed Deployment.

1. Place certificate files in a folder named saml. The saml folder should be in the /var/opt/tableau hierarchy. For example:

   /var/opt/tableau/tableau_server/data/saml

2. Run the following command to enable SAML delegation and to specify the location of the certificate and key files:

   tsm data-access set-saml-delegation --enabled enabled --cert-key <cert-key> --cert-file <cert-file>

   Where <cert-key> and <cert-file> are file paths to the private key and certificate file, respectively.

   For example,
tsm data-access set-saml-delegation --enabled enabled --cert-key /var/opt/tableau/tableau_server/data/saml/saml_key.der --cert-file /var/opt/tableau/tableau_server/data/saml/saml_cert.crt

You can specify other options. For example, you can specify user name format and how credentials are normalized. See tsm data-access set-saml-delegation.

3. When you have finished, run `tsm pending-changes apply`.

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Manage Content Access

Set a Site’s Web Authoring Access

Tableau Server administrators can specify at the site level whether to allow users to edit published views in the web environment.

By default web authoring functionality is enabled for all sites. Users with the Web Edit capability to edit workbooks directly on the server. Turn off web authoring if you want users to be able to view and interact with published workbooks but not make any changes to the core information.

The steps below describe how to set web authoring for an entire site. For more granular control over which users can use web editing, you can use projects, groups, and permissions. See Set Web Edit, Save, and Download Access on Content.
Turn web authoring on or off for a site

1. In a web browser, sign in to the server as an administrator and go to the site in which you want web authoring to be enabled. In that site, click **Settings**.

2. Select **Allow users to use web authoring** to enable the functionality.

   Clear the check box to turn off web authoring for that site.

3. If your site is already in production, and you want the change to take effect immediately, restart the server.

   Otherwise, the change takes effect after server session caching expires or the next time users sign in after signing out.

Notes

- When you enable web authoring, make sure that, on the appropriate workbooks or views, the permission rule for a user or group allows the **Web Edit** capability.

- If you turn off web authoring on a production site and do not complete the last step to restart the server, users might continue to have authoring access until their session caches expire or they sign out.

See which sites allow web authoring

To confirm which sites allow web authoring, on the site-selection menu at the top, select **Manage All Sites**, and then go to the **Sites** page.
Set Web Edit, Save, and Download Access on Content

If you’re enabling web authoring functionality on your site, you can configure more precisely which users on the site have access to this functionality. Using site roles and permissions rules at the content level, you can grant or deny Web edit, Save, or Download capabilities on projects, workbooks, and data sources.

Note: This document strives to use the phrase Web edit only to specify the name of the capability in permissions rules, and web authoring to refer to the general functionality of creating and modifying workbooks on the server. However, you might otherwise see these two phrases used interchangeably.

Why allow users to work on the server directly

As an administrator, your initial thought about allowing people to populate a site with content, seemingly indiscriminately, might be one of skepticism. However, with a few controls, you can limit where this is done, while providing important benefits that centralized content management offers both you and your users.

Web authoring pros and cons

For publishers and business users, some benefits of web authoring include the following:
- It provides analyst teams who work collaboratively with a central location in which to provide input.

- It enables people who do not have Tableau Desktop to connect to data sources and create workbooks.

- It enables people to access content when they are away from their Tableau Desktop computer or VPN, whether on a computer or a hand-held device.

- It can provide a framework for enabling consistency across Tableau reports.

  By making template workbooks available on the server, analysts can download or create new workbooks with data connections, branding, and formatting already in place.

For administrators, benefits can include the following:

- Fewer Tableau Desktop deployments to manage and support.

- Fewer computers that need to have database drivers installed.

- Capacity to govern content.

- More accurate monitoring of what people are doing with Tableau.

Some disadvantages to web editing include the following:

- For analysts, web editing functionality is not as extensive as in Tableau Desktop (although it continues to evolve toward that parity).

- For administrators, more people working on the server might mean upgrading systems.

- Without publishing guidelines, content proliferation on the server is expected.

  This can confuse the people who rely on published Tableau dashboards and data sources, degrade server performance and data quality, and potentially affect data security.
Managing permissions to help users avoid content proliferation

To help users to avoid content proliferation on the server, many Tableau administrators use projects to allow varying levels of access to content. For example, one project can allow all users to edit and save workbooks; another can allow only approved publishers to save new content.

To get a better idea how this works, see the following resources:

- Configure Projects, Groups, and Permissions for Managed Self-Service
- Projects and Content Permissions in Everybody’s Install Guide
- Governed Self-Service at Scale, a Tableau whitepaper by Rupali Jain.

To view the PDF, you might need to provide your Tableau website credentials. These are the same ones you use for the community forums or to submit support cases.

Coordinate edit and save capabilities with site roles for the appropriate level of access

To edit, save, and download workbooks, users must have a site role that allows those actions, along with the capabilities—defined in permissions rules—that grant or deny editing-related access.

Site role access

- When the appropriate permissions are set at the content level, the Publisher site role allows both Save (overwrite) and Save As/Download.

Publishers can save (overwrite) only workbooks that they own. If a publisher edits a workbook owned by another user, only the Save As command is available, and they can save the workbook with a new name.

- The Interactor site role allows downloading but not saving (neither overwriting existing nor saving changes to a new workbook).
Capabilities set on content

These capabilities coordinate with the site role to allow levels of editing and saving.

- **Web Edit** determines whether users can edit workbooks directly on the server.

- **Download Workbook/Save As** determines whether users have access to the **Save As** option while they are editing a workbook, or can download a workbook as a Tableau packaged workbook (.twbx).

  **Save As** allows users to save changes they’ve made to an existing workbook as a new workbook on the server. It also determines whether they can open a workbook on the server from Tableau Desktop.

  Tableau treats saving like publishing, so to save new workbooks, users must have a site role of **Publisher**.

- **Save** determines whether users can save changes to an existing workbook on the server (overwrite a workbook).

  To overwrite a workbook with changes made on the server, a user must have a site role of **Publisher**.

**Configure site roles and permissions**

Use the tables in the following sections to coordinate the site role with saving and editing capabilities.

Set these capabilities at the project level, using permissions rules.

Allow publishers to edit, save changes to, and download existing and new workbooks.

Apply these capabilities to users who have the **Publisher** site role.

<table>
<thead>
<tr>
<th>Capability</th>
<th>For the project</th>
<th>For specified workbooks in the project</th>
</tr>
</thead>
</table>
**Web Edit**  |  -  |  Allowed  
**Download Workbook/Save As**  |  -  |  Allowed  
**Save**  |  Allowed  |  Allowed

Allow publishers to edit, save changes to, and download new workbooks, but not overwrite existing workbooks

Apply these capabilities to users who have the **Publisher** site role.

<table>
<thead>
<tr>
<th>Capability</th>
<th>For the project</th>
<th>For specific workbooks in the project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Edit</td>
<td>-</td>
<td>Allowed</td>
</tr>
<tr>
<td>Download Workbook/Save As</td>
<td>-</td>
<td>Allowed</td>
</tr>
<tr>
<td>Save</td>
<td>Allowed</td>
<td>Denied</td>
</tr>
</tbody>
</table>

**Important:** In the “save but not overwrite” case above, allowing Save access at the project level is required for publishers to be able to publish workbooks from Tableau Desktop. To deny overwriting existing workbooks, you must deny the **Save** capability explicitly on each relevant workbook, and not lock permissions to the project. If project permissions are locked, permissions set on the project apply to all workbooks in it.

Allow interactors to download workbooks

Apply these capabilities to users who have the **Interactor** site role.

<table>
<thead>
<tr>
<th>Capability</th>
<th>For the project</th>
<th>For specific workbooks in the project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Edit</td>
<td>-</td>
<td>Allowed</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>Download Workbook/Save As</td>
<td>-</td>
<td>Allowed</td>
</tr>
<tr>
<td>Save</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

The **Save** and **Save As** capabilities will be denied by the Interactor site role, even if you allow them on the content.

**See also**

Set Permissions on Individual Content Resources

Set Users' Site Roles

Permissions Reference

**Configure Projects, Groups, and Permissions for Managed Self-Service**

Tableau Online and Tableau Server each provide an environment for easy open publishing and collaborative analysis of visualizations created in Tableau Desktop or web authoring. With that flexibility comes the challenge of making sure the *right* content is easy to find for the people who rely on it for their work. Likewise, making sure the access you allow doesn’t create performance or management nightmares on the site.

To address these challenges, many administrators set up their Tableau sites for what we’ll refer to as managed self-service. This is just a way of saying that the site allows areas of open collaboration and web editing, alongside areas in which access to data and reports is more controlled. As the site administrator, you put guidelines in place to help users figure out where to go for the type of work they need to do.

To get started with a managed self-service approach, the following sections discuss how you as the site administrator can meet the following objectives:
Create projects on the Tableau Server or Tableau Online site to match the ways people need to work with content.

For example, some projects are open to all for collaboration; others are visible only to authorized publishers.

Create user groups based on the type of access users need to the content.

Create a clear and scalable permissions strategy.

Note: The information provided here is adapted and simplified from practices of existing Tableau Zen Masters and customers who have shared their experiences. Links to their talks are available at the bottom of this page.

Start by creating a project team and adopting a permissions strategy

We recommend that you recruit users from various segments of your Tableau population, to create a project team of people who have differing uses for Tableau content.

Your permissions strategy will help your environment scale as you add new Tableau users. Make sure it incorporates two important practices: manage permissions only for groups, and set permissions only at the project level. Setting permissions at the individual user level and on individual content resources becomes unmanageable quickly. If you need to deviate from this practice, make sure you document and communicate your strategy to other administrators and project leaders.

Steps to coordinate projects and groups

To get projects and permissions (content) to work together with groups (people) in a managed self-service environment, you generally take the following steps:

1. Plan your permissions: Find common themes in the type of access users need. This helps determine projects and groups.

2. Remove permissions that will cause ambiguities
3. Create groups

4. Assign permissions to the groups

5. Create projects and adjust permissions

6. Lock permissions in each project

If you decide to follow the guidelines described here, you might want to Automate working with groups and projects.

1. Plan your permissions

Before you create groups and start assigning permissions, create a list of people who need access to content, and arrange them in groups according to what they’ll want to do.

For example, someone who publishes or moves a data source to a certified content project would need different level of access than someone who only consumes published reports. (We use the term “certified” to mean “trusted” — these are the data sources or reports that your Tableau community can trust to be a source of truth for your organization.)

Keep in mind also that you can set permissions differently for each project. So someone who is a data steward for the Ops department might not get the equivalent access to the Marketing content.

This exercise, done outside of the Tableau environment, can be the most challenging part of setting up a site.

Use a closed permissions model for managed content

General models for setting permissions are open or closed. In an open model, users get a high level of access, and you explicitly deny capabilities. This model can work when your organization is very small, and everyone has a similar level of responsibility.

In a closed model, users get only the access they need to do their jobs. This is the model security professionals advocate, and the examples in this article will attempt to show.
2. Remove permissions that will cause ambiguities

Every site has a **Default** project and an **All Users** group. The Default project works as a template for new projects in the site. Creating groups and setting baseline permissions here helps you to know and manage exactly who gets what level of access for each new project.

In the managed self-service context, setting baseline permissions means *removing* the permissions from the **All Users** group, so that the permissions are enabled only on groups you create and have control over.

1. In the menu at the top of the page, select **Projects**.

2. Open the permissions for the **Default** project. Open the **Actions** menu (…), and then select **Permissions**.

3. Next to the **All Users** group name, select . . ., and then select **Edit**.
4. In the drop-down lists under Project, Workbooks, and Data Sources, select None.

5. Select Delete to apply the changes.

3. Create groups

You create groups to match what people need to do with a set of content. In this case “a set of content” refers to the workbooks and data sources in a project.

When you create your groups, use descriptive names that make sense for your organization. For example, one possible set of groups might be as follows:

- **Project leaders.** Users who can perform all available capabilities on data sources, with the possible exception of setting permissions on them.

  People in this group can be site administrators, or publishers whose job it is to approve or certify data models or reports.

- **Analysts/Publishers.** This group is for users who can publish workbooks to
production and other open projects, use web editing on some projects, and connect to data sources certified by the data stewards. This group is not allowed to set permissions on content or move it between projects.

- **Business Users.** This group is the most likely to include people who do not use Tableau Desktop, but use data to answer questions and make business decisions. They can view and interact with workbooks only in specific projects, and they can’t publish, edit, save, or delete anything.

- **Administrators.** Depending on the size of your deployment, managing site or server administrators as a group helps you keep track of who has that level of access.

**Note:** Users with the Server Administrator or Site Administrator site role have access to everything on the site, regardless of the groups you add them to.

If you have multiple roles per department, creating corresponding groups manually can be labor intensive. For alternatives, see Automate working with groups and projects later in this article.

**Learn more:** Add Users to a Group

4. Assign permissions to the groups

After you create groups, you can assign permissions in one of the following ways:

- In the Default project, apply a core set of permissions on each group that will stay more or less the same for all projects. You can then make minor adjustments in specific projects.

  Or

- Keep the Default project clean and apply permissions only on projects you create.
For the example we’re using, it makes more sense to set permissions templates in the Default project. You will want to explicitly deny some capabilities across the board, and then allow them on only a few projects where you want to allow more open access.

Set permissions templates

1. While you have the Default project open, on the **Actions** menu (...), select **Permissions**.

   The **Permissions** pane shows only the All Users group that has no permissions.

2. Create a permission rule for each group as follows:
   a. Select **Add a user or group rule**, and then select one of your groups.

      This adds the group to the **User/Group** column, open for editing.

   b. Select a permission role in the **Project**, **Workbook**, and **Data Source** columns.

      Permission roles are predefined sets of capabilities that make setup easier.

   c. Refine permissions in any of the columns by selecting the expand icon (++) to display individual capabilities and set them explicitly.

3. **Lock permissions to the project.**

Example

For the groups defined in 3. Create groups, here is one way you might set default permissions.

**Project leaders roles**

- Project: **Project Leader**
- Workbooks: **Editor**
- Data Sources: **Editor**
This gives site administrators and data stewards full access to a project and its content. If you’re an IT admin, this enables you to delegate Tableau content administration to people who are closer to that content.

**Analyst Publisher roles**

- Project: **Publisher**
- Workbooks: **Editor**
- Data Sources: **Connector**

**Business Users roles**

- Project: **Viewer**
- Workbooks: **Interactor**
- Data Sources: **Denied**

**Default project settings for individual capabilities**

- Under Workbooks, set **Web Edit** and **Download Full Data** to **Deny**.

  This assumes you want to allow web editing and downloading data only on select projects. When you create those projects, you can refine the permissions.

- If you want to put more than a couple of users in the Project leaders group for each project, consider denying the **Set Permissions** capability for that group. An alternative for delegating the task of setting permissions is to set individual users’ site roles to Site Administrator instead of Publisher.

- Leave capabilities in the Edit category set to **None**.

- If you want to allow other capabilities only as an exception, set those to **Deny** here as well.
5. Create projects and adjust permissions

After the Default project is set with your custom permissions template, you can create projects that allow the content use cases you identified. For each project, you can adjust the default permissions as appropriate.

Example project structure

One way to structure projects could be to reflect the following use cases:

**Workbooks shared for open collaboration on the server**

Anyone in the department can publish to the open-collaboration project while their content is in development. Colleagues can collaborate using web editing on the server. Some people call this a sandbox, some call it staging, and so on. On this project you can allow web editing, saving, downloading, and so on.

Here you want not only to enable collaboration, but also to enable people who don’t have Tableau Desktop to contribute and provide feedback.

**Shared reports that cannot be edited**

This could be a project that people who create workbooks and data sources (Analysts and Data Stewards) could publish to when they want to make content available to business users for viewing, with confidence that their work cannot be “borrowed” or modified.

For this type of project, you would deny all capabilities that allow editing or getting the data off of the server for reuse. You would allow viewing and interacting capabilities.
Vetted data sources for Analysts to connect to

This would be where Data Stewards publish the data sources that are meet all of your data requirements and become the “source of truth” for your organization. Project leaders on this project can certify these data sources, so that they rank higher in search results and are included in recommended data sources.

You would allow authorized Analysts (that is, the Publishers group described earlier) to connect their workbooks to data sources in this project, but not download or edit them. You would deny capabilities to the Business Users group, so those users would not even see this project.

Inactive content

Another possibility is to segregate workbooks and data sources that the site’s administrative views show haven’t been used for a period of time. You could give content owners a time limit before their content is removed from the server.

Whether you do this or delete directly from the working projects is up to your organization. In an active environment, don’t be afraid to be intentional about removing content that is not being used.

Source for workbook templates

This is a project that people can download from but not publish or save to, where authorized publishers or project leaders make template workbooks available. Templates that have your organization’s approved fonts, colors, images, and even data connections built in can save authors a lot of time and keep your reports looking consistent.

Help project leaders manage content and users find it

- Devise a scalable project-naming scheme that makes sense in your organization.

  For example, basic structure might be `<Department> - <ContentUse>`; such as `Ops - Production`.

- Use the project’s Description field.
The description you enter when you create a project appears on the project thumbnail, as well as the project-specific Details tab.

6. Lock permissions in each project

After you refine the capabilities for each group in a project, you can lock the project's permissions. Do this on the Default project, too.

1. With a project’s **Permissions** page open, select the button next to **Permissions for workbooks and data sources** are
2. In the dialog box that appears, select **Locked to the project**.

Locking permissions prevents publishers from setting permissions explicitly as part of the publishing process in Tableau Desktop. Instead, content inherits permissions set on the project it’s published to, and only administrators and project leaders can set permissions.

**Automate working with groups and projects**

Creating multiple groups and projects and setting permissions manually can get tedious. To automate these processes, as well as make them repeatable for future updates, you can perform these tasks using **REST API** or **tabcmd** commands.

**Next steps**

Besides projects, groups, and permissions, other data governance themes include:

**User education**

Help **all** of your Tableau users become good data stewards. The most successful Tableau organizations create Tableau user groups, have regular training sessions, and so on.
For a common approach to orienting users to the site, see Dashboard-based Custom Portals.

For publishing and data certification tips, see the following topics:

- Use Certification to Help Users Find Trusted Data
- Prepare for Publishing a Workbook (links to Tableau Help)
- Best Practices for Published Data Sources (links to Tableau Help)

Optimize extract refresh and subscription activity

If you use Tableau Server, create policies for extract refresh and subscription schedules, to avoid them dominating the site’s resources. The TC customer presentations by Wells Fargo and Sprint address this subject in detail. In addition, see the topics under Tuning & Operations.

If you use Tableau Online, see the following topics to become familiar with the ways people can refresh extracts:

- Keep Data Fresh
- Use Tableau Bridge to Expand Data Freshness Options

Monitoring

Use administrative views to keep an eye on the site’s performance and content use.

Administrative Views

Learn how other Tableau customers address governance and self-service.

The following list contains links to data governance presentations that Tableau customers gave at the Tableau Conference in Austin, TX, November 2016.

Server Admins: Don’t Fear Web Authoring—Robb Yeager of Sprint
Building a COE at Wells Fargo, Next Level Governance at Large Scale—Angie Greenhaw and Brian Mooneyham of Wells Fargo

Maintaining Governance in a Self-Service Environment—Mark Jackson of Piedmont Healthcare

To get access to these and related free talks, you complete a simple registration form on the TCLive website.

Use Projects to Manage Content Access

When Tableau Desktop users publish a workbook or data source to a site on Tableau Server, they can select a project to publish it to. If they don’t specify a project, their content is published to the Default project.

As an administrator, you can create projects to hold and organize related content resources, or to delegate content management. Content resources on Tableau Server are workbooks, views, and data sources, and the projects that hold them.

In this article

Why use projects
Project-level administration
How the Default project acts as a permissions template
Prevent publishers from changing permissions on content they own
Next steps

Why use projects

Projects help you to create a scalable process for managing access to the content published to Tableau Server. Advantages they have include:
They enable administrators to delegate content management to project leaders who work with the content more closely, without having to give them administrator access to site or server settings.

Project leaders can create nested projects under their top-level project, enabling them to maintain their team’s content within a single hierarchy.

They can make the site easier to navigate for self-service users.

They segment the Tableau Server site into areas that give users access based on how they use the data published to those areas, or on the Tableau user group they work with.

You can hide projects from groups who don’t need to use them, create a distinguishable project-naming scheme, and take advantage of project descriptions to clarify how to use the project.

They enable you to track permissions effectively.

You can create groups based on the level of content access users in the group need, and set default permissions on projects, so you know exactly which capabilities new users get by default, and likewise which capabilities all users get when a new project is created.

When to create project hierarchies (example)

Many organizations have several or more distinct groups of Tableau users, each with its own priorities and leaders. These groups might share some organization-wide content (or even draw from an org-wide pool of data sources), but primarily they use data and reports that are specific to their team. In this or similar scenario, an example for using project hierarchies might look as follows:

1. You, as a site or server administrator, can create top-level projects for each of your distinct Tableau teams.

2. On each top-level project, you assign the Project Leader role to team leads, and
change project ownership.

Project leaders effectively are the content administrators, so it’s important that they understand how permissions work in Tableau, along with Tableau content management best practices.

3. Each project leader takes over their project, creating the structure within the project that works for their team.

That is, they can create child projects they need, based on how their team members collaborate and share data and reports.

The benefit to you as the site administrator is that you can focus on system health. The benefit to your Tableau users is that people who know the best practices for working with Tableau and data can manage these things for their teams, without having to submit IT requests to change permissions or add projects.

Why not use sites?

If you manage your own Tableau Server deployment, you can create as many sites as you want. However, for managing data and reports across your company, projects allow the flexibility you need to administer shared data and reports, and users who might belong to multiple groups. Many Tableau administrators configure projects as described in the previous section, to expose only what’s necessary to users who need to work with it. Projects work better than sites for evolving content from development to staging to production.

Sites work well when content can remain completely separate during all phases, and there is little to no user overlap. A good (and common) example for using multiple sites is to create a site for each of multiple external clients, whose published content you manage as a consultant or vendor. Our own Tableau Online is an example of this on a large scale. Another example might be to use a separate site for sensitive content that you want only specific Human Resources or medical staff to use.
Project-level administration

As a server or site administrator, you can delegate administration of projects and their content, without exposing access to your site or server settings. You can do this by changing the ownership of a project or granting a group or user the Project Leader permissions role on a project.

The **project owner** is always one individual user. By default, the user who creates a project is its owner. The project owner has administrative access to the project and content in it—including making someone else the owner and assigning Project Leader permissions.

The **Project Leader** permissions role provides a way to allow multiple users administrative access to a project, its child projects, and all workbooks and data sources in those projects. A project leader does not have to be an administrator or project owner. You can assign Project Leader permissions to users with a site role of Interactor or Publisher.

Project ownership and project leader access in project hierarchies

In a multi-level project hierarchy, a user or group that is given the Project Leader permissions role to a project, at any level within the hierarchy, is implicitly given Project Leader access to all of that project’s child projects and their content items.

To remove the Project Leader access, you must do so at the parent level in the hierarchy on which the ownership or role was explicitly assigned.

Similarly, the owner of a project at any level has project leader access to all content in that project, as well as to any of its child projects, even if they do not own the child projects.

Only a project owner or administrator can change ownership of a content resource, and this can be done regardless of whether the project permissions are locked at the top level.

Actions project-level administrators can take on projects

In addition to clarification points in the following list, server or site administrators can perform all of these actions.
• Create and delete projects as follows:

Server or site administrators can create or delete top-level or nested projects anywhere on the site.

Project owners and project leaders can create and delete child (nested) projects in projects they own, or on which they have Project Leader permissions.

• Project owners can change ownership of their projects. Both project owners and project leaders can assign the Project Leader permissions to groups or users.

• Set permissions for a project, as well as the child projects, workbooks, and data sources in it.

• Lock permissions to apply the top-level project’s default settings to all workbooks, data sources, and child projects and their content. Permissions can be modified only from the project on which they’re locked.

In a locked project hierarchy, only the owner or project leader of the top-level project in the hierarchy can change permissions.

• Move workbooks and data sources to another project to which they have Project Leader or owner access.

Moving the project can affect permissions. For information, see Move a project or project hierarchy.

• Run, add, or remove extract refresh schedules.

See also Permissions in Project Hierarchies

How the Default project acts as a permissions template

Tableau creates a Default project with every site.
The Default project serves as a template for new top-level projects you create on the site. When you create a new top-level project, settings and permissions from the Default template are applied to the new project, including permissions set on the project's workbooks and data sources.

**Note:** Nested projects (projects you create within other projects) take the permissions set at their parent project, not the Default project.

Before you create other top-level projects, you can take steps to set up the Default project, to help you to know exactly:

- Which type of user gets what level of access for each new project.

- How you might need to modify permissions for each new project.

For more information, see Steps to coordinate projects and groups in the topic Configure Projects, Groups, and Permissions for Managed Self-Service.
Prevent publishers from changing permissions on content they own

Administrators and project leaders can prevent users from changing the permissions for workbooks and data sources in a project hierarchy. For example, you can disable the option to set permissions during the publishing process, and prevent publishers and content owners from changing them after publishing. To do this, you lock content permissions to the project. For more information, see the links below.

Next steps

- Add Projects and Move Content Into Them
- Permissions in Project Hierarchies
- Set Project Default Permissions and Lock the Project

Add Projects and Move Content Into Them

A content resource (workbooks and data sources) can live in only project. Server and site administrators can add or remove top-level projects on a site, and move published content from one project to another. Project leaders can add or remove child projects and move content between projects on which they have Project Leader access.

This article contains the steps for creating and moving projects. To learn about projects and when or why to use them, see Use Projects to Manage Content Access. Before you create project hierarchies, become familiar with Permissions in Project Hierarchies.

In this article

Add a top-level or child (nested) project
Move a workbook or data source to another project
Move a project or project hierarchy
Delete a project
Required access level for moving content
Add a top-level or child (nested) project

1. While you’re signed in to Tableau Server as an administrator or project leader, select **Content > Projects**.

2. In the **Projects** page, do one of the following:
   
   - (Administrators only) To add a top-level project, click **New Project**.
   
   - To add a child project under a top-level project, select the top-level project, and on its **Projects** tab, click **New Project**.

3. Enter a name and description for the project, and then click **Create**.
You can include formatting and hyperlinks in the project description. Select Show formatting hints for syntax.

**Note:** To edit a project description later, select it to open it, navigate to Details, and then click Edit Description.

Move a workbook or data source to another project

The following procedure is specific to workbooks; however, you follow similar steps for data sources on the Data Sources page.

1. Go to the Workbooks place, select one or more workbooks, and then select Actions > Move.

2. Select the new project for the workbook, and then click Move.

Move a project or project hierarchy

1. In the thumbnail or list view, navigate to the project you want to move, select the Actions button, and then select Move.
2. Select the destination project and then click **Move Project**.
How moving projects affects Project Leader permissions

When you move a project, Project Leader permissions adapt to the new project environment.

- When the target project hierarchy is **locked**, previous Project Leader permissions are removed, and new Project Leader permissions are granted according to those set at the top-level of the target hierarchy.

- When the target project hierarchy is **unlocked** (managed by owner), previous implicitly granted Project Leader permissions are removed, explicitly set Project Leader permissions are retained, and new Project Leader permissions are granted according to those set at the top-level of the target hierarchy.

Delete a project

When you delete a project, all of the workbooks and data sources in the project are also deleted from the site. If you want to delete a project but not its content, move the content to another project, and then delete the project.

**Important**

- You cannot undo deleting a project.

- Deleting a project deletes all content in it, including child projects and their content.

- You cannot delete the Default project.

To delete a project:

1. Click **Content > Projects**. On the **Projects** page, do the following:

   a. Select the check box next to a top-level project.

      Or navigate to a child project, and on its **Projects** page, select the check box for the child project you want to delete.

   b. Select **Actions > Delete**.
2. Confirm that you want to delete the project.

Required access level for moving content

Moving content is effectively like removing it from one project and publishing it to another. For non-administrators, the permissions needed on the source project are different than those needed on the destination project.

**Note:** The term *permissions role* refers to the collection of related capabilities you can assign as a set.
Required site role

To move content, users must have one of the following site roles:

- Server Administrator (Tableau Server only)
- Site Administrator
- Publisher

Users with a Server Administrator or Site Administrator site role do not need any additional capabilities.

Required permissions for the project that users move content to

Users with the Publisher site role need to have the Publisher or higher permissions role set on the project that is the move destination.

Required permissions for the project that users move content from

Users with Publisher site role must have these additional capabilities on the project:

- Ownership of the project or the Project Leader or Publisher permissions role.

  OR

- In a project hierarchy, project owner of, or Project Leader permissions role on the project or a parent project higher in the hierarchy.

If they have the Publisher permissions role on the project, and they are not the project owner (or owner or leader of a parent project), they must have the following capabilities:

**Data sources:** Data source owner

**Workbooks:** Workbook owner or Move capability set explicitly on the workbook.

You can grant the Move capability individually, or along with related capabilities allowed through the Editor permissions role.
Set Project Default Permissions and Lock the Project

As an administrator or project leader, you can organize a related workbooks and data sources in projects. You can then set a project’s permissions—including the default permissions that will be set on child projects you create under that project, and workbooks and data sources published anywhere in that project hierarchy. You can further lock the top-level project, at which point only administrators and project leaders can change permissions, and only at the top-level project.

Administrators and users with the Project Leader permission can lock permissions to a project. For more information, see Permissions in Project Hierarchies.

In this article

- Set the default permissions for a project
- Lock or unlock permissions on a project
- Permissions roles you can set at the project level
- Capabilities you can set for a project

Prerequisite

The remaining sections in this article assume that you have already created your user groups. For a best practice recommendation for creating groups, see Steps to coordinate projects and groups in the topic Configure Projects for Managed Self-Service.

For related information, see the topics listed at the end of this article.

Set the default permissions for a project

These steps focus on our recommended practice of assigning permissions to groups; however, you can assign permissions to individual users the same way.

Before you can complete these steps, you must create your groups, as mentioned in Prerequisite.
1. While you’re signed in to Tableau Server as an administrator or project leader, select **Content > Projects**.

2. In the **Projects** page, on the actions menu (….) for a project, select **Permissions**.

3. Click **Add a user or group rule**, select **Group**, and then select the group or name from the list.

For an existing group, open the actions menu, and then select **Edit**.
4. Select a permissions role for **Project**, **Workbooks**, or **Data Sources**, and then click **Save**.

To create a custom set of capabilities, select the **Project**, **Workbooks**, or **Data Sources** labels to expand the permissions view. Click capabilities to set them to **Allowed**, **Denied**, or **Unspecified**. Click **Save**.

The following image shows how to set project permissions. The same general steps apply for workbooks and data sources.
Note: To change the settings after saving, open the group’s or user’s actions menu (…), and then select Edit.

5. View the effective permissions:

Select a group name or user name in the permission rules to see the resulting user permissions.

Expand the Project, Workbooks, or Data Sources permissions views to see individual capabilities.
Hover over a capability box to see a tooltip with details on whether a capability is allowed or denied.

6. Follow the same steps to configure additional permission rules for more groups.

Lock or unlock permissions on a project

1. Sign in to your Tableau Server site as an administrator or project leader, in the Projects area, click a project’s Actions (…) menu, and then select Permissions.
2. Click **Edit Content Permissions**.

3. In the **Content Permissions in Project** dialog box, select **Locked to the project**, and then click **Save**.
4. To unlock content permissions for the projects, click **Edit Content Permissions** again, and select **Managed by owner**.
The default permissions are reapplied to workbooks and data sources in the project, and their permissions are now editable.

Permissions roles you can set at the project level

Permissions roles are built-in templates that assign a common set of capabilities on content. You can assign them only at the project level, for the project itself, and the workbook and data sources in it.

<table>
<thead>
<tr>
<th>Template</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Leader</td>
<td>Allows the user or group full access to the project, its child projects, and content published into that project hierarchy.</td>
</tr>
<tr>
<td>Editor</td>
<td>Allows the user or group to connect to, edit, download, delete, and set permissions for data sources or workbooks in the project. They can also publish data sources, and as long as they are the owner of a data source they publish, can update connection</td>
</tr>
</tbody>
</table>
information and extract refresh schedules. This permission is relevant for views when the view they access connects to a data source.

<table>
<thead>
<tr>
<th>Capability</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Publisher</td>
<td>Allows the user or group to publish workbooks and data sources to the project.</td>
</tr>
<tr>
<td>Connector</td>
<td>Allows the user or group to connect to data sources in the project.</td>
</tr>
<tr>
<td>Viewer</td>
<td>Allows the user or group to view the workbooks and views in the project.</td>
</tr>
<tr>
<td>None</td>
<td>Sets all capabilities for the permission rule to Unspecified.</td>
</tr>
<tr>
<td>Denied</td>
<td>Sets all capabilities for the permission rule to Denied.</td>
</tr>
</tbody>
</table>

Capabilities you can set for a project

The three capabilities you can set specifically for a project are: View, Save, and Project Leader.

<table>
<thead>
<tr>
<th>Capability</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>Allows the user or group to view the workbooks and views in the project. The View capability must also be allowed for the individual workbooks and views in the project.</td>
</tr>
<tr>
<td>Save</td>
<td>Allows the user or group to publish workbooks and data sources to the server and overwrite content on the server. The Save capability must also be allowed for the individual workbooks and data sources in the project. When allowed, the user with a site role that supports publishing can re-publish a workbook or data source from Tableau</td>
</tr>
</tbody>
</table>
Desktop, thereby becoming the owner and gaining all permissions.

Subsequently, the original owner’s access to the workbook is determined by that user’s group permissions and any further permissions the new owner might set.

This permission also determines the user’s or group’s ability to overwrite a workbook after editing it on the server. For related information, see Set Web Edit, Save, and Download Access on Content.

| Project Leader | Allows the user or group to set permissions for all items in the project, lock project permissions, and edit default permissions. |

See also

Content Permissions and Ownership

Use Projects to Manage Content Access

Configure Projects, Groups, and Permissions for Managed Self-Service

Permissions in Project Hierarchies

As an administrator or project leader, you can decide how permissions are set in a project hierarchy you manage. On the top-level project in a hierarchy, you can set the project to one of two available states: locked and managed by owner. Locking permissions can help you maintain a predictable permissions model within a project hierarchy. For example, it denies publishers the option to set explicit permissions on workbooks and data sources they own.

In this article

- Project permissions states: locked and managed by owner
How permissions are evaluated in project hierarchies

Project permissions states: locked and managed by owner

You can lock permissions in project hierarchies at the top level of the hierarchy. When you lock permissions, the top-level project’s permissions settings are applied to all workbooks, views, and data sources in the hierarchy.

In a locked project hierarchy, only administrators and project leaders can modify permissions, and they can do so only on the top level. Publishers cannot set permissions on their content during or after publishing from Tableau Desktop.

When permissions are *managed by the owner* (“unlocked”), they become editable by content owners and other users with the appropriate access level. During publishing, users have the option to set permissions explicitly on the workbook or data source they are publishing, unless you explicitly deny this capability on groups they are members of or individual users.

To clarify, owners always get full read, write, and other access to the content they publish, but in a locked state, they cannot change permissions on it.

**Note**: If a workbook, data source, or project with editable permissions is moved to a locked project, the permissions set on the locked project are applied to that content. The moved content’s permissions are then locked.

When to use each state

In general permissions are easier to manage when you lock projects. You need to go only to one place to see how they’re set for all content in that project’s hierarchy.

However, this requires that all child projects have the same permissions as set at the top level. The following common scenarios are possible only when you keep the project hierarchy *unlocked*:
• You want to create projects for different types of access your team needs for content in child projects. This strategy is described in Configure Projects, Groups, and Permissions for Managed Self-Service, and it requires setting unique permissions on each child project.

If you keep permissions unlocked, you can still set permissions at a parent project level that you want all child projects to take by default. And you can use groups to set capabilities explicitly, including denying the Set Permissions capability, so that users cannot set permissions on content they own.

• In web editing, you want to allow publishers to save changes to workbooks but not overwrite them.

For more information, see Allow publishers to edit, save changes to, and download new workbooks, but not overwrite existing workbooks.

How permissions are evaluated in project hierarchies

The following diagram shows how ownership and Project Leader permissions are applied in a hierarchy.

- PL = Project Leader
- O = Owner
- Dotted line indicates implicit permissions “inherited” at a parent level.
- * = Owner was assigned explicitly by the owner of the parent project.
How permissions are applied in locked project hierarchies

You can lock permission only at the top-level in a project hierarchy, and the following behavior applies:

- All child projects you create within a top-level project use the default permissions set at that top-level project (not the Default project), and you cannot modify permissions at the child-project level.

- Workbooks, views, and data sources also use the default permissions set on the top-level project.

- Users, including content owners, cannot edit permissions for individual workbooks, views, or data sources.

When users publish workbooks and data sources from Tableau Desktop, they cannot set permissions in the Publish dialog box, and their content gets the default permissions for the project they publish it to.

- Users or groups that are assigned the Project Leader permissions role at the top-level
project get Project Leader access to all child projects and their workbooks and data sources.

- Administrators and content owners can change ownership at any level in the hierarchy.

- Administrators and project leaders can edit permissions at the top-level project, and those changes propagate to all child projects.

How permissions are applied in unlocked project hierarchies

If the top-level project’s permissions are not locked, you can set permissions on its child projects in the same way you can on top-level projects, with the exception of locking the child project.

When you create a child project, it takes default permissions from the parent project. Users implicitly get the same permissions they have on the parent project.

If you set permissions at the child level, those settings take precedence over permissions set at the parent level.

See also

For a couple of best-practice steps for how to implement permissions, see the following:

- Configure Projects, Groups, and Permissions for Managed Self-Service

- Structure Content Projects, Groups, and Permissions (links to Everybody’s Admin Guide)

Add a Project Image

Projects can have images that are displayed in thumbnail view in Tableau Server.
Set a project image

1. Sign in to a site on Tableau Server. On the Content tab, select Projects, and then select a project.

2. Navigate to Details, and then click Edit Description.
3. In the **About** field, add the URL for your image, using the following syntax:

!http://www.example.com/image.png!
Select **Show formatting hints** to see how you can format description text.

4. Click **Save**.
Content Permissions and Ownership

In Tableau Server, you set content permissions to specify who is allowed to work with which content resources on a particular site.

For example, you can tightly restrict who has access to your company’s financial information, but widely share organizational development content.

Content resources on which you will generally assign permissions include the following:

- Projects
- Workbooks
- Data sources

Views in a workbook inherit permissions from the workbook, although you might have an edge case for which you want to set permissions for a single view.
What makes up a user’s permissions

When Tableau determines which tasks (or capabilities) a user is allowed to perform on a content resource, it takes the following pieces into account:

**Site roles:** At the time that you add users to a site, you must apply a site role to them. This is the only setting that you apply to users (as opposed to content) to affect permissions. The site role determines whether the user can publish, interact with, or only view published content on that site. See Set Users’ Site Roles.

**Permission rules and templates:** You assign content access through permission rules. These rules describe the capabilities that you want a user or group to be able to perform on a set of content. Examples of capabilities include editing a view or connecting to a data source. Tableau provides a set of templates for common permission roles, such as Editor, Project Leader, and so on.

**Content ownership:** By default, the person who publishes a data source or workbook to the server is the owner of that content. Ownership changes when another publisher updates the content on the server, or republishes it from Tableau Desktop. An administrator or project leader can change ownership or set defaults for the project.

**User permissions:** are the effective permissions that are the result of evaluating rules and settings, and which ultimately determine what a user can do with the content.

Who can set permissions

Users can have the **Set Permissions** capability. By default, these are:

- Administrators, for all content on the site.
- Project leaders, for all content in the project and its child projects.
- Content owners, generally those who create projects or publish data sources or workbooks.
Quick Start: Permissions

You can use permission rules to control access to content on a site. A permission rule is a set of capabilities that defines the level of access a group or user has on a content item. Content items are projects you create, and the workbooks and data sources published to them.

The most efficient way to manage permissions is to create permission rules for groups, and assign the permissions at the project level.

1 Add users to groups

Create groups for users based on who should have the same permissions, and then add users to those groups. Within a site, select Groups. Select a group name, and then select Add Users.
2 Select a project

On the Content page for a site, navigate to the Projects page. On the Actions menu, select Permissions to view the permission rules for the project. By default, only an All Users group is defined, with pre-defined permissions roles applied to the group.

The image below shows an All Users group that was modified to remove those pre-defined permissions, in preparation for creating new groups for your own environment.

3 Create a Permission Rule

Click Add a user or group rule, select Group, and then find and select the group.

Select a permission role template to apply an initial set of capabilities for the group. Click a capability to set it to Allowed or Denied, or leave it Unspecified. Click Save when you are done.
Whether a user can set permissions is based on their site role and how their **Set Permissions** capability is set.

4 View User Permissions

After you save the permission rule for the group, you can view the effective permissions for that content.

Click a group name to see the group’s users and their permissions. Hover over a capability box to see a tooltip with details on whether a capability is allowed or denied.
Custom indicates a user's capabilities have been changed from the initial settings for their site role or content role.

Site roles

A user’s site role determines the maximum permissions allowed for that user.

- Server and site administrators can access all site content with full permissions.
- Owners always get full access to the content they’ve published, but can change permissions for their workbooks and data sources only when the parent project permissions are not locked.

For more information, see Set Users’ Site Roles.

Permissions evaluation

- **Denied** takes precedence over **Allowed**.
- **Unspecified** results in **Denied** if no other permissions are specified.
- Specific user permissions on content take precedence over group permissions on content. In other words, user permissions trump group permissions.
For a couple of best-practice steps for how to implement permissions, see the following:

- Configure Projects, Groups, and Permissions for Managed Self-Service
- Projects and Content Permissions (links to Everybody’s Admin Guide)
Manage Content Ownership

When you publish a data source or workbook on Tableau Server or when you create a project, you become its owner. A content owner, a project leader, or an administrator can change ownership of a content asset. After ownership is reassigned, the original owner has no special connection to the content item, and their ability to access it is determined by their permissions on the project or that specific item.

In this article
Considerations for changing content ownership
Who can change or be given ownership, by content type
Change a workbook or data source owner
Change a project owner

Considerations for changing content ownership

- If you want to remove a user from Tableau Server, make sure they do not own any content assets before you delete them.

  If the user does own content, you must first reassign ownership of those assets before you can delete the user. Otherwise, their site role is set to Unlicensed, but they are not deleted, and they continue to show as the content owner.

- If you change the ownership of a workbook or data source that includes embedded credentials (for connecting to underlying data), the embedded credentials will be deleted.

  You can update the embedded credentials by editing the connection information on Tableau Server. For more information, see Edit Connections. Alternatively, the new owner can download the workbook or data source, open the item in Tableau Desktop
to update the embedded credentials, and then re-publish the workbook or data source.

Who can change or be given ownership, by content type

Whether you can change or be given ownership depends on your permissions and your relationship to the content asset, as described in the following table.

<table>
<thead>
<tr>
<th>Content asset type</th>
<th>Who can change ownership</th>
<th>Who can be given ownership</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Top-level projects</strong></td>
<td>Server administrator&lt;sup&gt;1&lt;/sup&gt;</td>
<td>Server administrator</td>
</tr>
<tr>
<td></td>
<td>Site administrator</td>
<td>Site administrator</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Publisher</td>
</tr>
<tr>
<td><strong>Child (nested) projects</strong></td>
<td>Server administrator</td>
<td>Any administrator or user of the site, excluding Guest users.</td>
</tr>
<tr>
<td></td>
<td>Site administrator</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Project leader or owner</td>
<td></td>
</tr>
<tr>
<td><strong>Workbooks and data sources</strong></td>
<td>Server administrator</td>
<td>Any administrator or user of the site, excluding Guest users.</td>
</tr>
<tr>
<td></td>
<td>Site administrator</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Workbook or data source owner</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Project leader or owner of the project that contains the item</td>
<td></td>
</tr>
</tbody>
</table>

<sup>1</sup> The Server Administrator site role applies to Tableau Server only; not Tableau Online.
Change a workbook or data source owner

1. On the Content page for a site, select Workbooks or Data Sources.

2. Select one or more workbooks or data sources, and then select Actions > Change Owner.

3. Type the name of a user or select a user from the list.
4. Click **Change Owner**.

Change a project owner

1. On the Content page for a site, select **Projects**.

2. Select one or more projects, and then select **Actions > Change Owner**.
3. Type the name of a user or select a user from the list.
4. Click **Change Owner**.

View or Edit Permission Rules and User Permissions

When you specify permissions for a project, workbook, view, or data source, you use a *permission rule* to express who is allowed to work with that resource. Permission rules are the explicit capabilities that can be set for an individual user, or for a group—for each resource.

You work with these rules in the **Permissions** window, described in the steps below. You define permission rules in the upper section. The **User Permissions** section below it shows the effective, or resulting, permissions after Tableau evaluates your rule.

View or edit a permission rule for a selected content resource

1. On the Content page for a site, select **Projects, Workbooks, Views, or Data Sources** to display all items of that content type.

2. Select the check box for the item, and then select **Actions > Permissions**.
This shows the current permission rules.

3. To edit a rule, do the following:
   a. Select the actions menu (. . .) next to the rule name, and then select **Edit**.
   b. Click a capability in the rule to set it to **Allowed** or **Denied**, or **Unspecified**.
      
      See How the permission rules settings work together below for more information.
   c. Click **Save**.

4. To see the resulting permissions for the selected group or user, do the following:
   a. In the permission rules area, select the group or user name.
   b. In the **User Permissions** area, hover over a capability box to show a tooltip that indicates the capability name, its resulting setting, and how the result was determined.
The following image shows permissions on a selected workbook for Andrew Allen in the Finance group. The tooltip indicates that a rule assigned to a group he is a member of denies the web edit capability.

How the permission rules settings work together

The sections shown in the Permissions Rules area depend on the type of content you select. The settings described here are those that appear when you select a workbook or view. (Some of these settings are not available until you open a rule for editing, as described in the steps above.) To learn how to set permission rules at the project level, see Set Project Default Permissions and Lock the Project.

- **User / Group**: Lists users or groups that a rule applies to. If the permissions are not locked at the project level, you can click Add a user or group rule to configure permissions for those users on the selected content.

- **Permissions**: Lists available permission-role templates for the selected content element. Each template contains a predefined set of capabilities for the rule. If the capabilities selected for the user or group do not match a predefined template, the template name changes to Custom. The word Custom appears regardless of how the capabilities are modified.

- **View / Interact / Edit**: These are categories for the groups of capabilities that you
can set to **Allowed**, **Denied**, or **Unspecified**. Setting **Unspecified** evaluates to **Denied** if no other permissions are specified for a user or group on the content.

Although some of the names here are the same as or similar to site or permission role names, the categories here are independent of those things, and they only indicate groups of capabilities that are typically configured as a set.

User Permissions

Effective user permissions for a content resource are determined by the following:

- The maximum capabilities allowed for a user’s site role. The site role acts as the "ceiling" for what permissions are allowed. For more information, see Set Users’ Site Roles.
- Whether the user owns the content item
- The evaluation of each user or group permission rule that applies to that user for that content item

For example, if a user is granted Editor-level permissions for a workbook (which allows all available capabilities), but has the site role of Viewer and does not own the workbook, the user will only be allowed the capabilities of **View**, **Export Image**, **Summary Data**, **View Comments**, **Add Comments**, and **Save**.

In the following example, a permission rule has been created for the Finance group. The permission role template of **Editor** was initially applied to the group, which granted all capabilities. The administrator then set the **Save** capability to **Denied**, so the name for the set of permissions applied to the group became **Custom**. The **User Permissions** section for the Finance group shows that most of the users in the group have all capabilities, except for the **Save** capability. One user has even fewer capabilities because that user has a site role of Viewer.
Note that the **All Users** group permission rule in this example has been set to **None**, which leaves all of the permissions as **Unspecified** for the **All Users** group. This approach requires the administrator to specifically assign permissions for only the groups or users that should see the content.

**Set Permissions on Individual Content Resources**

As an administrator, or as a user granted the **Set Permissions** capability on a specific resource, you can change permissions on that resource (data source or workbook).

Generally we recommend setting permissions at the project level for all content in the project. A reason you might want to deviate from this practice, for example, is to explicitly deny the some capabilities on a data source or workbook that has a data source or user filter that enables a form of row-level security.
In this article

Capabilities you can set on data sources

Capabilities you can set on workbooks and views

Set permissions on a content resource

Set Permissions on Individual Content Resources

Capabilities you can set on data sources

Use permission rules to set the following capabilities for a data source:

<table>
<thead>
<tr>
<th>Capability</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>View the data source on the server.</td>
</tr>
<tr>
<td>Connect</td>
<td>Connect to the data source. The <strong>Connect</strong> permission allows a user to connect to a data source from an editor (in Tableau Desktop or Tableau Server web editing). <strong>Note:</strong> If a workbook author embeds credentials in a workbook or view, users who also have the <strong>Web Edit</strong> permission will be able to access to the workbook’s data source regardless of their <strong>Connect</strong> permissions.</td>
</tr>
<tr>
<td>Save</td>
<td>Publish data sources to the server and overwrite data sources on the server.</td>
</tr>
<tr>
<td>Download Data Source</td>
<td>Download the data source from the server.</td>
</tr>
</tbody>
</table>

**Note:** Cube data sources, like those for Microsoft Analysis...
Services or Oracle Essbase connections, must be used locally. To download the published data source to Tableau Desktop, the user must have the Download capability. You must explicitly grant the Download permissions regardless of the permissions role you apply. For more information, see Cube Data Sources.

Delete
Delete the data source.

Set Permissions
Grant or deny permissions for the data source.

Permission role templates for data sources

<table>
<thead>
<tr>
<th>Template</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector</td>
<td>Allows the user or group to connect to the data source on the server.</td>
</tr>
<tr>
<td>Editor</td>
<td>Allows the user or group to connect to, download, delete, and set permissions on data sources on the server. They can also publish data sources, and as long as they are the owner of a data source they publish, they can update connection information and extract refresh schedules. (The latter two capabilities are no longer available if an administrator or project leader changes data source ownership.)</td>
</tr>
<tr>
<td>None</td>
<td>Sets all capabilities for the permission rule to Unspecified.</td>
</tr>
<tr>
<td>Denied</td>
<td>Sets all capabilities for the permission rule to Denied.</td>
</tr>
</tbody>
</table>
Capabilities you can set on workbooks and views

The list of capabilities and the available permission role templates vary depending on whether you are setting permissions for a workbook or a view. For information about capability definitions, see Permissions Reference.

Editing view-level permissions

When a Tableau Desktop author publishes a workbook with the Show Sheets as Tabs enabled, these tabbed views take on the workbook permissions rules. Changes you make to the workbook permissions affect all of its tabbed views.

To edit an individual view’s permissions, save the workbook again without tabs (or hide sheets). The default permissions are applied to the workbook, and you can then edit view permissions.

We recommend that you set view-level permissions sparingly, as an exception. Try to manage permissions at the project level as much as possible. When permissions are locked to a project, views in a workbook use the workbook permissions.

Permission role templates for workbooks and views

<table>
<thead>
<tr>
<th>Template</th>
<th>Applies to…</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viewer</td>
<td>workbooks, views</td>
<td>Allows the user or group to view the workbook or view on the server.</td>
</tr>
<tr>
<td>Interactor</td>
<td>workbooks, views</td>
<td>Allows the user or group to view the workbook or view on the server, edit workbook views, apply filters, view underlying data, export images, and export data. All other permissions are inherited from the user’s or group’s project permissions.</td>
</tr>
<tr>
<td>Editor</td>
<td>workbooks, views</td>
<td>Sets all capabilities for the rule to Allowed.</td>
</tr>
<tr>
<td>Single Value</td>
<td>Multiple Values</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>None</td>
<td>workbooks</td>
<td>Sets all capabilities for the rule to <strong>Unspecified</strong>.</td>
</tr>
<tr>
<td>None</td>
<td>views</td>
<td>Sets all capabilities for the rule to <strong>Denied</strong>.</td>
</tr>
</tbody>
</table>

**Set permissions on a content resource**

1. In the Data Sources, Workbooks, or Views page, select the check boxes for each resource, and then select **Actions > Permissions**.

   The following image shows how this looks on the Data Sources page.

   ![Image showing Data Sources page with selections and permissions options]

   **Note:** If you select multiple items and some of the items are read-only, you cannot view the permissions. Instead, select one item at a time.

2. Click **Add a user or group rule**, select **Group** or **User**, and then select the group or user name from the list.
3. Select a permission role template to apply an initial set of capabilities for the group or user, and then click **Save**.

4. To further customize the rule, click the actions menu ( . . . ) next to the rule name, and then click **Edit**. Click a capability in the rule to set it to **Allowed** or **Denied**, or leave it unspecified. Click **Save** when you are done.
5. Configure any additional rules you want for other users or groups.

6. View the resulting permissions.

   Click a group name or user name in the permission rules to see the resulting permissions. Hover over a capability box to see a tooltip that shows whether a capability is allowed or denied, and what determined that result.
How Permissions are Evaluated

Permissions in Tableau Server are assigned to content resources—projects, workbooks, data sources, and sometimes to individual views. You use permission rules to specify who can work with a content resource.

What users can access and the actions available for each content type, are determined by the following:

- **Site role.** A user’s site role determines whether a user can publish, interact with, or only view resources. For more information, see Set Users’ Site Roles.

- **Content permissions.** Every project, workbook, data source, or view can have a unique set of permission rules.

  A permission rule includes the user or group, and the set of capabilities you want to grant users for a resource (such as the ability to edit a view). Each permission role template (such as Editor, Interactor, Viewer) specifies a predefined set of capabilities for the rule. If the capabilities that are selected do not match a predefined template, the permission role template changes to Custom.

  Available capabilities vary depending on the resource. Capabilities can be set to **Allowed**, **Denied**, or **Unspecified**. **Denied** always takes precedence over **Allowed**, and **Unspecified** results in **Denied** if no other permission rules allow a capability for a user.

- **Ownership.** Content owners always get full access to the content they’ve published. In projects with locked permissions, content owners cannot edit permissions for their workbooks and data sources.
Users with the **Set Permissions** capability can change permissions for content items in projects that aren't locked. Administrators, content owners, and users with the **Project Leader** capability automatically have the **Set Permissions** capability.

You can set permission rules for an individual user or group for each resource. This diagram illustrates how permission rules are evaluated in Tableau Server.

A user’s effective permissions for a given content resource are determined by the following:

- The maximum capabilities allowed through the site role.
- Whether the user owns the content item.
- The result after Tableau evaluates permission rules applied to that user and all groups the user is a member of.

Notes on permissions

- Server and site administrators can access all the resources in a site with full permissions.
- You cannot set permissions at the site level; permissions are assigned to resources only.
- As content owners, publishers get full access to their content. Administrators can manage permissions on content after it’s published.

Another option for practicing content governance is to lock permissions at the project level. This does not change publishers’ ownership of the content they publish after the
project is locked. However, it enables default permissions for their users and prevents publishers from being able to change those default permissions during the publishing process. See Permissions in Project Hierarchies.

- Individual user permissions on resources take precedence over group permissions on resources. In other words, user permissions trump group permissions.

- Workbook permissions serve as templates for view permissions. When content permissions are locked to the project, and when a workbook uses tabbed views, views inherit their workbook permissions. When permissions are not locked, and when a workbook is saved without tabs, the workbook and view permissions can be edited independently.

- Project default permissions serve as templates for content in a project. When content permissions are locked to the project, the workbooks and data sources always use the default permissions. When permissions are not locked, workbook and data source permissions can be edited independently.

- For each content item, every site user is automatically included in the All Users group. As a result, the All Users permission rule affects how permissions are evaluated for users when you create additional group permission rules for that content item.

If you use Tableau Server in an environment where openly sharing knowledge and information across the organization is important, set the permission rule for the All Users group in the Default project to the Publisher permission template. Users can publish to and consume content from new projects.

If you use Tableau Server in an environment where restricting access is important, set the permission rule for the All Users group in the Default project to the role of None. Then, add explicit permissions for groups and users to allow them to publish and work with content in new projects.
The order of precedence in which Tableau evaluates permissions

1. **Server and Site Administrator**: Administrators can access all site content with full permissions.

2. **User - Unlicensed, Viewer license, or Guest**: If a user is Unlicensed, has a Viewer license (different than Viewer site role), or is a Guest, there are certain capabilities they are never allowed to perform. If the capability is explicitly denied for the user because of licensing, they are denied.

3. **Project Owner**: If the user owns the project that contains the content, the capability is allowed. Otherwise,

4. **Project Leader**: If the user has the Project Leader capability, or is in a group that has the Project Leader capability, they are allowed. If the user is explicitly denied the Project Leader capability, they are denied. Otherwise,

5. **User - Authorizable Owner**: If the user is the owner of the content, they are allowed. Otherwise,

6. **User - Capability Denied**: If the user has been explicitly denied the capability for the content, they are denied. Otherwise,

7. **User - Capability Allowed**: If the user has been explicitly allowed the capability for the content, they are allowed. Otherwise,

8. **Group - Capability Denied**: If the user belongs to a group that has been explicitly denied the capability for the content, they are denied. Otherwise,

9. **Group - Capability Allowed**: If the user belongs to a group that has been explicitly allowed the capability for the content, they are allowed. Otherwise,

10. The user is denied access to the content.
Permissions Reference

Administrators and other authorized users can allow or deny capabilities on content published to Tableau Server. When permissions are not locked to a project, Tableau Desktop users who publish to that project can set permissions on their content during the publishing process.

In this article

Site and permissions roles that allow full content access

Capabilities available for each content type

How data access is evaluated for workbooks that connect to Tableau data sources

Site and permissions roles that allow full content access

The following list describes roughly who gets full access to content on the server and the scope of that access:

- Server administrators (Tableau Server only) are allowed all capabilities on all content published to the server.
- Site administrators are allowed all capabilities on content published to the site.
- Groups to which you apply the Project Leader role are allowed all capabilities on content in the project.
- Publishers become the owners of content they publish, and they are allowed all capabilities on that content, with the exception of setting permissions in locked projects.

Capabilities available for each content type

The following table shows how permissions apply to content and describes each capability.
<table>
<thead>
<tr>
<th>Permission</th>
<th>Applies to…</th>
<th>When allowed, users can…</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>workbooks</td>
<td>Open the item on Tableau Server.</td>
</tr>
<tr>
<td></td>
<td>data sources</td>
<td><strong>Note</strong>: When a workbook is configured to show sheets as tabs, all views use the workbook permissions, even if different permissions are specified on an individual view.</td>
</tr>
<tr>
<td></td>
<td>views</td>
<td></td>
</tr>
<tr>
<td></td>
<td>projects</td>
<td></td>
</tr>
<tr>
<td>Web Edit</td>
<td>workbooks</td>
<td>Edit views in workbooks. For information, see Set Web Edit, Save, and Download Access on Content.</td>
</tr>
<tr>
<td></td>
<td>views</td>
<td></td>
</tr>
<tr>
<td>Save</td>
<td>workbooks</td>
<td>Overwrite the resource on the server.</td>
</tr>
<tr>
<td></td>
<td>data sources</td>
<td>When allowed, the user can re-publish a workbook or data source from Tableau Desktop, thereby becoming the owner and gaining access to all permissions. Subsequently, the original owner’s access to the workbook is determined by that user’s group permissions and by any further permissions the new owner might set.</td>
</tr>
<tr>
<td></td>
<td>views</td>
<td>This capability also determines the user’s or group’s ability to overwrite a workbook after editing it on the server.</td>
</tr>
<tr>
<td></td>
<td>projects</td>
<td>Special consideration for the <strong>All Users</strong> group: To help protect an owner’s content from being overwritten by another user (via publishing from Tableau Desktop or saving a web-edited workbook on the server), when a user publishes into a project where the <strong>All Users</strong> group has permissions, the <strong>Save</strong> permission for the <strong>All Users</strong> group is changed from</td>
</tr>
<tr>
<td>Permission</td>
<td>Applies to...</td>
<td>When allowed, users can...</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>Download Workbook/Save As</td>
<td>workbooks</td>
<td>Allowed to Unspecified by default. You can then manually modify this capability if you want to allow it.</td>
</tr>
<tr>
<td></td>
<td>data sources</td>
<td>Download a Tableau packaged workbook (.twbx) from the server.</td>
</tr>
<tr>
<td></td>
<td>workbooks</td>
<td>Save an edited workbook as a new workbook on the server.</td>
</tr>
<tr>
<td></td>
<td>views</td>
<td>Download the data source from the server.</td>
</tr>
<tr>
<td>Delete</td>
<td>workbooks</td>
<td>Delete the content resource.</td>
</tr>
<tr>
<td></td>
<td>data sources</td>
<td></td>
</tr>
<tr>
<td></td>
<td>views</td>
<td></td>
</tr>
<tr>
<td>Filter</td>
<td>workbooks</td>
<td>Modify filters in the view, keep only filters, and exclude data.</td>
</tr>
<tr>
<td></td>
<td>views</td>
<td></td>
</tr>
<tr>
<td>Add Comments</td>
<td>workbooks</td>
<td>Add comments to views in a workbook.</td>
</tr>
<tr>
<td></td>
<td>views</td>
<td></td>
</tr>
<tr>
<td>View Comments</td>
<td>workbooks</td>
<td>View the comments associated with the views in a workbook.</td>
</tr>
<tr>
<td></td>
<td>views</td>
<td></td>
</tr>
<tr>
<td>Download Sum</td>
<td>workbooks</td>
<td>View the aggregated data in a view, or in the user's</td>
</tr>
<tr>
<td>Permission</td>
<td>Applies to...</td>
<td>When allowed, users can...</td>
</tr>
<tr>
<td>------------</td>
<td>---------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>mary Data</td>
<td>views</td>
<td>selection within the view, and download that data as a text file in .csv format.</td>
</tr>
<tr>
<td>Download Full Data</td>
<td>workbooks views</td>
<td>View the underlying data behind each row in a view, as restricted by any marks the user has selected, and download the data as text file in .csv format.</td>
</tr>
<tr>
<td>Download Image/PDF</td>
<td>workbooks views</td>
<td>Download each view to PNG or PDF format.</td>
</tr>
<tr>
<td>Share Customized</td>
<td>workbooks views</td>
<td>Users can save customizations they’ve made to a view using the interaction features (such as filters). They can then share their custom view for others to see. Customization in this way does not change the view’s default state or its underlying data.</td>
</tr>
<tr>
<td>Move</td>
<td>workbooks</td>
<td>Move workbooks between projects. <strong>Note:</strong> Only administrators can move data sources between projects.</td>
</tr>
<tr>
<td>Set Permissions</td>
<td>workbooks data sources views</td>
<td>Specify permissions for the resource. Views in a workbook take permissions set at the workbook level.</td>
</tr>
<tr>
<td>Connect</td>
<td>data sources</td>
<td>Connect to a published data source from Tableau</td>
</tr>
</tbody>
</table>
How data access is evaluated for workbooks that connect to Tableau data sources

When a Tableau Desktop user publishes a workbook that connects to a data source published to the server (a Tableau data source), the user can set the data authentication type to **Embed password** or **Prompt users**. In this specific publishing case, data authentication refers to the workbook’s connection to the Tableau data source. This is independent of the authentication set in the published data source’s connections to the underlying data.

When a user subsequently opens the workbook on the server, the workbook-specific data authentication is evaluated along with the **Connect** capability, set in permissions rules. Whether the user can see the data in the workbook is determined as follows:

- When the workbook publisher selects the **Embedded password** option, they effectively embed their own **Connect** permissions, allowing anyone who can view the workbook to see the data it connects to. How the viewer’s **Connect** capability is set on the data source does not affect this outcome.

- When the publisher selects the **Prompt users** option, viewers who are allowed the **Connect** capability are on the data source are prompted for their credentials. Users who are denied the **Connect** capability are denied access the data.

The following table summarizes the two bullet points above.
### Manage Data

#### Tableau Server Data Sources

When your Tableau users want to share data connections they’ve defined, they can publish data sources to Tableau Server. When a data source is published to the server, other users can connect to it from their own workbooks, as they do other types of data. When the data in the Tableau data source is updated, all workbooks that connect to it pick up the changes.

Looking for Tableau Server on Windows? See Tableau Server Data Sources.

A Tableau Server data source consists of metadata that describes the following:

- **The connection information**: Defines whether the data is in a live database or an extract, and which of that data to bring in to Tableau.
• **Customization and cleanup**: Includes information that facilitates efficient use of the data. For example, calculations, sets, groups, bins, parameters, custom field formatting, and so on.

• **Data access and refresh instructions**: Includes the location of the underlying database server (whether on-premises or in the cloud), network paths for file-based data, security information such as credentials or access tokens, and related information.

In addition to helping your users create data consistency and reliability, using Tableau data sources offers advantages to you as the administrator. Because multiple workbooks can connect to one data source, you can minimize data source proliferation and save on storage space and processing time. When someone downloads a workbook that connects to a Tableau data source that in turn has an extract connection, the extract stays on the server, reducing network traffic. Finally, if a connection requires a database driver, you need to install and maintain the driver only on the server, instead of on each user’s computer. (If you use Tableau Online, all supported drivers are available to data sources published to your site.)

**Managing data sources**

You can perform some or all management tasks on a data source if you have one of the following levels of access:

• Site or server administrator

• Project leader or owner of the project the data source is published to

• Data source owner

Unless you’re a site or server administrator, you might not have access to all management tasks. For example, if the Permissions menu is not available, chances are that the project permissions are locked, which denies setting permissions on individual workbooks and data sources.
Tip: A best practice is to designate a person or team to manage all data sources published to a project or to the whole site, so that all data sources can be maintained under the same set of guidelines.

To perform the management tasks that you have access to, do the following:

1. Sign in to the site, go to the **Data Sources** page, select the check box next to the data source, and in the upper-left of the Data Sources page, select **Actions**.

   ▼ Actions
   - New Workbook
   - Download
   - Tag...
   - Permissions...
   - Move...
   - Change Owner...
   - Refresh Extracts...
   - Edit Connection...
   - Revision History...
   - Delete...

- **Edit and view permissions**: Permissions can specify which users or groups can connect to, modify, or download data sources. As mentioned at the beginning of this section, if this action is not available, it’s likely that the project permissions are locked.

- **Edit connection information**: Update embedded credentials or other metadata that describes the connections to the underlying data.

- **Create a new workbook or download**: You can start a new workbook in the browser environment by connecting to a Tableau data source. Or download the data
source to use locally.

- **Change the data source owner**

- **View the data source’s revision history**

- **Refresh extracts:** If a data source includes an extract, you can assign the extract to a refresh schedule.

  For information, see Refresh Data on a Schedule.

- **Add or remove keyword tags:** Tags can contain a single word or multiple words, delimited by a comma.

- **Delete:** Deleting a data source affects workbooks that connect to the data source.
  Before you delete a data source, ensure that there are no workbooks that connect to the data source or edit the workbooks to use another data source.

- **Move:** Move a data source from one project to another. This requires specific settings on each project. For information, see Required access level for moving content.

In addition, for data sources that are proxy connections, administrators can stay aware of how users authenticate to the database, and whether the appropriate drivers are installed. For information, see Install Drivers on Linux and Data Security.

**Extract Upgrade to .hyper Format**

Beginning in version 10.5, new extracts use the .hyper format instead of the .tde format. Extracts in the .hyper format take advantage of an improved data engine, which supports the same fast analytical and query performance as the data engine before it, but for even larger extracts.

This article covers what you and your users should be aware of as a result of the extract format change in version 10.5 and how the change impacts your users.
Changes to extracts in version 10.5

With the introduction of Tableau's improved data engine technology, Hyper, comes a change to the extract format. As mentioned above, extracts created in version 10.5 use the .hyper format. Though Tableau version 10.5 can continue to read .tde extracts, it cannot create new .tde extracts. The impact of this format change means that when certain extract tasks are performed either by a user or by Tableau Server, such as an extract refresh or append data, the .tde extract is automatically upgraded and converted to a .hyper extract.

After an extract is upgraded to a .hyper extract, your users can interact with the .hyper extract as they would a .tde extract. However, your users should be aware of the following backward limitations:

- The upgrade can't be reversed. An upgraded extract can't be converted back to a .tde extract.
- The upgraded extract can't be opened in Tableau Desktop 10.4 and earlier.
- Workbooks that contain upgraded extracts can't be published as an older version from Tableau Desktop to Tableau Server.
- Workbooks that contain upgraded extracts can't be downgraded as an older version in Tableau Desktop.
Tasks that cause extract upgrade

There are three distinct ways a .tde extract can get upgraded to a .hyper extract: 1.) during an extract refresh (full or incremental), 2.) when appending data to an extract, and 3.) when an extract is upgraded manually using Tableau Desktop 10.5. After an extract has been upgraded, the original .tde extract is automatically removed from Tableau Server if it's not being reference by other workbooks.

The followings tasks on Tableau Server automatically upgrades and converts a .tde extract to a .hyper extract:

- Manual refresh
- Scheduled extract refresh
- Automated refresh task that's performed through tabcmd
- Automated refresh task that's performed through the Extract Command-Line Utility
- Automated append data to an extract using tabcmd
- Automated append data to an extract using the Extract Command-Line Utility

Impact of extract upgrade

Tableau recommends that the Tableau Desktop version in your environment be upgraded to match Tableau Server 10.5. If they don't match, extract compatibility can be an issue depending on the task your users want to perform.

To get a better understanding of when extract upgrades can occur and potential compatibility issues your users might experience, review a detailed explanation of the extract compatibility scenarios on the Tableau Support page or a summary explanation below.

**Note:** In the tables below, "10.4" represents Tableau 10.4 and earlier and "10.5" represents Tableau 10.5 and later.
Performing tasks on Tableau Server 10.5

When working with extracts created in Tableau Desktop 10.4 and earlier, your users should be aware of the following extract-related compatibility scenarios around common tasks performed on Tableau Server 10.5.

<table>
<thead>
<tr>
<th>Task</th>
<th>10.4 workbook</th>
<th>10.5 workbook</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>.tde extract</td>
<td>.tde extract</td>
</tr>
<tr>
<td>Publish from Tableau Desktop 10.4</td>
<td>√</td>
<td>Not possible</td>
</tr>
<tr>
<td>Publish from Tableau Desktop 10.5</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Refresh, scheduled refresh, or append</td>
<td>Workbook version remains unchanged, but extract upgrades to .hyper format</td>
<td>Workbook version remains unchanged, but extract upgrades to .hyper format</td>
</tr>
<tr>
<td>Edit/save in web authoring</td>
<td>Workbook version changes to 10.5, extract remains in .tde format</td>
<td>√</td>
</tr>
<tr>
<td>Edit/save in web authoring then refresh or append</td>
<td>Workbook version changes to 10.5, and extract upgrades to .hyper format</td>
<td>Workbook version remains unchanged, but extract upgrades to .hyper format</td>
</tr>
</tbody>
</table>
Downloading from Tableau Server 10.5

When working with extracts created in Tableau Desktop 10.4 and earlier, your users should be aware of the following extract-related compatibility scenarios when downloading from Tableau Server 10.5.

<table>
<thead>
<tr>
<th>Task</th>
<th>10.4 workbook</th>
<th>10.5 workbook</th>
</tr>
</thead>
<tbody>
<tr>
<td>Download and open in Tableau Desktop 10.4</td>
<td>✓</td>
<td>Can't open workbook; you see a &quot;this workbook uses a .hyper extract and is not compatible with this version; open the workbook in version 10.5 or later&quot; error message, and then asked to locate the extract</td>
</tr>
<tr>
<td>Download and open in Tableau Desktop 10.5</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Export As Version from Tableau Desktop 10.5</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Automating refresh and append tasks

When working with extracts created in Tableau Desktop 10.4 and earlier, your users should be aware of the following extract-related compatibility scenarios around automating refresh and append tasks using tabcmd or the Tableau Command-Line Utility.

<table>
<thead>
<tr>
<th>Task</th>
<th>10.4 workbook</th>
<th>10.5 workbook</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>.tde extract</td>
<td>.tde extract</td>
</tr>
<tr>
<td>Refresh or append</td>
<td></td>
<td></td>
</tr>
<tr>
<td>using 10.4 tabcmd</td>
<td>Workbook version remains unchanged, but extract upgrades to .hyper format</td>
<td>Workbook version remains unchanged, but extract upgrades to .hyper format</td>
</tr>
<tr>
<td>using 10.5 tabcmd</td>
<td>Workbook version remains unchanged, but extract upgrades to .hyper format</td>
<td>Workbook version remains unchanged, but extract upgrades to .hyper format</td>
</tr>
<tr>
<td>Publish</td>
<td></td>
<td></td>
</tr>
<tr>
<td>using 10.4 tabcmd</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>using 10.5 tabcmd</td>
<td>Not possible</td>
<td>Not possible</td>
</tr>
<tr>
<td>Refresh or append</td>
<td></td>
<td></td>
</tr>
<tr>
<td>using 10.4 Tableau Command-Line Utility</td>
<td>✓</td>
<td>Not possible</td>
</tr>
<tr>
<td>using 10.5 Tableau Command-Line Utility</td>
<td>Workbook version remains unchanged, but extract upgrades to .hyper format</td>
<td>Workbook version remains unchanged, but extract upgrades to .hyper format</td>
</tr>
</tbody>
</table>
Why keep an extract in .tde format?

If the version of Tableau Desktop cannot be upgraded to match Tableau Server 10.5, your users will need to keep their extracts in .tde format.

How to keep an extract in .tde format

To keep an extract in the .tde format, the extract should not be upgraded. To help your users keep their extracts from upgrading, advise them against performing any of the tasks listed above in the Extract Upgrade to .hyper Format section. Then, consider the following suggestions to maintain a .tde version of an extract:

- Disable existing extract refresh schedules on Tableau Server until you can identify which extracts should and shouldn’t be upgraded.

  **Note:** To access an extract’s refresh schedule, you must be a data source owner, or be a server administrator, site administrator, or project leader.

- Continue to perform extract refresh and append data tasks using Tableau Server 10.5, but maintain an earlier version of Tableau Desktop and the ability to connect to the original data in case you need to recreate the .tde extract.

- Use an earlier version of Tableau Desktop to perform extract tasks, such as extract refresh or append data.

What to expect after extract upgrade

After an extract upgrade has taken place, your users should expect some additional changes when working with extracts in version 10.5. For more information, see After an extract upgrade section in Tableau Help.

View Data Source Attributes

On the Data Sources page, you can filter the view to data sources or connections and see attributes for each.
View data sources by name

To filter by data source name, in the View list, select **Data Sources**.

![Image of Data Sources list]

Attributes in the data source name list include the following:

- **Icon/Name**—The icon next to the data source name indicates whether the data source is published ( ) or embedded in a workbook ( ).
  
  - Select the name of a published data source to open its data source page, with tabs for viewing connections and workbooks that connect to it.
  
  - Select the name of an embedded data source to open the workbook associated with it, as well as tabs for viewing other data it connects to.

- **Connects To**—Indicates the name of the database server or data file. This could be a database outside of Tableau Server, an extract, or a published data source.

- **Live or Last Extract**—This column tells you whether the connection to the data is live, or, if it is a connection to an extract, when the extract was last updated.

View a list of connections

To filter by connection type, in the View list, select **Connections**.
Connection attributes include:

- **Connects to**—Indicates the name of the database server or data file. This could be a database outside of Tableau Server, an extract, or a published data source.

- **Connection type**—Shows the type of data. **Tableau Server** indicates that the connection is to a data source published on the site. **Tableau Data Engine** means the data source has an extract stored on the Tableau data server.

**Use Certification to Help Users Find Trusted Data**

In a self-service environment with multiple publishers, it’s common for a project on Tableau Server to contain a variety of data sources that are named similarly, or are based on the same or similar underlying data, or are published without any descriptive information about them. When this is the case, analysts who create views in Tableau Desktop or web authoring can’t be confident knowing which data source is the right one to connect to.

To help your users find the data that’s trusted and recommended for their type of analysis, you can **certify** the data sources that comply with your organization’s data standards.

Certification complements the Recommendations Trainer feature, by offering a way to promote data sources through curation.

**In this article**

How certification helps users find trusted data
Who can certify data sources
Certify a data source

How certification helps users find trusted data

When you certify a data source, a certification badge appears next to it on the Data Sources page, and in data source lists, such as when connecting to data in web authoring or from Tableau Desktop.

Certified data sources rank higher in search results and are added to recommended data sources.

In addition, you can provide notes about the certification status, which appear in a tooltip when users hover over the certification badge. The tooltip also shows who certified the data source and the data source’s connection type.

For more information, see the Certify a data source steps below.
Create guidelines for selecting data sources to certify

As with most Tableau functionality, certification is flexible. You can define for your organization the criteria you use to determine when to certify a data source. As you do this, you might want to document and share your guidelines. As new data sources are published, the guidelines can help you and other administrators or project leaders to be consistent with your certification choices. They can also help users understand what certification means.

Whether you use the same certification criteria across all projects, or define unique criteria for each project, the important thing is to be clear about what certification means in your environment.

Who can certify data sources

To access data source certification, a user must have either of the following permissions levels on Tableau Server:

- Site Administrator site role.
- Publisher site role with the Project Leader permissions role on the project that contains the data sources.

Certify a data source

1. Sign in to Tableau Server and select the Data Sources tab.

2. Select the name of the data source, and then select the Details tab.

3. Next to Certification, select Edit Certification Status, and then do the following:
   a. Select the This data is certified check box.
   b. Add a note that gives users context for the certification status, intended use for the data, or other helpful information.
Information you add to the Note section appears in the certification badge tooltip, mentioned earlier in How certification helps users find trusted data.

Keep Data Fresh

Manage Refresh Tasks

Administrators can change the priority of scheduled extract refreshes relative to other server tasks, manually refresh extracts, or delete their schedules.

1. Sign in to the site that has the schedules you want to manage, and then click Tasks.

2. Select one or more scheduled extract refreshes.

3. From the Actions menu, do any of the following:

   • Select Change Schedule, and choose a new schedule from the list.

   • Select Run Now to refresh manually.

     Note: If an extract does not have a scheduled refresh, you can refresh it on demand from the Data Connections page.

   • Select Change Priority, and enter a number between 1 and 100 to move the extract up or down in the priority list.
Select **Delete** to completely remove the schedule for the selected data sources.

See also

Enable Extract Refresh Scheduling and Failure Notification

**Refresh Data on a Schedule**

You can set scheduled refresh tasks for published data source extracts and published workbooks that connect to extracts.

**Note:** When an extract refresh is performed on extracts created in Tableau 10.4 and earlier (that is, a .tde extract), the extract is upgraded to .hyper extract automatically. While there are many benefits of upgrading to a .hyper extract, your users won't be able open the extract with earlier versions of Tableau Desktop. For more information, see Extract Upgrade to .hyper Format.

1. When you’re signed in to Tableau Server, display **Content > Data Sources** or **Content > Workbooks**, depending on the type of content you want to refresh.

2. Select the check box for the data source or workbook you want to refresh, and then select **Actions > Extract Refresh**.

3. In the Refresh Extracts dialog, select **Schedule a Refresh**, and complete the following steps:
- Select the schedule you want.

- If available, specify whether you want a full or incremental refresh.

By default, and if this option is not shown, a full refresh is run. Incremental refresh is available only if you configured it in Tableau Desktop before publishing the extract. For information, see Refreshing Extracts in the Tableau Help.

- Click the Schedule Refresh button.

Note: If you want to add a new schedule, you can do so on the Schedules page.

Quick Start: Manage Incremental Extracts

When you publish a workbook that has an incremental extract, you can associate it with up to two refresh tasks that Tableau Server will handle for you: An incremental refresh of the extract and a full refresh. After you publish the workbook, you or a Tableau Server
administrator can modify any tasks that are associated with the workbook. You can also delete tasks or add more.

1 Publish and Assign a Schedule

In Tableau Desktop, after you create a workbook that uses an extract, go to Server > Publish Workbook, and click Scheduling & Authentication. Next, choose schedules for your refreshes and click OK.

After you publish in Tableau Desktop and choose your refresh schedules, Tableau Server handles the refresh tasks for you.
2 Select the Workbook

To modify a workbook's scheduled task, sign in to Tableau Server and on the **Workbooks** page, select the workbook:

![Workbook Selection](image)

3 Access the Refresh Schedule

Click **Refresh Schedule**.

![Refresh Schedule](image)

Select the check box for the refresh task you want to modify:
4 Edit, Delete, or Add More Tasks

Select the action you want to take—for example, **Change Schedule**—and make your selection. You can also delete the task, change its priority, or add more refresh tasks.

**Quick Start: Refresh Extracts on a Schedule**

For published workbooks that connect to data extracts, you can set up the server to refresh the extracts on a recurring schedule, so all workbooks connected to them always show the most up-to-date data.

To schedule refreshes you need to have administrator or data owner permissions.
Note: This topic applies to extracts published to Tableau Server. For Tableau Online, how you refresh extracts depends on the underlying data they connect to. For more information, see Keep Data Fresh.

1 Set up a schedule on the server

Sign in to the server, go to the Schedules page, and click New Schedule.

Tableau provides a few refresh schedules. You create additional schedules you need.

2 Enable scheduled extract refreshes and failure emails

As a server or site administrator, you can enable schedules, as well as email notification when extract refreshes fail.

Select Settings, and then go to the General page.

- Under Email Notification, select Send email to data source and workbook owners when scheduled refreshes fail.
- Under Embedded Credentials, select both check boxes to allow publishers to
3 Publish a workbook with an extract

In Tableau Desktop, select **Server > Publish Workbook**. Sign in to the server if you’re not already. In the **Publish Workbook to Tableau Server** dialog box, click **Schedules & Authentication**. Under **Extract Schedule**, select the schedule from the list.
If the original data requires authentication, you will also need to select how you want people to access it.

4 Monitor refresh performance

You can monitor scheduled tasks by viewing **Background Tasks for Extracts** on the **Status** page.
Automate Refresh Tasks

You can associate extract refresh tasks with schedules in Tableau Server to automate refreshing extracts. You can also automate extract refreshes using tabcmd, a command line utility that you can download for use with Tableau Server. In particular, you can use the refreshextracts command in combination with other commands in your own script. For example:

```
tabcmd login - http://mytabserver -u jsmith -p P@ssw0rd!
refreshextracts --datasource salesq4
```

**Note:** When an extract refresh is performed on extracts created in Tableau 10.4 and earlier (that is, a .tde extract), the extract is upgraded to .hyper extract automatically. While there are many benefits of upgrading to a .hyper extract, your users won’t be able open the extract with earlier versions of Tableau Desktop. For more information, see Extract Upgrade to .hyper Format.

For information about downloading the tabcmd utility, see tabcmd.

Handle Extract Refresh Alerts

When Tableau Server cannot complete a scheduled refresh, an alert appears to indicate that the refresh has failed. If a scheduled refresh fails five consecutive times, Tableau Server suspends the refresh. When a refresh is suspended, Tableau Server does not try to run it again until someone takes an action that attempts to correct the cause of the failure.

**Note:** The number of consecutive failures for a refresh is set to five by default, but can be changed by a Tableau Server administrator, using the `backgrounder.failure_threshold_for_run_prevention` option. For more information, see tsm configuration set Options.
You will see the Alerts menu only if an extract refresh failed and you are:

- A system or site administrator
- The author of the workbook or data source that couldn't be refreshed
- The author of a workbook that connects to a data source that couldn’t be refreshed

When you open the Alerts menu you can see more information about the refresh failure(s):

When a Data source is listed as Embedded it means that the data source definition (which includes things like the data source credentials or the database name) is embedded, or resides, within the workbook itself, originally created in Tableau Desktop.

When a data source name or workbook name is listed as the Data source (for example, Data source: sales_data), it means that the data source is a Tableau Server data source. The data source definition resides on Tableau Server.

In the Data pane on Tableau Desktop, you can determine whether the data source is on Tableau Server or is local. If the data source is on the server, a Tableau icon is displayed next to the data source name instead of a database icon:
Resolving Extract Refresh Problems

To resolve refresh issues, you can take any of these actions, based on the cause indicated in the alert:

- **Errors related to access token validation or user credentials**

  You can resolve some extract refresh problems by clicking the **Connection Details** in the alert. Select the check box next to the problematic data source, click **Actions > Edit Connection**, and then enter the missing information. Click **Save** when you’re done. After you update the connection information, Tableau Server restarts the refresh schedule.

  If you originally embedded the credentials or other data connection information when you published the workbook or data source from Tableau Desktop, you can also republish the workbook or data source. As part of the publishing process, you can choose to set a new refresh schedule. If you don't choose a new schedule, Tableau Server restarts the existing schedule.

- **Errors that indicate the database was unreachable**

  Confirm that the database is online and that you can sign in to access the data. You can use the **Try again** link in the alert to restart the refresh schedule.

  If the problem cannot be corrected by editing the data connection, you will need to resolve it in Tableau Desktop and republish the workbook.

**Tip:** Administrators can edit data connections at any time on the **Data Connections** page, accessible from each site by clicking the **Content** tab and Data Connections.
Edit Connections on Tableau Server

Administrators and data source owners can manage the information that describes how a published data source connects to the underlying data. This information includes the database server name or address, the server port, the database user name, and whether or not the password is embedded in the connection.

**Note:** Ability to edit connections is determined primarily by your site role, rather than by your permissions on the data source. To edit connections, your site role must be Server Administrator, Site Administrator, or Publisher. If your role is Publisher, you also must be the data source owner.

1. Sign in to the site that has the data sources you want to modify, and open the Data Sources page.

2. Select the name of the data source with the connection you want to update.

   Use the search box or filters on the left to narrow the data source list. The values you type into the Server and Database username fields are treated as regular expressions.

3. In the Connections view, select the check box for the connection, and then click Actions > Edit Connection.
4. Update the connection information.

For **Server name**, if you want to use an IP address, make sure the database or its driver supports that type of connection. If it doesn’t, enter the database server name.

For connections to Google, Salesforce, and web data connector (WDC) data, see Authentication types for Google, Salesforce, and WDC data later in this topic.
5. Click Save.

6. Refresh the browser page for your changes to take effect.

Authentication types for Google, Salesforce, and WDC data

Google BigQuery, Google Analytics, Salesforce.com, and many web data connector (WDC) connections use the OAuth authentication standard, which uses secure access tokens instead of “raw” user name and password credentials. With OAuth, database credentials do not need to be stored in Tableau, and all users connect through this access token, including Tableau Desktop users who want to create or edit workbooks that connect to the data source.

The following sections describe Google and Salesforce connection options. Web data connector options vary, but all involve signing in through the provider’s web-based sign-in form to establish the access token.

Google authentication options

When you edit Google BigQuery or Google Analytics connections, select either of the following options in the Edit Connection dialog box:

- Select Embed Google BigQuery (or Google Analytics) credentials in the connection to authenticate through a designated account, and then select an existing account from the list or select authenticate account now... to add a new one.

  When you add a new account, the Google sign-in page appears. After you provide your database credentials, Google prompts you to confirm Tableau access to the data. When you click Accept, Google returns an access token to use for connecting to the data.

  Note: If you create extracts of your Google data source, select this first option, so that you can schedule refresh tasks.

- Select Prompt user for Google BigQuery/Analytics credentials to require users
to connect through their own individual access tokens or sign in each time they connect.

Salesforce.com authentication options

When you edit Salesforce.com connections, you can select any of the following options in the Edit Connection dialog box:

- Select **Embed a Salesforce username and password** to use a traditional authentication method.

- Select **Embed Salesforce credentials in the connection** to use an OAuth connection and schedule refresh tasks, and then select an existing account from the list or click **Add a Salesforce Account** to add a new one.

When you add a new account, the Salesforce.com sign-in page appears. After you provide your database credentials, Salesforce.com prompts you to confirm Tableau access to the data. When you allow Tableau access, Salesforce.com creates an access token through which it connects to the data.
• Select **No Salesforce authentication** to require users to sign in to Salesforce.com each time they connect. (This option does not allow scheduled extract refreshes.)

**Monitor progress**

When you save your changes in the Edit Connection dialog box, the dialog displays the progress. If you close the dialog box, the modifications continue to run in the background until completed. Tableau Server will make as many changes as possible. Any failures will be skipped, but they will not impede other changes. For example, if you try to change the server name and add a password to several connections, the server names will be changed, and the passwords on workbooks will be changed. However, because you cannot add a password to a data source, the passwords for the data sources will not be changed.

For information about checking the progress of these tasks, see Background Tasks for Extracts.
Cube Data Sources

Cube (multidimensional) data sources have certain characteristics that make them unique in Tableau.

Cube data sources do not support pass-through connections. This means that when a cube data source is published, you cannot make a connection from Tableau Server using the data source. It also means you cannot create a workbook using the data source in Tableau Server.

Publishing a cube data source to Tableau Server gives you the ability to store the data source on the server. However, to use the data source, you must download the data source to Tableau Desktop and use it locally. To download a published data source you need:

- The **Download/Web Save As** permission for the data source. For more information, see Set Permissions on Individual Content Resources.
- Correct drivers installed and ports opened on computer running Tableau Desktop.

Web Data Connectors in Tableau Server on Linux

Web data connectors (WDCs) are web pages that provide a data connection that is accessible over HTTP for data sources that don’t already have a connector in Tableau. Web data connectors allow users to connect to almost any data that is accessible over the web and to create extracts for their workbooks. Data sources for a web data connector can include internal web services, JSON data, REST APIs, and other sources that are available over HTTP or HTTPS. Users can create their own web data connectors or use connectors that were created by others.

For information about how to use a web data connector in Tableau Desktop, see Web Data Connector in the Tableau Desktop documentation.

For information about how to create a web data connector, see the Web Data Connector documentation on Github.
Before you run connectors on Tableau Server

- Manage connectors in a safe list
- Refresh the extract for a connector
- Troubleshooting

Before you run connectors on Tableau Server

As a security measure, Tableau Server for Linux won't run web data connectors unless you approve the connector, as explained in this topic.

**Note:** You must be a server administrator to approve web data connectors for use on Tableau Server.

Web data connectors require your approval because they contain executable code and typically make requests to third-party websites. Before a user can use a web data connector with Tableau Server, you must either add the domain and port used by the connector to a safe list (whitelist) and also include the domains that a connector can send requests to and receive requests from on a secondary safe list (secondary whitelist). Before you do this, we recommend that you vet and test the connector so that you know what the connector does and what sites it connects to. For more information, see Testing and Vetting Web Data Connectors.

When you add a connector to the safe lists (whitelists), you configure Tableau Server to allow connections to a particular URL where the connector is hosted and from a URL which the connector can query. This is the only way to allow Tableau Server to run web data connectors. The connectors can then be hosted on a server inside your organization's firewall or on an external domain. Importing web data connectors is not supported for Tableau Server on Linux.
Manage connectors in a safe list

To add a web data connector to the safe list, use the `tsm configuration set -k webdataconnector.whitelist.fixed` command. To list web data connectors, use the `tsm configuration get -k webdataconnector.whitelist.fixed` command. These commands let you perform the following tasks:

- Add connectors to the safe list and secondary safe list.
- List connectors on the safe list.
- Remove a connector from the safe list.
- Remove all connectors from the safe list.
- Configure a secondary safe list, that is, a list of domains that a particular connector can send requests to and receive requests from.

Add connectors to the safe list and secondary safe list

To add a connector to the safe list and secondary safe list, you update the `web-dataconnector.whitelist.fixed` key with a value that is formatted as JSON data on a single line (with double-quotes escaped). This value must include all web data connector domains that you want to use on Tableau Server, and their associated safe lists. After updating the key, you need to apply your pending changes using the `tsm pending-changes apply` command.

The format used for this JSON for a single web data connector is as shown below:

```
""{"<scheme>://<host>:<port>/<path>": {"properties": { "secondary_whitelist": ["(<scheme>://<host>/<path>)] } } }
'"'
```

To add two or more web data connectors, you would add a set of braces to enclose the safelists and secondary safe lists for the complete set of web data connectors, and repeat the JSON syntax shown above. So, to add two web data connectors, the syntax is as follows (broken across lines for readability):
"\"<scheme>://<host>:<port>/<path>\": {"properties": { "secondary_whitelist": ["(\"<scheme>://<host>/<path>\")"] } },
"<scheme>://<host>:<port>/<path>\": {"properties": { "secondary_whitelist": ["(\"<scheme>://<host>/<path>\")"] } } }
"

Notes on formatting:

- The <scheme> value (http or https) must match between the safe list and the secondary safe list.
- If a web data connector uses multiple domains to send requests to and receive requests from, set the secondary safe list value to ["(.*)"].
- For many web data connectors, the <port> value is 443 or 80, but you can check the value for your connector by looking at the data source details on Tableau Server.
- The <path> parameter can use a wildcard (*).
- The entire JSON value must be on a single line, with all double-quotes (") escaped using a backslash (\).
- Be sure to use straight quotes (" and "), not curly or "smart" quotes, when formatting the JSON value.

For example, use these commands to add the San Francisco Film Locations web data connector, or any other web data connectors from data.world, to the safe list and secondary safe list:

tsm configuration set --key webdataconnector.whitelist.fixed --value "'\"https://tableau.data.world:443\": {"properties": { "secondary_whitelist": ["(https://data.world/){.*}\"] } }'"

tsm pending-changes apply

The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the -r flag.
option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

List connectors on the safe list

To list web data connectors, use the following command:

tsm configuration get --key webdataconnector.whitelist.fixed

Remove a connector from the safe list

To remove one or more web data connectors from the safe list, list the web data connectors as shown above, edit the JSON provided to remove the web data connectors that you want to remove, and use tsm configuration set to update the JSON as shown above.

For more information, see tsm configuration set Options.

Remove all connectors from the safe list

To remove all web data connectors from the safe list, set the value for web-dataconnector.whitelist.fixed to a null string:

```bash
tsm configuration set --key webdataconnector.whitelist.fixed --value ""
tsm pending-changes apply
```

The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the -r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

Refresh the extract for a connector

When a user creates a workbook that uses a web data connector, Tableau creates an extract from the data returned by the connector. If the user then publishes the workbook, the publish process sends the workbook and the data extract to the server.
Tableau can refresh an extract that was created by a web data connector, the same as it can refresh any extract. If the connector requires credentials to sign in to the web-based data source, you need to ensure that the credentials are embedded with the data source, and that the web data connector is on the safe list for the server. Tableau Server cannot refresh the extract if the connector requires credentials and they are not embedded with the data source. This is because the refresh can occur on a schedule or in some other background context, and the server cannot prompt for credentials.

Currently, there is no way to re-authenticate a data source from Tableau Server directly. If the data source has credentials that expire, or was published without embedding the credentials, the workbook and data extract need to be published again with the new embedded credentials.

If the background process that performs the refresh operation fails, it creates an alert and a log entry that indicates this issue. Users will be able to see that the timestamp on the extract does not change.

To disable refresh for all web data connectors, use the `tsm configuration set` command to change the `webdataconnector.refresh.enabled` setting to `false`, as in the following example:

```
tsm configuration set --key webdataconnector.refresh.enabled --value false
```

Troubleshooting

If the server experiences problems with adding connectors to the safe list or importing connectors, you can examine the `tabadmincontroller.log` file. Be sure to check the log files on both the initial server node and on the other nodes that are running the gateway process. For more information about log files, see Server Log File Archive File Locations.

If the issue is that Tableau Server will not refresh an extract that was created by a web data connector, make sure that the `webdataconnector.refresh.enabled` configuration setting has been set to `true`. 
If you have re-imported a changed web data connector on the server (overwriting an existing one), but users who work with the web data connector are not seeing the changes, the users might be getting a cached version of the older version. By default, the cache is reset after eight hours; after a cache reset, older versions of the web data connector will no longer be used. If you want to force the cache to reset, you can restart the server.

If you have deleted an imported connector from the server but users are still able to work with the connector, the connector is probably still in the server’s cache. A web data connector can stay available in the cache for up to eight hours. To clear the cache, restart the server. If you delete a web data connector from a server in a distributed environment, make sure that the connector has been successfully deleted from all computers where a gateway process is running.

Testing and Vetting Web Data Connectors

Web Data Connectors (WDCs) contain JavaScript that typically connects to data on another site. Because of this, you should test and vet web data connectors before users use them as data sources for a workbook, and before you use them with Tableau Server.

This topic includes some suggestions for testing and vetting web data connectors.

- Examine the source
- Test the web data connector in an isolated environment
- Monitor the traffic created by the connector
- Test the performance and resource usage of the connector

Examine the source

The code in a web data connector is in JavaScript, so you can open the file (and any external files that the connector uses) and examine the source code.

Many connectors reference external JavaScript libraries, such as the jQuery library or API libraries for third parties. Validate that the URL for external libraries points to a trusted location.
for the library. For example, if the connector references the jQuery library, make sure that the library is on a site that is considered standard and safe. If it is practical for you to change the source code of the connector, use HTTPS protocol (https://) to reference external libraries (if the source site supports HTTPS) to help verify the site’s authenticity.

To the extent possible, make sure you understand what the code is doing. In particular, try to understand how the code is constructing requests to external sites, and what information is being sent in the request.

**Note:** Experienced JavaScript programmers often compress (minify) their code to reduce the size of the code for download. Dense blocks of code that use cryptic function and variable names are not uncommon. While this can make it more difficult to examine the code, it is not a sign that the code was written to be deliberately difficult to understand.

**Test the web data connector in an isolated environment**

If possible, test the web data connector in an environment that is isolated from your production environment and from user computers. For example, add a web data connector to a safe list on a test computer or virtual machine that's running a version of Tableau Server that is not used for production.

**Monitor the traffic created by the web data connector**

When you test a web data connector, use a tool like Fiddler, Charles HTTP proxy, or Wire-shark to examine the requests and responses that the connector makes. Make sure that you understand what sites the connector makes requests to and what content the connector is requesting. Similarly, examine the responses and their content to be sure that the connector is not reading data or code that is not directly related to the connector’s purpose.
Test the performance and resource usage of the web data connector

When you test a web data connector, use tools to monitor its CPU and memory usage. Remember that the web data connector will run on Tableau Server, which is an environment in which many processes are already running. You want to make sure that when the connector fetches data, the connector does not have an undue impact on server performance.

Check whether the connector writes to disk. If it does, check how much disk space it occupies, and examine the output to make sure you understand what it’s writing and why.

Navigate Site Admin Pages

When you sign in to Tableau Server as a site administrator, you see menus for site and content management that are not available to other users.

A site administrator can:

- Manage content and assign permissions.
- View and manually run schedules for extract refreshes and subscriptions.
- Manage extract refreshes and subscriptions.
- Add and manage site users (if allowed by the server administrator; see Add or Update Sites).
- Add and manage site groups.
- Monitor site activity.

For information about navigating content (workbook, data source, project) pages in Tableau Server, see Navigate Tableau on the Web and Manage Web Content in the Tableau Help.

Site administrator pages

Site administrators manage all site content and access on a per-site basis.
If you work with a single site on Tableau Online, or your Tableau Server deployment has one site, you’ll see menus for managing that site’s content, users, groups, schedules, and tasks, and for monitoring its status.

If you work with multiple sites, the menus across the top include a site menu on the left side.

The site menu displays the name of the current site. To go to another site, select the site menu, and then select the site name.

Create, Interact with, and Embed Views on the Web

You can create and interact with views on Tableau Server. For more information, see the following topics in the Tableau Help for users.
Using Tableau on the Web

Get Started with Web Authoring

Connect to Published Data Sources While Web Editing

Build Data Views on the Web

Create a Dashboard

Create a Story

Embed Views and Dashboards in Web Pages
Security

Authentication

Authentication verifies a user’s identity. Everyone who needs to access Tableau Server—whether to manage the server, or to publish, browse, or administer content—must be represented as a user in the Tableau Server identity store. The method of authentication may be performed by Tableau Server (“local authentication”), or authentication may be performed by an external process. In the latter case, you must configure Tableau Server for external authentication technologies such as Active Directory, SAML, or OpenID. In all cases, whether authentication takes place locally or is external, each user identity must be represented in the Tableau Server identity store, which is managed by the repository.

Looking for Tableau Server on Windows? See Authentication.

Access and management permissions are implemented through site roles. Site roles define which users are administrators, and which users are content consumers and publishers on the server. For more information about administrators, site roles, groups, Guest User, and user-related administrative tasks, see Users and Site Roles for Users.
Note: In the context of authentication, it’s important to understand that users are not authorized to access external data sources through Tableau Server by virtue of having an account on the server. In other words, in the default configuration, Tableau Server does not act as a proxy to external data sources. Such access requires additional configuration of the data source on Tableau Server or authentication at the data source when the user connects from Tableau Desktop.

Local authentication

If the server is configured to use local authentication, then Tableau Server authenticates users. When users sign-in and enter their credentials, either through Tableau Desktop, tabcmd, API, or web client, Tableau Server verifies the credentials.

To enable this scenario, you must first create an identity for each user. To create an identity, you specify a username and a password. To access or interact with content on the server, users must also be assigned a site role. User identities can be added to Tableau Server in the server UI, using tabcmd Commands, or using the REST API.

You can also create groups in Tableau Server to help manage and assign roles to large sets of related user groups (e.g., “Marketing”).

When you configure Tableau Server for local authentication, you cannot set password policies or account lockout on failed password attempts. If you require these account safeguards, then you should use Active Directory or other supported authentication.

Active Directory

If Tableau Server is configured to use Active Directory authentication, then credentials are managed and verified by Active Directory. When a user logs onto Tableau Server from Tableau Desktop or a web client, the credentials are passed through to Active Directory, which then verifies them and sends an access token to Tableau Server. Tableau Server will then manage user access to Tableau resources based on the site roles stored in the local identity store.
In this scenario, Tableau Server must be installed in a domain in Active Directory. Tableau Server will sync user and group metadata from Active Directory to the identity store. You do not have to manually add users. However, after the data is synchronized, you will need to assign site and server roles. You can assign these individually, or at the group level. Tableau Server does not synchronize any data back to Active Directory. Tableau Server manages content and server access according to the site role permission data is stored in the repository.

If you are already using Active Directory to manage users in your organization, then we recommend selecting Active Directory authentication during Tableau setup to make user provisioning and management easier. For example, by synchronizing Active Directory groups, you can set minimum site role Tableau permissions for users that are synchronized in the groups. You can synchronize specific Active Directory groups, or you can synchronize them all. For more information, see Synchronize All Active Directory Groups on the Server.

Be sure to review User Management in Active Directory Deployments to understand how multiple domains, domain naming, NetBIOS, and Active Directory user name format influence Tableau user management.

LDAP identity store

You can also configure Tableau Server to use LDAP as a generic way to communicate with the identity store. For example, OpenLDAP is one of several LDAP server implementations with a flexible schema. Tableau Server can be configured to query the OpenLDAP server. See Identity Store. Authentication in this scenario maybe be provided by the native LDAP solution, or with one of the single sign-on solutions discussed later in this topic.

Data connection authentication

You can configure Tableau Server to support a number of different authentication protocols to various different data sources. Data connection authentication may be independent of Tableau Server authentication. For example, you may configure user authentication to Tableau Server using local authentication, while configuring Kerberos delegation, OAuth, or SAML authentication to specific data sources. See Data Connection Authentication.
Other authentication options for Tableau Server

Tableau Server supports several types of single sign-on (SSO) and other authentication methods. In most SSO scenarios, users don't have to explicitly sign in to Tableau Server. Instead, the credentials they've used to authenticate already (for example, by signing in to your corporate network) are used to authenticate them to Tableau Server, and they can skip the step of entering a username and password to access Tableau Server. In the following authentication methods, users' identities as established externally are mapped to a user identity defined in the Tableau Server identity store.

When you configure Tableau Server to use an SSO or alternative authentication solution, all authentication is handled by the external solution. However, Tableau Server will manage user access to Tableau resources based on the site roles stored on Tableau Server.

Tableau Server supports these authentication methods:

- **SAML.** You can configure Tableau Server to use SAML (security assertion markup language) for SSO. With SAML, an external identity provider (IdP) authenticates the user's credentials, and then sends a security assertion to Tableau Server that provides information about the user's identity.

  You can use SAML to access Tableau Server if you have configured Active Directory or local authentication on Tableau Server. For more information, see SAML.

- **Kerberos.** If Kerberos is enabled in your environment and if the server is configured to use Active Directory authentication, you can provide users with access to Tableau Server based on their Windows identities. For more information, see Kerberos.

- **Mutual SSL.** Using mutual SSL, you can provide users of Tableau Desktop, Tableau Mobile, and other approved Tableau clients a secure, direct-access experience to Tableau Server. With mutual SSL, when a client with a valid SSL certificate connects to Tableau Server, Tableau Server confirms the existence of the client certificate and authenticates the user, based on the user name in the client certificate. If the client does not have a valid SSL certificate, Tableau Server can refuse the connection. For
more information, see Configure Mutual SSL Authentication for Tableau Server on Linux.

- **OpenID.** OpenID Connect is a standard authentication protocol that lets users sign in to an identity provider (IdP) such as Google. After they've successfully signed in to their IdP, they are automatically signed in to Tableau Server. To use OpenID Connect on Tableau Server, the server must be configured to use local authentication. Active Directory authentication is not supported. For more information, see OpenID Connect.

- **Trusted Authentication.** Trusted authentication lets you set up a trusted relationship between Tableau Server and one or more web servers. When Tableau Server receives requests from a trusted web server, it assumes that the web server has already handled whatever authentication is necessary. Tableau Server receives the request with a redeemable token or ticket and presents the user with a personalized view which takes into consideration the user's role and permissions. For more information, see Trusted Authentication.

**Related topic**

- **REST API:** Signing In and Out (Authentication)

**SAML**

SAML (Security Assertion Markup Language) is an XML standard that allows secure web domains to exchange user authentication and authorization data. You can configure Tableau Server to use an external identity provider (IdP) to authenticate users over SAML 2.0. No user credentials are stored with Tableau Server, and using SAML enables you to add Tableau to your organization's single sign-on environment.

You can use SAML server wide, or you can configure sites individually. Here's an overview of those options:
• **Server-wide SAML authentication.** A single SAML IdP application handles authentication for all Tableau Server users. Use this option if your server has only the Default site.

  In addition, if you want to use site-specific SAML, you must configure server-wide SAML before you configure individual sites.

• **Server-wide local authentication and site-specific SAML authentication.** In a multi-site environment, users who are not enabled for SAML authentication at the site level can sign in using local authentication.

• **Server-wide SAML authentication and site-specific SAML authentication.** In a multi-site environment, all users authenticate through a SAML IdP configured at the site level, and you specify a server-wide default SAML IdP for users that belong to multiple sites.

User authentication through SAML does not apply to permissions and authorization for Tableau Server content, such as data sources and workbooks. It also does not control access to underlying data that workbooks and data sources connect to.

**Note:** Tableau Server supports both service provider initiated and IdP initiated SAML in browsers only. Connections from Tableau Desktop or the Tableau Mobile app require that the SAML request be service provider initiated.

The following image shows the steps to authenticate a user with single sign-on in a typical service provider initiated flow:
Step 1  User navigates to the Tableau Server sign-in page or a published workbook.

Step 2  Tableau Server starts the authentication process and redirects the request to the registered IdP.

Step 3  The IdP requests the user’s username and password and authenticates the user to the IdP.

Step 4  The IdP returns a SAML success response to Tableau Server.

Step 5  Tableau Server verifies that the username in the response matches what is stored in Tableau Server and displays the page the user requested in Step 1.

SAML Requirements

Before you configure SAML on Tableau Server, make sure your environment meets the requirements.

In this article

Certificate and identity provider (IdP) requirements
Certificate and identity provider (IdP) requirements

To configure Tableau Server for SAML, you need the following:

- **Certificate file.** A PEM-encoded x509 certificate file with a .crt extension. This file is used by Tableau Server, not the IdP. If you have an SSL certificate, you can use the same certificate with SAML. See About the certificate and key files later in this topic for details.

- **Certificate key file.** An RSA or DSA private key file that is not password protected, and that has the .key extension. This file is used by Tableau Server, not the IdP. The certificate key file must not have the passphrase. If you have an SSL certificate key file, you can use it for SAML as long as it does not have a passphrase. See About the certificate and key files later in this topic for details.

- **IdP account that supports SAML 2.0 or later.** You need an account with an external identity provider. Some examples are PingFederate, SiteMinder, and OpenAM. The IdP must support SAML 2.0 or later.

- **IdP provider that supports import and export of XML metadata.** Although a manually created metadata file might work, Tableau Technical Support cannot assist with generating the file or troubleshooting it.

About the certificate and key files

If you are using a PEM-encoded x509 certificate file for SSL, you can use the same file for SAML. When it’s used for SSL, the certificate file is used to encrypt traffic. When it’s used for
SAML, the certificate is used for authentication.

Tableau Server does not support certificate and certificate key files for SAML if the certificate and key require a chain file. If your SSL certificate and key files require a chain file, you need to generate new certificate and key files to use for SAML.

User management requirements

When you enable SAML, user authentication is performed outside of Tableau, with the IdP. However, the user management is performed either by Active Directory or by Tableau Server (called local authentication even though Tableau Server is not performing authentication in this scenario).

Note: The REST API and tabcmd do not support SAML single-sign (SSO). To sign in, you must specify the name and password of a user who has been created on the server. The user could have a local or Active Directory account, depending on how you have configured Tableau Server. For Tableau Online, you can specify the TableauID credentials of the user. REST API or tabcmd calls will have the permissions of the user you sign in as.

When you configure user authentication, you select the option that reflects how you want to use SAML:

- **For site-specific SAML**: If you have multiple sites on Tableau Server and want to set up each site for a particular IdP or IdP application, configure Tableau Server to use local authentication rather than Active Directory. For site-specific SAML, Tableau Server relies on the IdP for authentication and does not use passwords.

- **For server-wide SAML**: If you configure server-wide SAML with a single IdP, you can configure Tableau Server to use local authentication or Active Directory for user management. If you select Active Directory, you must disable the **Enable automatic logon** option.
SAML compatibility notes and requirements

- **External authentication types**: Tableau Server supports using one external authentication type at a time.

- **Mutual SSL**: Tableau Server does not support mutual SSL (two-way SSL) and SAML together. If you want to use mutual SSL, you can configure it on the IdP.

- **Encryption and site-specific SAML assertions**: Although Tableau Server does not support encrypted SAML assertions from the IdP, all SAML requests and responses are sent over HTTPS.

- **User identity in Tableau Server for tabcmd users**: As described in User management requirements section above, to use tabcmd, you must sign in as a user defined on the server. You cannot use SAML accounts with tabcmd.

- **IdP must support forms-based authentication**: You can allow SAML sign-in from Tableau Desktop or Tableau Mobile client applications. To do this, your IdP must support forms-based authentication.

  If it does not, you can disable SAML sign-in for Tableau clients using the tsm authentication saml configure command. For more information, see tsm authentication saml <commands>.

  See also Using SAML SSO with Tableau client applications.

- **Distributed installations**: Clusters configured for SAML must have the same SAML certificate, SAML key, and SAML IdP metadata files on each Tableau Server that runs an Application Server process.

  For more information, see Configure Server-Wide SAML for Tableau Server on Linux.

- **Login URL**: For users to be able to sign in, your IdP must be configured with SAML Login endpoint that sends a POST request to the following URL:
http(s)://<tableuserver>/wg/saml/SSO/index.html.

- **Logout URL:** To enable users to sign out after signing in with SAML (single logout, or SLO), your IdP must be configured with a SAML Logout endpoint that sends a POST request to the following URL:

  http(s)://<tableuserver>/wg/saml/SingleLogout/index.html.

- **Post-logout redirect URL:** By default, when a user signs out of Tableau Server, the sign-in page is displayed.

  To display a different page after sign-out, use the `tsm authentication saml configure` command with the `-su` or `--signout-url` option.

  - To specify an absolute URL, use a fully-qualified URL starting with `http://` or `https://`, as in this example:

    tsm authentication saml configure --signout-url https://example.com

  - To specify a URL relative to the Tableau Server host, use a page starting with a `/` (slash):

    tsm authentication saml configure --signout-url /our-logoutpage.html

- **Active Directory Federation Service (AD FS):** You must configure AD FS to return additional attributes for Tableau authentication with SAML. The **Name ID** and **username** attributes can be mapped to the same AD attribute: `SAM-Account-Name`.

Using SAML SSO with Tableau client applications

Tableau Server users with SAML credentials can sign in to the server from Tableau Desktop or the Tableau Mobile app. For full compatibility, we recommend that the Tableau client application version matches that of the server. To connect using site-specific SAML, users must run version 10.0 or later of the Tableau client application.
Connecting to Tableau Server from Tableau Desktop or Tableau Mobile uses a service provider (SP) initiated connection.

Preserving the Tableau client RelayState value

When a user signs in to Tableau Server, Tableau Server sends a SAML request (AuthnRequest) to the IdP, which includes the Tableau application’s RelayState value. If the user has signed in to Tableau Server from a Tableau client such as Tableau Desktop or Tableau Mobile, it’s important that the RelayState value is returned within the IdP’s SAML response back to Tableau.

When the RelayState value is not returned properly in this scenario, the user is taken to their Tableau Server home page, rather than being redirected back to the application they signed in from.

Work with your Identity Provider and internal IT team to confirm that this value will be included as part of the IdP’s SAML response, and then preserved by any network appliance (such as a proxy or load balancer) that resides between your IdP and Tableau Server.

XML data requirements

You configure SAML using XML metadata documents that are generated by Tableau Server and by the IdP. During the authentication process, the IdP and Tableau Server exchange authentication information using these XML documents. If the XML does not meet the following requirements, errors can occur when you configure SAML or when users try to sign in.

- **HTTP POST**: Tableau Server accepts only HTTP POST requests for SAML communications. HTTP Redirect is not supported.

  The SAML metadata XML document that is exported by Tableau Server should contain the following elements, with the Binding attribute set to HTTP-POST.

  - Verify the following element which specifies the URL that the IdP redirects to after successful authentication:
<md:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="http(s)://TABLEAU-SERVER/wg/saml/SSO/index.html index="0" isDefault="true"/>

- Verify the following element which specifies the URL that the IdP will use for the logout endpoint:

<md:SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="http(s)://<tableauserver>/wg/saml/SingleLogout/index.html/>

- Verify the following element which specifies the URL for signin in:

<md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="http(s)://<tableauserver>/wg/saml/SSO/index.html/>

- **Attribute named *username***: You must configure the IdP to return an assertion that includes the `username` value in the `saml:AttributeStatement` element. The assertion's attribute type must be `xs:string` (it should *not* be typed as `xs:any`).

The following example shows what this might look like.

```xml
<saml:Assertion assertion-element-attributes>
  <saml:Issuer>issuer-information</saml:Issuer>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    ...
  </Signature>
  <saml:Subject>
    ...
  </saml:Subject>
  ...
```
To change the SAML attribute that passes the `username` value, use the following command:

```
tsm authentication saml map-assertions --user-name <USER-NAME>
```

- **Matching usernames**: The user name stored in Tableau Server must match the user name stored in the IdP. For example, if the user name for Jane Smith is stored in PingFederate as `jsmith`, it must also be stored in Tableau Server as `jsmith`. 
If you are configuring SAML as part of the initial Tableau Server setup, make sure the account you plan to use exists in your IdP before you run setup. During Tableau Server setup you create the server administrator account.

If you use Active Directory authentication with Tableau Server and have multiple Active Directory domains (that is, users belong to multiple domains, or your Tableau Server installation includes multiple domains), the IdP must send both the user name and domain for a user, and they must match exactly in Tableau Server. Although these can be sent as either domain/username or username@domain.com, we recommend using domain/username.

Configure Server-Wide SAML for Tableau Server on Linux

Configure server-wide SAML when your Tableau Server has only the Default site, or as the first step to configuring site-specific SAML in a multi-site environment.

Looking for Tableau Server on Windows? See Configure Server-Wide SAML.

This article makes the following assumptions:

- You are familiar with the options for configuring SAML authentication on Tableau Server, as described in the SAML overview.

- You have verified that your environment meets the SAML Requirements, and obtained the SAML certificate files described in those requirements.

SAML configuration steps

1. Put certificate and IdP metadata files in place — This section includes information for configuring SAML for a Tableau Server cluster.

2. Choose the configuration method — use the SAML entity (recommended for first-time configuration) or TSM CLI commands.

3. Generate Tableau Server metadata and configure the IdP — Add Tableau Server as a
Service Provider.

4. Test the configuration — Sign in to the Tableau Server web UI.

1. Put certificate and IdP metadata files in place

In this phase, you create required XML metadata files, and gather the IdP metadata file and copies of the SAML certificate and key files into a new `saml` directory on Tableau Server.

To review the certificate file requirements again, see SAML Requirements.

1. **Create a new `saml` directory in the `tableau_server/data` directory, in which you keep the metadata and certificate files. For the rest of this process, we’ll use the following location:**

   `/var/opt/tableau/tableau_server/data/saml`

   **Note:** If you use the same certificate files for SSL, you could alternatively use the existing certificate location for configuring SAML, and add the IdP metadata file to that directory when you download it later in this procedure. For more information, see About the certificate and key files in the SAML requirements.

2. **Place a copy of the SAML certificate and key files in the new `saml` directory. (Keep the certificate files in a safe location outside of the Tableau Server directory tree as well.)**

   When you create a `samlSettings` configuration template (or use the CLI commands) to configure the server to support SAML, you’ll specify this location for the SAML entity keys (or CLI parameters) that specify the certificate and key files.

3. **On the IdP’s website or in its application, export the IdP’s metadata XML file.**

   Confirm that the metadata XML from the IdP includes a `SingleSignOnService` element, in which the binding is set to `HTTP-POST`, as in the following example:
4. Copy the IdP’s metadata file to the Tableau Server saml directory.

5. If you’re running a Tableau Server cluster, complete the following steps on each node that runs an application server process (vizportal.exe):
   
a. Repeat the earlier step to create a /var/opt/tableau/tableau_server- /data/saml directory (or use the existing certificate location if you use the same certificate files for SSL).

b. Copy the SAML certificate, key, and IdP metadata file from the initial node’s saml directory to the same location on the subsequent nodes.

After you copy the files to the subsequent nodes, you do not need to do any additional configuration on those nodes.

6. If you choose the JSON configuration method (recommended), for the idpMetadataFile, certFile, and keyFile key values, enter the path to the saml directory you created, along with the appropriate file name.

   For example, for idpMetadataFile, you might enter /var/-
   opt/tableau/tableau_server/data/saml/idp-metadata.xml

   For more information, see the SAML configuration entity reference.

2. Choose the configuration method

   You can configure SAML one of the following ways:

   • Create a configuration file that you pass as a parameter value.

   For initial SAML configuration, we recommend that you use a .json configuration file, based on a template that we provide. This is a multi-step process, but it is more con-
venient for providing multiple required configuration values. You can also save it for future configuration updates.

For the configuration template and steps to customize it for your environment, see samlSettings Entity.

- **Use the TSM CLI and specify each parameter on the command line.**

To use the CLI, run the following commands:

1. Using the location you created if you followed the steps in 1. Put certificate and IdP metadata files in place, and including all parameters that are required for initial configuration, configure the SAML settings for the server:

   ```
   ```

2. If SAML is not already enabled on Tableau Server; for example, you're configuring it for the first time, or you have disabled it, enable it now:

   ```
   tsm authentication saml enable
   ```

3. Apply the changes:

   ```
   tsm pending-changes apply
   ```

   The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can
suppress the prompt using the \texttt{-r} option, but this does not change the restart behavior. For more information, see \texttt{tsm pending-changes apply}.

3. Generate Tableau Server metadata and configure the IdP

1. Run the following command to generate the required XML metadata file for Tableau server.

   \texttt{tsm authentication saml export-metadata \text{-f <file-name.xml>}}

   You can specify a file name, or omit the \texttt{-f} parameter to create a default file named \texttt{samlmetadata.xml}.

2. On your IdP’s website or in its application:
   
   \begin{itemize}
   
   \item Add Tableau Server as a Service Provider.
   
   Refer to your IdP’s documentation for information about how to do this. As part of the process of configuring Tableau Server as a Service Provider, you will import the Tableau Server metadata file you generated from the \texttt{export-metadata command}.
   
   \item Confirm that your IdP uses \texttt{username} as the attribute to verify users.
   
   \end{itemize}

4. Test the configuration

1. In your web browser, open a new page or tab, and enter the Tableau Server URL.

   \begin{figure}
   \centering
   \includegraphics[width=\textwidth]{tableau-server-url}
   \caption{The browser redirects you to the IdP’s sign-in form.}
   \end{figure}
2. Enter your single sign-on user name and password.

![Login Form]

- Please sign on first and we'll send you right along.

  **Username**
  
  [jsmith]
  
  **Password**
  
  [●●●●●●●●]

  - Sign On
  - Cancel

The IdP verifies your credentials and redirects you back to your Tableau Server start page.

Configure Site-Specific SAML for Tableau Server on Linux

Use site-specific SAML when you want to enable single sign-on, and you also use multiple SAML identity providers (IdPs) or IdP applications. When you enable site SAML, you can specify the IdP or IdP application for each site.

Looking for Tableau Server on Windows? See Configure Site-Specific SAML.

If you want all server users to use SAML and sign in through the same IdP application, see Configure Server-Wide SAML for Tableau Server on Linux.

In this topic

- Prerequisites for enabling site-specific SAML
- Configure the server to allow site-specific SAML
- Enable SAML on a site
- Configure SAML settings for a site
Prerequisites for enabling site-specific SAML

Before you can enable SAML single sign-on at the site level, complete the following requirements:

- Configure the default server-wide authentication method that can be used with site-specific SAML. This method will apply to users that do not belong to a site, or that belong to multiple sites.

  With site-specific SAML, you can use server-wide local authentication or server-wide SAML authentication. You cannot use Active Directory.

- Make sure your environment and your IdP meet the general SAML Requirements.

- Note the location of the SAML certificate files. For more information, see [Put metadata and certificate files in place](#) in the topic on configuring server-wide SAML.

- Configure the IdP to add Tableau Server as a service provider. You can find this information in the documentation the IdP provides.

Configure the server to allow site-specific SAML

After you complete the prerequisites listed above, you can run the following commands to configure the server to support site-specific SAML. Using these commands, you provide the SAML certificate location and file names, along with the other required attributes. For more information, see [tsm authentication](#).

```
  tsm authentication saml configure --idp-entity-id <id>  --idp-return-url <url> --cert-file </path/to/file.crt> --key-file </path/to/file.key>
  tsm authentication sitesaml enable
  tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is
stopped, but in that case there is no restart. You can suppress the prompt using the –r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

If you want to review what commands and settings will be carried out when you run pending-changes apply, you can run the following command first:

```bash
tsm pending-changes list --config-only
```

The `sitesaml enable` command exposes the **Authentication** tab on each site’s **Settings** page in the Tableau Server web UI. After you configure the server to support site SAML, you can continue to **Enable SAML** on a site to work through the settings on the **Authentication** tab.

Enable SAML on a site

1. Sign in to Tableau Server as a site administrator, and select the site for which you want to enable SAML.

2. Click **Settings**.

3. Click the **Authentication** tab.

4. On the **Authentication** tab, select **Use site-specific SAML**.

Continue to Configure SAML settings for a site.

Configure SAML settings for a site

This section continues from **Enable SAML** section, and it takes you through each of the configuration steps shown on the **Authentication** page in Tableau Server.
Note: To complete this process, you will also need the documentation your IdP provides. Look for topics that refer to configuring or defining a service provider for a SAML connection or adding an application.

Step 1: Export metadata from Tableau

To create the SAML connection between Tableau Server and your IdP, you need to exchange required metadata between the two services. To get metadata from Tableau Server, do either of the following:

- Select Export metadata to download an XML file that contains the Tableau Server SAML entity ID, Assertion Consumer Service (ACS) URL, and X.509 certificate.

  The entity ID is site-specific and generated from the value that you specified when you enabled SAML on the server. For example, if you specified https://tableau_server, you might see the following entity ID for the site:

  https://tableau_server-/saml/service/public/sp/metadata?alias=48957410-9396-430a-967c-75b6e02a0

- Select Download signing and encryption certificate if your IdP expects the required information in a different way. For example, if it wants you to enter the Tableau Server entity ID, ACS URL, and X.509 certificate in separate locations.
See the IdP’s SAML configuration steps to confirm the correct option.

Steps 2 and 3: External steps

For Step 2, to import the metadata you exported in step 1, sign in to your IdP account, and use the instructions provided by the IdP’s documentation to submit the Tableau Server metadata.

For Step 3, the IdP’s documentation will guide you also in how to provide metadata to a service provider. It will instruct you to download a metadata file, or it will display XML code. If it displays XML code, copy and paste the code into a new text file, and save the file with a .xml extension.

Step 4: Import IdP metadata to the Tableau site

On the Authentication page in Tableau Server, import the metadata file that you downloaded from the IdP or configured manually from XML it provided.

Step 5: Match attributes

Attributes contain authentication, authorization, and other information about a user. In the Identity Provider (IdP) Assertion Name column, provide the attributes that contain the
information Tableau Server requires.

- **Username or Email**: (Required) Enter the name of the attribute that stores user names or email addresses.

- **Display name**: (Optional but recommended) Some IdPs use separate attributes for first and last names, and others store the full name in one attribute.

  Select the button that corresponds to the way your IdP stores the names. For example, if the IdP combines first and last name in one attribute, select **Display name**, and then enter the attribute name.

**Step 6: Manage users**

Select existing Tableau Server users, or add new users you want to approve for single sign-on.

When you add or import users, you also specify their authentication type. On the Users page, you can change users’ authentication type any time after adding them.

For more information, see Add Users to a Site or Import Users and Set the User Authentication Type for SAML.

**Important**: Users that authenticate with site-specific SAML can belong only to one site. If a user needs to access multiple sites, set their authentication type to the server default. Depending on how site-specific SAML was configured by the server administrator, the server default is either local authentication or server-wide SAML.

**Default authentication type for embedded views**

Part of enabling SAML on your site is to specify how users sign in to access views
embedded in web pages.

- **Allow users to choose their authentication type**

  When you select this option, two sign-in options appear where a view is embedded: a sign-in button that uses single sign-on authentication and a link to use TableauID as an alternative.

  **Tip:** With this option, users need to know which alternative to choose. As part of notification you send your users after you add them to the single sign-on site, let them know which type of authentication to use for a variety of sign-in scenarios. For example, embedded views, Tableau Desktop, Tableau Bridge, Tableau Mobile, and so on.

- **TableauID**

  This option requires users to sign in using a TableauID even if SAML is enabled on the site. Generally it’s reserved for administrators for troubleshooting issues with embedded views and SAML.

- **Single sign-on with SAML**

  If your IdP doesn’t support signing in from an iframe, select **Authenticate in a separate pop-up window**. When a user goes to the web page with the embedded view, the pop-up window appears when they select the sign-in button.

  If your IdP supports signing in from an iframe, select **Authenticate using an inline frame (less secure; not supported by all IdPs)**. Iframe embedding can provide a more seamless user experience. For example, if a user is already authenticated with your IdP, and iframe embedding is enabled, the user seamlessly authenticates with Tableau Server when browsing to pages that contain embedded Tableau views.
Caution: Because iframes can be vulnerable to clickjacking attacks, not all IdPs support signing in through an iframe. With clickjacking, the attacker tries to lure users into clicking or entering content. They do this by displaying the page to attack in a transparent layer over an unrelated page. For Tableau Server, an attacker might try to capture user credentials or to get an authenticated user to change settings. For more information, see Clickjacking on the Open Web Application Security Project website.

Kerberos

Kerberos is a three-way authentication protocol that relies on the use of a trusted third-party network service called the Key Distribution Center (KDC) to verify the identity of computers and provide for secure connections between the computers through the exchange of tickets. These tickets provide mutual authentication between computers or services, verifying that one has permission to access the other.

Tableau Server supports Kerberos authentication in an Active Directory Kerberos environment, with authentication to Tableau Server being handled by Kerberos.

Note: The Kerberos support in Tableau Server is for user authentication. It does not handle internal permissions and authorization related to Tableau Server content, such as workbooks.

Kerberos Requirements

You can configure Kerberos authentication for Tableau Server running in either Active Directory or LDAP directory environments.
General requirements

- External Load Balancer/Proxy Server: If you are going to use Tableau Server with Kerberos in an environment that has external load balancers (ELBs) or proxy server, you need to set these up before you configure Kerberos in the Tableau Server Configuration utility. See Configuring Reverse Proxies for Tableau Server.

- iOS Browser Support: An iOS user can use Kerberos authentication with mobile Safari if a Configuration Profile specifying the user’s Kerberos identity is installed. See Configuring an iOS Device for Kerberos Support in the Tableau Mobile Help. For more information about browser support for Kerberos SSO, see Browser Support for Kerberos SSO.

- Tableau Server requires constrained delegation, where the Tableau data access account is specifically granted rights to the target database SPNs. Unconstrained delegation is not supported.

- The supported data sources (SQL Server, MSAS, PostgreSQL, Hive/Impala, and Teradata) must be configured for Kerberos authentication.

- A keytab file that is configured with the service provider name for the Tableau Server for user authentication. If you are using Kerberos authentication for datasources then those credentials should be included in the single keytab file that you will specify during Kerberos configuration on Tableau Server.

Active Directory requirements

You must meet the following requirements to run Tableau Server with Kerberos in an Active Directory environment:

- Tableau Server must use Active Directory (AD) for authentication.

- The domain must be an AD 2003 or later domain for Kerberos connections to Tableau Server.

- Smart Card Support: Smart cards are supported when users sign into their
workstations with a smartcard and this results in a Kerberos TGT being granted to the user from Active Directory.

- Single-Sign On (SSO): Users must be granted a Kerberos Ticket Granting Ticket (TGT) from Active Directory when they sign into their computers. This is standard behavior for domain-joined Windows computers and standard for Mac computers that use AD as their network account server. For more information on using Mac computers and Active Directory, see Join your Mac to a network account server in the Apple Knowledge Base.

**Kerberos delegation**

For Kerberos delegation scenarios the following are required:

- Kerberos must be configured on Tableau Server. See Enable Kerberos Service Account Access.

- If the domain is AD 2003 or later, single domain Kerberos delegation is supported. The users, Tableau Server, and backend database must be on the same domain.

- If the domain is AD 2008, there is limited cross domain support. Users from other domains can be delegated if the following conditions are met. Tableau Server and the backend database must be on the same domain, and a two way trust is required between the domain where Tableau Server resides and the user’s domain.

- If the domain is 2012 or later, full cross-domain delegation is supported. AD 2012 R2 is preferred because it has a dialog for configuring constrained delegation, while 2012 non-R2 requires manual configuration.

Configure Kerberos

You can configure Tableau Server to use Kerberos. This allows you to provide a single sign-on experience across all the applications in your organization. Before you configure Tableau Server for Kerberos make sure you meet the Kerberos Requirements.
To configure Kerberos, you must first enable Kerberos, and then specify a keytab file for user authentication. The keytab file must be named `kerberos.keytab`. The keytab file you specify must be configured with the service provider name for the Tableau Server for user authentication. If you are using Kerberos authentication for data sources, those credentials should be included in the single keytab file that you will specify during Kerberos configuration on Tableau Sever.

1. Copy the keytab file to the computer running Tableau Server and run the following command to set permissions on the file:

   ```bash
   chmod 644 "/path/kerberos.keytab"
   ```

   If you are running Tableau Server on in a distributed cluster deployment, then you will need to manually distribute the keytab file to each node and then set the permissions. Copy the keytab file to the same directory on each node in the cluster. After you have copied the keytab file to each node and set permissions on the file, then run the following TSM commands on one node. The configuration will propagate to each node.

2. Type the following command to specify the location and name of the keytab file:

   ```bash
   tsm authentication kerberos configure --keytab-file 
   </path/kerberos.keytab>
   ```

3. Type the following command to enable Kerberos:

   ```bash
   tsm authentication kerberos enable
   ```

4. Run `tsm pending-changes apply` to apply changes.

   The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`. 

- 484 -
Confirm Your SSO Configuration

Once Tableau Server has restarted, test your Kerberos configuration from a web browser on a different computer by typing the Tableau Server name in the URL window:

You should be automatically authenticated to Tableau Server.

Browser Support for Kerberos SSO

To use browser-based Kerberos Single Sign-on (SSO), the following must be true:

- Kerberos must be enabled on Tableau Server
- The user must have permission to access Tableau Server (they can log in using a user name and password)
- The user must be authenticated to Active Directory through Kerberos on the client computer. Specifically, this means that they have a Kerberos Ticket Granting Ticket (TGT).

**Note:** If Kerberos SSO fails you can still sign in to Tableau Server with your user name and password.

Desktop clients

The following combinations of operating system and Tableau Desktop or browser support Kerberos authentication to Tableau Server. In some cases, additional configuration is required.

Windows
- Tableau Desktop 8.3 or newer
- Internet Explorer - supported, may require configuration, see Note 1
- Chrome - supported, may require configuration, see Note 1
- Firefox - requires configuration, see Note 2
- Safari - not supported

Mac OS X
- Tableau Desktop 8.3 or newer
- Safari - supported
- Chrome - see Note 3
- Firefox - see Note 2
- Internet Explorer - not supported

Mobile clients

The following combinations of operating system and Tableau App or browser support Kerberos authentication to Tableau Server:

Mac iOS
- Tableau App 8.3 or newer - see Note 4
- Safari - see Note 4
- Chrome - not supported

Android - see Note 5
- Tableau App 8.3 or newer
- Android Browser
- Chrome

Note 1: Internet Explorer and Google Chrome on Windows desktop

Kerberos SSO is supported on both Internet Explorer and Google Chrome but requires configuration in Windows Internet Options:
- Enable Integrated Windows Authentication.
- Verify that Tableau Server URL is in the local intranet zone. The Tableau Server URL must be in the local internet zone. In many cases Internet Explorer’s automatic detection of intranet zones will automatically configure this. If the URL has not been detected and configured, then you will need to manually add the URL to the local intranet zone.

To enable Integrated Windows Authentication:

1. In Windows Control Panel, open Internet Options.
2. On the Advanced tab scroll down to the Security section.
4. Click Apply.

To verify or add the Tableau Server URL to the local intranet zone:

1. In Windows Control Panel, open Internet Options.
2. On the Security tab, select Local intranet, and then click Sites.
3. On the Local intranet dialog box, click Advanced.
4. The internal Tableau Server URL should be listed in the Websites field. In some organizations, IT administrators will use a wildcard (*) to specify internal URLs. For example, the wildcard, https://*.example.lan, includes all servers in the internal example.lan namespace in the local intranet zone.
If the Tableau Server URL or a wildcard URL for the Tableau Server namespace is listed in the **Websites** field, then click **Close**. You do not need to update the local intranet zone.

- If the Tableau Server URL or a wildcard URL is not specified in the **Websites** field, Enter the Tableau Server name (URL) in the **Add the website to the zone** field, click **Add**, and then click **OK**.

**Note 2: Firefox on Windows and Mac OS X desktop**

Kerberos SSO is supported in Firefox on both Windows and Mac, but requires that you configure two URL settings to enable Kerberos. Both of those settings are described in the following procedure:

1. In Firefox, enter `about:config` in the address bar.

2. Click **I'll be careful, I promise** when warned about changing advanced settings.

3. Enter `negotiate` in the Search box.
4. Double-click `network.negotiate-auth.allow-non-fqdn`, and then set the value to `true`.

5. Double-click `network.negotiate-auth.trusted-uris` and enter the Tableau Server fully qualified domain name (FQDN). For example, `tableau.example.com` or `example.com`.

**Note 3: Chrome on Mac OS X desktop**

According to Chrome documentation, Kerberos SSO works a Mac when you launch Chrome from a terminal window with the following command:

```
open -a "Google Chrome.app" --args --auth-server-whitelist="tableau-server.example.com"
```

where `tableauserver.example.com` is the URL for Tableau Server in your environment.

However, we have found inconsistent results in our testing and therefore recommend using Safari or Firefox if you want to use Kerberos SSO on a Mac. See the Integrated Authentication section at [HTTP authentication](http://The Chromium Projects site) for more information.
**Note:** Chrome on Mac OS X continues to be supported with Tableau Server, but users may be prompted to log in with their user name and password (single sign-on may not work).

---

**Note 4: Mobile Safari and Tableau App on Mac iOS**

Kerberos SSO is supported if iOS is configured for Kerberos. The iOS device must have a Kerberos authentication configuration profile installed. This is usually done by an enterprise IT group. Tableau Support cannot assist with configuring iOS devices for Kerberos.

**Note 5: Android platform**

Kerberos SSO is not supported on the Android operating system because there is no platform-level support for Kerberos. You can still use your Android device and the Tableau App or a supported browser to connect to and log in to Tableau Server.

**Troubleshoot Kerberos**

The troubleshooting suggestions in this topic are divided into issues related to Single sign-on (SSO) on the server and issues with the delegated data sources.

**Single Sign-on to Tableau Server**

**Kerberos Authentication Failed (unable to connect automatically to Tableau Server)**

If you are using Kerberos for SSO and a user is prompted to sign in to Tableau Server when they connect with either a web browser or with Tableau Desktop, try these steps from the client computer:
Troubleshooting on the client computer

- **Account permissions**—Try to sign in to Tableau Server using the user's name and password. If they can't sign in to Tableau Server using their user name and password, they do not have permission to access Tableau Server and Kerberos authentication will fail.

- **Other accounts**—Try to connect with SSO to Tableau Server using other user accounts. If all users are affected, the problem may be in the Kerberos configuration.

- **Computer location**—Kerberos will not work when connecting from localhost. Clients must be connecting from a computer other than the Tableau Server computer.
• **URL address**—You cannot use Kerberos SSO when connecting using an IP address. In addition, the server name you use to access Tableau Server must match the name used in the Kerberos configuration (see Key table entry, below).

• **TGT (Ticket Granting Ticket)**—Confirm that the client computer has a TGT from the Active Directory domain. Kerberos requires a TGT to sign in. Constrained delegation, with the proxy granting a ticket, is not supported.

To confirm the client computer has a TGT, type:

```
• klist tgt at a command prompt on a Windows computer
  or
  klist at a terminal prompt on a Mac computer
```

The output should show a TGT for the user/domain trying to authenticate to Tableau Server.

The client computer may not have a TGT in the following circumstances:

• The client computer is using a VPN connection

• The client computer is not joined to the domain (for example, it is a non-work computer being used at work)

• The user signed into the computer with a local (non-domain) account

• The computer is a Mac that is not using Active Directory as a network account server

• **Browser**—Check which browser the user is using to access the server

  • Internet Explorer (IE) and Chrome work "out of the box" on Windows
  
  • Safari works "out of the box" on Mac
  
  • Firefox requires additional configuration
For more information about browser support for Kerberos Single Sign-On (SSO), see Browser Support for Kerberos SSO.

Troubleshooting on the server

If you cannot solve the problem from the client computer, your next steps are to troubleshoot on the computer running Tableau Server. The administrator can use the request ID to locate the sign-in attempt in the Apache logs on Tableau Server.

- **Log files**—Check the Apache error.log for an error with the exact time/date of the failed sign-in attempt.

- **Key table entry**—If the error.log entry says "No key table entry matching HTTP/<servername>.<domain>.<org>@", for example:

  [Fri Oct 24 10:58:46.087683 2014] [:error] [pid 2104:tid 4776] [client 10.10.1.62:56789] gss_acquire_cred() failed: Unspecified GSS failure. Minor code may provide more information (, No key table entry found matching HTTP/servername.domain.com@)

This error is a result of a mismatch between any of the following:

- **Tableau Server URL** - The URL used by the client computer to access the server.

  This is the name that you type into Tableau Desktop or a browser address bar. It could be a shortname (http://servername) or a fully-qualified domain name (http://servername.domain.com)
- DNS reverse lookup for the server IP address

This looks up a DNS name using an IP address.

At a command prompt type:

    ping servername

with the IP address returned by pinging the server, do a reverse DNS lookup type:

    nslookup <ip address>

The Tableau Server computer name needs to match in:

- .keytab file
- Service Principal Name (SPN) for the server

Test Configuration and tabconfig.log

Look in tabconfig.log for any problems or errors. For example:

    2014-10-17 10:58:16.545 -0700 ERROR root: No SPN entries found

Data source SSO

Delegated data source access failures

Check the vizqlserver log files for "workgroup-auth-mode."

Look for "workgroup-auth-mode" in the log files. It should say "kerberos-impersonate" not "as-is".

Kerberos delegation multi-domain configuration

Tableau Server has the ability to delegate users from other Active Directory domains. If your database uses MIT Kerberos, you may need to adjust your Kerberos principal to database user mapping. Specifically, you will need to update krb5.conf with rules for each Kerberos
realm that users will connect from. Use the `auth_to_local` tag in the `[realms]` section to map principal names to local user names.

For example, consider a user, `EXAMPLE\jsmith`, whose Kerberos Principal is `jsmith@EXAMPLE.LAN`. In this case, Tableau Server will specify a delegated user, `jsmith@EXAMPLE`. Tableau Server will use the Active Directory legacy domain alias as the Kerberos Realm.

The target database may already have a rule such as the following to map the user, `jsmith@EXAMPLE.LAN` to the database user, `jsmith`.

```plaintext
EXAMPLE.LAN = {
    RULE:[1:$1@$0](.*@EXAMPLE.LAN)s/@.*/
    DEFAULT
}
```

To support delegation, you must add another rule to map `jsmith@EXAMPLE` to a database user:

```plaintext
EXAMPLE.LAN = {
    RULE:[1:$1@$0](.*@EXAMPLE.LAN)s/@.*/
    RULE:[1:$1@$0](.*@EXAMPLE)s/@.*/
    DEFAULT
}
```

See the MIT Kerberos Documentation topic, `krb5.conf`, for more information.

**Cross-domain constrained delegation**

In some cross-domain scenarios where the KDC is running on a Windows Server prior to Windows 2012, delegation may fail. Errors you may see include:

- SQL Server Network Interfaces: The system cannot contact a domain controller to service the authentication request. Please try again later.
- SQL Server Native Client: Cannot generate SSPI context
- The Domain Controller returns: KRB-ERR-POLICY error with a status `STATUS_CROSSREALM_DELEGATION_FAILURE (0xc000040b)`.
"Cross-domain" refers to a scenario where Tableau Server is running in a different domain than the data source with different service accounts. For example:

- Tableau Server runs on DomainA with DomainA service account.
- SQL Server runs on DomainB with DomainB service account.

Traditional constrained delegation only works if both servers are in the same domain. The user can come from other domains.

If you are seeing the errors noted above, then to enable this scenario, your Active Directory administrator should remove any traditional constrained delegation which is configured on the delegating account. Removing delegation can be achieved with Active Directory management tools or by removing the values associated with the Active Directory property, msDS-AllowedToDelegateTo.

If you wish to preserve an existing single domain delegation alongside cross-domain delegation, then you must configure both using resource-based constrained delegation.

The article, How Windows Server 2012 Eases the Pain of Kerberos Constrained Delegation, Part 2, provides a detailed description of how Microsoft KDC and Active Directory handle delegated scenarios. To configure these scenarios, refer to the Microsoft Kerberos documentation.

Configure Mutual SSL Authentication for Tableau Server on Linux

Using mutual SSL, you can provide users of Tableau Desktop, and other approved Tableau clients a secure, direct-access experience to Tableau Server. With mutual SSL, when a client with a valid SSL certificate connects to Tableau Server, Tableau Server confirms the existence of the client certificate and authenticates the user, based on the user name in the client certificate. If the client does not have a valid SSL certificate, Tableau Server can refuse the connection.

Mutual SSL authentication is not supported on Tableau Mobile.

To configure Tableau Server for mutual SSL, you need the following:
SSL certificate file—A valid PEM-encoded x509 certificate with the extension .crt.

SSL certificate key file—A valid RSA or DSA private key file (with the extension .key by convention).

SSL certificate chain file—(Optional for Tableau Server, required for Tableau Desktop on the Mac) Some certificate providers issue two certificates for Apache. The second certificate is a chain file, which is a concatenation of all the certificates that form the certificate chain for the server certificate. All certificates in the file must be x509 PEM-encoded and the file must have a .crt extension (not .pem).

1 Require SSL for external server communication

To configure Tableau Server to use SSL for external communication between Tableau Server and web clients, run the `external-ssl enable` command as follows, providing the names for the server certificate’s .crt and .key files:

```
tsm security external-ssl enable --cert-file <file.crt> --key-file <file.key>
```

- For `--cert-file` and `--key-file`, specify the location and file name where you saved the server’s CA-issued SSL certificate (.crt) and key (.key) files.

- The above command assumes the you are signed in as a user that has the **Server Administrator** site role on Tableau Server. You can instead use the `-u` and `-p` parameters to specify an administrator user and password.

- If the certificate key file requires a passphrase, include the `--passphrase` parameter and value.

2 Use mutual SSL

Add mutual authentication between the server and each client, and allow for Tableau client users to be authenticated directly after the first time they provide their credentials.
1. Run the following command:

```bash
tsm authentication mutual-ssl configure -cert-file <file.crt>
```

For `--cert-file`, specify the location and file name of the server’s CA certificate (.crt) file, as in the previous step for external SSL.

See the remaining sections in this article for any additional options you might want to include with the `mutual-ssl configure` command.

2. Run the following commands to enable mutual SSL and apply the changes:

```bash
tsm authentication mutual-ssl enable

 tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `--r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Additional options for mutual SSL

You can use `mutual-ssl configure` to configure Tableau Server to support the following options.

For more information, see `tsm authentication mutual-ssl <commands>`.

Fallback authentication

When Tableau Server is configured for mutual SSL, authentication is automatic and a client must have a valid certificate. You can configure Tableau Server to allow a fallback option, to accept user name and password authentication.

```bash
tsm authentication mutual-ssl configure -f true
```
User name mapping

When Tableau Server is configured for mutual SSL, the server authenticates the user directly by getting the user name from their client certificate. The name that Tableau Server uses depends on how the server is configured for user authentication:

- **Local Authentication**—uses the UPN (User Principal Name) from the certificate.
- **Active Directory (AD)**—uses LDAP (Lightweight Directory Access Protocol) to get the user name.

You can override either of these defaults to set Tableau Server to use the common name.

```
 tsm authentication mutual-ssl configure -m cn
```

For more information, see Mapping a Client Certificate to a User During Mutual Authentication

Certificate Revocation List (CRL)

You might need to specify a CRL if you suspect that a private key has been compromised, or if a certificate authority (CA) did not issue a certificate properly.

```
 tsm authentication mutual-ssl configure -r <revoke-file.pem>
```

Mapping a Client Certificate to a User During Mutual Authentication

When you use mutual (two-way) SSL authentication, the client presents its certificate to Tableau Server as part of the authentication process. Tableau Server then maps user information in the client certificate to a known user identity. The strategy that Tableau Server uses to perform client mapping depends on the content of your organization’s client certificates.

This topic discusses the ways information in a client certificate can map to a user identity and how to change the way Tableau Server performs that mapping. To understand how the mapping happens and whether you need to change it, you must know how client certificates are structured in your organization.
User-name mapping options

Change the certificate mapping

Address user-name ambiguity in multi-domain organizations

User-name mapping options

Tableau Server uses one of the following approaches to map a client certificate to a user identity:

- **Active Directory.** If Tableau Server is configured to use Active Directory for user authentication, when Tableau Server receives a client certificate, it passes the certificate to Active Directory, which maps the certificate to an Active Directory identity. Any explicit user name information in the certificate is ignored.

  **Note:** This approach requires client certificates to be published for the user accounts in Active Directory.

- **User principal name (UPN).** A client certificate can be configured to store the user name in the user principal name field. Tableau Server reads the UPN value and maps it to a user in Active Directory or to a local user.

- **Common name (CN).** A client certificate can be configured to store the user name in the common name field of the certificate. Tableau Server reads the CN value and maps it to a user in Active Directory or to a local user.

If you configure the server for Active Directory authentication and UPN or CN user-name mapping, put the user name in one of the following formats:

username, domain/username, or username@domain.

For example: asmith, example.org/asmith, or asmith@example.org.

If the server uses local authentication, the format of the name in the UPN or CN fields is not predetermined, but the name in the field must match a user name on the server.
Change the certificate mapping

You use the following command to map a client certificate to a user identity in Tableau Server:

```bash
tsm authentication mutual-ssl configure -m <value>
```

Possible values are `ldap` for Active Directory mapping, `upn` for UPN mapping, or `cn` for CN mapping.

When you first install and configure Tableau Server, the server sets the default user-name mapping to match the server's authentication type:

- If the server is configured to use Active Directory, it also uses Active Directory for mapping the certificate to the user identity.
- If the server is configured to use local authentication, the server gets the user-name value from the UPN field in the certificate.

If the default behavior for how Tableau Server maps a user name to an identity is not correct for your server configuration, run the following set of commands to change the mapping to use the CN value:

```bash
tsm authentication mutual-ssl configure -m cn
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Address user-name mapping ambiguity in multi-domain organizations

Under some circumstances, the user name in a certificate’s UPN or CN field can be ambiguous. This ambiguity can lead to unexpected results when the user name is mapped to a user
identity on the server.

For example, if Tableau Server is presented with a user name that does not include a domain, the server maps the user name to an identity using the default domain. This can cause an incorrect user-name mapping, potentially assigning a user a different user’s identity and permissions.

This can occur particularly in environments where the following conditions apply:

- Your organization supports multiple Active Directory domains.
- The server is configured to use Active Directory authentication.
- The server is configured to use UPN or CN mapping.
- Some users have the same user name but different domains. For example, asmith@example.org and asmith@example.com.
- The user name in the certificate’s UPN or CN fields does not include the domain as part of the user name—for example, it shows asmith.

To avoid incorrect user-name mapping, make sure the client certificates include fully qualified user names with the domain, using the format asmith@example.org or example.org/asmith.

See also

tsm authentication mutual-ssl <commands>

Open ID Connect

You can configure Tableau Server to support OpenID Connect for single sign-in (SSO). OpenID Connect is a standard authentication protocol that lets users sign in to an identity provider (IdP) such as Google. After they’ve successfully signed in to their IdP, they are automatically signed in to Tableau Server.
Configuring OpenID Connect involves several steps. The topics in this section provide general information about using Tableau Server with OpenID Connect, and provide a sequence for configuring the IdP and Tableau Server.

Authentication overview

This section describes the OpenID Connect authentication process with Tableau Server.

**Step 1:** A user requests a resource on Tableau Server.

**Step 2:** Tableau Server redirects the request to the IdP gateway for authentication.

**Step 3:** The user is prompted and successfully authenticates with the IdP. The IdP redirects the session back to Tableau Server. Included with the URL is an authorization code for the user.

**Step 4:** Tableau Server presents the user’s authorization code back to the IdP. As an IdP client, the Tableau Server includes its own client credentials to prevent spoofing.

**Step 5:** The IdP returns an access token and an ID token to Tableau Server.

- The access token contains an encrypted key to verify that the IdP isn't being spoofed. The process the key is used for is referred to as JSON Web Token (JWT) validation.
The key for JWT is specified in the `jwks_uri` in the IdP discovery document. If your IdP does not support JWT validation, you can disable it. See tsm authentication `openid <commands>`.

- The ID token is a set of attribute key/value pairs for the user. The key/value pairs are called `claims`. Here is an example IdP claim for a user:

```
"sub" : "7gYhRR3HiRRCaRcgvY50ubrtjGQBMJW4rXbpPFp-g2cptHP62m2sqowM7G1LwjN5",
"email" : "alice@tableau.com",
"email_verified" : true,
"name" : "Alice Adams",
"given_name" : "Alice",
"family_name" : "Adams",
"phone_number" : "+359 (99) 100200305",
"profile" : "https://tableau.com/users/alice"
```

**Step 6: Tableau identifies user from IdP claim.** Tableau Server searches the user account records stored in the repository. By default, Tableau Server will use the subject identifier, or `sub`, claim to identify a user account. If no user record is storing the `sub` claim value, then Tableau Server will search for username matches using the `email` claim. When a username match is found, Tableau Server will write the corresponding `sub` claim to the user record in the repository. You can configure Tableau Server to use different claims for this process. See Requirements for Using OpenID Connect.

**Step 7: Tableau Server authorizes the user.**

How Tableau Server Works with OpenID Connect

OpenID Connect is a flexible protocol that supports many options for the information that's exchanged between a service provider (here, Tableau Server) and an IdP. The following list provides details about the Tableau Server implementation of OpenID Connect. These details can help you understand what types of information Tableau Server sends and expects, and how to configure an IdP.
Tableau Server supports only the OpenID Authorization Code Flow as described in the OpenID Connect final specification.

Tableau Server relies on using discovery or a provider URL to retrieve the OpenID Provider metadata. Alternatively, you can host a static discovery document on Tableau Server. For more information see Configure OpenID Connect.

Tableau Server supports only the `client_secret_jwt` Client Authentication method specified in the OpenID Connect specification. In addition, Tableau Server supports only RSA Asymmetric Encryption for handling the JWT. However, you can turn off JWT validation. For more information, see tsm authentication openid <commands>.

Tableau Server expects a `kid` value in the `id_token` attribute’s JOSE Header. This value is matched with one of the keys found in the JWK Set document, whose URI is specified by the `jwks_uri` value in the OpenID discovery document. A `kid` value must be present even if there is only one key in the JWK Set document.

Tableau Server does include OpenID support for the JWK `x5c` parameter or for using X.509 certificates.

For more information about OpenID Connect, see the following:

- OpenID Connect Core 1.0 incorporating errata set 1
- OpenID Connect Discovery 1.0 incorporating errata set 1

Requirements for Using OpenID Connect

This topic describes the requirements to use OpenID Connect with Tableau Server.

IdP account

You must have access to an identity provider (IdP) that supports the OpenID Connect (OIDC) protocol. You must also have an account with the IdP. OpenID Connect is supported by many identity providers. The OIDC protocol is an open and flexible standard, and as
such, not all implementations of the standard are identical. As you configure Tableau Server for OIDC, work with your IdP.

The Google IdP implementation has been extensively tested with Tableau Server and is the model IdP for the configuration documented in these topics.

Local authentication

To use OpenID Connect on Tableau Server, the server must be configured to use local authentication—that is, the server must be configured so that you explicitly create users on the Tableau Server, rather than importing them from Active Directory. Active Directory authentication is not supported with OpenID.

IdP claims: mapping users

To sign in successfully to Tableau Server, a given user must be provisioned in OpenID and then mapped to a user account on Tableau Server. OpenID uses a method that relies on claims to share user account attributes with other applications. Claims include user account attributes such as email, phone number, given name, etc. To understand how Tableau Server maps IdP claims to user accounts, see Authentication overview.

Tableau Server relies on the IdP claims to map user accounts from the IdP to those hosted on Tableau Server. By default, Tableau Server expects the IdP to pass the email claim. Depending on your IdP, you may need to configure Tableau Server to use a different IdP claim.

If you are using Google as an IdP, then use the default, email claim to map IdP identities to Tableau Server user accounts. If you are not using Google as an IdP, then work with your IdP to determine the claim for which you should configure Tableau Server.

Default: using email claim to map users

By default, the user's user name in Tableau Server must match the email claim in the IdP ID token. Therefore, in the default configuration, you must use email addresses (also referred to as UPN) as the username in Tableau Server. If you use Google as the IdP, the user name in Tableau Server must be the user's Gmail address (alice@gmail.com).
Using a complete email address helps to guarantee the uniqueness of the user name in Tableau Server, even when two users have the same email but are on different email hosts.

**Note:** When you create a user identity in Tableau Server, you specify a user name, password, and optionally an email address. For using OpenID Connect in the default configuration, the user name (expressed as an email address) is the value that must match the user’s name in the IdP. The optional email address in the Tableau Server user identity is not used for OpenID authentication.

**Ignoring the domain name**

You can configure Tableau to ignore the domain portion of an email address when matching the IdP email claim to a user account on Tableau Server. In this scenario, the email claim in the IdP might be alice@example.com, but this will match a user named alice in Tableau Server. Ignoring the domain name might be useful if you already have users defined in Tableau Server that match the user names portion of the email claim, but not the domain portions.

**Important:** We do not recommend ignoring the user domain name without taking precautions. Specifically, verify that user names are unique across the configured domains that you’ve created in your IdP.

Setting Tableau Server to ignore the user domain name has the potential to result in unintended user log on. Consider the case where your IdP has been configured for multiple domains (example.com and tableau.com). If two users with the same first name, but different user accounts (alice@tableau.com and alice@example.com) are in your organization, then the first one to complete the OpenID provisioning sequence will claim the sub mapping in the IdP. If the wrong user is mapped, then the other user will be unable to log on until the associated sub value is reset.
To configure Tableau Server to ignore domain names in user names from the IdP, set `tsm authentication openid configure --ignore-domain` to `true`. For more information, see `tsm authentication openid <commands>`.

When you change the `tsm authentication openid configure --ignore-domain` option to ignore the domain in user names, all user names in Tableau Server must have a domain name.

**Using custom claims to map users**

As referenced in Authentication overview, the `sub` claim is often included in IdP claims. Typically, the `sub` claim is a unique string that identifies a given user account. The benefit of using a `sub` claim is that it will not change, even if you or another admin updates other user attributes or IdP claims (email, phone number, etc) associated with that account. By default, Tableau Server identifies and verifies OpenID users according to the `sub` claim in the IdP ID token.

The OpenID `sub` claim value must be mapped to the corresponding user in Tableau Server. Since the `sub` claim is an arbitrary string, a different claim is used to associate accounts during the first sign-in session. The first time a user signs in to Tableau Server with OpenID, Tableau will match the OpenID user account to a corresponding user account on Tableau. By default, Tableau will use the IdP claim, `email`, to identify the Tableau user. Tableau will then update that user's record with the `sub` claim from OpenID. Since the ID token always includes the `sub` claim along with other claims, on subsequent sessions, Tableau will identify that user with the `sub` claim only.

For some organizations, mapping user names with the email address is not reliable or not supported by the IdP. Beginning with Tableau Server 10.2, you can map user accounts from any arbitrary IdP claim to the Tableau Server username.

The IdP claim you are using must map exactly to a corresponding Tableau Server username. In the example below, the username is `kwilliams`. 
To change the IdP claim that is used to map identity on Tableau Server, use the `tsm authentication openid map-claims --user-name` command. For more information, see `tsm authentication openid <commands>`.

### Changing the `sub` claim

As described above, the `sub` claim is the identifier that Tableau Server uses to identify users after the initial mapping session. The `sub` claim is written to the corresponding user account in Tableau Server. If your IdP does not provide a `sub` claim, then you can specify an arbitrary claim to use instead. Like `sub`, the claim value you specify must be unique and should not change when other user claims are updated.
To specify a different IdP claim for default sub claim, use the command `tsm authentication openid map-claims --id <command>`.

For more information, see `tsm authentication openid <commands>`.

OpenID Connect Authentication Request Parameters

The OpenID authentication request sent from Tableau Server passes information using a limited set of parameters, as listed in this topic. If your OpenID IdP requires parameters that are not on the list below, it is not compatible for use with Tableau Server.

- **scope.** This value specifies a profile that tells the IdP what user information claims to return. This value can be configured by a Tableau Server administrator. The default value is "openid email profile". For more information, see Configure the scope value later in this document.

- **response_type.** OpenID Connect supports multiple flows. This value tells the IdP which flow Tableau Server expects. Tableau supports only the authorization code flow, and the value is always set to "code".

- **client_id.** This value specifies the server’s ID (Provider client ID in the Tableau Server Configuration dialog box), which lets the IdP know where the request came from. It is provided by the IdP when the service is registered. The value is configurable by a Tableau Server administrator.

- **redirect_uri.** This value specifies the URL that the IdP redirects to after the user has authenticated using OpenID Connect. The URL must include the host and protocol (for example, http://example.tableau.com), but Tableau provides the URL endpoint.

- **nonce.** Tableau Server generates a nonce value to verify that the client that it redirected to matches the entity that comes back from the IdP.
Configure the scope value

The scope value indicates to the IdP the information that Tableau Server requests about the user. By default, Tableau Server sends the value "openid profile email". This indicates that Tableau uses OpenID to authenticate (this part of the scope attribute value must always be included) and that Tableau Server is requesting the user profile and email information during the exchange of the user authorization code.

If this default scope is not appropriate for your scenario, you can have Tableau Server request custom information about the user. To do so, you configure the IdP with a custom profile (for example, something like "tableau-scope"). You can then configure Tableau Server to send the scope request using the name of the custom profile.

To change the scope value that Tableau Server requests, use the following TSM CLI command:

```
tsm authentication openid configure --custom-scope-name custom-scope-name
```

**Note:** Tableau Server always includes "openid" as part of the scope value (even if you don't include it in the custom-scope-name option).

Configure OpenID Connect

This topic provides an end-to-end description for setting up OpenID Connect authentication.

Before you begin

We recommend that you configure Tableau Server to use SSL for external communications. This helps to maintain secure communications between Tableau Server and the IdP during the exchange of authentication information.

If your IdP uses a self-signed certificate (or the IdP certificate is not signed by a trusted CA), then you can add the IdP public certificate to the Java truststore. See the Knowledge Base
article, Configuring Tableau Server for OpenID Connect Using Uncommon or Self-Signed Certificates.

If you are configuring OpenID Connect during the initial configuration of Tableau Server (before running tsm initialize), there is no option to set up SSL. In that case, we recommend that you finish the installation, then return to the configuration to set up SSL and then configure OpenID.

Note If you want to use external SSL for Tableau Server, it's generally more convenient to do that before you configure OpenID Connect. If you configure SSL after you've already configured OpenID, you need to return to the IdP and update the configuration that you made previously. For example, you need to change the protocol for the Tableau Server external URL from http:// to https://.

Step 1: Configure the identity provider

Before you can use OpenID Connect with Tableau Server, you must have an account with an identity provider (IdP) and a project or application with the IdP. When you configure Tableau Server, you will need to be able to provide the following information:

- Provider client ID. This is the identifier that the IdP assigned to your application.
- Provider client secret. This is a token that is used by Tableau to verify the authenticity of the response from the IdP. This value is a secret and should be kept securely.
- Provider configuration URL. This is the URL at the provider’s site that Tableau Server should send authentication requests to.

Redirect URL

Some IdPs will require a redirect URL for your Tableau Server. You can manually construct your URL for the IdP using the following syntax:
<protocol>://<host>/vizportal/api/web/v1/auth/openIdLogin

For example, https://tableau.example.com/vizportal/api/web/v1/auth/openIdLogin.

Example IdP process

The following procedure provides an outline of the steps that you follow with the provider. As an example, the procedure discusses using Google as a provider. However, each provider has a somewhat different flow, so the specifics of the steps (and their order) might vary depending on your provider.

1. Register at the provider’s developer site and sign in. For example, for Google, you can go to the Developers Console at this URL: https://console.developers.google.com

2. Create a new project, application, or relying party account.

3. In the developer dashboard, follow the steps for getting an OAuth 2.0 client ID and client secret. Record these values for later.

   Note: Keep the client secret in a secure place.

4. On the developer site, find the URL of the endpoint that the IdP uses for OpenID Connect discovery. For example, Google uses the URL https://accounts.google.com/.well-known/openid-configuration. Record this URL for later.

   Alternatively, if your IdP has provided you with a static discovery document, copy that file to a local directory on the Tableau Server for later.

Step 2: Configure Tableau Server

The procedure in this section describes how to use TSM command line interface to configure OpenID Connect. You can also use a configuration file for the initial configuration of
OpenID Connect. See openIDSettings Entity.

1. Use the `configure` command of `tsm authentication openid <commands>` to set the following required options:

   --client-id <id>: Specifies the provider client ID that your IdP has assigned to your application. For example, “laakjwdlnaoiloadjkwha”.

   --client-secret <secret>: Specifies the provider client secret. This is a token that is used by Tableau to verify the authenticity of the response from the IdP. This value is a secret and should be kept securely. For example, “fwahfkjaw72123=”.  

   --config-url <url> or --metadata-file <file_path>: Specifies location of provider configuration json file. If the provider hosts a public json discovery file, then use --config-url. Otherwise, specify a path on the local computer and file name for --metadata-file instead.

   --return-url <url>: The URL of your server. This is typically the public name of your server, such as “http://example.tableau.com”.

   For example, run the command:

   

   There are additional, optional configurations that you can set for Open ID Connect using either openIDSettings Entity or `tsm authentication openid <commands>`. In addition, if you need to configure IdP claim mapping, see Options for openid map-claims.

2. Type the following command to enable Open ID Connect:

   tsm authentication openid enable

3. Run `tsm pending-changes apply` to apply changes.
The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `–r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Step 3: Sign in to Tableau Server using OpenID Connect

After Tableau Server has been configured to use OpenID Connect, users who access the server and aren't already signed in are redirected to the IdP site, where they are prompted to sign in. Users enter the credentials that they have with the IdP. In many cases, the user is also asked to authorize the IdP to share information with Tableau Server, as in the following example:
When a user signs in using OpenID Connect, the IdP sends a unique user identifier (known in OpenID as the sub value) as part of the information that's redirected to Tableau Server. This sub value is associated with the user's Tableau user identity.

**Note:** Tableau Server does not support using OpenID Connect to sign in from mobile devices.

Restricting sign-in to server administrators for command-line tools

Command-line tools for working with Tableau Server (tabcmd, TSM, and tableau.com) do not support sign-in using OpenID Connect. When OpenID Connect is enabled for the server, these tools still require sign-in using a Tableau Server username and password.

Even if users normally authenticate using OpenID Connect, each user has a Tableau Server username and password. This means that users could use command-line tools like tabcmd. As a security measure, you can make sure that only server administrators can use command-line tools. To do this, use `tsm configuration set wgserv-er.authentication.restricted to true`. When this setting is true, only server administrators can sign in to Tableau Server using a username and password; all other users must sign in to the server using a single sign-on (SSO) option like OpenID Connect. The effect is that users who are not administrators also cannot then use command-line tools. To make this change, run the following sequence of TSM commands:

```
  tsm configuration set -k wgserver.authentication.restricted -v true
  tsm pending-changes apply
```

Troubleshoot OpenID Connect

Use the following topics to troubleshoot OpenID Connect (OIDC) issues in Tableau Server.
OpenID Connect protocol is supported by many identity providers. The OIDC protocol is an open and flexible standard, and as such, not all implementations of the standard are identical. Most issues that administrators encounter when configuring Tableau Server for OIDC are the result of how different identity providers implement OIDC. If you encounter errors as you set up OIDC with Tableau Server, we recommend that you work with your IdP to resolve them.

Signing In from the Command Line

Even if Tableau Server is configured to use OpenID, it is not used if you sign in to Tableau Server using tabcmd, the REST API, or the Tableau Data Extract command line utility (provided with Tableau Desktop).

Login Failed

Login can fail with the following message:

Login failure: Identity Provider authentication successful for user <username from IdP>. Failed to find the user in Tableau Server.

This error typically means that there is a mismatch between a username stored in Tableau Server and the username provided by the IdP. To fix this, make sure that they match. For example, if Jane Smith’s username is stored in the IdP as jsmith it must be stored in Tableau Server as jsmith as well.

Error 69: "Unable to Sign In"

An error 69 may be returned when you attempt to sign in to Tableau Server with a web browser and receive an error, "Unable to Sign In. Sign in failed. Contact your Tableau Server administrator." The URL that returns this message is https://example.com/#/error/signin/69?redirectPath=%2.

If you receive this error and you have configured OIDC for the Salesforce IdP, then you must set the vizportal.openid.client_authentication parameter. See tsm configuration set Options for more information.
OpenID Error Log

OpenID authentication takes place outside Tableau Server, so troubleshooting authentication issues can be difficult. However, sign-in attempts are logged by Tableau Server. You can create a snapshot of log files and use them to troubleshoot problems.

**Note:** To log OpenID-related events, `vizportal.log.level` must be set to `debug` with `tsm configuration set Options`.

Check for OpenID errors in the following files in the unzipped log file snapshot:

```
/vizportal/vizportal-<n>.log
```

**Trusted Authentication**

When you embed Tableau Server views into webpages, everyone who visits the page must be a licensed user on Tableau Server. When users visit the page they are prompted to sign in to Tableau Server before they can see the view. If you already have a way of authenticating users on the webpage or within your web application, you can avoid this prompt and save your users from having to sign in twice by setting up trusted authentication.

Looking for Tableau Server on Windows? See [Trusted Authentication](#).

Trusted authentication simply means that you have set up a trusted relationship between Tableau Server and one or more web servers. When Tableau Server receives requests from these trusted web servers it assumes that your web server has handled whatever authentication is necessary.

If your web server uses SSPI (Security Support Provider Interface), you do not need to set up trusted authentication. You can embed views and your users will have secure access to them as long as they are licensed Tableau Server users and members of your Active Directory.
How Trusted Authentication Works

The diagram below describes how trusted authentication works between the client’s web browser, your web server(s) and Tableau Server.

1. **User visits the webpage**: When a user visits the webpage with the embedded Tableau Server view, the webpage sends a GET request to your web server for the HTML for that page.

2. **Web server POSTS to Tableau Server**: The web server sends a POST request to the trusted Tableau Server (for example, `https://tabaserver/trusted`, not `https://tabserver`). That POST request must have a `username` parameter. The `username` value must be the username for a licensed Tableau Server user. If Tableau Server is hosting multiple sites and the view is on a site other than the Default site, then the POST request must also include a `target_site` parameter.
Tableau Server creates a ticket: Tableau Server checks the IP address or host name of the web server (192.168.1.XXX in the above diagram) that sent the POST request. If the web server is listed as a trusted host then Tableau Server creates a ticket in the form of a unique string. Tickets must be redeemed within three minutes after they are issued. Tableau Server responds to the POST request with that ticket. Or if there is an error and the ticket cannot be created, then Tableau Server responds with a value of -1.

Web server passes the URL to the browser: The web server constructs the URL for the view and inserts it into the HTML for the page. The ticket is included (for example, https://tabserver/trusted/<ticket>/views/requested_view_name). The web server passes the HTML back to the client’s web browser.

Browser requests view from Tableau Server: The client web browser sends a GET request to Tableau Server that includes the URL with the ticket.

Tableau Server redeems the ticket: Tableau Server redeems the ticket, creates a session, logs the user in, removes the ticket from the URL, and then sends the final URL for the embedded view to the client.

The session allows the user to access any of the views that the user would have if they logged onto the server. In the default configuration, users authenticated with trusted tickets have restricted access such that only views are available. They cannot access workbooks, project pages, or other content hosted on the server.

To change this behavior, see the wgserver.unrestricted_ticket option at tsm configuration set Options.

How is a trusted ticket stored?

As of version 10.2, Tableau Server stores trusted tickets in the Tableau Server repository using the following process:
1. Tableau Server generates a two-part ticket: the first part is a Base64-encoded unique ID (UUID) and the second part is a 24-character random secret string.
2. Tableau Server hashes the secret string and stores it with the unique ID in the repository. Hashing takes the secret string as input, and uses an algorithm to compute a unique string. This unique string protects the security of the secret string from unauthorized users.
3. Tableau Server sends the Base64 UUID and the original 24-character random string to the client.
4. The client returns the Base64 UUID and the original 24-character secret string to Tableau Server as part of the request for a view.
5. Tableau Server locates the string pair with the Base64 UUID, and then hashes the secret string to verify that it matches the hash stored in the repository.

This process ensures that any trusted ticket content stored on Tableau Server cannot be used to impersonate users or access content protected by authentication. However, because the full trusted ticket is sent over HTTP between Tableau Server and the client, the process relies on secure and encrypted transmission of HTTP data. Therefore, we recommend that you only deploy trusted tickets over SSL/TLS or another layer of network encryption.

Configure Trusted Authentication

This topic provides an end-to-end description for setting up trusted authentication.

Looking for Tableau Server on Windows? See Add Trusted IP Addresses or Host Names to Tableau Server.

Follow these steps:

**Step 1: Specify trusted web servers**

The first step in setting up trusted authentication is to configure Tableau Server to recognize and trust requests from one or more web servers:
1. Enter the following command:

   tsm authentication trusted configure -th <trusted IP address or host name>

   In the command above, <trusted IP address> should be a comma-separated list of the IPv4 addresses or host names of your web server(s), with each host name or IP address in quotes.

   **Note:** The values you specify completely overwrite any previous setting. Therefore, you must include the full list of hosts in the tsm authentication trusted configure -th command. (You cannot amend the list of hosts by running the tsm authentication trusted configure -th command repeatedly.)

   For example:

   tsm authentication trusted configure -th "192.168.1.101", "192.168.1.102", "192.168.1.103"

   or

   tsm authentication trusted configure -th "webserv1", "webserv2", "webserv3"

   **Notes:**
   The each host name or IP address in the list should be in quotes, followed by a comma and one space after each comma.
   The web servers you specify must use static IP addresses, even if you use host names.

2. If you have one or more proxy servers between the computer that is requesting the
trusted ticket (one of those configured in step 2, above) and Tableau Server, you also need to add them as trusted gateways. See Configure a reverse proxy server for steps.

3. Type the following command to save the changes to all the server configuration files:

   `tsm pending-changes apply`

   The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `–r` option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

   There are other optional trusted authentication configurations (legacy support, logging, and timeout settings) that you can make by passing a json file to Tableau Server. See trustedAuthenticationSettings Entity.

### Step 2: Configure your web server to receive tickets from Tableau Server

After you've added trusted IP addresses to Tableau Server, you're ready to configure your web server to get tickets from Tableau Server via POST requests (step 3 in the diagram). The POST request must be sent to `http://<server name>/trusted`, not `http://tabserv`. For example `http://tabserv/trusted`.

| Note: If SSL is enabled you must use https instead of http. For example: | https://tabserver/trusted. |

Here's the data you can use in a POST request to Tableau Server:
- username=<username> (required): The username for a licensed Tableau Server user. If you are using Local Authentication the username can be a simple string (for example, username=jsmith). If you are using Active Directory with multiple domains you must include the domain name with the user name (for example, username=MyCo\jsmith).

- target_site=<site id> (required if view not on Default site): Specifies the site containing the view if Tableau Server is running multiple sites and the view is on a site other than the Default site (for example, target_site=Sales). The value you use for <site id> should be the Site ID that was provided when the site was created. This value is case sensitive. If the Site ID is Sales, then the target_site=Sales.

- client_ip=<IP address> (optional): Used to specify the IP address of the computer whose web browser is accessing the view (for example, client_ip=123.45.67.891). It is not the IP address of the web server making the POST request of Tableau Server. If you decide to use this parameter, see Optional: Configure Client IP Matching for more information.

Tableau Server’s response to the POST request will be a unique string (the ticket). If Tableau Server isn’t able to process the request, the return will be -1. See Ticket Value of -1 Returned from Tableau Server for tips on how to correct this. Also, in order for users to successfully authenticate when they click an embedded view, their browsers must be configured to allow third-party cookies.

The ticket format has changed in Tableau Server 10.5. The ticket format is now a string composed of two parts. Each part is a 128 bit string that is encoded before it is returned to the client. The first part is a universally unique ID (UUID v4) that is Base64-encoded. The second part is a 24-character secure random string. The concatenation of these parts can be expressed as Base64(UUIDv4):SecureRandomString. An example of a ticket might look like this: 9D1ObyqDQmSIOyQpKdy4Sw==:dg62gCsSE0QRArXNT0p6mlJ5.

Next, you need to add code that allows the web server to construct an URL for the view that includes the view’s location and the ticket.

**Step 3: Add code to web server for URL construction**
After you create the POST request, you need to write code that provides the web server with the view’s location and the ticket from Tableau Server. It will use this information to display the view. How you specify it depends on whether the view is embedded, and if Tableau Server is running multiple sites.

**Tableau Server View Examples**

Here’s an example of how to specify a view that users only access via Tableau Server (the view is not embedded):

http://tabserver/trusted/<ticket>/views/<workbook>/<view>

If Tableau Server is running multiple sites and the view is on a site other than the Default site, you need to add t/site ID to the path. For example:

http://t-tabserver/trusted/<ticket>/t/Sales/views/<workbook>/<view>

Use the same capitalization that you see in the Tableau Server URL.

**Embedded View Examples**

Here are some examples of how to specify embedded views. Because there are two approaches you can take with embed code, both ways are provided below. Regardless of which you use, there is some information unique to trusted authentication that you must provide. For more information, search for "Writing Embed Code" in the Tableau User Help.

**Script Tag Examples**

This example uses the ticket object parameter:

```html
<script type="text/javascript" src="http://myserver/javascripts/api/viz_v1.js"></script>
<object class="tableauViz" width="800" height="600" style="display:none;">
  <param name="name" value="MyCoSales/SalesScoreCard" />
  <param name="ticket" />```
value="9D1ObyqDQmSIOyQpKdy4Sw==:dg62gCsSEOQRArXNTOp6mlJ5" />
</object>

Here’s what the above example looks like for a multi-site Tableau Server, where the view is published on the Sales site:

<script type="text/javascript" src="http://myserver/javascripts/api/viz_v1.js"></script>
<object class="tableauViz" width="800" height="600" style="display:none;">
  <param name="site_root" value="/t/Sales" />
  <param name="name" value="MyCoSales/SalesScoreCard" />
  <param name="ticket" value="9D1ObyqDQmSIOyQpKdy4Sw==:dg62gCsSE0QRArXNTOp6mlJ5" />
</object>

Instead of using ticket, you can use the path parameter to state the full path of the view explicitly. When path is used, you do not also need the name parameter, which is usually a required parameter in Tableau JavaScript embed code:

<script type="text/javascript" src="http://myserver/javascripts/api/viz_v1.js"></script>
<object class="tableauViz" width="900" height="700" style="display:none;">
  <param name="path" value="true://myserver/javascripts/api/viz_v1.js"></object>

Here’s the same example, but for a multi-site server. Note that /t/<site ID> is used here:

<script type="text/javascript" src="http://myserver/javascripts/api/viz_v1.js"></script>
<object class="tableauViz" width="900" height="700"
Iframe Tag Example

<iframe src="http://tabserver/trusted/9D1ObyqDQmSI0yQpKdy4Sw==:dg62gCsSE0QRArXNTOp6mlJ5/" width="800" height="600"></iframe>

Step 4: (Optional) Configure client IP matching

By default, Tableau Server does not consider the client web browser IP address when it creates or redeems tickets. To change this, you need to do two things: specify an IP address using the `client_ip` parameter in the POST request that obtains the ticket, and follow the steps below to configure Tableau Server to enforce client IP address matching.

1. Open a bash shell as an administrator and type the following command:

   ```bash
   tsm configuration set -k wgserver.extended_trusted_ip_checking -v true
   ```

2. Then type the following command:

   ```bash
   tsm pending-changes apply
   ```

   The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt
using the \textit{\texttt{-r}} option, but this does not change the restart behavior. For more information, see \texttt{tsm pending-changes apply}.

**Step 5: Test trusted authentication configuration**

The steps below detail a method for testing the retrieval of a trusted ticket from your web server. This simple test can help evaluate connectivity between the web server and Tableau Server, and whether or not trusted authentication has been configured correctly.

For information about troubleshooting common errors, see \texttt{Troubleshoot Trusted Authentication}.

**Step 1: Add a test user**

Create a user on the Tableau Server that you can use to test trusted ticket functionality. See \texttt{Add Users to the Server}. Set the role of the user to \texttt{Interactor}.

**Step 2: Create a test HTML page**

Copy the following code and save as an html file on the web server that you specified in Step 1.

```html
<html>
<head>
<title>Trusted Ticket Requester</title>
<script type="text/javascript">
  function submitForm()
  {
    document.getElementById('form1').action =
    document.getElementById('server').value + "\trusted";
  }
</script>
<style type="text/css">
  \.style1 {width: 100%;}
  \.style2 {width: 429px;}
  \#server { width: 254px; }
</style>
```
<H3>Trusted Ticketer</H3>
<form method="POST" id="form1" onSubmit="submitForm()">
<table class="style1">
<tr>
<td class="style2">Username:</td>
<td><input type="text" name="username" value=""></td>
</tr>
<tr>
<td class="style2">Server:</td>
<td><input type="text" id="server" name="server" value="http://"></td>
</tr>
<tr>
<td class="style2">Client IP (optional):</td>
<td><input type="text" id="client_ip" name="client_ip" value=""></td>
</tr>
<tr>
<td class="style2">Site: (leave blank for Default site, else NameOfSite if using sites)</td>
<td><input type="text" id="target_site" name="target_site" value=""></td>
</tr>
</table>
</form>
Step 3: Retrieve a trusted ticket from Tableau Server

The following procedure will return a trusted ticket from Tableau Server.

1. Open the web page that you created in Step 3. The web browser may prompt you to allow scripts to run. This operation requires Javascript.

2. In the text boxes, enter the following:
   - **Username**: The test user that was created in Step 2.
   - **Server**: the address of your Tableau Server, e.g., `https://server_name`. 
- Client IP (optional): The IP address of the user's computer (only if configured for client trusted IP matching).
- Site: The name of the Tableau Server site where test user is hosted.

3. Click **Go**. One of the following will be returned:
   - **A unique ticket**: A trusted ticket is a string composed of a base64-encoded UUID and a 24-character random string, for example, 9D1OlxmDQmSI0yQpKdy4Sw==:dg62gCsSE0QRArXNTOp6mlJ5.
   - **-1**: If the value, -1 is returned, then there is an error in the configuration. See Ticket Value of -1 Returned from Tableau Server.

**Step 4: Test access with trusted ticket**

Now that you have a ticket, you can use it to access content on Tableau Server.

Construct a URL with the unique ticket that you generated in Step 4 to verify access with the trusted ticket. The URL syntax is different if you are accessing a Tableau Server with a single site vs a server that hosts multiple sites.

**Single (Default) site server url**

https://<Tableau Server name>/trusted/<unique ticket from Step 4>/views/<workbook>/<view>

**Non-default site server url**

https://<Tableau Server name>/trusted/<unique ticket from Step 4>/t/<site name where test user is hosted>/views/<workbook>/<view>

Variables in the URLs are encapsulated with angle brackets (< and >). All other syntax is literal.

**Troubleshoot Trusted Authentication**

This section includes some common issues and errors you might encounter when you're configuring trusted authentication. Trusted authentication information is written to /var/-opt/tableau/tableau_server/data/tabsvc/logs/vizqlserver/vizql-
To increase the logging level from info to debug, use the tsm configuration set Options setting vizqlserver.trustedticket.log_level.

To test your trusted authentication deployment, see Step 5: Test trusted authentication configuration.

Ticket Value of -1 Returned from Tableau Server

Tableau Server returns -1 for the ticket value if it cannot issue the ticket as part of the trusted authentication process. The exact reason for this message is written to the vizql*.log files in the following folder:
/var/opt/tableau/tableau_server/data/tabsvc/logs/vizqlserver

Here are some things to confirm:

- **All web server host names or IP addresses are added to trusted hosts**
  
  The IP address or host name for the computer sending the POST request must be in the list of trusted hosts on Tableau Server. See Configure Trusted Authentication to learn how to add IP addresses or host names to this list.

- **IP addresses are IPv4**
  
  If you are using IP addresses to specify trusted hosts, they must be in Internet Protocol version 4 (IPv4) format. An IPv4 address looks like this: 123.456.7.890. IPv6 addresses (for example, fe12::3c4a:5eab:6789:01c%34) are not supported as a way of inputting trusted hosts.

- **Username in POST request is a valid Tableau Server user**
  
  The username you send in the POST request must be a licensed Tableau Server user with a Viewer or Interactor license level. You can see a list of users and their license levels by signing in to Tableau Server as an administrator and clicking the Licensing link on the left side of the page.

- **Username in POST request includes domain**
If Tableau Server is configured to use Local Authentication, the username that you send in the POST can be a simple string. However, if the server is configured for Active Directory you must include the domain name with the user name (domain\username). For example, the username parameter might be: `username=dev\jsmith`

- **Content-Type is specified**

  If you are designing an ASP.NET or C# application, you need to declare the content type in your HTTP request. For example, `http.setRequestHeader("Content-Type","application/x-www-form-urlencoded; charset=UTF-8")`. If you do not specify content type and Tableau Server returns a -1, the log files contain the error: "missing username and/or client_ip".

HTTP 401 - Not Authorized

If you receive a 401- Not Authorized error, you may have configured Tableau Server to use Active Directory with SSPI. If your web server uses SSPI, you do not need to set up trusted authentication. You can embed views and your users will have access to them as long as they are licensed Tableau server users and members of your Active Directory.

See tsm authentication sspi <commands>.

If you see a 401 error (or a 302 - Redirect error) after you have deployed Tableau Server 10.5, then it's likely the trusted ticket code you have written to construct the URL for the client has not been updated to account for the two-part ticket URL format.

See Step 2: Configure your web server to receive tickets from Tableau Server.

HTTP 404 - File Not Found

You may receive this error if your program code references a Tableau Server URL that does not exist. For example, your web server may construct an invalid URL that cannot be found when the webpage tries to retrieve it.

Invalid User (SharePoint or C#)

You may encounter this error if you've configured Tableau Server for trusted authentication.
The example code for the SharePoint .dll references the following GET request:

```csharp
```

The above request will return the display name of the current Windows Active Directory user. If you want to use the login ID, then you will need to change the code to:

```csharp
```

After you make the change, recompile the SharePoint .dll.

## Attempting to Retrieve the Ticket from the Wrong IP Address

You may encounter this error if you've configured Tableau Server for trusted authentication.

The client web browser IP address is not considered by default when redeeming the ticket. If Tableau Server is configured to enforce client IP address matching, make sure that the client's web browser IP address that is sent in the POST to Tableau Server is the same as when the browser tries to retrieve the embedded view. For example, in the Trusted Authentication diagram, if the POST request in step 3 sends the parameter `client_ip=74.125.19.147`, then the GET request in step 5 must come from that same IP address.

See Step 4: (Optional) Configure client IP matching to learn how to configure Tableau Server to enforce client IP address matching.

## Cookie Restriction Error

When a user signs in to Tableau Server, a session cookie is stored in their local browser. The stored cookie is how Tableau Server maintains that the signed in user has been authenticated and can access the server. Because the cookie is set with the same domain or subdomain as the browser’s address bar, it is considered a first-party cookie. If a user's browser is configured to block first-party cookies, they will be unable to sign in to Tableau Server.

When a user signs in to Tableau Server via an embedded view, or in an environment where trusted authentication has been configured, the same thing happens: a cookie is stored. In this case, however, the browser treats the cookie as a third-party cookie. This is because the cookie is set with a domain that's different from the one shown in the browser's address bar.
If a user’s web browser is set to block third-party cookies, authentication to Tableau Server will fail. To prevent this from occurring, web browsers must be configured to allow third-party cookies.

An error occurred communicating with the server (403)

If Tableau Server is configured for trusted authentication, you may receive this error after opening a new view in a browser and attempting to navigate back to views you’d opened earlier. Tableau Server provides protection against unauthorized reuse of VizQL sessions through the tsm configuration set Options vizqlserver.protect_sessions, which is set to true by default. Because Tableau Server is configured for trusted authentication, you may not also need to enable vizqlserver.protect_sessions. To disable it, use tsm configuration set Options to change it to false.

Data Connection Authentication

The topics in this section describe how to configure authenticated connections to various data sources.

More information

- Tableau Server on Linux - Connecting to a Windows Shared Directory (Tableau Community)
- Setting an Oracle Connection to Use TNSNames.ora or LDAP.ora (Tableau Support)

Enable Kerberos Service Account Access

You can configure Tableau Server to use a service account to access a database on behalf of users on Tableau Desktop. In this scenario, Tableau Server connects to databases with a service account (also referred to as a "RunAs service account") in an Active Directory environment. Workbooks and data sources that use this service account are published as "Authentication: Server RunAs account" in Tableau Desktop.
Enabling Kerberos service account access is a requirement for Kerberos delegation. See Enable Kerberos Delegation.

Requirements

Kerberos service account access requires Active Directory.

- The Tableau Server information store must be configured to use LDAP - Active Directory.
- The Linux computer where Tableau Server is installed must be joined to Active Directory domain.
- MIT Kerberos is not supported.
- RunAs service account: This is an Active Directory user account that is used as the security principal to connect to the database on behalf of Tableau Server. The RunAs service account must have read access to the target database. The RunAs service account must be a user account in the Windows Active Directory domain. If your users are in a different Active Directory domain than Tableau Server and the data source, then domain trust must be configured. See Domain Trust Requirements.

Configuration process

This section provides an example of the process to enable Kerberos service account access. The scenario also includes example names to help describe the relationships between the configuration elements.

1. Create a domain user account to act as the RunAs service account. This account must have read access to the target database.

   In the example here, the RunAs service account is User principal named tab-srv@example.com.

2. Create a keytab file for the RunAs service account.

   For example, the following commands create a keytab (tabsrv-runas.keytab) using the ktutil tool:

   sudo ktutil
ktutil:  addent -password -p tabsrv@example.com -k 2 -e
<encryption scheme>

Encryption schemes for this command include RC4-HMAC,aes128-cts-hmac-shal-96,and aes256-cts-hmac-shal-96. Consult your IT team for the correct encryption scheme for your environment and data source.

ktutil:  wkt tabsrv-runas.keytab

Tableau Server will use the RunAs service account and the associated keytab to authenticate and make a direct connection to the database.

3. Copy the keytab into the Tableau Server data directory and set proper ownership and permissions.

    mkdir /var/opt/tableau/tableau_server/keytab
    sudo cp -p tabsrv-runas.keytab /var/opt/tableau/tableau_server/keytab
    sudo chown $USER /var/opt/tableau/tableau_server-keytab/tabsrv-runas.keytab
    chgrp tableau /var/opt/tableau/tableau_server-keytab/tabsrv-runas.keytab
    chmod g+r /var/opt/tableau/tableau_server/keytab/tabsrv-runas.keytab

4. Run the following TSM commands to enable RunAs access, set the RunAs service account, and associate the keytab file with the service account:

    tsm configuration set -k features.RunAsAuthLinux -v true
    tsm configuration set -k native_api.datasource_runas_principal -v tabsrv@example.com
    tsm configuration set -k native_api.datasource_runas_keytab_path -v /var/opt/tableau/tableau_server-keytab/tabsrv-runas.keytab

5. Run the following TSM command apply the changes to Tableau Server:
The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Enable Kerberos Delegation

Kerberos delegation enables Tableau Server to use the Kerberos credentials of the viewer of a workbook or view to execute a query on behalf of the viewer. This is useful in the following situations:

- You need to know who is accessing the data (the viewer's name will appear in the access logs for the data source).
- Your data source has row-level security, where different users have access to different rows.

Requirements

To enable Kerberos delegation, you must first configure Kerberos service account access. See Enable Kerberos Service Account Access.

Kerberos delegation requires Active Directory.

- The Tableau Server information store must be configured to use LDAP - Active Directory.
- The Linux computer where Tableau Server is installed must be joined to Active Directory domain.
- MIT Kerberos is not supported.
- Delegation service account: Grant delegation rights for this account to the target database Service Principal Names (SPNs). This account is delegated authority to access resources on behalf of the initiating source user. This account must be a user account in the Windows Active Directory domain. If your users are in a different Active Directory
domain than Tableau Server and the data source, then domain trust must be configured. See Domain Trust Requirements.

Configuration process

This section provides an example of the process to enable Kerberos delegation. The scenario also includes example names to help describe the relationships between the configuration elements.

1. Enable Kerberos Service Account Access

2. Tableau Server will need a Kerberos service ticket to delegate on behalf of the user that is initiating the call to the database. You must create a domain account that will be used to delegate to the given database. In this example, the user configured as the delegation account is delegate_user_account.

   If you are operating in Active Directory, then the delegation account must be configured with Active Directory User and Computers:

   - Open the **Properties** page for the delegated user, click the **Delegation** tab and select **Trust this user for delegation to specified services only** and **Use any authentication protocol**.

3. Set service principal names (SPN) in your directory to associate the Tableau Server as a Service principal with the delegation account. The following example shows how to set the SPNs and how to create a single keytab file (kerberos.keytab) for both delegation and SSO functionality.

   The following commands set the SPN and create a keytab file. Running these commands require administrative privilege on the Kerberos key distribution center (KDC).

   ```bash
   setspn -s HTTP/tableau-server EXAMPLE\delegate_user_account
   setspn -s HTTP/tableau-server.example.com EXAMPLE\delegate_user_account
   ```
ktpass /out kerberos.keytab /mapuser delegate_user_account@EXAMPLE.COM /princ HTTP/tableau-serv-
er.example.com@EXAMPLE.COM /pass * /crypto ALL

4. Run the following TSM commands to enable Kerberos delegation, set the delegation service account, and associate the keytab file with the service account:

```bash
tsm configuration set -k features.LinuxMITKerberos -v true
tsm configuration set -k native_api.datasource_impersonation_runas_principal -v HTTP/tableau-serv-
er.example.com@EXAMPLE.COM
tsm configuration set -k native_api.datasource_impersonation_runas_keytab_path -v /var/-
opt/tableau/tableau_server/keytab/kerberos.keytab
```

5. Run the following TSM command apply the changes to Tableau Server:

```bash
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `–r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

6. (Optional) If you are operating in an environment with multiple domains then you will need to configure `krb5.conf` to map principal names to local user names for each Kerberos realm. See Kerberos delegation multi-domain configuration.

7. Enable delegation for data connections:

   - **SQL Server**—See Enabling Kerberos Delegation for SQL Server in the Tableau Community.
   - **MSAS**—See Enabling Kerberos Delegation for MSAS in the Tableau Community.
Community.

- **PostgreSQL**—See Enabling Kerberos Delegation for PostgreSQL in the Tableau Community.

- **Teradata**—See Enabling Kerberos Delegation for Teradata in the Tableau Community.

OAuth Connections

For Google BigQuery, Google Analytics, Salesforce, OneDrive, Dropbox, and QuickBooks Online, an alternative to storing your sensitive database credentials with Tableau Server is to create connections using the **OAuth 2.0** standard.

From Tableau, when you sign in to data with a provider that uses OAuth, you are redirected to the provider’s sign-in page. After you provide your credentials and authorize Tableau to access your data, the data provider sends Tableau an **access token** that uniquely identifies requests from Tableau. For more information, see Overview of the OAuth process below.

Using OAuth connections provides the following benefits:

- **Security**: Your database credentials are never known to or stored in Tableau Server, and the access token can be used only by Tableau.

- **Convenience**: Instead of having to embed your data source ID and password in multiple places, you can use the token provided for a particular data provider for all published workbooks and data sources that access that data provider.

In addition, for live connections to Google BigQuery data, each workbook viewer can have a unique access token that identifies the user, rather than sharing a single user name and password credential.
Overview of the OAuth process

The following steps describe a workflow in the Tableau environment that calls the OAuth process.

1. You take an action that requires access to a cloud data source.

   For example, you open a workbook that’s published to Tableau Server.

2. Tableau directs you to the cloud data provider’s sign-in page. The information that is sent to the data provider identifies Tableau as the requesting site.

3. When you sign in to the data, the provider prompts you to confirm your authorization for Tableau Server to access the data.

4. Upon your confirmation, the data provider sends an access token back to Tableau Server.

5. Tableau Server presents your workbook and data to you.

The following workflows can use the OAuth process:
• Creating a workbook and connecting to the data source from Tableau Desktop or from Tableau Server.

• Publishing a data source from Tableau Desktop.

• Signing in to Tableau Server from an approved client, such as Tableau Mobile or Tableau Desktop.

Access tokens for data connections

You can embed credentials based on access tokens with data connections, to enable direct access after the initial authentication process. An access token is valid until a Tableau Server user deletes it, or the data provider revokes it.

It is possible to exceed the number of access tokens your data source provider allows. If that's the case, when a user creates a new token, the data provider uses length of time since last access to decide which token to invalidate to make room for the new one.

Access tokens for authentication from approved clients

By default, Tableau Server sites allow users to access their sites directly from approved Tableau clients, after users provide their credentials the first time they sign in. This type of authentication also uses OAuth access tokens to store the users' credentials securely.

For more information, see Disable Automatic Client Authentication

Configure the Server for OAuth Support

Instead of individual usernames and passwords, OAuth works through limited-purpose access tokens. Before you can obtain access tokens needed to create an OAuth connection in Tableau, you need to configure your server so that the data provider sending the token can recognize Tableau Server as a trusted destination. The following section describes how to prepare for setting up OAuth regardless of your data provider. The topics listed below it contain the steps for configuring your server for specific data providers.
Preparing for Configuring OAuth Support

Before you begin the configuration steps specific to your data provider, complete the following prerequisites:

- Obtain the fully qualified domain name of each Tableau Server node that will host views that connect to this data source. For example:

  https://sales.your_domain.com

  If you use Salesforce.com, you will need to provide an https address.

- Make sure at least one of your data-provider accounts is enabled for API access.

  For **Google** data types, you need access to the developers console on the [Google Cloud Platform](https://cloud.google.com/).  

  For **Salesforce.com**, you need access to the Force.com platform.

  For **QuickBooks Online**, you need access to the Intuit platform.

Configure Settings for Your Data Provider

When you complete the OAuth-preparation steps, you can configure the appropriate settings with your data provider.

- Set up OAuth for Google
- Set up OAuth for Salesforce.com
- Set up OAuth for QuickBooks Online

Set up OAuth for Google

This topic describes how to set up your Google BigQuery, Google Sheets, and Google Analytics data sources for OAuth. Complete these steps for each Tableau Server instance.
Note Before you complete these steps, make sure you have completed the pre-requisites described in Preparing for Configuring OAuth Support.

Set up OAuth by following these two procedures:

- Get required information from Google and enable API access.
- Use the information you obtained to configure your server.

**Obtain a Client ID and Enable Google APIs**

Note These steps reflect the settings in the Google Cloud Platform console at the time of this writing. For more information, see Using OAuth 2.0 for Web Server Applications in the Google Developers Console Help.

1. Sign in to Google Cloud Platform, and then click Go to my console.

2. On the drop-down menu next to the Google Cloud Platform title, select Create project.

3. In the new project form that appears, complete the following:

   - Give the project a meaningful name that reflects the Tableau Server instance for which you’ll use this project.
   - Determine whether you want to change the project ID.

   Note After you create the project, you will not be able to change the project
ID. For information, click the question mark icons.

4. Open the new project, and navigate to **APIs Manager > Credentials**.

5. Click the **OAuth consent screen** tab and then enter a meaningful name for **Product name** shown to users.

6. Click the **Credentials** tab, then click the **Create credentials** drop-down list, and then select **OAuth client ID**. Complete the following:

   - Select **Web Application**.
   - For **Authorized JavaScript Origins**, enter the local computer name of your Tableau Server.
   - For **Authorized Redirect URI**, replace the existing text with the Internet address for your server, and add the following text to the end of it: `auth/add_oauth_token`. For example:
     
     ```
     https://your_server_url.com/auth/add_oauth_token
     ```

7. Copy the Authorized Redirect URI, and paste it in a location that you can access from your Tableau Server computer.

8. Click **Create**.
9. Copy the following values that Google returns, and paste them in a location that you can access from your Tableau Server computer:

   - Client ID
   - Client secret

10. In APIs Manager > Dashboards, verify that BigQuery API, Google Drive API (to enable Google Sheets) or Analytics API is enabled. To enable APIs, click ENABLE API at the top of the page.

### Configure Tableau Server for Google OAuth

Using the information you obtained by completing the steps in Obtain a Client ID and Enable Google APIs, configure your Tableau Server:

- On the Tableau Server computer, open the bash shell and run the following commands:

  tsm configuration set -k oauth.google.client_id -v <your_client_ID>
  tsm configuration set -k oauth.google.client_secret -v <your_client_secret>
  tsm configuration set -k oauth.google.redirect_uri -v <your_authorized_redirect_URI>
  tsm pending-changes apply

The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the -r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.
Managing access tokens

After you configure the server for OAuth, you can allow users to manage their own access tokens in their profile settings, or you can manage the tokens centrally. For more information, see Allow Saved Access Tokens.

Set up OAuth for Salesforce.com

This topic describes how to set up your Salesforce.com data sources for OAuth. Complete these steps for each Tableau Server instance.

**Note:** Before you complete these steps, make sure you have completed the pre-requisites described in Preparing for Configuring OAuth Support.

Set up OAuth by following these two procedures:

- Create a Connected App in Salesforce
- Use the information you obtained to configure your server.

Create a Connected Salesforce App

1. Sign in to your Salesforce.com developer account, click your user name in the upper-right, and then select *Setup*.

2. In the left navigation column, under App Setup, select *Create > Apps*. 

- 548 -
3. In the Connected Apps section, click **New**.

![Connected Apps](image)

4. Complete the **Basic Information**, and in the API section, select **Enable OAuth Settings**.

5. In the new OAuth settings that appear, for **Callback URL**, type the fully qualified domain name of your server, using the https protocol, and append the following text to the URL: `auth/add_oauth_token`.

   For example:

   `https://www.your_server.com/auth/add_oauth_token`

6. Move the following items from Available OAuth Scopes to Selected OAuth Scopes:

   - Access and manage your data (api)
   - Access your basic information (id)
   - Perform requests on your behalf at any time (refresh_token)
7. Click **Save**.

After you save the app, Salesforce populates the API section with the following IDs that you will use to configure Tableau Server:

- Consumer Key
- Consumer Secret
- Callback URL

**Configure Tableau Server for Salesforce.com OAuth**

1. On the Tableau Server computer, open the bash shell and run the following commands:

   `tsm configuration set -k oauth.salesforce.client_id -v <your_client_ID>`

   `tsm configuration set -k oauth.salesforce.client_secret -v <your_client_secret>`

   `tsm configuration set -k oauth.salesforce.redirect_uri -v <your_authorized_redirect_URI>`

2. (Optional) To change the default login server, type the following command:
tsm configuration set -k oauth.salesforce.server_base_url -v <URL>

3. Enter the following command to apply changes:

    tsm pending-changes apply

    The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `--r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Managing access tokens

After you configure the server for OAuth, you can allow users to manage their own access tokens in their profile settings, or you can manage the tokens centrally. For more information, see Allow Saved Access Tokens.

Set up OAuth for QuickBooks Online

This topic describes how to set up your QuickBooks Online data sources for OAuth authentication. Complete these steps for each Tableau Server instance.

**Note:** Before you complete these steps, make sure you have completed the prerequisites described in Preparing for Configuring OAuth Support.

Setting up OAuth for QuickBooks Online consists of two tasks:

- Create a Connected App on the Intuit developer platform.

- Use the information you get as part of the Connected App to configure your server.
Create a Connected Intuit App

1. Sign in to your Intuit developer account, and then click My Apps.

2. In the Just start coding section, click Select APIs.

3. Select Accounting and click Create App.

4. In the Get your app ready for submission section, click the link to get your production keys.

   Important: You must use production keys rather than development keys.

5. Copy the app token, OAuth consumer key, and OAuth consumer secret.

Configure Tableau Server for QuickBooks Online

- On the Tableau Server computer, open the bash shell and run the following commands:

  tsm configuration set -k oauth.quickbooks.oauth_callback_uri -v http://YOUR-SERVER/auth/add_oauth_token

  tsm configuration set -k oauth.quickbooks.consumer_key -v <your_consumer_key>

  tsm configuration set -k oauth.quickbooks.consumer_secret -v <your_consumer_secret>

  tsm pending-changes apply

The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the
server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

### Managing access tokens

If you run an extract refresh job for your QuickBooks Online data source, Tableau Server attempts to renew access tokens for you. To help ensure that your access tokens do not expire, run your extract refresh jobs more than once a month. Otherwise, the access tokens from QuickBooks Online expire and your extract refresh jobs fail. If your access tokens do expire, you can edit your saved credentials from the **Settings** page.

The saved credentials can be managed centrally or by your users. For more information, see Allow Saved Access Tokens.

#### Allow Saved Access Tokens

After you configure Tableau Server for OAuth, you can decide to allow users to manage their own OAuth credentials, or you want to manage them centrally. If you want users to manage their own, you need to enable user profile settings from the server.

**Note:** If you have not yet configured your server to enable OAuth data connections, see the related topics listed below.

1. Sign in to Tableau Server as a server administrator.

2. **Single-site:** Click **Settings > General**.

   **Multisite:** In the site menu, click **Manage All Sites** and then click **Settings > General**.

3. In the **Saved Credentials** section, select the following:
• Allow users to save passwords for data sources (allows users to save their individual credentials with data sources).

• Allow users to save OAuth access tokens for data sources

4. Click **Save**.

After you select these check boxes, users will see a **Manage Credentials** section in their profile settings, where they can add access tokens for OAuth data connections.

Managing credentials centrally

Server administrators alternatively can manage OAuth credentials centrally. This can work well, for example, if multiple users work from the same data, and you have a dedicated user account for your data provider.

To manage credentials centrally, you do the following:
• Clear the check boxes described in the preceding procedure.

• Edit connection information as data sources are published.

When you edit the connection, you embed credentials that use an OAuth access token instead of an individual’s user name and password.

When the settings for saving passwords and access tokens are not enabled, the Manage Credentials section is excluded from users’ profile settings.

See also

Set up OAuth for Google

Set up OAuth for Salesforce.com

Set up OAuth for QuickBooks Online

Configure SAP HANA SAML delegation

You can configure Tableau Server to use SAML delegation to provide a single sign-on (SSO) experience for SAP HANA. This scenario is not dependent on SAML authentication to Tableau Server. You do not need to use SAML sign on with Tableau Server in order to use HANA SAML delegation. You can sign in to Tableau Server using whatever method you choose.

With SAML delegation for SAP HANA, Tableau Server functions as an identity provider (IdP).

Before you begin

Configuring SAML delegation with SAP HANA requires configuration on both Tableau Server and on SAP HANA. This topic provides configuration information about configuring Tableau Server. Before you configure Tableau Server, you must complete the following:

• Acquire a SAML certificate and key file for Tableau Server. The certificate file must be a PEM-encoded x509 certificate with the file extension .crt or .cert. This file is used by
Tableau Server, and must also be installed on HANA. The private key must be a DER-encoded private key file that is not password protected, and that has the file extension .der. This file is only used by Tableau Server.

- Install the certificate in HANA.
- Install the latest version of SAP HANA driver (minimum version is 1.00.9) on Tableau Server.
- Configure network encryption from Tableau Server to SAP HANA (recommended).

For more information about generating the certificate/key pair, encrypting the SAML connection, and configuring SAP HANA, see How to Configure SAP HANA for SAML SSO with Tableau Server in the Tableau Community.

Configure Tableau Server SAML for SAP HANA

The following procedure describes how to configure SAML for SAP HANA on Tableau Server using `tsm data-access set-saml-delegation`. You can also configure SAML for SAP HANA using the `sapHanaSettings Entity`.

If you are running Tableau Server in a distributed deployment, then you may need to copy the certificate and key files to other nodes in the cluster. See Copying Files in a Distributed Deployment.

1. Place certificate files in a folder named `saml`. The `saml` folder should be in the `/var/opt/tableau` hierarchy. For example:

   `/var/opt/tableau/tableau_server/data/saml`

2. Run the following command to enable SAML delegation and to specify the location of the certificate and key files:

   `tsm data-access set-saml-delegation --enabled enabled --cert-key <cert-key> --cert-file <cert-file>

   Where `<cert-key>` and `<cert-file>` are file paths to the private key and certificate file, respectively.

   For example,
You can specify other options. For example, you can specify user name format and how credentials are normalized. See tsm data-access set-saml-delegation.

3. When you have finished, run `tsm pending-changes apply`.

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

Authorization

`Authorization` refers to how and what users can access on Tableau Server after authentication has been verified. Authorization includes:

- What users are allowed to do with content hosted on Tableau Server, including projects, sites, workbooks, and views.
- What users are allowed to do with the data sources that are managed by Tableau Server.
- What tasks users are allowed to perform to administer Tableau Server, such as configuring server settings, running command line tools, creating sites, and other tasks.

Authorization for these actions is managed by Tableau Server and determined by a combination of the user’s site role and permissions associated with specific entities such as workbooks and data sources.

Site Roles

Site roles are permission sets that are assigned to a user, such as System Administrator, Publisher, or Viewer. The site roles define collections of capabilities (delete, save, view, and
others) that can be granted to users or groups on Tableau Server.

Site roles define who is an administrator. Administrators can be assigned at the site or server level. Site roles also determine whether non-admin users are allowed to publish to the server from Tableau Desktop. In general, site roles determine the maximum capabilities that can be granted for each non-admin user. For example, if a user’s site role is Interactor, the user cannot publish to the server, no matter what other permissions the user has, because the Interactor role denies permission to publish.

For more information about site roles, see Set Users’ Site Roles.

Permissions

Permissions determine whether a given user is allowed or denied to perform a specific action on a specific resource.

As an administrator setting up Tableau Server, it’s important that you understand how permissions are evaluated. Understanding the Tableau permissions process will enable you to set up and configure permissions on sites, projects, and other resources so that you can control how content and data is shared, published, viewed, extracted, and imported.

Four important concepts to understand about permissions in Tableau are:

- **Permissions are resource-based.** Permissions are assigned to individual resources and are granted to users or groups. Permissions are evaluated for projects, workbooks, views, and data sources.

- **Permissions are implicitly denied, and non-admin users must explicitly be allowed to access resources.** The process by which Tableau Server determines the “allow” or “deny” permission is explained in detail in the topic, How Permissions are Evaluated.

- **Permissions inheritance exists only in locked projects and in workbooks with tabbed views.** When content permissions are locked to the project, its workbooks, views, and data sources will always use the default permissions in the project. In the case of workbooks saved with the option **Show sheets as tabs**, views will use the workbook permissions. For more information, see Content Permissions and Ownership.
• **In a project that is not locked, initial permissions are a one-time copy of the container item’s permissions.** A workbook, view, or data source will start with the default permissions, but authorized users can subsequently edit permissions on those resources. For more information on default permissions and projects, see Set Project Default Permissions and Lock the Project.

Tableau Server provides a flexible permissions infrastructure that allows you to manage access to all content for countless scenarios. See Content Permissions and Ownership for more detailed information.

**Data Access and External Authorization**

There are scenarios where Tableau Server and Desktop rely on external authorization to enable access to data. For example:

- Users connecting to external data sources may require authorization that is outside the scope of Tableau Server’s authority. If users publish an external data source, then Tableau Server will manage access and capabilities of data source. But if users embed an external data source in a workbook, then it’s up to the users who publishes the workbook to determine how other users who open the workbook will authenticate with the data source.
- Running Tableau Server in an organization with Active Directory where Tableau has been configured with a Run As user account results in a dependency on Active Directory and NTFS for authorization. For example, if you configure Tableau Server to use the Run As account to impersonate users connecting to SQL, then object-level authorization is reliant on NTFS and Active Directory.
- How users authenticate and are authorized by specific database solutions may differ. As noted, Tableau Server can be configured to provide access authorization when a data source is configured, but some databases will authorize access according to their own authentication scheme.

**Data Security**

Tableau provides several ways for you to control which users can see which data. For data sources that connect to live databases, you can also control whether users are prompted to
provide database credentials when they click a published view. The following three options work together to achieve different results:

- **Database login account:** When you create a data source that connects to a live database, you choose between authenticating to the database through Windows NT or through the database’s built-in security mechanism.

- **Authentication mode:** When you publish a data source or a workbook with a live database connection, you can choose an **Authentication mode**. Which modes are available depends on what you choose above.

- **User filters:** You can set filters in a workbook or data source that control which data a person sees in a published view, based on their Tableau Server login account.

The table below outlines some dependencies with the above options:

<table>
<thead>
<tr>
<th><strong>Database Connection Options</strong></th>
<th><strong>Data Security Questions</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Database login account uses...</td>
<td>Is database security possible per Tableau Server user?</td>
</tr>
<tr>
<td>Authentication mode</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Window NT Integrated Security (Windows Authentication)</strong></td>
<td><strong>Server Run As account</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Impersonate via server Run As account</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Viewer Credentials</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Prompt user: Viewers are prompted for their database cre-</strong></td>
</tr>
<tr>
<td><strong>Database Connection Options</strong></td>
<td><strong>Data Security Questions</strong></td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Database login account uses...</td>
<td>Is database security possible per Tableau Server user?</td>
</tr>
<tr>
<td><strong>Username and Password</strong></td>
<td>Authentication mode</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Because it can create unexpected results, Tableau recommends that you not use this authentication mode with user filters.

User filters, the embedded credentials option and the impersonation modes have similar effects—when users click a view, they are not prompted for database credentials and they see only the data that pertains to them. However, user filters are applied in the workbook by
authors, and the impersonation authentication modes rely on security policies defined by administrators in the database itself.

Manage Server Secrets

Tableau Server needs to store a number of secrets it uses to perform various functions, typically securing internal communication, communicating with other applications or the operating system, or providing secure communication with clients. In this context, the term secret may refer to a password, a token, or other string that is used to authenticate one entity to another.

There are two categories of secrets that are required to run Tableau Server. They differ according to how the secrets are generated:

- **Secrets that are generated by administrators.** These include credentials and associated secrets for the Run As User account and the SMTP credentials used by Tableau Server.
- **Secrets that are automatically generated by various processes in the system.** For example, a secret is required to protect communication between the Cluster Controller and ZooKeeper processes. And a number of different passwords are required for each service and programmatic user that communicates with Postgres.

Most secrets are encrypted while at rest. When a secret is needed, it is decrypted at run time.

This topic describes how secrets storage works and what you need to do to properly manage storage of secrets on Tableau Server.

Understanding how secrets storage works

During installation Tableau Server generates and stores a master key in a Java keystore. The master key is used to encrypt a configuration encryption key that is used across the system.

Whenever a new secret is created or updated, the secret is encrypted with the configuration encryption key. The encrypted value is then stored with its corresponding configuration
parameter in a YAML file on the server. Parameters that hold an encrypted value use the format, ENC(<encrypted string>), where <encrypted string> is a Base64-encoded encrypted string.

At run time, when a given secret needs to be accessed, the encrypted values are read into memory and decrypted with the configuration encryption key.

Tableau Server encrypts secrets using 256-bit AES in GCM mode. The keys used for secure storage are different than the asset keys that are used to encrypt embedded database credentials before they are stored in the repository.

Who has access to the master key?

In a default installation, the Java key store for Tableau Server will be replicated into the /tabsvc/keystores folder for each service on that node under /var/opt/tableau/tableau_server/data/tabsvc/config.

For example,

/var/opt/tableau/tableau_server-
/data/tabsvc/config/tabadminagent_<version_number>/tabsvc/keystores/tableauserver.jks.

If you use a custom install directory, then the keystore files will be found under

<install directory>/tableau_server/data/tabsvc/config/<service name_.version_number>/tabsvc/keystores

By default, the following users and groups have access to this directory:

- root
- tsmagent (user)
- tableau (user)
- members of the 'tableau' group
Import and export configuration information

Tableau Services Manager introduces the capability to import and export configuration information using tsm configuration.

**Note:** This version of Tableau Server does not support restoring configuration information from a backup. Instead, we recommend using the export and import configuration commands to backup and restore configuration information.

While configuration secrets are encrypted when stored on disk internally, when the configuration is exported to a file, secrets are written into the file in plain text. It is up to the administrator to take measures to protect this file. There are a variety of options available:

- Write the file to an encrypted file system.
- Write the file to a directory that is restricted to specific users or groups by file system permissions.
- PGP encrypt the output. Create a named pipe and provide it as the file argument, then use that as input to gpg encrypt it. The advantage is the secrets are never written to disk in plain text. gpg must be available and a private key available. An example of this method is shown in the following section.

Securing secrets for import and export operations

The example in this section describes one way of handing secrets to store them on a separate computer.

See the following external references for more information:

- [File encryption on the command line](#)
- [mkfifo(1) - Linux man page](#)

**Example: encrypt and export**

The following is an example of how to secure the file when exporting the configuration.

```bash
mkfifo -m 600 /tmp/secure1 &
(gpg --symmetric --batch --yes --passphrase-file ~/.secrets/pgppassphrase.txt --cipher-algo
```
AES256 --output encrypted.enc < /tmp/secure1 && tsm settings export -f /tmp/secure1 && rm /tmp/secure1

The details of this operation are:

- Create a named pipe with access limited by file permissions to rw for current user.
  
  mkfifo -m 600 /tmp/secure1

- Call gpg to encrypt the data sent to the named pipe, backgrounding it to a separate process. It will block waiting for data. The result will be a file containing the encrypted data.
  
  gpg --symmetric --batch --yes --passphrase-file ~/.secrets/pgppassphrase.txt --cipher-algo AES256 --output encrypted.enc < /tmp/secure1 &

- Call tsm to export the configuration, providing the named pipe as the file argument.
  
  tsm settings export -f /tmp/secure1

- Delete the named pipe.
  
  rm /tmp/secure1

The encrypted data is in the file "encrypted.enc."

**Example: decrypt and import**

The following is an example of how to decrypt and import the configuration.

mkfifo -m 600 /tmp/secret2 && (gpg --decrypt --batch --yes --passphrase-file ~/.secrets/pgppassphrase.txt encrypted.enc > /tmp/secret2 &)
&&

The details of this operation are:
- Create a named pipe with access limited by file permissions to rw for current user.

  mkfifo -m 600 /tmp/secure2

- Decrypt the configuration and send it to the named pipe. Background this to a separate process, it will block waiting to be read.

  gpg --decrypt --batch --yes --passphrase-file ~/.secret-s/pgppassphrase.txt encrypted.enc > /tmp/secret2 &

- Execute the tsm configuration import command, logging in as needed.

  tsm settings import -f /tmp/secret2

- Delete the named pipe.

  rm/tmp/secure1

The pending configuration contains the imported configuration.

Run tsm pending-changes apply to commit changes. The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

Worker and cluster nodes

When adding a new node to your Tableau Server cluster, you will first need to generate the node configuration file (tsm topology). The node configuration file contains a copy of the master keystore file used for encrypting the configuration secrets.

**Important:** We strongly recommend that you take additional measures to secure the node configuration file using similar mechanisms as discussed above when exporting a configuration file with secrets.
When installing and configuring Tableau Server on the new node, you will need to provide the node configuration file to the initialize-tsm command. You can use a similar technique as described above to decrypt the contents of the file that was previously encrypted and send it via a named pipe to the initialize-tsm command.

Secrets storage event logging

The following events related to secrets storage are logged:

- Generating new encryption keys
- Encryption key is rolled or changed
- Encrypting a new value in the configuration file

For more information about log files and where they are stored, see Work with Log Files.

Managing secrets

As a Tableau Server administrator the most important task related to secrets storage is to periodically update secrets. In some cases (server troubleshooting or auditing), you may need to retrieve a password.

For other operations, such as upgrading versions, backing up and restoring, or adding new nodes to a cluster—as noted above—Tableau Server manages secrets storage and related processes automatically.

Updating secrets

You should update secrets periodically, according to your company's security policy.

To update the master key and automatically generated secrets, you will need to follow the steps described below. For the list of secrets and configuration key and corresponding values see the table below.

**Note:** The following procedure stops the server. Run this procedure during non-business hours.

**Step 1: Generate a new key**
Run `tsm security regenerate-internal-tokens` to generate a new master key that will be added to the master key store file. This command also prepares changes to the configuration information, such as generating new service passwords and internal SSL certs used to communicate with Postgres.

**Step 2: Copy the keystore file to other nodes**

If you are running Tableau Server in a multi-node cluster, you must manually copy the keystore file to each node in the cluster before you apply changes.

Use a secure process to transfer the master keystore file. Place the file in the `keystores` directory:

- **Default installation:** `/var/opt/tableau/tableau_server-/data/tabsvc/config/tabadminagent_0.<server_version>/tabsvc/keystores`
- **Custom installation:** `<install directory>/tableau_server-/data/tabsvc/config/tabadminagent_0.<server_version>/tabsvc/keystores`

**Step 3: Apply changes**

Verify that you that the keystore file is on each node and then run `tsm pending-changes apply` on the node where you ran Step 1. The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

**Step 4: Start Tableau Server**

Run `tsm start` to restart Tableau Server.

**Retrieving passwords**

In some cases, you may need to retrieve a password for troubleshooting or other operations. For example, you may need the Postgres readonly user credentials that are generated and
encrypted by Tableau Server. In these cases, you can run a tsm command that will retrieve and decrypt the password for you.

The account you are using to run the tabadmin command must be a member of the Administrators group.

To retrieve a password, open Command Prompt and issue a `tsm configuration get` command for one of the parameters listed in the table below.

For example, to retrieve a password for the readonly Postgres user, type the following command:

```
$ tsm configuration get -k pgsql.readonly_password
```

The command will return the password in clear text:

```
$ tsm configuration get pgsql.readonly_password
password
```

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>clustercontroller.zookeeper.password</code></td>
<td>Password for cluster controller to connect to zookeeper.</td>
</tr>
<tr>
<td><code>filestore.zookeeper.password</code></td>
<td>Password for filestore to connect to zookeeper.</td>
</tr>
<tr>
<td><code>jdbc.password</code></td>
<td>Password for the rails Postgres user.</td>
</tr>
<tr>
<td><code>oauth.google.client_secret</code></td>
<td>Client secret of the Google Cloud Platform account.</td>
</tr>
<tr>
<td><code>oauth.quickbooks.consumer_secret</code></td>
<td>Consumer secret of the Intuit developer account.</td>
</tr>
<tr>
<td><code>oauth.salesforce.client_secret</code></td>
<td>Client secret of the Salesforce developer account.</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>pgsql.adminpassword</code></td>
<td>Password for the <code>tblwgadmin</code> Postgres user.</td>
</tr>
<tr>
<td><code>pgsql.readonly_password</code></td>
<td>Password for the <code>readonly</code> Postgres user.</td>
</tr>
<tr>
<td><code>pgsql.remote_password</code></td>
<td>Password for the <code>tableau</code> Postgres user.</td>
</tr>
<tr>
<td><code>redis.password</code></td>
<td>Password for the <code>tableau</code> Postgres user.</td>
</tr>
<tr>
<td><code>servercrashupload.proxy_server_password</code></td>
<td>Password for custom proxy server used to upload crash reports.</td>
</tr>
<tr>
<td><code>service.runas.password</code></td>
<td>Password of the <code>Run As</code> users. Stored temporarily.</td>
</tr>
<tr>
<td><code>ssl.key.passphrase</code></td>
<td>Optional passphrase used to protect the Apache SSL key.</td>
</tr>
<tr>
<td><code>svcmonitor.notification.smtp.password</code></td>
<td>SMTP Server password supplied by the administrator through TabConfig.exe.</td>
</tr>
<tr>
<td><code>tabadminservice.password</code></td>
<td>Password for the service that allows server admins to download log files through the web interface.</td>
</tr>
<tr>
<td><code>vizportal.openid.client_secret</code></td>
<td>This is the password (&quot;provider client secret&quot;) used for OpenID Connect SSO.</td>
</tr>
</tbody>
</table>
### Network Security

There are three main network interfaces in Tableau Server:

- **Client to Tableau Server**: The client can be a web browser, Tableau Mobile, Tableau Desktop, or the `tabcmd` utility.

- **Tableau Server to your database(s)**: To refresh data extracts or handle live database connections, Tableau Server needs to communicate with your database(s).

- **Server component communication**: This applies to distributed deployments only.

#### Client to Tableau Server

A Tableau Server client can be a web browser, a device running Tableau Mobile, Tableau Desktop, or `tabcmd` commands. Communications between Tableau Server and its clients use standard HTTP requests and responses. We recommend configuring Tableau Server for HTTPS for all communications. When Tableau Server is configured for SSL, all content and communications between clients are encrypted using SSL, and the HTTPS protocol is used for requests and responses.

By default, passwords are communicated from browsers and `tabcmd` to Tableau Server using 1024-bit public/private key encryption. This level of encryption is not considered robust enough for secure communications. Additionally, this method, where a public key is sent to the recipient in the clear and without network layer authentication is susceptible to man-in-the-middle attacks.
To adequately secure network traffic from clients to Tableau Server, you must configure SSL with a certificate from a trusted certificate authority.

Client access from the Internet

We recommend a gateway proxy server to enable secure client access from the internet to your Tableau Server. We do not recommend running Tableau Server in a DMZ or otherwise outside your protected, internal network.

Configure a reverse proxy server, with SSL enabled, to handle all inbound traffic from the internet. In this scenario, the reverse proxy is the only external IP address (or range of addresses if multiple reverse proxies are load-balancing inbound requests) that Tableau Server will communicate with. Reverse proxies are transparent to requesting clients, thereby obfuscating Tableau Server network information and simplifying client configuration.

For configuration information, see Configuring Reverse Proxies for Tableau Server.

Clickjack Protection

By default, Tableau Server has **clickjack protection** enabled. This helps prevent certain types of attacks in which the attacker overlays a transparent version of a page on top of an innocuous-looking page in order to lure a user into clicking links or entering information. With clickjack protection enabled, Tableau Server imposes certain restrictions on embedding views.

For more information, see Clickjack Protection.

**Tableau Server to your database**

Tableau Server makes dynamic connections to databases to process result sets and refresh extracts. It uses native drivers to connect to databases whenever possible and relies on a generic ODBC adapter when native drivers are unavailable. All communications to the database are routed through these drivers. As such, configuring the driver to communicate on non-standard ports or provide transport encryption is part of the native driver installation. This type of configuration is transparent to Tableau.

When a user stores credentials for external data sources on Tableau Server, they are stored encrypted in Tableau Server’s internal database. When a process uses those credentials to
query the external data source, the process retrieves the encrypted credentials from the internal database and decrypts them in process.

Tableau Server to the Internet

In some cases, where users connect to external data sources, such as the Tableau map servers, then Tableau Server will need to connect to the internet. We recommend that you run all components of Tableau inside your protected network. Therefore, connections to the internet may require that you configure Tableau Server to use a forward proxy.

Communication with the repository

You can configure Tableau Server to use Secure Sockets Layer (SSL) for encrypted communications on all traffic that is exchange with the Postgres repository to and from other server components. By default, SSL is disabled for communications between server components and the repository.

Server component communication in a cluster

There are two aspects to communication between Tableau Server components in a distributed server installation: trust and transmission. Each server in a Tableau cluster uses a stringent trust model to ensure that it is receiving valid requests from other servers in the cluster. Computers in the cluster running a gateway process accept requests from third parties (clients), unless they are fronted by a load balancer, in which case the load balancer receives the requests. Servers not running a gateway process only accept requests from other trusted members of the cluster. Trust is established by a whitelist of IP address, port, and protocol. If any of these are invalid, the request is ignored. All members of the cluster can communicate with each other.

When a user stores credentials for external data sources on Tableau Server, they are stored encrypted in Tableau Server's internal database. When a process uses those credentials to query the external data source, the process retrieves the encrypted credentials from the internal database and decrypts them in process.
Clickjack Protection

Tableau Server includes protection against clickjack attacks. *Clickjacking* is a type of attack against web pages in which the attacker tries to lure users into clicking or entering content by displaying the page to attack in a transparent layer over an unrelated page. In the context of Tableau Server, an attacker might try to use a clickjack attack to capture user credentials or to get an authenticated user to change settings on your server. For more information about clickjack attacks, see [Clickjacking](https://owasp.org/www-community/attacks/Clickjacking) on the Open Web Application Security Project website.

**Note:** Clickjack protection was available in previous versions of Tableau Server, but was disabled by default. New installations of Tableau Server 9.1 and later will always have clickjack protection on unless you explicitly disable it.

Effects of clickjack protection

When clickjack protection is enabled on Tableau Server, the behavior of pages loaded from Tableau Server changes in the following ways:

- **Tableau Server adds the** [X-Frame-Options: SAMEORIGIN](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options) **header to certain responses from the server.** In the current versions of most browsers, this header prevents the content from being loaded into an `<iframe>` element, which helps prevent clickjacking attacks.

- **The top-level page from Tableau Server cannot be loaded in `<iframe>` elements.** This includes the sign-in page. One consequence is that you cannot host Tableau Server pages in an application that you create.

- **Only views can be embedded.**

- **If an embedded view requires data source credentials,** a message is displayed in the `<iframe>` element with a link to open the view in a secure window where the user can safely enter credentials. Users should always verify the address of the opened window before entering credentials.
Views can be loaded only if they include the `:embed=y` parameter in the query string, as in this example:

http://<server>/views/Sales/CommissionModel?:embed=y

**Note:** When clickjack protection is enabled, embedded views that use the URL copied from the browser address bar might not load. These view URLs usually contain the hash symbol `#` after the server name (for example, http://myserver/#/views/Sales/CommissionModel?:embed=y) are blocked when clickjack protection is enabled on Tableau Server.

Disabling clickjack protection

You should leave clickjack protection enabled unless it is affecting how your users work with Tableau Server. If you want to disable clickjack protection, use the following `tsm` commands:

1. `tsm configuration set -k wgserver.clickjack_defense.enabled -v false`

2. `tsm pending-changes apply`

   The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Content Security Policy

Tableau Server supports the Content Security Policy (CSP) standard. CSP is intended to be an additional layer of security against cross-site scripting and other malicious web-based attacks. CSP is implemented as a HTTP response header that allows you to specify where external resources, such as scripts and images, can be safely loaded from.
See the Mozilla website for more information about CSP.

**Note:** If your Tableau Server is configured to use SAML for user authentication, then do not enable CSP. Running CSP on Tableau Server with SAML is not supported.

Configure and enable CSP

CSP is configured and enabled using the tsm configuration set Options command. If you are running Tableau Server in a distributed deployment, run these commands on the initial node in the cluster. The configuration will be applied across the cluster after you run tsm pending-changes apply.

**Step 1: Set default directives**

Tableau Server includes the set of default directives in the table below.

To set a directive, use the following tsm syntax:

```tsm configuration set -k content_security_policy.-directive.<directive_name> -v <value>
```

For example, to set the `connect_src` directive, run the following command:

```tsm configuration set -k content_security_policy.-directive.connect_src -v 'http://example.com'
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content_security_policy.-directive.default_src</td>
<td>'none'</td>
<td>Serves as a fallback for the other fetch directives. Valid values for <code>default_src</code>.</td>
</tr>
<tr>
<td>content_security_policy.-directive.connect_src</td>
<td><code>*</code></td>
<td>Restricts the URLs which</td>
</tr>
<tr>
<td>Directive</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>connect_src</code></td>
<td>*</td>
<td>Specifies valid sources for JavaScript.</td>
</tr>
<tr>
<td><code>script_src</code></td>
<td>*</td>
<td>Specifies valid sources for JavaScript.</td>
</tr>
<tr>
<td><code>style_src</code></td>
<td>* <code>unsafe-inline</code></td>
<td>Specifies valid sources for stylesheets.</td>
</tr>
<tr>
<td><code>img_src</code></td>
<td>* <code>data:</code></td>
<td>Specifies valid sources of images and favicons.</td>
</tr>
<tr>
<td><code>font_src</code></td>
<td>* <code>data:</code></td>
<td>Specifies valid sources for fonts loaded using @font-face.</td>
</tr>
<tr>
<td><code>frame_src</code></td>
<td>* <code>data:</code></td>
<td>Specifies valid sources for nested browsing contexts loading using elements such as <code>&lt;frame&gt;</code> and <code>&lt;iframe&gt;</code>.</td>
</tr>
</tbody>
</table>
Valid values for `frame_src`.

<table>
<thead>
<tr>
<th>content_security_policy.-directive.object_src</th>
<th>data:</th>
<th>Specifies valid sources for the <code>&lt;object&gt;</code>, <code>&lt;embed&gt;</code>, and <code>&lt;applet&gt;</code> elements. Valid values for <code>object_src</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>content_security_policy.-directive.report_uri</td>
<td>/vizql/csp-report</td>
<td>Instructs the user agent to report attempts to violate the CSP. These violation reports consist of JSON documents sent via an HTTP POST request to the specified URI. Valid values for <code>report_uri</code>.</td>
</tr>
</tbody>
</table>

Step 2: Add additional directives (optional)

The default directives included with Tableau Server are a subset of directives that are supported by CSP.

For a full list of supported CSP directives, go to https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy.

You can add directives to the existing default set, by using adding the new directive in the `content_security_policy.directive` namespace. You must include the `--force-keys` parameter when adding new directives. The syntax is as follows:

```
tsm configuration set -k content_security_policy.-directive.<new_directive_name> -v <value> --force-keys
```
For example, to add the `worker-src` directive, run the following command:

```bash
tsm configuration set -k content_security_policy.directive.worker-src -v 'http://*.example.com' --force-keys
```

Step 3: Specify report-only directives (optional)

You can configure CPS to report some directives and to enforce others. When you set `content_security_policy.enforce_enabled` to true, then all directives are enforced (even if `content_security_policy.report_only_enable` is also set to true).

To specify directives as "report-only" and not enforced, add the directives to the report_only_directive namespace. You must include the `--force-keys` parameter when adding new directives. The syntax is as follows:

```bash
tsm configuration set -k content_security_policy.report_only_directive.<directive_name> -v <value> --force-keys
```

For example, to report only on the `script_src` directive, run the following command:

```bash
tsm configuration set -k content_security_policy.report_only_directive.script_src -v 'http://*.example.com' --force-keys
```

Step 4: Enable CSP on Tableau Server

After you have configured directives, enable CSP on Tableau Server.

The following options are used to enable enforcement or report only mode for the directives you have set.

<table>
<thead>
<tr>
<th>Option</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>content_security_policy.enforce_enabled</code></td>
<td>false</td>
<td>Adds a CSP header to all requests so that any violation will be enforced by the browser.</td>
</tr>
</tbody>
</table>
content_security_policy.report_only_enabled  |  true  |  Adds a CSP header to all requests so that any violation will be recorded in our vizql-client logs, but will not be enforced by the browser.

To enable enforcement of the CSP directives that you’ve specified, run the following command:

```
tsm configuration set -k content_security_policy.enforce_enabled -v true
```

Step 5: Run tsm pending-changes apply

When you are finished configuring CSP, run tsm pending-changes apply.

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

View CSP report

To view CSP violations for a given viz, load the viz in a browser that includes developer tools. This example uses the Chrome browser.

1. Load a test viz with violations that is hosted on the Tableau Server deployment where you configured CSP.
2. Enter `CTRL+Shift+I` to open the developer tools in Chrome.
3. Click the `Network` tab.
4. In the `Filter` field, enter `csp-report`, and then click `Find All`.
- If there are no violations then the search will not return any CSP reports.

- If there are violations, click the Headers tab in the results pane and scroll to the bottom to view Request Payload.

SSL

SSL (Secure Sockets Layer) is a standard security technology that establishes an encrypted link between a web server and clients. To use SSL, you need to install an SSL certificate on Tableau Server.

You can configure Tableau Server to use SSL in the following ways:

- Use SSL for external HTTP traffic.

- Use mutual (two-way) SSL between clients (Tableau Desktop, web browsers, and tabcmd.exe) and Tableau Server.

- Use SSL for all HTTP traffic between internal server components and the repository.

If you are using mutual SSL, each client also needs a certificate.

**Note:** Tableau Server uses SSL for user authentication. SSL is not used to handle permissions and authorization to content (data sources and workbooks) hosted on Tableau Server.

Configure SSL for External HTTP Traffic to and from Tableau Server (Linux)

You can configure Tableau Server to use Secure Sockets Layer (SSL) encrypted communications on all external HTTP traffic. Setting up SSL ensures that access to Tableau Server is secure and that sensitive information passed between the server and Tableau clients—such as Tableau Desktop, the REST API, and so on—is protected.

Mutual SSL authentication is not supported on Tableau Mobile.
Looking for Tableau Server on Windows? See Configure External SSL.

After you have configured Tableau Server for SSL, the server accepts requests to the non-SSL port (default is port 80) and automatically redirects them to the SSL port 443. Tableau Server supports only port 443 as the HTTPS port. Tableau Server cannot support SSL on a computer where another application is using port 443.

In this article

- Preparing the environment
- SSL certificate requirements
- Enable SSL on Tableau Server

Preparing the environment

Before you enable SSL on Tableau Server, you need to acquire an SSL certificate that meets Tableau Server requirements, and then place its files in an appropriate location:

1. Purchase an Apache SSL certificate from a trusted Certificate Authority (CA). For example, Verisign, Thawte, Comodo, GoDaddy.

   The CA certificate must meet specific additional requirements. See SSL certificate requirements below.

   If you’re new to SSL, see the example at Generate a Key and Certificate Signing Request for tips on how to get started with the process of acquiring a certificate.

2. When you get the certificate files from the CA, save them to a location accessible by Tableau Server, and note the names of the certificate .crt and .key files and the location where you save them. You’ll need to provide this information to Tableau Server when you enable SSL.
A common practice is to place a copy of the certificate files in a location that's within the Tableau Server directory tree. For example:

/var/opt/tableau/tableau_server/data/ssl

Name the directory whatever is appropriate depending on whether you’ll use the certificate files only for SSL or also for configuring SAML authentication.

3. Some web browsers require additional configuration to accept certificates from certain providers. Refer to the documentation that the CA provides, and make any necessary browser-specific changes.

SSL certificate requirements

When you acquire an SSL certificate for external communication to and from Tableau Server, follow these guidelines and requirements:

- If your organization issues certificates with a local PKI, or if you are using certificates that are not issued by a trusted certificate authority, you’ll need a certificate authority (CA) certificate file to identify the trusted CA.

  The CA certificate file must be a valid PEM-encoded X509 certificate with the extension .crt. If you have multiple trusted certificate authorities, you can copy and paste the entire contents of each CA certificate, including the "BEGIN CERTIFICATE" and "END CERTIFICATE" lines, into a new file, and then save the file as CAs.crt.

- Use a SHA-2 (256 or 512 bit) SSL certificate. All major browsers will display warnings when connecting to SHA-1 certificates.

  As of the end of 2017, your up-to-date browsers may no longer connect to a server that presents an SHA-1 certificate.

- In addition to the certificate file, you must also acquire a corresponding SSL certificate key file. The key file Must be a valid RSA or DSA private key file (with the extension .key by convention).
You can choose to passphrase-protect the key file. The passphrase you enter during configuration will be encrypted while at rest. However, if you want to use the same certificate for SSL and SAML, you must use a key file that is not passphrase protected.

- A certificate chain file is required for Tableau Desktop on the Mac. The chain file is a concatenation of all of the certificates that form the certificate chain for the server certificate. All certificates in the file must be x509 PEM-encoded and the file must have a .crt extension (not .pem).

- For multiple host names or sub-domains, Tableau Server supports wildcard certificates.

Note: If you plan to configure Tableau Server for single-sign on using SAML, see About the certificate and key files in the SAML requirements to help determine whether to use the same certificate files for both SSL and SAML.

Enable SSL on Tableau Server

This section continues from Preparing the environment, earlier in this article. To configure external SSL on port 443, using the certificate file you acquired from a trusted CA and copied into the directory you created, run the following commands:

```
tsm security external-ssl enable --cert-file </location/and/file.crt> --key-file </location/and/file.key>
tsm pending-changes apply
```

See the command reference at tsm security external-ssl enable to determine whether you want to include additional options for external-ssl enable. Tableau has specific recommendations for the --protocols option.
The `external-ssl enable` command imports the information from the .crt and .key files. If you run this command on a node in a Tableau Server cluster, it also distributes the information to any other gateway node.

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

**Add SSL port to the local firewall**

If you are running a local firewall, then you must add the SSL port to the firewall on Tableau Server. The example below describes how to configure the firewall running on RHEL/CentOS distributions. The example uses `Firewalld`, which is the default firewall on CentOS.

1. Start firewalld:
   
   ```shell
   sudo systemctl start firewalld
   ```

2. Add port 443 for SSL. Run the following command:
   
   ```shell
   sudo firewall-cmd --permanent --add-port=443/tcp
   ```

3. Reload the firewall and verify the settings. Run the following commands:
   
   ```shell
   sudo firewall-cmd --reload
   sudo firewall-cmd --list-all
   ```

**Example: SSL Certificate - Generate a Key and CSR**

**Important:** This example is intended to provide general guidance to IT professionals who are experienced with SSL requirements and configuration. The procedure described in this article is just one of many available methods you can use to generate the required files. The process described here should be treated as an example and not as a recommendation.
When you configure Tableau Server to use Secure Sockets Layer (SSL) encryption, this helps ensure that access to the server is secure and that data sent between Tableau Server and Tableau Desktop is protected.

Looking for Tableau Server on Windows? See Example: SSL Certificate - Generate a Key and CSR.

Tableau Server uses Apache, which includes OpenSSL. You can use the OpenSSL toolkit to generate a key file and Certificate Signing Request (CSR) which can then be used to obtain a signed SSL certificate.

Steps to generate a key and CSR

To configure Tableau Server to use SSL, you must have an SSL certificate. To obtain the SSL certificate, complete the steps:

1. Generate a key file.
2. Create a Certificate Signing Request (CSR).
3. Send the CSR to a certificate authority (CA) to obtain an SSL certificate.
4. Use the key and certificate to configure Tableau Server to use SSL.

You can find additional information on the SSL FAQ page on the Apache Software Foundation website.

Configure a certificate for multiple domain names

Tableau Server allows SSL for multiple domains. To set up this environment, you need to modify the OpenSSL configuration file, openssl.conf, and configure a Subject Alternative Name (SAN) certificate on Tableau Server. See For SAN certificates: modify the OpenSSL configuration file below.

Generate a key

Generate a key file that you will use to generate a certificate signing request.
1. Change to the **Apache** directory for Tableau Server:

   `/opt/tableau/tableau_server/packages/apache.<version>`

2. Run the following command to create the key file:

   ```bash
   openssl genrsa -out <yourcertname>.key 4096
   ```

   **Note:** This command uses a 4096-bit length for the key. You should choose a bit length that is at least 2048 bits because communication encrypted with a shorter bit length is less secure. If a value is not provided, 512 bits is used.

**Create a certificate signing request to send to a certificate authority**

Use the key file you created in the procedure above to generate the certificate signing request (CSR). You send the CSR to a certificate authority (CA) to obtain a signed certificate.

**Important:** If you want to configure a SAN certificate to use SSL for multiple domains, first complete the steps in [For SAN certificates: modify the OpenSSL configuration file](#) below, and then return to here to generate a CSR.

1. Run the following command to create a certificate signing request (CSR) file:

   ```bash
   openssl req -new -key yourcertname.key -out yourcertname.csr
   ```

2. When prompted, enter the required information.
Note: For **Common Name**, type the Tableau Server name. The Tableau Server name is the URL that will be used to reach the Tableau Server. For example, if you reach Tableau Server by typing `tableau.example.com` in the address bar of your browser, then `tableau.example.com` is the common name. If the common name does not resolve to the server name, errors will occur when a browser or Tableau Desktop tries to connect to Tableau Server.

**Send the CSR to a certificate authority to obtain an SSL certificate**

Send the CSR to a commercial certificate authority (CA) to request the digital certificate. For information, see the Wikipedia article Certificate authority and any related articles that help you decide which CA to use.

**Use the key and certificate to configure Tableau Server**

When you have both the key and the certificate from the CA, you can configure Tableau Server to use SSL. For the steps, see **Configure External SSL**.

For SAN certificates: modify the OpenSSL configuration file

In a standard installation of OpenSSL, some features are not enabled by default. To use SSL with multiple domain names, before you generate the CSR, complete these steps to modify the `openssl.cnf` file.

1. Navigate to the Apache `conf` folder for Tableau Server.
   
   For example: `/opt/tableau/tableau_server-<version>/conf`

2. Open `openssl.cnf` in a text editor, and find the following line: `req_extensions = v3_req`
This line might be commented out with a hash sign (#) at the beginning of the line.

```c
UTF8Strings
# so use this option with caution!
string_mask = nombstr

# req_extensions = v3_req # The extensions to add to a
certificate request
r[ req_distinguished_name ]
countryName = Country Name (2 letter code)
```

If the line is commented out, uncomment it by removing the # and space characters from the beginning of the line.

3. Move to the [v3_req] section of the file. The first few lines contain the following text:

```c
# Extensions to add to a certificate request
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
```

After the `keyUsage` line, insert the following line:

```c
subjectAltName = @alt_names
```

If you're creating a self-signed SAN certificate, do the following to give the certificate permission to sign the certificate:

a. Add the `cRLSign` and `keyCertSign` to the `keyUsage` line so it looks like the following: `keyUsage = nonRepudiation, digitalSignature, keyEncipherment, cRLSign, keyCertSign`

b. After the `keyUsage` line, add the following line: `subjectAltName = @alt_names`

4. In the [alt_names] section, provide the domain names you want to use with SSL.
DNS.1 = [domain1]
DNS.2 = [domain2]
DNS.3 = [etc]

The following image shows the results highlighted, with placeholder text that you would replace with your domain names.

```
[ v3_req ]
# Extensions to add to a certificate request
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
subjectAltName = @alt_names

[alt_names]
DNS.1 = domain1
DNS.2 = domain2
DNS.3 = etc
```

5. Save and close the file.

6. Complete the steps in Create a certificate signing request to send to a certificate authority section, above.

Configure SSL for Internal Postgres Communication

This article applies to Tableau Server on Linux. For Tableau Server on Windows, see Configure Internal SSL.

You can configure Tableau Server to use SSL (TLS) for encrypted communication between the Postgres repository and other server components. By default, communication internal to Tableau Server is not encrypted.

While you enable support for internal SSL, you can also configure support for direct connections to the repository from Tableau clients, such as Tableau Desktop, Tableau Mobile,
and web browsers. For information, see Configure Postgres SSL to Allow Direct Connections from Clients.

Enable SSL for internal traffic

To enable SSL for internal traffic among the server components, run the following commands:

```bash
    tsm security repository-ssl enable

    tsm pending-changes apply
```

`repository-ssl enable` generates the server’s certificate and key files, which it places in the following directory:

```
/var/opt/tableau/tableau_server/data/tabsvc/config/pgsql_<version>/security
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

By default, Tableau Server requires SSL for traffic between the repository and other server components, as well as for direct connections from Tableau clients (for connections through the `tableau` or `readonly` users).

For the default state, you must also complete the steps described in Configure Postgres SSL to Allow Direct Connections from Clients.

If you run this command on a node in a cluster, it copies the required certificate file to the same location on each other node.
Options for repository-ssl enable

-i, --internal-only

Optional.

In this state, Tableau Server uses SSL between the repository and other server components, and it supports but does not require SSL for direct connections through tableau or readonly users.

If you include this option, you do not need to complete the steps to allow direct connections.

Configure Postgres SSL to Allow Direct Connections from Clients

When Tableau Server is configured to use SSL for internal communication with the Postgres repository, you can also require SSL for Tableau clients that connect directly to the repository. Direct connections include those using the tableau user or the readonly user. Examples of Tableau clients include Tableau Desktop, Tableau Mobile, and the REST API.

1. Run the following commands:

   tsm security repository-ssl enable

   tsm pending-changes apply

   This enables internal SSL support, generates new server certificate and key files, and requires all Tableau clients to use SSL to connect to the repository.

   The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the -r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

2. For each client computer that will connect directly to the repository, do the following:
• Copy the `server.crt` file to the client computer. You can find this file in the following directory:

```
/var/opt/tableau/tableau_server-
data/tabsvc/config/pgsql_<version>/security
```

**Note:** Do not copy `server.key` to the client computer. This file should reside only on the server.

• Import the certificate into the computer’s certificate store.

For information, use the documentation from the operating system manufacturer.

See also

For additional repository-ssl commands and options, see tsm security.

Configure Mutual SSL Authentication for Tableau Server on Linux

Using mutual SSL, you can provide users of Tableau Desktop, and other approved Tableau clients a secure, direct-access experience to Tableau Server. With mutual SSL, when a client with a valid SSL certificate connects to Tableau Server, Tableau Server confirms the existence of the client certificate and authenticates the user, based on the user name in the client certificate. If the client does not have a valid SSL certificate, Tableau Server can refuse the connection.

Mutual SSL authentication is not supported on Tableau Mobile.

To configure Tableau Server for mutual SSL, you need the following:

- **SSL certificate file**—A valid PEM-encoded x509 certificate with the extension .crt.

- **SSL certificate key file**—A valid RSA or DSA private key file (with the extension .key by convention).
SSL certificate chain file—(Optional for Tableau Server, required for Tableau Desktop on the Mac) Some certificate providers issue two certificates for Apache. The second certificate is a chain file, which is a concatenation of all the certificates that form the certificate chain for the server certificate. All certificates in the file must be x509 PEM-encoded and the file must have a .crt extension (not .pem).

1 Require SSL for external server communication

To configure Tableau Server to use SSL for external communication between Tableau Server and web clients, run the `external-ssl enable` command as follows, providing the names for the server certificate's .crt and .key files:

```
tsm security external-ssl enable --cert-file <file.crt> --key-file <file.key>
```

- For `--cert-file` and `--key-file`, specify the location and file name where you saved the server’s CA-issued SSL certificate (.crt) and key (.key) files.

- The above command assumes the you are signed in as a user that has the Server Administrator site role on Tableau Server. You can instead use the `-u` and `-p` parameters to specify an administrator user and password.

- If the certificate key file requires a passphrase, include the `--passphrase` parameter and value.

2 Use mutual SSL

Add mutual authentication between the server and each client, and allow for Tableau client users to be authenticated directly after the first time they provide their credentials.

1. Run the following command:

```
tsm authentication mutual-ssl configure --cert-file <file.crt>
```

For `--cert-file`, specify the location and file name of the server’s CA certificate (.crt) file, as in the previous step for external SSL.
See the remaining sections in this article for any additional options you might want to include with the `mutual-ssl configure` command.

2. Run the following commands to enable mutual SSL and apply the changes:

```
tsm authentication mutual-ssl enable

 tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Additional options for mutual SSL

You can use `mutual-ssl configure` to configure Tableau Server to support the following options.

For more information, see `tsm authentication mutual-ssl <commands>`.

**Fallback authentication**

When Tableau Server is configured for mutual SSL, authentication is automatic and a client must have a valid certificate. You can configure Tableau Server to allow a fallback option, to accept user name and password authentication.

```
tsm authentication mutual-ssl configure -f true
```

**User name mapping**

When Tableau Server is configured for mutual SSL, the server authenticates the user directly by getting the user name from their client certificate. The name that Tableau Server uses depends on how the server is configured for user authentication:
• **Local Authentication**—uses the UPN (User Principal Name) from the certificate.

• **Active Directory (AD)**—uses LDAP (Lightweight Directory Access Protocol) to get the user name.

You can override either of these defaults to set Tableau Server to use the common name.

```
tsm authentication mutual-ssl configure -m cn
```

For more information, see Mapping a Client Certificate to a User During Mutual Authentication

**Certificate Revocation List (CRL)**

You might need to specify a CRL if you suspect that a private key has been compromised, or if a certificate authority (CA) did not issue a certificate properly.

```
tsm authentication mutual-ssl configure -r <revoke-file.pem>
```

**Mapping a Client Certificate to a User During Mutual Authentication**

When you use mutual (two-way) SSL authentication, the client presents its certificate to Tableau Server as part of the authentication process. Tableau Server then maps user information in the client certificate to a known user identity. The strategy that Tableau Server uses to perform client mapping depends on the content of your organization’s client certificates.

This topic discusses the ways information in a client certificate can map to a user identity and how to change the way Tableau Server performs that mapping. To understand how the mapping happens and whether you need to change it, you must know how client certificates are structured in your organization.

• **User-name mapping options**

• **Change the certificate mapping**

• **Address user-name ambiguity in multi-domain organizations**
User-name mapping options

Tableau Server uses one of the following approaches to map a client certificate to a user identity:

- **Active Directory.** If Tableau Server is configured to use Active Directory for user authentication, when Tableau Server receives a client certificate, it passes the certificate to Active Directory, which maps the certificate to an Active Directory identity. Any explicit user name information in the certificate is ignored.

  **Note:** This approach requires client certificates to be published for the user accounts in Active Directory.

- **User principal name (UPN).** A client certificate can be configured to store the user name in the user principal name field. Tableau Server reads the UPN value and maps it to a user in Active Directory or to a local user.

- **Common name (CN).** A client certificate can be configured to store the user name in the common name field of the certificate. Tableau Server reads the CN value and maps it to a user in Active Directory or to a local user.

If you configure the server for Active Directory authentication and UPN or CN user-name mapping, put the user name in one of the following formats:

username, domain/username, or username@domain.

For example: asmith, example.org/asmith, or asmith@example.org.

If the server uses local authentication, the format of the name in the UPN or CN fields is not predetermined, but the name in the field must match a user name on the server.

**Change the certificate mapping**

You use the following command to map a client certificate to a user identity in Tableau Server:
tsm authentication mutual-ssl configure -m <value>

Possible values are ldap for Active Directory mapping, upn for UPN mapping, or cn for CN mapping.

When you first install and configure Tableau Server, the server sets the default user-name mapping to match the server’s authentication type:

- If the server is configured to use Active Directory, it also uses Active Directory for mapping the certificate to the user identity.

- If the server is configured to use local authentication, the server gets the user-name value from the UPN field in the certificate.

If the default behavior for how Tableau Server maps a user name to an identity is not correct for your server configuration, run the following set of commands to change the mapping to use the CN value:

```bash
tsm authentication mutual-ssl configure -m cn

```

```bash
tsm pending-changes apply

```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Address user-name mapping ambiguity in multi-domain organizations

Under some circumstances, the user name in a certificate’s UPN or CN field can be ambiguous. This ambiguity can lead to unexpected results when the user name is mapped to a user identity on the server.

For example, if Tableau Server is presented with a user name that does not include a domain, the server maps the user name to an identity using the default domain. This can
cause an incorrect user-name mapping, potentially assigning a user a different user’s identity and permissions.

This can occur particularly in environments where the following conditions apply:

- Your organization supports multiple Active Directory domains.
- The server is configured to use Active Directory authentication.
- The server is configured to use UPN or CN mapping.
- Some users have the same user name but different domains. For example, asmith@example.org and asmith@example.com.
- The user name in the certificate’s UPN or CN fields does not include the domain as part of the user name—for example, it shows asmith.

To avoid incorrect user-name mapping, make sure the client certificates include fully qualified user names with the domain, using the format asmith@example.org or example.org/asmith.

See also

`tsm authentication mutual-ssl <commands>`

How Mutual SSL Authentication Works

Mutual (or two-way) SSL authentication provides a combination of an encrypted data stream, mutual authentication of both server and client, and direct access convenience. To use mutual SSL with Tableau Server, you need the following:

- A trusted CA-issued SSL certificate for Tableau Server.
- A certificate on each client that will connect to Tableau Server.
- A Tableau Server configured to use mutual SSL.

Tableau Server and the client verify that each other has a valid certificate, and Tableau Server authenticates the user, based on the user name in the client certificate.
The following image shows a little more detail about the sequence of events that occurs with mutual SSL.

1. The user navigates to Tableau Server.
2. Tableau Server sends its SSL certificate to the client computer.
3. The client computer verifies the Tableau Server certificate.
4. The client computer sends its certificate to Tableau Server.
5. Tableau Server verifies the client certificate.
6. Tableau Server references the user name in the client certificate to authenticate the user.

System User and sudo Privileges

This topic describes system user and sudo privilege in the context of Tableau Server.

Privilege separation

Following standard security best practices, Tableau Server for Linux runs processes with the least privilege possible. During installation, two users are created: tsmagent, and tableau. They are both added to group tableau, which is also created.
Example user entries in the `/etc/passwd` file are as follows:

- `tableau:x:993:991:Tableau Server:/sbin/nologin`
- `tsmagent:x:992:991:Tableau Services Manager Agent:/sbin/nologin`

The `tsmagent` user is used only to run the `tsmagent` process, which is part of TSM and performs all actions that require elevated privileges (see the next section for details on those privileges, provided by `sudo`). All other processes run as the unprivileged `tableau` user. This means that if one of the Tableau Server processes (such as a process displaying vizzes to users) were compromised in some fashion, it would only be able to impact Tableau Server, not the rest of the Linux system. For more information about system users and groups, in the context of installation and LDAP configuration, see Identity Store.

**sudo privileges in version 10.5**

**Important:** The current implementation of Tableau Server relies on `sudo` privileges. This approach will be changed in the next release. Updating the `sudoers` file conflicts with some system management configuration best practices and security policies. Therefore, the next version of Tableau Server will eliminate the use of the privileged user (`tsmagent`) and will not include a Tableau-specific `sudoers` file. Due to the changes required by removing the `sudoers` file, upgrading from Tableau Server 10.5 to the next version will require additional steps, which will be fully documented and supported by Tableau.

Tableau Services Manager provides facilities to configure and manage a variety of aspects of Tableau Server, and these changes can take place after the initial installation of TSM. Some of these changes require elevated privileges, such as modifying the host firewall configuration if the gateway port is changed. To provide these elevated privileges, Tableau Server on Linux installs a `sudoers` file during installation. By using `sudoers`, all needs for elevated privilege are listed explicitly. The `sudoers` file is generated from a template at installation time (some commands have different names in different Linux distributions), and made operational by placing it in the `/etc/sudoers.d` directory. By convention, the default `/etc/sudoers`
configuration file for sudo will include any file found in the /etc/sudoers.d directory. This is done by with lines in /etc/sudoers like this:

```plaintext
## Read drop-in files from /etc/sudoers.d (the # below is not a comment)
#include /etc/sudoers.d
```

**Note:** These lines must be added to the end of the sudoers config file, because sudo uses the last matching entry for a user.

If for some reason the `#includedir` directive is not present (by accident or by policy), then the new sudoers directives must be incorporated by some alternative method in order for Tableau Server on Linux to install and function properly.

All of the commands are run as NOPASSWD, since they are run by an automated process, and the requiretty option is turned off for the same reason.

The command specifications can be broken into different groups:

- Commands related to the management of the TSM services via systemd: creating and maintaining service files as well as starting, stopping enabling, and disabling services.

- Commands related to the management of the host firewall, including reading Tableau firewalld rule files as well as adding, removing, and reloading firewalld rules.

- Installing the FlexNet licensing service.

- Running PostgreSQL commands as the tableau user (not root). These privileges are required by postgres, which restricts access to the database directory to the owner (mode 700). As a result, these files cannot be accessed by the shared tableau group permission.
• Creating, inserting, and deleting TSM TLS certificates in the default system certificate store.

For more information on the sudoers file, see the example sudoers file below. Alternatively, you can view the sudoers file that is included with the installation by viewing the following file:

/etc/sudoers.d/tableau_server_<buildnumber>

Example sudoers file

# Regarding exclusion rules:
# The rules containing filepath wildcards are accompanied by exclusion rules to prevent matching tabs/spaces or ..
# The rules containing argument wildcards are accompanied by an exclusion rule to prevent matching tabs/spaces

# don't require a tty, since we only run sudo with NOPASSWD
Defaults:tsmagent !requiretty

# tsmagent user can write or delete a file to a very specific directory with very specific names
#tsmagent ALL = (root) NOPASSWD: /bin/rm /etc/sys
temd/system/tabsvc_0.service
#tsmagent ALL = (root) NOPASSWD: /bin/rm /etc/sys
temd/system/tabadminagent_0.service
#tsmagent ALL = (root) NOPASSWD: /bin/rm /etc/sys
temd/system/tabadmincontroller_0.service
#tsmagent ALL = (root) NOPASSWD: /bin/rm /etc/sys
temd/system/appzookeeper_0.service
#tsmagent ALL = (root) NOPASSWD: /bin/rm /etc/sys
temd/system/licenseservice_0.service
#tsmagent ALL = (root) NOPASSWD: /bin/rm /etc/fire
walld/services/tableau-* .xml, !/bin/rm /etc/fire
walld/services/tableau-* * .xml, !/bin/rm
/etc/firewalld/services/tableau-*..*.xml
#tsmagent ALL = (root) NOPASSWD: /usr/bin/tee
/etc/systemd/system/tabsvc_0.service
tsmagent ALL = (root) NOPASSWD: /usr/bin/tee /etc/systemd/system/tabadminagent_0.service
tsmagent ALL = (root) NOPASSWD: /usr/bin/tee /etc/systemd/system/tabadmincontroller_0.service
tsmagent ALL = (root) NOPASSWD: /usr/bin/tee /etc/systemd/system/appzookeeper_0.service
tsmagent ALL = (root) NOPASSWD: /usr/bin/tee /etc/systemd/system/licenseservice_0.service
tsmagent ALL = (root) NOPASSWD: /usr/bin/tee /etc/firewalld/services/tableau-*.xml, !/usr/bin/tee /etc/firewalld/services/tableau-* *.xml, !/usr/bin/tee /etc/firewalld/services/tableau-*..*.xml

# tsmagent user can read a specific file
tsmagent ALL = (root) NOPASSWD: /bin/cat /etc/firewalld/services/tableau-*.xml, !/bin/cat /etc/firewalld/services/tableau-* *.xml, !/bin/cat /etc/firewalld/services/tableau-*..*.xml

# tsmagent user can make changes to firewalld
tsmagent ALL = (root) NOPASSWD: /usr/bin/firewall-cmd --add-service tableau-*, !/usr/bin/firewall-cmd --add-service tableau-*

 tsmagent ALL = (root) NOPASSWD: /usr/bin/firewall-cmd --add-service tableau-* --permanent, !/usr/bin/firewall-cmd --add-service tableau-* --permanent
tsmagent ALL = (root) NOPASSWD: /usr/bin/firewall-cmd --remove-service tableau-*, !/usr/bin/firewall-cmd --remove-service tableau-*

 tsmagent ALL = (root) NOPASSWD: /usr/bin/firewall-cmd --remove-service tableau-* --permanent, !/usr/bin/firewall-cmd --remove-service tableau-* --permanent
tsmagent ALL = (root) NOPASSWD: /usr/bin/firewall-cmd --reload
# tsmagent user can reload the services so that changes are noticed

tsmagent ALL = (root) NOPASSWD: /bin/systemctl daemon-reload

# tsmagent user can start/stop/enable/disable the services it creates

tsmagent ALL = (root) NOPASSWD: /bin/systemctl start tabsvc_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl start tabadminagent_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl start tabadmincontroller_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl start app-zookeeper_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl start licenseservice_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl enable tabsvc_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl enable tabadminagent_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl enable tabadmincontroller_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl enable app-zookeeper_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl enable licenseservice_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl stop tabsvc_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl stop tabadminagent_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl stop tabadmincontroller_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl stop app-zookeeper_0

tsmagent ALL = (root) NOPASSWD: /bin/systemctl stop licenseservice_0
tsmagent ALL = (root) NOPASSWD: /bin/systemctl disable tabsvc_0

# tsmagent user can install licensing service
# tsmagent user can change ownership of postgresql directory
# tsmagent user can manage PostgreSQL status
server/data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql enable
tsmagent ALL = (tableau) NOPASSWD: /var/opt/tableau/tableau_
server/data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql disable
tsmagent ALL = (tableau) NOPASSWD: /var/opt/tableau/tableau_
server/data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql configure
tsmagent ALL = (tableau) NOPASSWD: /var/opt/tableau/tableau_
server/data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql restore begin /var/opt/tableau/tableau_server/
data/tabsvc/backuprestore/*, !/var/opt/tableau/tableau_server/
data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql restore begin /var/opt/tableau/tableau_server/
data/tabsvc/backuprestore/* *, !/var/opt/tableau/tableau_server/
data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql restore begin /var/opt/tableau/tableau_server/
data/tabsvc/backuprestore/*..* *
tsmagent ALL = (tableau) NOPASSWD: /var/opt/tableau/tableau_
server/data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql restore commit /var/opt/tableau/tableau_server/
data/tabsvc/backuprestore/*, !/var/opt/tableau/tableau_server/
data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql restore commit /var/opt/tableau/tableau_server/
data/tabsvc/backuprestore/* *, !/var/opt/tableau/tableau_server/
data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql restore commit /var/opt/tableau/tableau_server/
data/tabsvc/backuprestore/*..* *
tsmagent ALL = (tableau) NOPASSWD: /var/opt/tableau/tableau_
server/data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql restore rollback /var/opt/tableau/tableau_server/
data/tabsvc/backuprestore/*, !/var/opt/tableau/tableau_server/
data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
control-pgsql restore rollback /var/opt/tableau/tableau_
server/data/tabsvc/backuprestore/* *, !/var/
  opt/tableau/tableau_server/data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/control-pgsql restore rollback /var/
  opt/tableau/tableau_server/data/tabsvc/backuprestore/*..*
tsmagent ALL = (tableau) NOPASSWD: /var/opt/tableau/tableau_server/data/tabsvc/services/pgsql_0.server-dev.0.0.0/bin/-
  control-pgsql ziplogs

# tsmagent user can enable and extract certificates on Centos
tsmagent ALL = (root) NOPASSWD: /usr/bin/update-ca-trust enable
tsmagent ALL = (root) NOPASSWD: /usr/bin/update-ca-trust extract
tsmagent ALL = (root) NOPASSWD: /bin/rm /etc/pki/ca-trust/-source/anchors/TableauServer-*..*.crt, !/bin/rm /etc/pki/ca-trust/-source/anchors/TableauServer-* *.crt, !/bin/rm /etc/pki/ca-trust/source/anchors/TableauServer-*..*.crt
tsmagent ALL = (root) NOPASSWD: /usr/bin/tee /etc/pki/ca-trust/-source/anchors/TableauServer-*..*.crt, !/usr/bin/tee /etc/pki/ca-trust/source/anchors/TableauServer-* *.crt, !/usr/bin/tee /etc/pki/ca-trust/source/anchors/TableauServer-*..*.crt

Security Hardening Checklist

The following list provides recommendations for improving the security ("hardening") of your Tableau Server installation.

1. Update to the current version

We recommend that you always run the latest version of Tableau Server. Additionally, Tableau periodically publishes maintenance releases of Tableau Server that include fixes for known security vulnerabilities. (Information regarding known security vulnerabilities can be found on the Security Bulletins page.) We recommend that you review maintenance release notifications to determine whether you should install them.
To get the latest version or maintenance release of Tableau Server, visit the Customer Portal page.

2. Configure SSL/TLS with a valid, trusted certificate

Secure Sockets Layer (SSL/TLS) is essential for helping to protect the security of communications with Tableau Server. Configure Tableau Server with a valid, trusted certificate (not a self-signed certificate) so that Tableau Desktop, mobile devices, and web clients can connect to the server over a secured connection. For more information, see SSL.

3. Disable older versions of TLS

Tableau Server uses TLS to authenticate and encrypt many connections between components and with external clients. External clients, such as browsers, Tableau Desktop, Tableau Mobile connect to Tableau using TLS over HTTPS. Transport layer security (TLS) is an improved version of SSL. In fact, older versions of SSL (SSL v2 and SSL v3) are no longer considered to be adequately secure communication standards. As a result, Tableau Server does not allow external clients to use SSL v2 or SSL v3 protocols to connect. We recommend that you only allow external clients to connect to Tableau Server with TLS v1.2.

Specially, we recommend that you disable TLS v1 and TLS v1.1 on Tableau Server. However, before you disable a specific version of TLS, verify that the browsers that your users connect to Tableau Server with support TLS v1.2. In some cases, you may need to preserve support for TLSv1.1.

The following tsm command enables TLS v1.2 (using the "all" parameter) and disables SSL v2, SSL v3, TLS v1, and TLS v1.1 (by prepending the minus [-] character to a given protocol).

```
tsm configuration set -k ssl.protocols -v 'all -SSLv2 -SSLv3 -TLSv1 -TLSv1.1'
```

```
tsm pending-changes apply
```
The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the -r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

4. Disable Triple-DES cipher suite

The Triple-DES cipher suite is no longer considered adequate to encrypt sessions on the internet. Specifically, running Triple-DES ciphers leaves the Tableau Server vulnerable to information disclosure and denial of service attacks. You can learn more at the National Vulnerability Database webpage for CVE-2016-2183.

Triple-DES is enabled by default on the version of OpenSSL that is running on Tableau Server. However, other deprecated cipher suites (MD5 and RC4) are disabled. To add Triple-DES to the list of disabled ciphers, run the following commands. (The !aNULL parameter forces clients to use a legitimate cipher.)

```text
  tsm configuration set -k ssl.ciphersuite -v
  'HIGH:MEDIUM:!aNULL:!MD5:!RC4:!3DES'
  tsm pending-changes apply
```

5. Configure SSL encryption for internal traffic

Configure Tableau Server to use SSL to encrypt all traffic between the Postgres repository and other server components. By default, SSL is disabled for communications between server components and the repository. We recommend enabling internal SSL for all instances of Tableau Server, even single-server installations. Enabling internal SSL is especially important for multi-node deployments. See Configure SSL for Internal Postgres Communication

6. Enable firewall protection

Tableau Server was designed to operate inside a protected internal network.
**Important:** Do not run Tableau Server, or any components of Tableau Server on the internet or in a DMZ. Tableau Server must be run within the corporate network protected by an internet firewall. We recommend configuring a reverse proxy solution for internet clients that need to connect to Tableau Server. See Configuring Reverse Proxies for Tableau Server.

A local firewall should be enabled on the operating system to protect Tableau Server in single and multi-node deployments. In a distributed (multi-node) installation of Tableau Server, communication between nodes does not use secure communication. Therefore, you should enable firewalls on the computers that host Tableau Server. See Configure Local Firewall.

To prevent a passive attacker from observing communications between nodes, configure a segregated virtual LAN or other network layer security solution.

7. **Restrict access to the server computer and to important directories**

Tableau Server configuration files and log files can contain information that is valuable to an attacker. Therefore, restrict physical access to the machine that is running Tableau Server. In addition, make sure that only authorized and trusted users have access to the Tableau Server files in the `/var/opt/tableau/tableau_server/` directory.

8. **Generate fresh secrets and tokens**

Any Tableau Server service that communicates with repository or the cache server must first authenticate with a secret token. The secret token is generated during Tableau Server setup. The encryption key that internal SSL uses to encrypt traffic to Postgres repository is also generated at during setup.

We recommend that after you install Tableau Server, you generate new encryption keys for your deployment.
These security assets can be regenerated with the `tsm security regenerate-internal-tokens` command.

Run the following commands:

`tsm security regenerate-internal-tokens`

`tsm pending-changes apply`

9. Disable services that you're not using

To minimize the attack surface of the Tableau Server, disable any connection points that are not needed.

**REST API**

The REST API interface is enabled by default. If no applications will make REST API calls to your installation of Tableau Server 9.3 (or later), disable it by using the following commands:

`tsm configuration set -k api.server.enabled -v false`

`tsm pending-changes apply`

You can disable REST API only on versions of Tableau Server 9.3 and later.

**JMX Service**

JMX is disabled by default. If it's enabled but you're not using it, you should disable it by using the following:

`tsm configuration set -k service.jmx_enabled -v false`

`tsm pending-changes apply`

10. Verify session lifetime configuration

By default, Tableau Server does not have an absolute session timeout. This means that client sessions can remain open indefinitely if the Tableau Server inactivity timeout is not exceeded. (The default inactivity timeout is 240 minutes.)
If your security policy requires it, you can set an absolute session timeout. Before you do that, you must enable session lifetime timeout. Run the following commands.

```
tsm configuration set -k wgserver.session.apply_lifetime_limit -v true
```

```
tsm configuration set -k wgserver.session.lifetime_limit -v 'value', where value is the number of minutes. The default is 1440, which is 24 hours.
```

```
tsm configuration set -k wgserver.session.idle_limit -v 'value', where value is the number of minutes. The default is 240.
```

```
tsm pending-changes apply
```

**11. Enable HTTP Strict Transport Security for web browser clients**

HTTP Strict Transport Security (HSTS) is a policy configured on web application services, such as Tableau Server. When a conforming browser encounters a web application running HSTS, then all communications with the service must be over a secured (HTTPS) connection. HSTS is supported by major browsers.

For more information about how HSTS works and the browsers that support it, see The Open Web Application Security Project web page, [HTTP Strict Transport Security Cheat Sheet](https://httpsec.info/trusted-seal/).

To enable HSTS, run the following commands on Tableau Server:

```
tsm configuration set -k gateway.http.hsts -v true
```

By default, HSTS policy is set for one year (31536000 seconds). This time period specifies the amount of time in which the browser will access the server over HTTPS. You should consider setting a short max-age during initial roll-out of HSTS. To change this time period, run

```
tsm configuration set -k gateway.http.hsts_options -v max-age=<seconds>. For example, to set HSTS policy time period to 30 days, enter tsm configuration set -k gateway.http.hsts_options -v max-age=2592000.
```

```
tsm pending-changes apply
```
12. Disable Guest access

Core-based licenses of Tableau Server include a Guest user option, which allows any user in your organization to see and interact with Tableau views embedded in web pages.

Guest user access is enabled by default on Tableau Servers deployed with core-based licensing.

Guest access allows users to see embedded views. The Guest user cannot browse the Tableau Server interface or see server interface elements in the view, such as user name, account settings, comments, and so on.

If your organization has deployed Tableau Server with core licensing and Guest access is not required, then disable Guest access.

You can disable Guest access at the server or site level.

You must be a server administrator to disable the Guest account at either the server or the site level.

**To disable Guest access at the server level:**

1. In the site menu, click **Manage All Sites** and then click **Settings > General**.

2. For **Guest Access**, clear the **Enable Guest account** check box.

3. Click **Save**.

**To disable Guest access for a site:**

1. In the site menu, select a site.

2. Click **Settings**, and on the Settings page, clear the **Enable Guest account** check box.

For more information, see Guest User.
Change List

<table>
<thead>
<tr>
<th>Date</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 2017</td>
<td>Ported and updated for Linux platform.</td>
</tr>
</tbody>
</table>

Manage Licenses

Licensing Overview

Tableau Server licenses have two aspects: the license model (term or perpetual) and the license metric (user-based or core-based). Term licenses are user-based, and perpetual licenses usually core-based.

License model: term or perpetual

Tableau Server can be licensed under two models: term licensing and a perpetual licensing. Term licenses, also known as subscription licenses, allow you to use and update Tableau Server for a specified period of time.

Perpetual licenses do not expire, so you can continue to use Tableau Server as long as you want. However, to get access to product updates and technical support you must purchase Support and Maintenance services.

License metric: user-based or core-based

In addition to the license model, your license is also defined by the metric that permits use of Tableau Server.

- A user-based license metric allows you to deploy Tableau Server on a single computer or on multiple computers in a cluster. Each user that accesses Tableau Server must be licensed. Administrators add users and license them.

- A core-based license metric imposes no constraints on the number of user accounts
in Tableau Server. Instead, the license specifies the maximum number of computer cores on which you can run Tableau Server. You can install Tableau Server on a single computer or across multiple computers as a multi-node cluster, as long as the total number of cores in all the computers does not exceed the total number that the license allows.

**Note:** Not all processes installed with Tableau Server impact the calculation of total number of cores used. A subset of processes are considered "licensed processes." Core licensing is calculated only on computers running licensed processes. If a computer has one or more licensed processes installed on it, the cores on that computer count toward the total cores used. For more information about licensed processes, see Licensed processes.

The topics in this section provide guidance about how to view and refresh Tableau licensing, as well as a topic that describes how to add user capacity on Tableau Server.

**Note:** As a Tableau Server administrator, you may also be tasked with managing Tableau Desktop deployment and tracking license usage on client computers. If your organization will be using Desktop License reporting, you need to configure Tableau Server to support this. For details, see Configure Desktop License Reporting. For additional information on licensing and management tasks related to Tableau Desktop, see The Tableau Desktop Deployment Guide.

**View Server Licenses**

Server administrators can view the license and product key information for Tableau Server.
Tableau Server site roles do not correspond to user licenses that you purchase from Tableau (if you are using user-based licensing instead of core-based server licensing). Those licenses allow a certain number of users on the server.

To view server licenses

You can view licenses from Tableau Server, on the Licenses page, or from the command line, using the TSM command line interface.

Viewing licenses from the TSM command line

On the initial node where you installed Tableau Server, type this command:

```
tsm licenses list
```

to see the Tableau Server licenses.

The output depends on what type of license you have on your server, and can be similar to the following:

License: term or perpetual, metric: user-based or core-based

<table>
<thead>
<tr>
<th>KEY</th>
<th>SEATS</th>
<th>LIC EXP</th>
<th>MAINT EXP</th>
<th>GUEST ACCESS</th>
<th>comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;PROD-LKEY-VALU-HERE&gt;</code></td>
<td><code>&lt;nn&gt;</code></td>
<td><code>mm/dd/yy</code></td>
<td><code>mm/dd/yy</code></td>
<td><code>true</code></td>
<td>user-based, term license</td>
</tr>
<tr>
<td><code>&lt;PROD-LKEY-VALU-HERE&gt;</code></td>
<td><code>&lt;nn&gt;</code></td>
<td><code>permanent</code></td>
<td><code>mm/dd/yy</code></td>
<td><code>true</code></td>
<td>user-based, perpetual license</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>KEY</th>
<th>CORES</th>
<th>LIC EXP</th>
<th>MAINT EXP</th>
<th>GUEST ACCESS</th>
<th>comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;PROD-CORE-LKEY-HERE&gt;</code></td>
<td><code>&lt;nn&gt;</code></td>
<td><code>mm/dd/yy</code></td>
<td><code>mm/dd/yy</code></td>
<td><code>true</code></td>
<td>core-based, term license</td>
</tr>
<tr>
<td><code>&lt;PROD-CORE-</code></td>
<td><code>&lt;nn&gt;</code></td>
<td><code>permanent</code></td>
<td><code>mm/dd/yy</code></td>
<td><code>true</code></td>
<td>core-based,</td>
</tr>
</tbody>
</table>
License Type: trial

<table>
<thead>
<tr>
<th>KEY</th>
<th>LIC EXP</th>
<th>MAINT EXP</th>
<th>GUEST ACCESS</th>
<th>comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>LKEY-HERE&gt;</td>
<td></td>
<td></td>
<td>false</td>
<td>perpetual license</td>
</tr>
</tbody>
</table>

License Type: subscription

<table>
<thead>
<tr>
<th>KEY</th>
<th>USERS</th>
<th>LIC EXP</th>
<th>MAINT EXP</th>
<th>GUEST ACCESS</th>
<th>comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;PROD-SUBS-LKEY-HERE&gt;</td>
<td>&lt;nn&gt;/&lt;nn&gt;</td>
<td>mm/dd/yy</td>
<td>mm/dd/yy</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

Viewing licenses from the Tableau Server web UI

How you navigate to the Licenses page in Tableau Server depends on whether you have a single site, or multiple sites:

- On a server with a single site, click **Settings** and **Licenses**:

- On a multi-site server, click **Manage All Sites** on the site menu, **Settings**, and **Licenses**:
**Note:** The **Manage All Sites** option only displays when you are signed in as a server administrator.

If you have a user-based Tableau Server license, you can review how these levels have been distributed.

If you have a core-based Tableau Server license, the Licenses page shows how many cores are allowed, how many have been licensed, and how many are in use (and on what server computers).

Also see:

- Licensing Overview
- Handle an Unlicensed Server.

**Add Capacity to Tableau Server**

You may need to add capacity to your Tableau Server installation to allow you to increase the number of users (if you have a user-based license) or the number of cores (if you have a core-based license).

Tableau Software will provide you with a new product key that adds capacity to your existing Tableau Server installation. You need to activate this key and use it together with your existing product key(s) to get the combined capacity you are licensed for.

Follow the steps below to add a product key to Tableau Server.
**Note:** This process requires a restart of Tableau Server.

1. Copy the product key *(file.tlf)* to your computer.

2. Run the following command:

   ```
   tsm licenses activate --license-key <file.tlf>
   ```

3. Restart the server:

   ```
   tsm restart
   ```

**Automate Licensing Tasks**

Use the tsm licenses category to automate licensing tasks.

**Handle an Unlicensed Server**

Tableau offers two licensing models: user-based and core-based. User-based licensing requires each active user account to be covered by a license. User-based licenses have a defined capacity, or number of users that it allows. Each user is assigned a unique user name on the server and is required to identify himself when connecting to the server.

Core-based licensing has no constraints on the number of user accounts in the system, but it does restrict the maximum number of processor cores that Tableau Server can use. You can install Tableau Server on one or more machines to create a cluster, with the restriction that the total number of cores in all the machines does not exceed the number of cores you have licensed and that all of the cores on a particular machine are covered by the license.

**Unlicensed User-Based Server**

The most common reason for a server that has user-based licensing to be unlicensed is an expired product key or an expired maintenance contract.
Unlicensed Core-Based Server

A core-based server can become unlicensed for a variety of reasons. A common problem is that any Tableau Server node has more cores than the license allows. When the server is unlicensed you may not be able to start or administer the server.

Add Users to the Server

In a single-site environment, server administrators can add users on the Users page.

After you add a site to Tableau Server, it becomes a multi-site server with a Server Users page (all server users from every site appear here) and a Site Users page. Only server administrators can access the Server Users page, and both site administrators and server administrators can access the Site Users page.
The **Server Users** page is the only place where you can assign users to multiple sites, delete users from the server, and if the server is using local authentication, reset user passwords.

The following procedure describes how to add users to the server. There are two approaches you can take: One at a time (described below) or in batches using the **Import** command, which relies on a CSV file (described in Import Users and CSV Import File Guidelines).
To add a user to the server

1. In the site menu, click Manage All Sites, click Users, and then click Add Users.

2. If you are using local authentication, click New User. If you are using Active Directory, click Active Directory User.

Note: If you are running Tableau Server on Linux then all external directory communication is configured and managed with a LDAP identity store. In the context of user and group synchronization, Tableau Server configured with LDAP identity store is equivalent to Active Directory. Active Directory synchronization features in Tableau Server function seamlessly with properly configured LDAP directory solutions.

Enter a user name.

- **Local authentication**: If the server is using local authentication, using an email address for the user name is the best way to avoid user name collisions (for example, jsmith@example.com instead of jsmith).

- **Active Directory**: If you are adding a user that is from the same Active Directory domain that the server is running on, you can type the AD user name without the domain. The server domain will be assumed.
Before adding users, be sure to review User Management in Active Directory Deployments to understand how multiple domains, domain naming, NetBIOS, and Active Directory user name format influence Tableau user management.

**Note:** Do not enter the user's full name in this field; it can cause errors during the importing process.

3. If the server is using local authentication, provide the following:

- **Display Name**—Type a display name for the user (e.g., *John Smith*).
- **Password**—Type a password for the user.
- **Confirm password**—Retype the password.
- **Email**—This is optional and can be added at a later time in the user profile settings.
- **Selected users are Server Administrators:** Specify whether the user should be a server administrator.
- **Name (Site Membership) / Site Role:** If the user is not a server administrator, you can assign a user to zero or more sites, along with a site role for each site. You do not have to choose site membership and site role at this time. If you don't specify site membership and site role for a new server user, the user will be added as a Server User only, with a site role of Unlicensed. For details
4. Click **Create**.

## Sign in to Tableau Server Admin Pages

This topic explains how to sign in to the Tableau Server as an administrator.

If you are running Tableau Desktop and want to sign in to Tableau Server to publish or access content and data sources, see [Sign in to Tableau Server in Desktop](#).

Here’s how to sign in to the Tableau Server admin pages:
1. Open your browser and enter the server URL. Here are some examples of what the URL might look like:

http://localhost/ (if you're working directly on the server computer)

http://MarketingServer/ (if you know the server's name)

http://10.0.0.2/ (if you know the server's IP address)

If the server is not using port 80, you need to include the port number in the URL, as in these examples:

http://localhost:8000/

http://MarketingServer:8080/

http://10.0.0.2:8888/

... where 8000 or 8080 or 8888 is the port that you configured.

Tableau Server displays a page where you can enter a user name and password:
2. Enter the credentials for the server administrator that you created when you finished installing Tableau Server.

You're then taken to the main page of the Default site, and you're ready to create users, sites, and manage content.

Navigate Server Admin Pages

As a server administrator, you can access all of the menus and pages in Tableau Server for server and site management. If your server is configured for multiple sites, the site menu is available for navigation. Click Manage All Sites in the site menu to access server administration pages.

The server administrator pages include server-wide settings that you will use to configure, monitor, and maintain Tableau Server.

Server Administrator Pages

In a single-site deployment, all server and site menus are available to you in the main menu. To create a site, click Settings > Add a Site.

On a multi-site server, when a site is selected, you will see these menus:

To access server administration pages, click the site menu, and then select Manage All Sites:
On a multi-site server, these are the server administration menus. The site menu text changes to **All Sites** to let you know you are managing server-wide settings.

To return to the site administration menus, click **All Sites**, and the select the site you want to manage.
Server administrators can:

- Monitor server status and activity.
- Generate log files.
- Add sites and edit site settings. Only server administrators can add sites to the server.
- Add users to the server, and assign users to sites.
- Add and manage site groups.

To manage site-related settings, you must first navigate to a specific site. Within each site, you can:

- Manage content and assign permissions.
- Manage schedules for extract refreshes and subscriptions.
- Monitor site activity and record workbook performance metrics.
• Manage storage space limits for content published by users.

• Allow web authoring.

• Enable revision history.

• Allow site administrators to add and remove users.

• Allow users to subscribe to workbooks and views, and allow content owners to subscribe others to workbooks and views.

• Enable offline snapshots for favorites (iOS only).

Manage Sites

Sites Overview

The topics in this section describe the Tableau Server concept of a site and aspects of working with multiple sites. Topics include authentication type each site uses, and what to know about user licenses and administrator-level access to sites.

In this article

What is a site
Authentication and sign-in credentials
The Default site
Why or why not add sites
Administrator-level access to sites
Licensing and user limits
See also
What is a site

You might be used to using the term site to mean “a collection of connected computers,” or perhaps as the short form of “website.” In Tableau-speak, we use site to mean a collection of users, groups, and content (workbooks, data sources) that’s walled off from any other groups and content on the same instance of Tableau Server. Another way to say this is that Tableau Server supports multi-tenancy by allowing server administrators to create sites on the server for multiple sets of users and content.

All server content is published, accessed, and managed on a per-site basis. Each site has its own URL and its own set of users (although each server user can be added to multiple sites). Each site’s content (projects, workbooks, and data sources) is completely segregated from content on other sites.

Authentication and sign-in credentials

By default, all sites on a server use the same identity store type. You configure these settings when you install Tableau Server. For information, Configure identity store settings.

Users who have access to more than one site on the same Tableau Server instance use the same credentials for each site. For example, if Jane Smith has a user name of jsmith and a password of MyPassword on Site A, she uses those same credentials on Site B. When she signs in to Tableau Server, she’ll be able to choose which site she wants to access.

The Default site

Tableau Server installs with a site named Default. If you maintain a single-site environment on Tableau Server, this becomes the site you work with, and on which your users share their Tableau analysis. If you add sites, Default becomes one of the sites you can select when you sign in to Tableau Server. Default differs from sites that you add to the system in the following ways:

- It can never be deleted but, just like sites that you add, it can be renamed.
- It stores the samples and data connections that ship with Tableau Server.
• The URL used for Default does not specify a site. For example, the URL for a view named Profits on a site named Sales is http://localhost/#/site/sales/views/profits. The URL for this same view on the Default site would be http://localhost/#/views/profits.

Why or why not add sites

On Tableau Server, users, projects, groups, data sources, and workbooks are managed per site. You can add users to multiple sites.

Each environment and its needs is unique. However, as a baseline, Tableau Zen Masters and Product Managers tend to recommend using sites for true multi-tenancy needs. In other words, create a new site only when you need to manage a unique set of users and their content completely separately from all other Tableau users and content.

Examples for which it makes sense to use sites

• You are a consultant who manages Tableau analysis for multiple clients, and you want to create a site for each client, to ensure that data from one client is not exposed to another.

• You want to allow Guest user access to a small and contained area of the server.

Examples for which projects can work better than sites

• A content-development process in which data sources and reports evolve from sandbox to production phases.

Migrating users and content from one site to another is a laborious process. Although you might have good reasons to use sites for this and similar processes, by creating sites, you as the site administrator compound your ongoing maintenance burden. For each configuration update you make to one site (for example creating new projects and setting permissions), you usually would need to duplicate the same work on each additional site.

• You want to separate areas of the server by functional area.
Among a group of Tableau users, it’s common that some users need to access content in multiple areas. Using sites would encourage publishing the same data sources and reports to multiple sites. This leads to data source proliferation and can negatively impact server performance. Using projects is a simpler way to work with this scenario.

For additional ideas, see the following resources:

- Why use projects in the topic User Projects to Manage Content Access.
- Discussions about sites on the Tableau Community forums. Here’s a link to get you started.

Administrator-level access to sites

Tableau Server includes two administrator-level site roles: Server Administrator and Site Administrator.

The **Server Administrator** site role allows full access to Tableau Server. You can find more information about this role in Server Administrator Overview. Server administrators also create sites as needed. (Site administrators don’t have permissions to do this.)

A server administrator can assign the **Site Administrator** site role to users to delegate creating and maintaining a specific site’s user and content framework. The content framework enables Tableau users to share, manage, and connect to data sources and workbooks.

By default, the Site Administrator site role allows creating and managing the site’s users and groups, creating projects to organize content on the site, assigning permissions to allow users (groups) to access the content they need, scheduling extract refreshes, and a few other tasks.

For each site the server administrator can also limit site administrator access, so that site administrators can manage groups and content, but not add or remove users or set users’ site roles.
In some organizations, the same person might be both a server administrator and site administrator for one or more sites. Even so, the tasks performed by a site administrator and a server administrator are distinct.

**Licensing and user limits**

Server users can be added to multiple sites, and their site roles and permissions set on each site. A user who belongs to several sites does not need a license for each site. Each server user only needs one license.

Server administrators can use the *Limit number of users* setting (select Site <name> > Settings) to specify a user limit for the site. Only licensed users are counted; server administrators are excluded. For example, if a site has 90 licensed users, 20 unlicensed users, and one server administrator, the user count is 90.

**See also**

- For site administrator recommendations for how to set up users on a site, how to structure a site for publishers and other content users, how to give users permissions to share and manage their content, and so on, see the Manage Sites section.

- For information about how users can get their content to Tableau Server, see *Publish Data Sources and Workbooks* in the Tableau user help.

**Add or Update Sites**

Tableau Server comes with one site named Default. Server administrators can add sites to the server and modify any site’s settings.

**Add a site**

1. Do one of the following:

   - If you’re adding a site to the server for the first time, select **Settings > Add a Site**, and then click **Add a Site**.
• If you’ve added sites before, in the site menu, click **Manage All Sites**, and then click **New Site**.

• If you’re editing an existing site, select it on the Sites page, and then select **Edit Settings**. Or, in a single-site deployment, click **Settings**, and then click the **General** tab.
2. Enter a **Site name** and **Site ID** for the site. (If you are editing the Default site, you cannot change the **Site ID**.)

   ![Sites screenshot]

   **Name**

   | Site Name: Enter Site Name |
   | Site ID: Enter Site ID |
   | URL: http://10.32.139.28/# |

   **Note** The “#/site” portion of the URL (for example, http://localhost/#/site/sales) cannot be changed. In multi-site server environments, these segments appear in the URL for sites other than the **Default site**.

3. For **Storage**, select either **Server Limit** or **GB**, and enter the number of GB you want as a limit for storage space for published workbooks, extracts, and other data sources.
If you set a server limit and the site exceeds it, publishers will be prevented from uploading new content until the site is under the limit again. Server administrators can track where the site is relative to its limit using the **Max Storage** and **Storage Used** columns on the Sites page.

<table>
<thead>
<tr>
<th>Name</th>
<th>Users</th>
<th>Site Admins</th>
<th>Max Users</th>
<th>Storage Used</th>
<th>Max Storage</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>10</td>
<td>1</td>
<td>Server limit</td>
<td>12.9 MB</td>
<td>Server limit</td>
<td>Active</td>
</tr>
<tr>
<td>MyFirstSite</td>
<td>0</td>
<td>0</td>
<td>Server limit</td>
<td>0 B</td>
<td>Server limit</td>
<td>Active</td>
</tr>
<tr>
<td>MySecondsSite</td>
<td>0</td>
<td>0</td>
<td>Server limit</td>
<td>0 B</td>
<td>Server limit</td>
<td>Active</td>
</tr>
</tbody>
</table>

4. Select whether only server administrators can add and remove users and change their site roles, or whether you want to allow site administrators to manage users on this site.

**Managing Users**

Who is allowed to add and remove users.

- Only server administrators
- Server and site administrators

Limit the number of users to:

- Server limit
- [Number of users]
If you are allowing site administrators to manage users, specify how many users they can add to the site by selecting one of the following:

- **Server limit**: For a server with user-based licensing, the limit is the number of available server seat licenses. For a server with core-based licensing, there is no limit. For more information, see View Server Licenses and Handle an Unlicensed Server.

- **<number of> users**: Allows a site administrator to add users up to a limit you specify. See Work with Sites for information on licensing and user limits.

5. Select **Allow users to use web authoring** if you want to enable browser-based authoring for site content.

When web authoring is disabled, users cannot edit published workbooks from the server web environment. To update a workbook published to the server, a Tableau Desktop user must re-publish it. For more information, see Set a Site’s Web Authoring Access.

6. Enable comments so users can share a conversation about data views, including snapshot images to highlight discoveries. For more information, see Comment on Views in Tableau user Help.

7. Enable data-driven alerts to let users automatically receive emails when data reaches key thresholds. For more information, see Send Data-Driven Alerts in Tableau user Help.

8. Select **Allow users to subscribe to workbooks and views** to let site users subscribe to views and receive regular emails of them. Select **Allow content owners to subscribe other users to workbooks and views** to let administrators, project leaders, and content owners set up subscriptions for other users. These options are visible only if you have configured subscription settings.

9. Under **Email Settings**, you can enter a custom **From Address** for alerts and
subscriptions. While the email address you enter should use valid syntax (such as bizdev@myco.com or noreply@sales), Tableau Server does not require it to correspond to a real email account. (Some SMTP servers may require an actual address, however).

If you want a standard email footer to appear, select Custom footer, and enter the text you want to display above Tableau Server links in email messages.

The email footer will look similar to the following:

10. Enable offline favorites for Tableau Mobile to let mobile users access cached snapshots of views.

11. Select Record workbook performance metrics to permit your site users to collect metrics on how workbooks perform, such as how quickly they load

To initiate recording, users must add a parameter to the workbook’s URL. For more information, see Create a Performance Recording.

12. Click Create or Save.
Note: After you add your first site to Tableau Server, the site menu becomes available. When you select All Sites and click Users, the Server Users page appears, showing all users. When you select a specific site and click Users, the Site Users page appears. To add users to the server or specific sites, see Manage Users and Groups.

Export or Import a Site

You can provision a new Tableau Server site by importing (migrating) information from another site. You do this by exporting the existing site’s (the source site) information to a file. Then you complete steps to verify and import that information to the target site.

In this article

Before you export

Site migration options
What information is or isn't preserved in a site export
Prepare the source and target sites
Tips for importing to a target with fewer users or schedules than the source site

Steps to migrate a site

1. Export a site
2. Generate the import mapping files
3. Verify that site settings are mapped correctly
4. Import the correctly mapped files to the target site

Site migration options

You can migrate a site in any of these ways:
• To another site on the same Tableau Server instance.

• To a site on a separate Tableau Server instance.

• From Tableau Online to your Tableau Server deployment.

Many of the tips listed in Prepare the source and target sites apply to the Tableau Online-to-Tableau Server migration. However, become familiar with What information is or isn’t preserved in a site export.

**Note:** After you prepare your Tableau Online site for export, contact your Tableau representative. Tableau will complete the export steps, and your representative will send you the export files, which you can then import to your site on Tableau Server.

• From Tableau Server on Windows to Tableau Server on Linux or vice-versa (or cross-platform Tableau Online-to-Tableau Server site migration).

**What information is or isn’t preserved in a site export**

The export file you create preserves workbooks, projects, data sources, and users. This includes permissions set on content, user favorites lists, site quota, and other site settings.

In addition:

• When you export a site on Tableau Server to import to another Tableau Server site, subscription and extract refreshes schedules are preserved.

• When you export a Tableau Online site to import to Tableau Server, only the default schedules are preserved.

On Tableau Online, users can create their own subscription schedules. These schedules currently are not supported to map to Tableau Server.
Usage data, which appears in the site’s administrative views, is not preserved. For example, view and data source counts, user actions, and performance data.

In addition, OAuth access tokens embedded in data connections are reset. For those data sources, you will need to edit the connections and re-authenticate to the underlying data.

Prepare the source and target sites

Before you export a site, complete the following checklist to prepare both environments. Some of these instructions depend on whether both sites are on the same server instance or on separate ones.

• **Verify version compatibility**
To import a site successfully, the source and target sites must be in the same product version family. For example, 10.5 through 10.5.x. You cannot import a site that is outside the target site’s version family. For example, 10.4 to 10.5. For version information, see the Tableau Release Notes.

• **Identify node**
The `tsm sites` commands will use your local file store to hold the export and import data. If you are running a multinode Tableau cluster, then you must run the `tsm sites` commands on a Tableau Server that is running the Data Engine process. For information about the Data Engine process and the processes that require it, see Process Reference.

• **Delete stale content items**
Make sure the source site contains only what you want to import to the new site. Delete unused workbooks, data sources, or projects.

• **Remove obsolete users**
Confirm that all server users are licensed, and remove accounts that are no longer in use. You can’t remove users during the import process, so if the two sites are on the same server instance, all users you export from the source site are imported to the target site.

• **Create or identify the target site**
You must import a site file to a site that already exists on the target Tableau Server instance.
Because the import process removes anything from the target site that is not included in the import file, we recommend that you import to an empty site. For more information, see Add or Edit Sites.

Notes

- If the target site is not empty, workbooks and data sources with names that are identical on both target and source sites are replaced. This can be verified by the time stamp.

- If your source site has workbooks that use published data sources, the data connections in these workbooks continue to refer to the source site name. To maintain those connections without having to republish the workbooks, make sure the target and source site names match.

- Locate site IDs

The tsm command you use to export or import a site requires a parameter that takes the site ID. You can get the site ID from the URL when you’re signed in to the site from a web browser.

If only one site exists on the server, that site is named Default. When you’re signed in to the Default site, the browser URL looks something like this:

https://server-name/#/projects

In the URL, the absence of the /site parameter indicates that it’s the Default site. The site ID for the Default site is "Default" without the quotation marks.

On a multi-site Tableau Server deployment, the browser URL includes #/site/ followed by the site ID. The following URL would appear if you navigate to the Views page on a site whose site ID is finance:

https://localhost/#/site/finance/views
• **Check user authentication**
  You can export from and import to sites that do not use the same user authentication method, but you will need to modify the mapping files used for the import. This step is built into the import process and described in 3. Verify that site settings are mapped correctly.

• **Create users on the target server if necessary**
  The site import process assigns users to the target site. If the source site is on Tableau Online or on a Tableau Server instance other than the target site, you must create users on the target server before you can perform the import. If the two sites are on the same Tableau Server instance, the target site has access to the existing users, and you can skip this step.

  User names are stored differently in Tableau Online, so when you export a Tableau Online site, you must edit the user-specific mapping file as part of the verification steps.

• **Configure the target server to deliver subscriptions**
  Subscriptions are imported, but you must configure the server to deliver them.

  For more information, see Set Up a Server for Subscriptions.

• **Check schedules**
  The **Schedules** page lists the existing schedules for extract refreshes and subscriptions.

<table>
<thead>
<tr>
<th>Name</th>
<th>Frequency</th>
<th>Task type</th>
<th>Tasks</th>
<th>Execution</th>
<th>Next run at</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afternoon-daily</td>
<td>Daily</td>
<td>Subscription</td>
<td></td>
<td>Parallel</td>
<td>Aug 4, 2016, 4:00 PM</td>
</tr>
<tr>
<td>End of the month</td>
<td>Monthly</td>
<td>Extract Refresh</td>
<td>0</td>
<td>Parallel</td>
<td>Aug 31, 2016, 11:00 PM</td>
</tr>
<tr>
<td>Monday morning</td>
<td>Weekly</td>
<td>Subscription</td>
<td></td>
<td>Parallel</td>
<td>Aug 8, 2016, 7:00 AM</td>
</tr>
<tr>
<td>Nightly</td>
<td>Daily</td>
<td>Extract Refresh</td>
<td>0</td>
<td>Parallel</td>
<td>Aug 5, 2016, 12:00 AM</td>
</tr>
<tr>
<td>Weekday mornings</td>
<td>Weekly</td>
<td>Subscription</td>
<td></td>
<td>Parallel</td>
<td>Aug 5, 2016, 9:00 AM</td>
</tr>
</tbody>
</table>
For migrations from one Tableau Server site to another, refreshes and subscriptions assigned to default schedules on the source site are mapped to the same schedules on the target site.

If the source site has schedules that do not exist on the target site, and the target site is on another Tableau Server instance, you must create the schedules on the target site that you want the source schedules to map to. You can edit the mapping files to make sure this is done as you expect.

For migrations from Tableau Online to Tableau Server, custom subscription schedules that users create are not mapped. Before you contact Tableau to request your site export, remove these custom schedules.

To remove them, go to Tasks > Subscriptions, and select the check boxes for the custom subscriptions. Then, on the Actions menu, select Unsubscribe.

Tips for importing to a target with fewer users or schedules than the source site

When a target site has fewer users or schedules than the source site, many-to-one importing is not supported. How you can address this depends on whether the source and target sites are on the same Tableau Server instance.
You can take any of the following approaches that apply to your situation:

- Remove extra users or schedules from the source site before you export.
  
  This is the preferred option if the two sites are on the same server instance.

- Add missing users or schedules to the target site before beginning the import.
  
  This is required if the target site is on another server instance.

- Add the missing users or schedules to the target site in the middle of the import process and manually update the mapping files.
  
  This is an option only if the sites are on the same instance.

- Manually map the users or schedules to different users and schedules in the target site during the import process.
  
  This is required if a user name differs between servers—for example, the exported user named adavis@company.com is defined on the target site as davisa.

1. Export a site

On the Tableau Server machine, type the following command:

```
tsm sites export --site-id <source-siteID> --file <filename>
```

Tableau Server must be running when you use the `export` command. During the export process, Tableau Server locks the site you’re exporting.

For example, to export a site with site ID `weather-data` to the file `export-file.zip`, type the following:

```
tsm sites export --site-id weather-data --file export-file
```
By default, Tableau Server saves <export-file>.zip to /var/opt/tableau/tableau_server/data/tabsvc/files/siteexports. For more information, see tsm File Paths.

2. Generate the import mapping files

To generate import files for the target site, you need the .zip file you created when you completed the steps in 1. Export a site.

1. On the Tableau Server machine, copy the exported .zip file to the directory Tableau Server expects to find the files for importing. For example:

   /var/opt/tableau/tableau_server-
   /data/tabsvc/files/siteimports

2. Run the following command (Tableau Server must be running):

   tsm sites import --site-id <target-siteID> --file <export-file.zip>

   This command generates a set of .csv files that show how source site settings will map to the target site. In the steps described in the next section of this article, you confirm these mappings and adjust them where needed.

   By default these .csv files are generated to a mappings directory created under siteimports. For example:

   /var/opt/tableau/tableau_server-
   /data/tabsvc/files/siteimports/working/import_<id>_<date-time>/mappings

   For more information, see tsm File Paths.

3. Verify that site settings are mapped correctly

   The .csv files you generated in the previous section describe how the source site’s resources will be assigned to the target site when the import is complete. Items in the files that Tableau
Server was unable to map, and that you need to edit, are indicated by a series of question marks (???). Before you can complete the import process, you must replace the question marks with valid assignments on the target site.

**Important:** Some requirements apply to mapping users, schedules, and published content resources, particularly when the source and target sites are on separate Tableau Server instances. For more information, see Prepare the source and target sites earlier in this article.

To verify mapping files

1. Navigate to the directory that contains the .csv map files generated by the `tsm sites import` command. By default:

```
/var/opt/tableau/tableau_server-
data/tabsvc/files/siteimports/working/imports_<id>_<date-time>/mappings
```

2. Use your preferred text editor to open one of the .csv files in the `mappings` directory, and do the following.

   a. Confirm that the mappings are correct.

   b. If an entry shows a series of question marks (???), replace them with a valid value.

      For descriptions of the settings in each of these files, use the tables in Mapping file content reference later in this article.

   c. Save the changes and preserve the CSV file formatting.

Repeat this process for the remaining .csv files.
4. Import the correctly mapped files to the target site

After you verify the site mappings in the .csv files, you can import the settings to the new site to complete the migration process.

1. Run the following command:

   `tsm sites import-verified --import-job-dir <import-id-directory> --site-id <target-siteID>`

   For example:

   `tsm sites import-verified --import-job-dir /var/-opt/tableau/tableau_server-/data/tabsvc/files/siteimports/working/import_ff00_20180102022014457 --site-id new-site`

2. When the success message appears, sign in to the new site and confirm that everything was imported as you expected.

   **Note:** The `tsm sites import` and `tsm sites export` commands can leave a site in a locked state if an error occurs. To unlock a site, use the `tsm sites unlock` command.

**Mapping file content reference**

The following tables list the columns in each of the mapping files created when you run the `tsm site import` command.

CSV file name: mappingsDomainMapperForGroups

<table>
<thead>
<tr>
<th>Column title</th>
<th>Can it be edited?</th>
<th>Description</th>
</tr>
</thead>
</table>

- 649 -
<table>
<thead>
<tr>
<th>source_name</th>
<th>No</th>
<th>A user group name on the source site.</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_domain_name</td>
<td>No</td>
<td>The user authentication type on the source site: either <strong>local</strong> (for local authentication) or a domain name (for Active Directory).</td>
</tr>
<tr>
<td>target_domain_name</td>
<td>Yes*</td>
<td>The user authentication type on the target site: either <strong>local</strong> for local authentication, or a domain name (such as example.com or example.lan) for Active Directory.</td>
</tr>
</tbody>
</table>

*For the **All Users** group, keep the **target_domain_name** value set to **local**, even if your target server is configured for Active Directory user authentication. The **All Users** group is a special default user group that must exist on every Tableau Server.

CSV file name: mappingsScheduleMapper

<table>
<thead>
<tr>
<th>Column title</th>
<th>Can it be edited?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_name</td>
<td>No</td>
<td>The names of custom and default extract or subscription schedules on the source site.</td>
</tr>
<tr>
<td>source_scheduled_action_type</td>
<td>No</td>
<td>The type of schedule, either <strong>Refresh Extract</strong>, for extract refreshes, or <strong>Subscriptions</strong>, for subscription deliveries on the source site.</td>
</tr>
<tr>
<td>target_name</td>
<td>Yes</td>
<td>The names of custom schedules on the target site. You can edit this value. For example, if the schedule is named <strong>Friday</strong></td>
</tr>
</tbody>
</table>
Update on the source site you can rename it **Friday Refresh** on the target site.

| target_scheduled_action_type | No* | The type of schedule, either **Refresh Extract**, for extract refreshes, or **Subscriptions**, for subscription deliveries on the target site. |

*In the rare case that you see question marks (???) in this column, replace them with either **Refresh Extract** or **Subscriptions**, to match the entry you see under **source_scheduled_action_type**.

CSV file name: mappingsSiteMapper

<table>
<thead>
<tr>
<th>Column title</th>
<th>Can it be edited?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_url_namespace</td>
<td>No</td>
<td>The site ID of the source site.</td>
</tr>
<tr>
<td>target_url_namespace</td>
<td>No</td>
<td>The site ID of the target site.</td>
</tr>
</tbody>
</table>

CSV file name: mappingsSystemUserNameMapper

<table>
<thead>
<tr>
<th>Column title</th>
<th>Can it be edited?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_name</td>
<td>No</td>
<td>The user name attribute of a user on the source site.</td>
</tr>
<tr>
<td>source_domain_name</td>
<td>No</td>
<td>The user authentication type on the source site: <strong>local</strong>, for local authentication, a domain name (such as example.com) for Active Directory, or <strong>external</strong> (for a Tableau</td>
</tr>
</tbody>
</table>
| **target_name** | **Yes** | The user name attribute for users who will be assigned to the target site upon import.

Confirm that all the user names in the list exist on the target server, and replace question marks (???) with user names that exist on the target server.

You cannot create user names by adding rows to the CSV file. Similarly, you cannot remove user names by deleting rows.

You can edit a user name in the **target_name** column to be different from its source user name, as long as the user already exists on the target server with that name.

For example, a user can have a **source_name** value of agarcia@company.com and a **target_name** value of ashleygarcia@company.com.

You can map a user on the source site to only one user name on the target site. |
| **target_domain_name** | **Yes** | The user authentication type on the target site: either **local**, for Local Authentication, or a domain name (such as example.com) for Active Directory. |

CSV file name: MappingsScheduleRecurrenceMapperWithAutoCreation

This file does not require updates.
Delete Sites

Server administrators can delete sites that have been added to Tableau Server. Deleting a site also removes workbooks and data sources that were published to the site, as well as users. If a user belongs to additional sites, they will not be removed. To permanently delete a user, go to the Server Users page.

**Note:** The Default site cannot be deleted.

1. On the site menu, click **Manage All Sites**, and then click **Sites**.

![Managing Sites](image)

2. Select the site you want to remove, and click **Delete**.

3. Click **Delete** in the confirmation dialog box that appears.

Site Availability

A site can become suspended or locked due to a site import failure, or because a server administrator chooses to suspend the site for a period of time.
When a site is suspended, only the server administrator can activate the site to make it available again.

To activate or suspend a site

1. In the site menu, click Manage All Sites, and then click Sites.

2. Select the site, and then select Actions > Activate or Suspend.

Enable Users to Save Revision History

Revision history enables your users to keep copies of workbooks and data sources (content resources) that change. Each time someone saves an update to a content resource, Tableau Server creates a new version and stores it with older versions.

Server administrators can enable revision history on a per-site basis, for all of its workbooks and data sources. You can determine the number of revisions saved in history. Revision history is enabled by default, with a limit of 25 revisions for each content resource.
Authorized users can take actions on earlier versions of their content. By enabling revision history, you give everyone the confidence to experiment with the content, knowing that older versions are always available.

**Note:** This topic applies to Tableau Server. Workbook revision history (and not data source revision history) is enabled on all Tableau Online sites. Users can save up to 10 revisions.

**Required permissions**

To access revision history, a user must have a site role of *Publisher*, plus the following permissions:

- Project: **View** and **Save**
- Workbooks in the project: **View**, **Save**, and **Download Workbook/Save As**
- Data sources in the project: **View**, **Save**, and **Download Data Source**

**Enable revision history and set number of revisions allowed**

1. Sign in to a site as a Server Administrator, and click **Settings**.
2. Under **Revision History**, select **Save a history of revisions**, and enter the maximum number of revisions you want to allow for each content resource.
3. Click **Save**.

When you lower the number of revisions, the most recent revisions are saved. For example, if you set the limit to 15, the 15 most recent versions of the workbook or data source are saved.

**Note:** A content resource’s revision history list might not show the changes to the limit until a background cleanup process runs on the server.
Clear all revisions

Server administrators can delete all previous revisions of published workbooks and data sources from a site. The most recent version of each published workbook and data source is always retained.

1. Sign in to a site as a Server Administrator, and click **Settings**.

2. Under **Revision History**, click **Clear Revision History**.

3. Click **Save**.

Security for previewing and restoring workbooks

When users select **Restore** or **Preview** for workbook revisions, user passwords are exchanged between the user’s browser and the server. Tableau Server encrypts these passwords using public/private key encryption. To ensure these public keys are provided by Tableau Server, you must configure the server to use SSL (https).

See also

*Potential revision history issues* in the User/Analyst section of the online help.

Customize a Site

You can customize how a site on Tableau Server looks to personalize it for your company or group. You can perform these customizations:

- Change the server name that appears in the browser tab, tooltips, and messages.
- Change the logos that appear in the web environment.
- Set the language used for the web environment and the locale used for views.
- Install custom fonts on Tableau Server and client computers that connect to Tableau Server.

Administrators and project leaders can also add images for projects in thumbnail view.
Language and Locale for Tableau Server

Tableau Server is localized into several languages. Server language and locale settings impact how this affects users. The **Language** setting controls user interface (UI) items such as menus and messages. The **Locale** setting controls items in views such as number formatting and currency.

Administrators can configure language and locale on a server-wide basis and individual users can configure their own settings (search for "Your Account Settings" in the Tableau Server Help). If a user configures their own language and locale, their settings override the server settings.

Default Settings

The default language for Tableau Server is determined during Setup. If the host computer is configured for a language Tableau Server supports, Tableau Server installs with that language as its default. If computer is configured for a language that is not supported, Tableau Server installs with English as its default language.

How Language and Locale are Determined

Another influence on which language and locale display when a user clicks a view is the user’s web browser. If a server user has not specified a **Language** setting on their User Account page, and their web browser is set to a language that Tableau Server supports, the browser’s language will be used—even if Tableau Server itself is set to a different language.

Here’s an example: Assume that Tableau Server has a system-wide setting of English as the **Language** for all users. Server user Claude does not have a language specified on his Tableau Server User Account page. Claude’s browser uses German (Germany) for its language/locale.

When Claude signs in to Tableau Server, the server UI displays in German and when he clicks a view, the view uses the Germany locale for numbers and currency. If Claude had set his user account **Language** and **Locale** to French (France), the UI and view would have
been displayed in French. His user account setting supersedes those of his web browser, and both of those have precedence over the Tableau Server system-wide setting.

Another setting to be aware of is the **Locale** setting in Tableau Desktop (**File > Workbook Locale**). This setting determines the locale of the data in the view, such as which currency is listed or how numbers are formatted. By default, **Locale** in Tableau Desktop is set to **Automatic**. However, an author can override that by selecting a specific locale. Using the above example, if the author of View A set **Locale** to **Greek (Greece)**, certain aspects of the data in View A would display using the Greek (Greece) locale.

Tableau Server uses these settings, in this order of precedence, to determine language and locale:

1. Workbook locale (set in Tableau Desktop)
2. Tableau Server User Account language/locale settings
3. Web browser language/locale
4. Tableau Server Maintenance page language/locale settings
5. Host computer’s language/locale settings

**Use Custom Fonts in Tableau Server**

You can use custom fonts with Tableau Server. When you do this the safest way to guarantee that users have the experience you intend is to keep the following in mind:

- The fonts need to be installed on the computer where Tableau Server is running.

- The fonts need to be installed on any client computers that will connect to Tableau Server. You need to have the fonts installed locally in order for your browser to properly display them.

- As a best practice, use "web safe" fonts that are installed by default on all major browsers. This increases the likelihood that the fonts will display properly on client
machines.

- Different browsers render the same fonts differently, so even when a client browser has the custom font installed, it may look different when viewed in different browsers. This can be especially noticeable with comments or titles where specific spacing is used for an intentional effect.

**Note:** For more information about installing fonts on Linux, refer to your Linux distribution's documentation and support.

### Extract Refresh Schedules

Tableau Desktop authors and data stewards can create and publish *extracts*. Extracts are copies or subsets of the original data. Because extracts are imported into the data engine, workbooks that connect to extracts generally perform faster than those that connect to live data. Extracts can also increase functionality.

### Before refreshing extracts

When an extract refresh is performed on extracts created in Tableau 10.4 and earlier (that is, a .tde extract), the extract is upgraded to .hyper extract automatically. While there are many benefits of upgrading to a .hyper extract, you will be unable to open the extract with previous versions of Tableau Desktop. For more information, see Extract Upgrade to .hyper Format.

### Setting up refresh schedules

As a server administrator, you can enable scheduling for extract refresh tasks, and then create, change, and reassign schedules. General scheduling options you change on the server are available as part of the publishing process when a Tableau Desktop user publishes an extract.

Schedules that you create have the following options:
Priority

The priority determines the order in which refresh tasks are run, where 0 is the highest priority and 100 is the lowest priority. The priority is set to 50 by default.

Execution mode

The execution mode indicates to the Tableau Server backgrounder processes whether to run refreshes in parallel or serially. Schedules that run in parallel use all available backgrounder processes, even if the schedule contains only one refresh task. Serial schedules run on only one backgrounder process.

By default, the execution mode is set to parallel, so that refresh tasks finish as quickly as possible. You might want to set the execution mode to serial (and set a lower priority) if you have a very large schedule that prevents other schedules from running.

Frequency

You can set the frequency to hourly, daily, weekly, or monthly.

For information, see Create or Modify a Schedule.

Refreshing extracts manually

In the Tableau Server web environment, both server and site administrators can run extract refreshes on-demand on the Schedules page:

- Select the schedule and click Actions > Run Now.

You can also refresh extracts from the command line using the tabcmd refreshextracts command. For more information, see tabcmd Commands.

Refreshing extracts from Tableau Desktop

Tableau Desktop users can refresh extracts they publish and own. They can do this the following ways:
At publish time: When an author publishes a workbook or data source that uses an extract, that author can add it to server refresh schedule. The refresh can be a full or an incremental refresh.

Incremental refreshes reference a column in the extract that has a data type of date, date/time, or integer; such as a timestamp. Tableau uses this column to identify new rows that need to be added to the extract. For more information, see Refreshing Extracts and Schedule Extract Refreshes as You Publish a Workbook in the Tableau Help.

User interface: In Tableau Desktop, you can use the Refresh from Source, Add Data From File, and Add Data From Data Source commands to upload an addition to or refresh an extract on Tableau Server. A user might want to do this if Tableau Server doesn’t have sufficient credentials to access the underlying data. For more information, see Updating Extracts on Tableau Server in the Tableau Help.

Data Extract command line utility: The Data Extract command line utility installs with Tableau Desktop. You can use it to append to or refresh a published extract. For more information, see Tableau Data Extract Command Line Utility in the Tableau Help.

See also

Enable Extract Refresh Scheduling and Failure Notification

Automate Refresh Tasks

Enable Extract Refresh Scheduling and Failure Notification

Before your publishers can schedule extract refreshes, you must enable scheduling on the server.

While you’re enabling scheduling, you can decide whether also to enable sending email to owners of data sources or workbooks that are refreshed when those extract refreshes do not complete successfully. You can read more about these emails below. When you enable
refresh failure notification, the owners of the content that has scheduled refreshes can opt out individually by changing their account settings.

1. Sign in as a server administrator, and select **Settings**.

2. On the **General** page, do the following:

   - Under **Refresh Failure Notifications**, select **Send email to data source and workbook owners when scheduled refreshes fail**.

     To clarify, if a scheduled refresh for a particular data source fails, the email goes only to the owner of that data source, not to owners of workbooks that connect to that data source.

   - Under **Embedded Credentials**, select both options to let publishers embed credentials and schedule extract refreshes. (Automatic refresh schedules require embedded credentials so Tableau Server can directly access data.)

     **Note:** On a multi-site server, failure notifications are a site setting, and embedded credentials are a server setting.

Managing schedules from the server

In your organization it might be more appropriate to manage embedded credentials and refresh schedules centrally from the server. If you do that, you might clear the check boxes in the **Embedded Credentials** section described in the steps above, so that Tableau Desktop publishers do not see schedule options during publishing.

Managing schedules centrally enables you to distribute extract refresh and subscription tasks, so you can run them when most people are offline. It also enables you to oversee which credentials are embedded in connections.

For more guidelines for managing schedules and refreshes on the server, see **Provide access to data sources** and **Keep data fresh** in *Everybody’s Install Guide*.
How refresh failure emails work

The email notification for a failed extract refresh lists the extract name and location on the server, gives the time of last successful refresh, the number of consecutive times the refresh has failed, and suggests the reason for the failure and possible solution.

After five consecutive failures, the refresh schedule is suspended until you or the data owner takes an action to address the cause of the failure, such as updating database credentials or a path to the original data file.

How the last successful refresh date is determined

The last successful refresh date and time are shown when that last refresh occurred within a number of days. By default it is 14 days, and this value is set in $wgserver.alerts.observed_days$. If the number of days since the last successful refresh exceeds the number specified in this setting, the message in the email shows “not in the last $N$ days.”

Create or Modify a Schedule

The Schedules page shows a list of schedules, including their name, type, what they’re for (scope), number of tasks, behavior (concurrent or serial processing), and when they are scheduled to run.

To create a new schedule

1. In a site, click **Schedules**.

2. Click **New Schedule**.
3. Specify a descriptive **Name** for the schedule. For example, End of week.

4. Select a **Task type** the schedule will handle—either refreshing extracts or delivering subscriptions.

5. Optionally you can define a **Default Priority** from 1 to 100, where 1 is the highest priority. This is the priority that will be assigned to the tasks by default. If two tasks are pending in the queue, the one with the higher priority runs first.

6. Choose whether a schedule will run in parallel or serially. Schedules that run in parallel run on all available backgrounder processes so that they can complete faster.
7. Finish defining the schedule. You can define an hourly, daily, weekly, or monthly schedule.

![New Schedule](image)

8. Click Create.

**To modify an existing schedule**

1. Navigate to the Schedules page.

2. Select an existing schedule, click the Actions drop-down arrow, and then select **Edit Settings**.
3. Finish editing the schedule, and click **Save**.

See also

Manage Refresh Tasks

Extract Refresh Schedules

**How Scheduled Server Tasks are Prioritized**

When processing scheduled extract refreshes and subscriptions, Tableau Server prioritizes background tasks in this order:

1. Any task already in process is completed first.

2. Any task that you initiate manually using **Run now** starts when the next backgrounder process becomes available.

3. Tasks with the highest priority (the lowest number) start next, independent of how long they have been in the queue.
For example, a task with a priority of 20 will run before a task with a priority of 50, even if the second task has been waiting longer. To change task priority, see Create or Modify a Schedule.

4. Tasks with the same priority are executed in the order they were added to the queue. The first task added to the queue starts first; then the second task starts.

5. When multiple tasks with the same priority are scheduled to run at the same time, they start in the order they were created or enabled. There is no distinction between extract refreshes, subscriptions, and data-driven alerts.

The following limitations also impact scheduled tasks:

- The number of concurrent tasks is limited to the number of backgrounder processes you have configured for Tableau Server.

- Separate refreshes for the same extract cannot run at the same time.

- Tasks associated with a schedule that is set to run serially run one at a time.

**Configure Workbook Performance after a Scheduled Refresh**

To improve the load times for workbooks, Tableau Server caches the results of queries included in workbooks. For most workbooks, query results are computed and cached when they are first viewed by a user on Tableau Server. However, for workbooks that connect to data extracts, Tableau Server can recompute query results when the corresponding extract refresh tasks run. This reduces the load time for these workbooks when they are first viewed, so this option is turned on by default for workbooks that have been viewed recently.

**Determine the performance impact**

Although this option reduces the initial load time for workbooks, recomputing query results also increases the load on Tableau Server. If your Tableau Server installation is already per-
formance-constrained, you might want to turn this option off or lower the threshold for workbook caching.

Here are some possible reasons why you might want to turn this option off or lower the threshold:

- The Background Tasks for Non Extracts administrative view displays many long-running jobs in the **Warming up external query cache on data change** category.
- The Background Task Delay administrative view displays long delays.
- CPU and memory consumption for the backgrounder processes is consistently high.

However, note that this is only one of the options that impacts the performance of background tasks. For more information about performance, see Performance.

**Turn off workbook caching for the server**

To decrease the load on Tableau Server, you can turn off workbook caching after a scheduled refresh at the server-level. If you turn this option off, Tableau Server caches query results for workbooks the first time the workbooks are viewed.

Use the following `tsm` configuration set option to turn off workbook caching after a scheduled refresh:

```
backgrounder.externalquerycachewarmup.enabled
```

For more information on how to use and apply `tsm` set options, see `tsm` configuration set Options.

**Turn off workbook caching for a site**

You can also turn off workbook caching after a scheduled refresh for an individual site. For example, you might do this if there is one site in particular that contains many slow workbooks which increase load on the server.
1. Select the site for which you want to turn off workbook caching in the sites drop-down.

2. Click Settings.

3. In the **Workbook Performance after a Scheduled Refresh** section, clear the check box.

**Note:** Although this option is available in the settings for an individual site, you must have server administrator permissions to view it.

### Configure the workbook caching threshold

Tableau Server only recomputes query results for workbooks that both have scheduled refresh tasks and have been viewed recently.

You can increase or decrease the number of workbooks that are cached after a scheduled refresh with the following tsm configuration set option:

```
backgrounder.externalquerycachewarmup.view_threshold
```

By default, the threshold is set to 2.0. The threshold is equal to the number of views that a workbook has received in the past seven days divided by the number of refreshes scheduled in the next seven days. (If a workbook has not been viewed in the past seven days, it is unlikely that it will be viewed soon, so Tableau Server does not spend resources recomputing queries for the workbook.)

### Ensure Access to Subscriptions and Data-Driven Alerts

To ensure that users see the Subscribe and Alert buttons in the Tableau Server toolbar and can receive related emails, do the following:

- **Ensure that users have an email address in Tableau Server:** Users can update their email address on their account settings page.

- **Embed database credentials or don't require them:** To email data in a view,
Tableau Server needs to access the data without user involvement. This can be accomplished by using a workbook with embedded database credentials, a Tableau Server data source, or data that doesn’t require credentials (such as a file that’s included with the workbook at publish time).

- **Ensure that users can access needed workbooks and views:** Access to content on the server is controlled by the View permission. To receive images of content in email messages, users also need the Download Image/PDF permission. For more information, see View or Edit Permission Rules and User Permissions.

- **Avoid trusted authentication for embedded views:** If you use restricted tickets (the default) to render an embedded view, the Subscribe and Alert buttons don’t appear.

**Set Up a Server for Subscriptions**

When users subscribe to a workbook or view, a snapshot of the view is email to them on scheduled basis, so they can see the latest updates without having to sign into Tableau Server. Administrators, project leaders, and content owners have the option to subscribe other users to workbooks and views. For more information, see Subscribe to Views.

**Note:** To create and receive data-driven alerts, users need access to related databases and views. See this list of requirements for details.


**In this article**

- Configure the server to send subscription emails
- Enable subscriptions in a site
Configure the server to send subscription emails

1. Configure SMTP Setup
2. Run the following commands:
   
   tsm configuration set –k subscriptions.enabled –v true

   tsm pending-changes apply

Enable subscriptions in a site

As an administrator, you decide which sites allow subscriptions.

1. At the top the browser window, click Settings.
2. Select **Allow users to subscribe to workbooks and views**.
3. Optionally, select **Allow content owners to subscribe other users to workbooks and sheets**, or enter an email "From" address and message footer.
   
   A site's "From" address and message footer are also used in emails for data-driven alerts.
4. Click Save.

To specify the subscription schedules available to users, see Create or Modify a Schedule.

Test subscriptions in a site

1. **Subscribe to a view**.
2. At the top the browser window, click **Schedules**.
3. Select the schedule you chose for the subscription, and then click **Actions > Run Now**.

Manage all user subscriptions

1. At the top the browser window, click **Tasks**, and then click **Subscriptions**.

   All user subscriptions for the current site appear, including information like subscriber name, view name, and delivery schedule.

2. Select any subscription you want to update. From the **Actions** menu, select **Change Schedule**, **Change Subject**, **Change Empty View Mode**, or **Unsubscribe**.

   (The empty-view option sends subscription emails only when data exists in a view. It's a good choice for high-priority alerts.)

Set Up a Server for Data-Driven Alerts

When data reaches important thresholds for your business, data-driven alerts automatically send email notifications to key people users specify. As a Tableau Server administrator, you set up data-driven alerts much like you do subscriptions. For information about how users create and manage these alerts, see **Send Data-Driven Alerts** in Tableau User Help.
**Note:** To create and receive data-driven alerts, users need access to related databases and views. See this list of requirements for details. If alerts are enabled for a site, any user on that site can create them.

**In this article**

Configure email for data-driven alerts  
Manage all data-driven alerts in a site  
Disable data-driven alerts for a site  
Control how often the server checks data-driven alerts  
Track the server’s alert-checking process  
Identify and fix failing alerts

**Configure email for data-driven alerts**

1. Complete the steps in Configure SMTP Setup so the server can send email.

2. While viewing a site, click **Settings** at the top of the browser window.

3. Under Email Settings, enter a site-specific "From" address or message footer.

   A site's "From" address and message footer are also used in emails for subscriptions.

4. Click **Save**.

**Manage all data-driven alerts in a site**

1. At the top the browser window, click **Tasks**, and then click **Alerts**.

2. Select any alerts you want to update.

3. From the **Actions** menu, do any of the following:
• Add or remove yourself as a recipient.
• Edit alerts to change data thresholds, delivery schedules, and the full list of recipients.
• Change alert ownership to different users, or delete alerts.

Disable data-driven alerts for a site

Data-driven alerts are supported for all sites by default, but administrators can disable them for specific ones.

1. While viewing a site, click **Settings** at the top of the browser window.
2. Under Data-Driven Alerts, uncheck **Let users create alerts and receive alert emails**.
3. Click **Save**.

**Tip**: To entirely disable data-driven alerts at the server level, change the tsm configuration set option, "features.AlertOnThresholdCondition", to false.

Control how often the server checks data-driven alerts

By default, Tableau Server checks every 60 minutes to confirm whether data conditions for alerts are true. If you notice performance impacts, you can customize this time interval with the tsm configuration set option, "dataAlerts.checkIntervalInMinutes".

Regardless of the dataAlerts.checkIntervalInMinute setting, the server also checks alerts whenever extracts in the related workbook are refreshed. To check an alert more frequently than the setting specifies, change the extract-refresh schedule.

Track the server's alert-checking process

In the Background Tasks for Non Extracts view, you can track the server’s alert-checking process by looking for these tasks:
- Find Data Alerts to Check

- Check If Data Alert Condition Is True

The "Find" task limits "Check" tasks to alerts that can currently send related emails. For example, if a user has chosen an email frequency of "Daily at most", after the alert condition becomes true, the server waits 24 hours before checking the alert again.

Each "Check" task uses one server backgrounder process, loading the related view to evaluate the alert condition. If all users see the same version of a view, it loads only once. But if users have applied filters to a view, or the data they see is limited by user-level security, the view loads once for each recipient.

Identify and fix failing alerts

As an administrator, you can use the "Check If Data Alert Condition is True" task to proactively identify failing alerts that users are unaware of. Hover over the red failure icon to display a tooltip with alert details. To determine the alert owner, look for the alert ID number in the data_alerts table of the Tableau Server Repository. (In the alert management area of a site, you can also look for the alert name following the number, but be aware that multiple different alerts may use the same name.)
Failing alerts are often caused by content changes on Tableau Server. Encourage users to recreate alerts if changes like the following occur:

- A workbook, view, or data field is removed or renamed.
- Database credentials embedded in workbooks expire. (Alerts require workbooks to use either embedded credentials, or none at all.)
- A data source becomes inaccessible.

**Tip:** To automatically get emailed when alerts fail, follow the steps in Collect Data with the Tableau Server Repository, and connect to the "background_jobs" table. From that table, create a custom view that includes the "Check If Data Alert Condition Is True" job name and its finish code. Then set up a data-driven alert to email you whenever a finish code equals 1 (failure).
Administrative Views

The Status page contains an embedded Tableau workbook with various administrative views. These views help you to monitor different types of server or site activity.

- Shows server and site activity for Tableau Server.
- Shows site activity for Tableau Online.

**Note:** You must install PostgreSQL drivers before you can see Administrative views. For more information, see Install Drivers on Linux.

Navigating to administrative views

To see administrative views, click **Status**. Site administrators can see administrative views for their site. Administrators of multiple sites can see views for the current site.

On a multi-site server, server administrators can see views for the entire server. Click the site menu, and then click **Manage All Sites** to access the server menus.

To see views for individual sites on a multi-site server, click the site menu, select the site name, and then click **Status**.
Pre-built Administrative Views

Administrative views are powerful monitoring tools that can help you optimize Tableau Server and better understand how your users are interacting with Tableau content. The administrative views listed to the right are included with Tableau Server. Click on the link for a view to learn more about how to interpret and act on the information the view provides.

Performance of Views

**Note:** This view is only available to server administrators. For information about how to navigate to administrative views, see Administrative Views.

The Performance of Views administrative view displays how long it takes for views to load and how many sessions are running at a time on the server.
You can compare spikes in the number of sessions with spikes in slow load times to identify the times of day when high user traffic is slowing down the server. You can also look at the individual views by load time to understand which views take the longest to load.

Some views might take a long time to load regardless of when they are viewed. You can identify which workbooks need to be optimized with the Stats for Load Times administrative view. Some simple ways to optimize workbooks includes the following:

- Display less information in each view.
- Break up views.
- Reduce the number of filters.
- Use data extracts.

Traffic to Views

The Traffic to Views view gives you the ability to see how much of your user traffic goes to views.

You can filter what information is displayed and the time frame it comes from by selecting the view, the workbook, and the time range. Server administrators can specify the site.
Two time lines at the top of the view show you how views are being used over a time range you specify (the default is the last 7 days):

- **What is the Total View Count by Day**—This shows total view count by day, based on the filters you set. Hover your mouse pointer over a point on the line to see the count of views. Select the point to update the other sections of the view based on your selection.

- **What is the Total View Count by Time**—This shows the view count by time of day. The filters and any selection impact this graph.

Two bar graphs at the bottom of the view show results that are filtered by the **Min View Count** filter at the top of the view. These show you the views that are most often accessed, and the users who most frequently access views Only those views and users with counts greater than or equal to the minimum view count value are displayed:

- **What Views are Seen the Most**—This is a list of the most visited views. Like the other sections of the view, the information is limited by filters and any selection you make.
• **Who Accesses Views Most Often**—This shows the users who most often access the views and is limited by filters and any selection you make.

**Traffic to Data Sources**

The Traffic to Data Sources view gives you the ability to see usage of data sources on your Tableau Server installation. This can help you determine which data sources are most heavily used and those that are less often used. You can filter the information you see by selecting the data source, the action taken on that data source, and the time range. Server administrators can specify the site.

![Graph showing data source usage](image)

A time line at the top of the view shows you how data sources are being used over a time range you specify (the default is the last 7 days):

• **What is the Total Data Source Usage by Day**—This shows total data source usage by day, based on the filters you set. Hover your mouse pointer over a point on the line to see the count. Select the point to update the other sections of the view based on your selection.

Two bar graphs at the bottom of the view show results that are filtered by the **Min Interactions** filter at the top of the view. These show you which data sources are most used, and who uses data sources most often. Only those data sources and users with interaction counts greater than or equal to the minimum interactions value are displayed:
• **What Data Sources are Used Most**—This is a list of the most used data sources. Like the other sections of the view, the information is limited by filters and any selection you make.

• **Who Uses Data Sources Most Often**—This shows the users who most often use the data sources. This is impacted by filters and any selection you make.

**Actions by All Users**

The Actions by All Users view gives you insight into how your Tableau Server installation is being used. You can filter the view by actions and by time range. Server administrators can filter by site. The Total Users count shows the number of users who have performed an action. This value is not affected by any filtering. The Active user count shows the number of active users who have performed one of the selected actions.

![Actions by All Users](image)

Up to three separate groups of time lines show you how users are using Tableau Server over a time range you specify (the default is the last 7 days). If no actions are selected for a particular group, that group does not display. Possible groups are:
- Access & Interactions—This shows you sign in (log on) activity, view access and data source use.
- Publish & Download—This shows publishing and downloading of workbooks and data sources.
- Subscriptions—This shows counts of subscription email sent for workbooks and views.

Use the legend at the bottom to view a subset of the displayed actions. Click a single action to highlight the line for the action, or Ctrl + Click on multiple actions to highlight more than one. To clear the selection and display all the selected actions, click on any action in the legend.

Actions by Specific User

The Actions by Specific User view gives you insight into how individual users are working in your Tableau Server installation. You can filter the view by user name, actions, and time range. Server administrators on multi-site installations can filter by site.
Up to three separate groups of time lines show you how a selected user is using Tableau Server over a time range you specify (the default is the last 7 days). If no actions are selected for a particular group, or if no actions were taken, that group does not display. Possible groups are:

- **Access & Interactions**—This shows you sign in (log on) activity, view access and data source use.
- **Publish & Download**—This shows publishing and downloading of workbooks and data sources.
- **Subscriptions**—This shows counts of subscription email sent for workbooks and views.

A bar graph at the bottom of the view shows which items the selected user is using.

Use the legend at the bottom to view a subset of the displayed actions. Click a single action to highlight the line for the action, or **Ctrl + Click** on multiple actions to highlight more than one. To clear the selection and display all the selected actions, click on any action in the legend.

**Actions by Recent Users**

The Actions by Recent Users view shows you which signed-in users have been active on Tableau Server over the past 24 hours. This can be useful if you need to perform some maintenance activity and want to know how many and which users this will affect, and what they are doing on Tableau Server.

The view **Active**, **Recently Active**, and **Idle** users that are currently signed in to Tableau Server. For this view, an active user is one who took an action in the last 5 minutes, a recently active user is one who last took an action within 30 minutes, and an idle user is one who last took an action more than 30 minutes ago. The actions are displayed in the lower section of the view.
Select a user to see only the actions that user performed recently. Hover over an action to see details of the action.

Background Tasks for Extracts

The Background Tasks for Extracts view displays extract-specific tasks that run on the server.
Understand this view

To better understand this pre-built administrative view, make note of the following:

- The table, What Extracts Ran on this Server, lists the extracts that ran in the time period specified in **Timeline**.
- You can click **Success** or **Error** to filter the table based on status.
- You can also click a specific task to update the How Much Time did Extracts Take graph for the selected task.
- The table, How Many Extracts Succeeded or Failed, updates for the status (success or failure) of the task, but the count of extracts that succeeded or failed does not change.

Tasks can have a status of successful or error:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🟢</td>
<td>Success—Server completed the task.</td>
</tr>
<tr>
<td>⚫</td>
<td>Error—Server was unable to complete the task.</td>
</tr>
</tbody>
</table>
For details on a task, hover over its icon:

Background Tasks for Non Extracts

The Background Tasks for Non Extracts view displays tasks that the server runs that are not related to refreshing extracts. For example, edited OAuth connections, subscription notifications, and so on.
A table lists the tasks that ran in the time range specified. Click **Success** or **Error** to filter the table based on status. Select a specific task in the **How Many Tasks Succeeded or Failed on this Site** table to update the **What Background Tasks Ran on this Site** graph for the selected task.

Tasks can have a status of successful or error.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✖️</td>
<td>Error—Server was unable to complete the task.</td>
</tr>
<tr>
<td>✨</td>
<td>Success—Server completed the task.</td>
</tr>
</tbody>
</table>

For details on a task, hover over its icon.

**Background Task Delay**

**Note:** This view is only available to server administrators. For information about how to navigate to administrative views, see Administrative Views.

The Background Task Delay view displays the delay for extract refresh tasks and for subscription tasks—that is, the amount of time between when they are scheduled to run and when they actually run. You can use the view to help you identify places you can improve server performance by distributing your task schedules and by optimizing tasks.
Here are possible reasons for the delays, and ways that you might reduce the delays:

- Many tasks are scheduled for the same time. In the example view, tasks that show long delays are clustered at the same time every day, which creates spikes in the wait time. Note that you can set the Timeline filter to a single day to view task delays by hour and identify the hours of the day which have many tasks scheduled at the same time. A solution to this issue can be to distribute the tasks to off-peak hours to reduce load on the server.

- Specific tasks take a long time to run and are preventing other tasks from running. For example, there might be an extract refresh job that is connecting to a slow data source or that is processing a large amount of data. Use the Background Tasks for Extracts administrative view to identify which extract refresh tasks are running slowly. You can then optimize the extract refresh task by filtering the data, aggregating the data, or creating multiple data sources for individual tables in a data source.
Other server processes are running at the same time and are consuming server resources and slowing down performance. Monitor the CPU and memory usage of server processes to see which processes are consuming the most resources and then adjust the configuration of processes on your server.

Stats for Load Times

The Stats for Load Times view shows you which views are the most expensive in terms of server performance. You can filter by view and time range. Server administrators can filter by site. You can also limit the view based on load time in seconds, using the sliding Load Time filter. Load times are for the server. Depending on your client browser and networking, actual load time may vary slightly.

The **Average Load Times** graph shows average load times for views based on the filters you set. Hover over a point to see details. Select a point on the line to update the rest of the view for the selection:
The **Exact Load Times** view shows exact time to load the listed views. A vertical line shows the average load time for each view. Select a mark to see details of a specific instance of the view loading:

**Stats for Space Usage**

The Stats for Space Usage view can help you identify which workbooks and data sources are taking up the most disk space on the server. Disk space usage is displayed by user, project, and by the size of the workbook or data source and is rounded down to the nearest number:
Use the **Min Size** filter to control which data sources and workbooks are displayed, based on the amount of space they take up.

Three bar graphs give you information about space usage on your Tableau Server:

- **What Users Use the Most Space**—This shows the users who own data sources and workbooks that are taking up the most space. Click a user name to filter the next two graphs for that user. Click the data source bar or the workbook bar for a user to filter the next two graphs for that type of object for that user. Click the selected user or bar to clear the selection.

- **What Projects Use the Most Space**—This shows the projects with the data sources and workbooks that are using the most space. If a user or object type is selected in the What Users Use the Most Space graph, this displays information specific to the selection.

- **What Workbooks and Data Sources Use the Most Space**—This shows the work-
books and data sources that are taking the most space. The bars are color-coded based on the length of time since the last refresh.

Move your cursor over any bar to display usage details:

<table>
<thead>
<tr>
<th>User</th>
<th>Site</th>
</tr>
</thead>
<tbody>
<tr>
<td>cs</td>
<td>Default</td>
</tr>
<tr>
<td>admin</td>
<td>Default</td>
</tr>
<tr>
<td>fa</td>
<td>Default</td>
</tr>
</tbody>
</table>

Click on a bar to select it and update the other areas of the view based on that selection.

Server Disk Space

**Note:** This view is only available to server administrators. For information about how to navigate to administrative views, see Administrative Views.

Use the Server Disk Space view to see how much disk space is in use on the computer or computers that run Tableau Server, where disk space refers only to the partition where Tableau Server is installed. You can also use this view to identify sudden changes in disk space usage.

For a distributed installation, the view displays information about each computer in the cluster.
The Server Disk Space view includes two graphs:

- **What is the most recent disk space usage?**—This graph shows disk space usage for the last 30 days both in gigabytes and as a percentage. Disk space refers only to the partition where Tableau Server is installed.

- **How has free disk space trended in the last month?**—This graph shows changes to disk space usage over the last month. Rest your pointer on a line to view the exact amount of free disk space for a point in time.

When Tableau Server is low on disk space, you can remove files to free space.

**Desktop License Usage**

**Note:** This view is only available to server administrators. For information about how to navigate to administrative views, see Administrative Views.
The Desktop License Usage view lets server administrators see usage data for Tableau Desktop licenses in your organization. This can help you manage licenses efficiently and determine if you need more or fewer licenses. This view can help you answer the following questions:

- Who is using a Tableau Desktop license in my enterprise?
- Have any licenses been shared or transferred?
- Is any license being used on a computer where it should not be?
- Does a specific user use their license?
- What types of licenses are being used in my enterprise?
- Do I need to convert any trial licenses?

**Note:** To get data about licenses, each copy of Tableau Desktop version 10.0 or later needs to be configured to send data to Tableau Server. This configuration can be done at installation time, using scripting or third-party software to install and configure Tableau, or after installation, by modifying the registry or property list file. For more information, see Configure Desktop License Reporting.

In order to view license data, Desktop License Reporting must be enabled on Tableau Server. See Enable and configure Desktop license reporting.

**Filters**

- **Product Keys.** Type a string to filter the dashboard to only those licenses that include the string anywhere in the product key. For example, to see only licenses that begin with TDTD, type TDTD and press Return to filter the view. Click the X after the string to reset the filter.

- **Action.** Use this filter to control what the dashboard displays, based on the action taken. Actions are Activate, Use, and Return (deactivate). If the Use action is not selected, nothing is displayed in the top bar graph.
• **Department.** Use this filter to control what departments the dashboard displays licenses for. The filter is populated based on the Department values specified when Tableau Desktop is registered.

• **Select time duration in days.** Use this slider to specify the time length in days that the dashboard displays information for. The default value is 183 days.

When you hover over the filter card in the first three filters, a drop-down icon appears. Click the icon to specify whether the view should include data that matches the filter (the default) or exclude data that matches the filter:

![Include values](image)

### Who has used Tableau in the last <nn> days?

This area of the dashboard shows a bar graph of three types of Tableau Desktop licenses (Perpetual, Trial, and Term) and the number of users who have used each license type during the specified time period. Hover over a license type segment to see an explanation of the license type. Click a segment to filter the rest of the dashboard for only that license type. This action filters both the tables that show licenses that have been used and those that have not been. For example, to see a list of term licenses that have been used during the time period, click the Term bar. The "used" and "not been used" lists are filtered to just show term licenses.

A table of detailed information shows under the bar graph. For each row in the table, action icons display on the right, above a timeline that shows you when the action last took place.

To see a list of the underlying data in a format that allows you to select and copy values like email or product key, click a row in the list of licenses and click the View Data icon:

![View Data](image)
The data displays in summary form. Click **Full data** to see all the data. From this view you can select and copy individual values, or download the data as a text file.

**What licenses have not been used in the last <nn> days**

This area of the dashboard shows a list of licenses that have not been used during the specified time period. A timeline shows the last use date. Hovering over a last use mark gives you information including the registered user of the copy of Tableau.

**Desktop License Expiration**

**Note:** This view is only available to server administrators. For information about how to navigate to administrative views, see Administrative Views.

The Desktop License Expiration view gives server administrators information about which Tableau Desktop licenses in your organization have expired or need maintenance renewal. This can help you manage licenses efficiently. This view can help you answer the following questions:

- What trial or term licenses have expired?
- What perpetual licenses have expired maintenance?
- What perpetual licenses have maintenance renewals coming up?

**Note:** In order to get data about licenses, each copy of Tableau Desktop version 10.0 or later needs to be configured to send data to Tableau Server. This configuration can be done at installation time, using scripting or third-party software to install and configure Tableau. For more information, see [Configure Desktop License Reporting](#).

In order to view license data, Desktop License Reporting must be enabled on Tableau Server. See [Enable and configure Desktop license reporting](#).
Filters:

- **Product Keys**—Type a string to filter the dashboard to only those licenses that include the string. For example, to only see licenses that begin with TDTD, type TDTD and press return to filter the view. Click the "x" after the string to reset the filter.

- **Department**—Use this filter to control what department(s) the dashboard displays licenses for. The filter is populated based on the Department values used when registering copies of Tableau Desktop.

- **Time Duration**—Use this filter to control the length of time for which the dashboard displays information.

The view includes the following tables, which are affected by the filters you set at the top of the view:

- **What keys have expired maintenance**—This table shows the product keys for which maintenance has expired, with a vertical line indicating the point at which the six month window for renewing maintenance closes. If maintenance for a key is expired for more than six months you need to purchase a new key in order to qualify for support or upgrades.

- **What trial and term licenses have expired**—This shows the trial or term product keys that have expired.

- **What is the maintenance schedule for my keys**—This shows the keys and their maintenance status.

Create Custom Administrative Views

In addition to the pre-built administrative views available on the Maintenance page on the Server, you can use Tableau Desktop to query and build your own analyses of server activity. To do this, you can connect to and query views in the Tableau Server repository using one of two built-in users: the "tableau" or "readonly" user.
To connect to the Tableau Server repository, see Collect Data with the Tableau Server Repository.

- The **tableau** user—The tableau user has access to special views and a subset of tables in repository database. These views and tables are provided so that administrators can create custom administrative views. Tableau makes an effort to limit changes to these tables and views so that custom views built with them do not break.

- The **readonly** user—The readonly user has access to a large number of the repository tables, providing more data about server usage. Administrators can use these to create custom administrative views too, but many of the tables are intended primarily to support the functioning of Tableau Server and may be changed or removed without warning. This means that views created from these tables can break when the database structure is changed.

  **Note:** The readonly user is available in Tableau Server 8.2.5 and later.

For examples of custom administrative views, see the Tableau Community.

Before you can connect using one of the built-in users, you must enable access to the Tableau Server database. After doing this you can use Tableau Desktop to connect to and query the database as the tableau user or the readonly user.

The `tsm configuration set option auditing.enabled` controls whether Tableau Server collects historical user activity and other information in the repository. It is enabled by default. Be aware that collecting historical events impacts the size of Tableau Server’s backup file (.tsbak).

- All hist_tables are controlled by the `tsm configuration set option wgserv-er.audit_history_expiration_days`, which controls how many days of event history are kept in the repository and has default value of 183 days.

- The _background_tasks table is cleaned automatically and keeps data for the last 30
days.

- All other tables with names that begin with a "_" prefix contain current data.

Performance

Tableau Server Performance Overview

When you take the time to understand the performance of Tableau Server, you make it easier to serve your users by improving the efficiency of Tableau Server. Although every server environment is unique, and there are many variables that can impact performance, the general steps that you take to understand and act on performance data in Tableau Server are the same.

- **Alerts.** Configure email notifications for important server events. For example, you can receive notifications when server processes become unavailable and when the server is running out of disk space.

- **Monitoring.** Collect and analyze data about Tableau Server to understand how well the server is performing.

- **Tuning.** Make adjustments to tasks, process configurations, and more to improve the performance of Tableau Server.

- **Troubleshooting.** Identify bottlenecks in resources, workbooks, and more to improve the performance of Tableau Server.

General Performance Guidelines

Hardware and Software

**Add more cores and memory:** Regardless of whether you’re running Tableau Server on one computer or several, the general rule is that more CPU cores and more RAM will give
you better performance. Make sure you meet the Tableau Server recommended hardware and software requirements.

If you are running Tableau Server in a virtual environment, use your VM host's best practices for vCPU allocation in relation to the number of physical CPU cores on the VM host.

**Configuration**

**Schedule refreshes for off-peak hours:** Backup tasks tend to stall other background tasks until the backup is completed. Use the Background Tasks for Extracts administrative view to see your refresh and backup task schedules. Your refresh tasks should be scheduled for off-peak hours that don't overlap with your backup window.

**Consider changing two session memory settings:**

- **VizQL session timeout limit:** The default VizQL session timeout limit is 30 minutes. Even if a VizQL session is idle, it is still consuming memory and CPU cycles. If you can make do with a lower limit, use tsm configuration set Options to change the vizqlserver.session.expiry.timeout setting.

- **VizQL clear session:** By default, VizQL sessions are kept in memory even when a user navigates away from a view. This consumes a good deal of session memory. Instead, you can end sessions when users move away from a view by changing the value of the vizqlserver.clear_session_on_unload setting to true (the default is false).

**Assess your process configuration:** Tableau Server is divided into six different components called server processes. While their default configuration is designed to work for a broad range of scenarios, you can also reconfigure them to achieve different performance goals. Specifically, you can control on which computers the processes run and how many are run. See Performance Tuning Examples for general guidelines for one-, two-, and three-node deployments.

**Server Resource Manager (SRM)**

The Server Resource Manager (SRM) monitors the system resources each Tableau process is using as well as tracking the total usage of Tableau Server on the system. If either a
specific process or the product as a whole takes up too much system resources, SRM can notify the processes to free the resources or restart those processes.

The thresholds that determine when SRM will notify or restart a process are set in the SRM configuration options. The Tableau development team has set the default settings based on internal testing and don’t recommend you change these settings directly.

If you are seeing excessive system resource usage, we recommend that you contact Tableau Support to help determine if these configuration options need to be modified to solve the problem or issue that you are seeing.

Performance Monitoring Overview

When you monitor a server, you collect and analyze data that signals whether the server is performing badly or running into problems. For example, if you notice that your server is using 100% of its processing capacity for long periods of time, you know that there’s a problem.

The data that you need to collect and analyze can be broken down into the following broad categories:

- Resource usage data—how Tableau Server uses hardware resources like diskspace, memory, and processors.

- Session and load time data—how users interact with Tableau Server, including how long it takes for views to load and how many concurrent users there are.

- Background task data—how Tableau Server runs tasks that are not directly tied to a user action. For example, background tasks include extract refresh tasks, subscription tasks, and more.

Some of this data, including load time data and extract refresh data, is already accessible from the administrative views that are built into Tableau Server. However, to collect resource usage data you need to use a external performance monitoring tools. To collect
additional load time data and background task data, you can connect to the Tableau Server repository.

For more information on the built-in administrative views, see Administrative Views.

**Note:** To use the sample workbook and to publish views to Tableau Server, you must have Tableau Desktop.

**Collect Data with the Tableau Server Repository**

The Tableau Server repository is a PostgreSQL database that stores data about all user interactions, extract refreshes, and more. You can enable access to the repository and use the data in it to help analyze and understand Tableau Server performance.

Looking for Tableau Server on Windows? See Collect Data with the Tableau Server Repository.

After you enable access to the Tableau Server repository, you can create views with data from the repository. The views that you create with this data are sometimes called custom administrative views. In addition to being used for performance monitoring, custom admin views can be used for tracking user activity, workbook activity, and more. For more information on the type of data that you can use for these views, see Create Custom Administrative Views. Alternatively, if you are only interested in performance data, you can use the preselected database tables in the sample performance workbook.

Enable access to the Tableau Server repository

You can use Tableau Desktop to connect to and query the Tableau Server repository using two built-in users. The user named tableau has access to several database views you can use as part of building your own analyses of Tableau Server activity. The user named readonly has access to additional database tables that you can use to create views for even more in-depth analysis and this is the user we recommend you use.
Before you can connect to the repository, you need to enable access for the `readonly` user to the database. Use the `tsm data-access repository-access enable` command to enable repository access. When you enable repository access, you also create a password for the `readonly` user. You will use this password to access to the repository.

Run the following command:

```
$ tsm data-access repository-access enable --repository-username readonly --repository-password <PASSWORD>
```

Connect to the Tableau Server repository

This section describes how to connect to a custom set of tables from Tableau Server repository.

1. In Tableau Desktop select **Data > Connect to Data**, and then select **PostgreSQL** as the database to connect to.

   **Note**: You might need to install the PostgreSQL database drivers. You can download drivers from [www.tableau.com/support/drivers](http://www.tableau.com/support/drivers).

2. In the PostgreSQL connection dialog box, enter the name or URL for Tableau Server in the **Server** box. If you have a distributed server installation, enter the name or IP address of the node where the repository is hosted.

   Connect using the port you have set up for the `pgsql.port`, which is 8060 by default.

3. Specify `workgroup` as the database to connect to.

4. Connect using the user and the password you specified.

5. Click **Connect**.
6. Select one or more tables to connect to.

7. Click **Go to Worksheet**.

**Performance Tuning**

This section describes how to use the performance data that you collect to identify ways to improve the performance of Tableau Server. Because no two server environments are identical, we can’t provide hard and fast rules for tuning server performance. However, you can draw conclusions about performance from patterns in the data that you collected.
For example, are there recurring spikes? Do any of the patterns that you notice in the administrative views correspond to similar patterns in Windows Performance Monitor? Observing patterns like this can guide you in testing and incremental tuning.

Most performance tuning for Tableau Server boils down to these general approaches:

- **Optimize for user traffic.** This tunes the server to respond to user requests and to display views quickly.

- **Optimize for extracts.** This tunes the server to refresh extracts for published data sources. You might want to optimize for extract refreshes if your organization has a lot of data and the data needs to be as up to date as possible.

Rendering views and refreshing extracts generate the most load on the server, so you should optimize for the task that your organization is most interested in.

**When to Add Nodes and Reconfigure**

Tableau Server can scale up and out as your needs and requirements evolve. Here are some guidelines to help you figure out whether it’s time to add more nodes to your system, reconfigure the server, or both:

- **More than 100 concurrent users:** If your deployment is user-intensive (>100 simultaneous viewers), it’s important to have enough VizQL processes—but not so many that they exceed your hardware’s capacity to handle them. Also, enabling the Tableau Server **Guest User account** can increase the number of potential simultaneous viewers beyond the user list you may think you have. The administrative view can help you gauge this. For more information, see Actions by Specific User.

- **Heavy use of extracts:** Extracts can consume a lot of memory and CPU resources. There’s no one measurement that qualifies a site as extract-intensive. Having just a few, extremely large extracts could put your site in this category, as would having very many small extracts. Extract heavy sites benefit from isolating the Backgrounder process on its own machine.

- **Frequent extract refreshes:** Refreshing an extract is a CPU-intensive task. Sites
where extracts are frequently refreshed (for example, several times a day) are often helped by more emphasis on the background process, which handles refresh tasks. Use the Background Tasks for Extracts administrative view to see your current refresh rate.

- **Downtime potential**: If your server system is considered mission critical and requires a high level of availability, you can configure it so there’s redundancy for the server processes that handle extracts, the repository, and the gateway.

**Performance Tuning Examples**

Performance tuning examples have been expanded and moved to separate topics. For more information, see Performance Tuning.

For recommended baseline configurations that serve as starting point for installations with one, two, and three nodes, see Recommended Baseline Configurations.

**Performance Recording**

This section describes how to create performance recordings and use the results to improve workbook performance. With performance recordings, you can view how long workbook events take. For example, you can see how long it takes to connect to a data source, run a query, render data, and more.

**Create a Performance Recording**

The Performance Recording feature in Tableau records performance information about key events as you interact with a workbook. You can then view performance metrics in a workbook that Tableau creates to analyze and troubleshoot different events that are known to affect performance:

- Query execution
- Geocoding
- Connections to data sources
- Layout computations
- Extract generation
- Blending data
- Server blending (Tableau Server only)

Tableau support may ask that you create a performance workbook as they work with you to diagnose performance issues.

Looking for Tableau Server on Windows? See Create a Performance Recording.

Enable Performance Recording for a Site

By default, performance recording is not enabled for a site. A server administrator can enable performance recording site by site.

1. Navigate to the site for which you want to enable performance recording.
2. Click Settings:

4. Click Save.
Start a Performance Recording for a View

1. Open the view for which you want to record performance.

When you open a view, Tableau Server appends ":iid=<n>" after the URL. This is a session ID. For example:

http://10.32.139.22/#/views/Coffee_Sales2013/USSalesMarginsByAreaCode?:iid=1

2. Type \texttt{:record\_performance=yes} & at the end of the view URL, immediately before the session ID. For example:

http://10.32.139.22/#/views/Coffee_Sales2013/USSalesMarginsByAreaCode?:record\_performance=yes&:iid=1

3. Click the \textbf{Refresh} button in the toolbar.

4. Load the view.

A visual confirmation that performance recording has started is the \textbf{Performance} option in the view toolbar:

![View a Performance Recording](image)

View a Performance Recording

1. Click \textbf{Performance} to open a performance workbook. This is an up-to-the-minute snapshot of performance data. You can continue taking additional snapshots as you continue working with the view; the performance data is cumulative.
2. Move to a different page or remove `record_performance=yes` from the URL to stop recording.

Interpret a Performance Recording

A performance recording workbook is a Tableau dashboard that contains three views: **Timeline**, **Events**, and **Query**.

For information on how to create a performance recording in Tableau Server, see Create a Performance Recording.

**Timeline**

The uppermost view in a performance recording dashboard shows the events that occurred during recording, arranged chronologically from left to right. The bottom axis shows elapsed time since Tableau started, in seconds.

In the Timeline view, the **Workbook**, **Dashboard**, and **Worksheet** columns identify the context for events. The **Event** column identifies the nature of the event, and the final column shows each event’s duration and how it compares chronologically to other recorded events:

![Timeline view example]

**Events**

The middle view in a performance recording workbook shows the events, sorted by duration (greatest to least). Events with longer durations can help you identify where to look first if you want to speed up your workbook.
Different colors indicate different types of events. The range of events that can be recorded is:

- Computing layouts

  If layouts are taking too long, consider simplifying your workbook.

- Connecting to data source

  Slow connections could be due to network issues or issues with the database server.

- Executing query

  - For live connections, if queries are taking too long, it could be because the underlying data structure isn’t optimized for Tableau. Consult your database server’s documentation. As an alternative, consider using an extract to speed performance.

  - For extracts, if queries are taking too long, review your use of filters. If you have a lot of filters, would a context filter make more sense? If you have a dashboard that uses filters, consider using action filters, which can help with performance.

- Generating extract

  To speed up extract generation, consider only importing some data from the original data source. For example, you can filter on specific data fields, or create a sample based on a specified number of rows or percentage of the data.

- Geocoding
To speed up geocoding performance, try using less data or filtering out data.

- Blending data

To speed up data blending, try using less data or filtering out data.

- Server rendering

You can speed up server rendering by running additional VizQL Server processes on additional machines.

Query

If you click on an Executing Query event in either the Timeline or Events section of a performance recording dashboard, the text for that query is displayed in the Query section.

If you are connected to a published data source, the query text is displayed in XML. If you are connected to the data source directly, the query is displayed in SQL like shown below:

```
SELECT "State"."ID" AS "ID",
    "StateSynonyms"."Name" AS "State_Name",
    "State"."ParentID" AS "State_ParentID"
FROM "StateSynonyms"
    INNER JOIN "State" ON ("State"."ID" = "StateSynonyms"."ParentID") AND ("State"."MapCode" = "StateSynonyms"."MapCode"
```

If it makes sense, you can use the query text to work with your database team on optimizing at the database level. Sometimes the query is truncated and you’ll need to look in the Tableau log to find the full query. Most database servers can give you advice about how to optimize a query by adding indexes or other techniques. See your database server documentation for details.

Sometimes for efficiency, Tableau combines multiple queries into a single query against the data. In this case, you may see an Executing Query event for the Null worksheet and zero queries being executed for your named worksheets.
Performance Resources

This topic describes external resources that you can use to monitor and tune performance.

Disclaimer: This topic includes information about third-party and community supported products. Please note that while we make every effort to keep references to third-party and community content accurate, the information we provide here might change without notice. For the most up-to-date information, please consult the documentation for products referenced here. To learn more about community supported tools, see Support levels for IT and developer tools.

- **TabJolt**. A load generation tool that you can use to understand how Tableau Server responds to user interactions over time. Use TabJolt to establish a baseline for server performance and test deployments before pushing them to production environments.

- **Replayer**. A tool that can replay log-based real user traffic from a Tableau Server against any other server or configuration. It replays Tableau Server single- or multi-user sessions. Here are some ways that Replay can be used:
  
  - Playback specific Tableau Server sessions, and filter the session based upon start time or RequestID.
  
  - Use it to simulate load conditions so that you can test how to scale and balance your Tableau Server installations.
  
  - Perform regression testing by running and comparing end-to-end user scenarios for Tableau Server upgrades.
  
  - Capture and report HTTP exceptions that occur in a single-user session.
  
  - Replay a defect, so that you can troubleshoot and verify that it is fixed.

- **Scout**. An exploratory tool that captures performance metrics across any workbooks.
on both Tableau Desktop and Tableau Server Here are some ways that Scout can be used:

- Find slow workbooks on Server.
- Validate performance improvements or regressions after making server configuration or topology changes.
- Validate that workbooks and dashboards are loading properly after upgrading to new Tableau Server.
- Validate that workbooks are still working properly after data source changes.

- **TabMon.** A monitoring tool that uses Windows Performance Monitor and Java Management Extensions to record performance data about Tableau Server to a PostGreSQL database.
- **Microsoft System Center.** A set of server management products for monitoring, configuration, automation, and more.
- **HP Sitescope.** An agentless application monitoring tool.
- **Zabbix.** An open-source, real-time monitoring tool.
- **Splunk.** A tool for monitoring and analyzing machine data, including logs.
- **Graylog.** An open-source log management tool.

**About Client-Side Rendering**

When you navigate to a view in Tableau Server, the processing required to display the view (the rendering) can either be performed by your client web browser or by Tableau Server depending on the complexity of the view. The complexity of the view is determined by the number of marks, rows, columns, and more. If a view is less complex, then it is faster for your web browser to render the view than it is to send a request to Tableau Server. If a view is more complex, then it is faster to send a request to Tableau Server and take advantage of the server's computing power.
As a server administrator, you can configure when client-side rendering happens both for web browsers on your computer and web browsers on mobile devices by adjusting the complexity threshold.

Requirements

- **Supported browsers**: Client-side rendering is supported in Internet Explorer version 9.0 or higher, Firefox, Chrome, and Safari. All of these web browsers include the HTML 5 `<canvas>` element, which is used by client-side rendering.

- **Polygons and the page history feature**: If a view uses the polygon mark type or the page history feature, server-side rendering is performed, even if client-side rendering is otherwise enabled.

Configure the complexity threshold for computers and mobile devices

Because computers have more processing power than mobile devices, Tableau Server performs more client-side rendering in your computer’s web browser than in your mobile device’s web browser. You can adjust how much client-side rendering happens for computers and mobile devices with the complexity thresholds. You might want to adjust the complexity thresholds if you notice that views display slowly on mobile devices. Alternatively, you might want to increase the thresholds to reduce the number of requests to Tableau Server.

By default, the complexity threshold for computer web browsers is 100. To adjust the complexity threshold for computer web browsers, use the following command:

```
tsm configuration set -k vizqlserver.browser.render_threshold -v [new value]
```

By default, the complexity threshold for web browsers on mobile devices is 60. To adjust the complexity threshold for web browsers on mobile devices, use the following command:

```
tsm configuration set -k vizqlserver.browser.render_threshold_mobile -v [new value]
```

For example, to change the mobile threshold to 40, you might enter the following command:
tsm configuration set -k vizqlserver.browser.render_threshold_mobile -v 40

For more information on how to use tsm option set, see tsm configuration set Options.

Disable client-side rendering

Client-side rendering is enabled by default and is recommended to improve the performance of views. However, you might want to disable client-side rendering temporarily for testing or if your server is being accessed primarily by computers or mobile devices with very little processing power.

Use the following command to disable client-side rendering:

tsm configuration set -k vizqlserver.browser.render -v false

For more information on how to use tsm option set, see tsm configuration set Options.

Testing with the URL Parameter

To test server-side rendering on a session basis, type ?:render=false at the end of the view’s URL. For example:

http://localhost/views/Supplies/MyView?:render=false

If client-side rendering is disabled on Tableau Server, enter ?:render=true to enable it for the session:

http://localhost/views/Supplies/MyView?:render=true

You can also test particular complexity thresholds on individual views to see if it’s appropriate to adjust the server-wide threshold for your server and network conditions. For example, you may find that lower complexity (such as 80) or higher complexity (such as 120) tipping points result in more responsiveness to user interactions. To test a threshold, you can keep the server’s default configuration (client-side-rendering enabled) and enter the test threshold number at the end of the view’s URL. For example:

http://localhost/views/Supplies/MyView?:render=80
Maintenance

Backup and Restore

This topic describes how to back up your Tableau Server deployment for recovery scenarios. Specifically, this topic explains how to recover a clone of a server from a collection of backup data and assets.

Note: If you need to restore only the repository on an otherwise functional Tableau Server, see Restore from a Backup. If you are running a distributed deployment, and your initial node has failed, see Recover from an Initial Node Failure.

Backup data types

There are three types of backup data that Tableau Server can generate. We recommend performing regular backups of each type in case you must restore a server in a recovery scenario:

- Respository data: Tableau Server data consists of the Tableau PostgreSQL database or repository, which contains workbook and user metadata, data extract files, and site configuration data. When you use TSM to create a backup, all of this data is saved in a single file with a .tsbak extension. This data is backed up with the tsm maintenance backup command.

- Topology data: defines how your Tableau Server processes are configured in both single-server and multiple node deployments. This data is backed up with the tsm settings export command.

- Configuration data: includes most of the server configuration information required to fully recover a server. SMTP, alerting, authentication, are all examples of con-
Configuration data that are exportable for backup. Configuration data is backed up with the `t.sm settings export` command.

Backup assets that require a manual process

Some configuration data is not included in the `t.sm settings export` command and must therefore be documented and restored manually. The following configuration data is excluded from the `t.sm settings export` operation. Your backup maintenance process should include documenting the following Tableau Server configuration data:

- System user accounts. Tableau Server setup creates two default accounts, `tsmagent` and `tableau`. These accounts are created for access to Tableau Server resources. If you have not changed these accounts, then you do not need to document them. For more information about these accounts, see System User and `sudo` Privileges.

- TSM group membership. There are two groups created by Tableau Server: `tableau` and `tsmadmin`. If you configured alternative groups when you installed Tableau Server, then you'll need to document the group names.

In all cases you should document the user accounts that are in these groups. To view membership in a group, run the following command `grep <group_name> /etc/-group`.

- Coordination Service deployment configuration. If you are running a multinode cluster, document which nodes are running the Coordination Services process. To view process configuration on your nodes, run `t.sm topology list-nodes -v`.

- Customization settings: If your organization uses custom header or sign-in logos for Tableau Server web pages, you should include a copy of those assets with your backup portfolio. See `t.sm customize`.

- Authentication assets: Most certificate files, key files, keytab files or other authentication-related assets are not backed up by TSM. The exception are the public
certificate and private key for the PostgreSQL database and certificate and key for the gateway process.

However, all other authentication-related assets are not backed up. For example, if you have enabled access to the PostgreSQL database with the tsm data-access repository-access enable command, be sure to document the name/password pairs for each account you’ve configured. These credentials are not backed up.

- LDAP assets: Keytab files, configuration files, and or other LDAP-related assets are not backed up by TSM.

Server secrets and repository passwords are crypto-related configurations that are not exported. However, you do not need to document configuration values. New secrets will be created as part of the restoration process when you initialize the new instance.

Back up Tableau Server for recovery

Tableau Server includes commands that you run to generate backup data for Tableau Server.

1. Backup topology and configuration data. Both of these data sets are included when you run the tsm settings export command. The data is exported as a json file. Specify the name and location of the json file by running the following command:

   tsm settings export -f \path\to\file.json

2. Back up repository data. Repository data is backed up with the tsm maintenance backup command. Specify the name and location of the backup file by running the following command:

   tsm maintenance backup -f <\path\to\backup_file> -d

See Back Up Tableau Server Data for more information.
Restoring core Tableau Server functionality

The procedure below uses the assets from the previous two sections to rebuild a Tableau Server in a recovery scenario.

Topology and configuration backup data must be from Tableau Server on Linux. You cannot restore configuration data from a backup file that was generated on Tableau Server on Windows. To restore a backup made from Tableau Server on Windows to Tableau Server on Linux, see Migrate Tableau Server from Windows to Linux.

You must have the following assets ready:

- Topology and configuration data: This is the json file that is generated by the `tsm settings export` command.

- Repository backup file: This is the file with a .tsbak extension that is generated by the `tsm maintenance backup` command.

- Backup assets: These assets include the list of documented configurations as noted in the previous section.

To restore Tableau Server

1. Install Tableau Server Package. If your organization used non-default system user accounts, as described in an early section of this topic, then you must specify the users during this step.

2. Activate and Register Tableau Server.

3. Configure Local Firewall (optional).

4. Import topology and configuration data. Copy the topology and configuration json backup file to the computer. Import the json file by running the following command:

   `tsm settings import -f /path/to/file.json`

5. Set identity store type.
• For local identity store, run the following command:

    tsm configuration set -k "wgserver.authenticate" -v "local"

• For any type of LDAP directory identity store, run the following command:

    tsm configuration set -k "wgserver.authenticate" -v "activedirectory"

LDAP directory will require authentication-related files (keytab file, Kerberos config file, and/or SSL certificates). Copy these files to the appropriate directories as part of this step. See identityStore Entity.

6. Finalize Installation.

7. Deploy a Coordination Service Ensemble. The ensemble configuration must match your previous configuration.

8. Restore repository data. See Restore from a Backup.

9. Repopulate TSM group membership. Add users to groups with this command:

    sudo usermod -G <group_name> -a <username>

Restore other functionality

If the previous server was configured with the following features, then you will need to re-enable and reconfigure them on the restored server:

• Authentication solutions: OpenID, external SSL, and trusted authentication. See Authentication.

• Site customizations: See tsm customize.

• Enable access to PostgreSQL repository: See tsm customize.
Database Maintenance

A Tableau Server administrator should perform regular database maintenance, monitor disk usage on the server, and clean up unnecessary files to free up space on the server. Taking these steps can help ensure that Tableau Server runs with maximum efficiency.

You can use the tsm command line tool to back up and restore your Tableau data. Tableau data includes Tableau Server’s own PostgreSQL database, which stores workbook and user metadata, data extract files, and server configuration data. Tableau Server log files capture activity and can help you diagnose problems. Logs are written to folders on the server and you can archive and remove them to save disk space.

**Note:** You can only use backups made with the `tsm maintenance backup` command when restoring Tableau Server data. Database backups made in other ways, and virtual machine snapshots are not valid sources for restoring Tableau Server.

Back Up Tableau Server Data

Regularly backing up Tableau Server is an important step in proper administration and maintenance of your server. Backups you create using Tableau Services Manager (TSM) are the only backups you can use to restore your Tableau Server data, so it is critical that you have an up-to-date backup created.


Tableau Server data consists of the Tableau PostgreSQL database or repository, which contains workbook and user metadata, data extract files, and configuration data. When you use TSM to create a backup, all of this data is saved in a single file with a .tsbak extension. If you are running a distributed installation of Tableau Server data from all the nodes is backed up.

The frequency of your backups depends on your environment, including how much use your server gets and how much and frequently the content and users change. Any changes or
updates that happen after your backup will be lost if there is a system failure and you need to restore Tableau Server. The more activity there is, the more often you need to back the server up.

In addition to regular backups, you should create a backup immediately before upgrading to a new version of Tableau Server.

For safety, after you create the backup, you should store the .tsbak file on a computer that is not a part of your Tableau Server installation.

In this topic

- Create a backup using the TSM command line interface (CLI)
- Create a pre-upgrade backup
- Script the backup process

Create a backup using the TSM CLI

Use the tsm maintenance backup command to create a backup of the data managed by Tableau Server. This data includes the Tableau PostgreSQL database (the repository) which contains workbook and user metadata, extract files, and server configuration data. The current date appended to the backup file:

```
tsm maintenance backup -f <backup_file> -d
```

For more information, see tsm maintenance backup

Create a pre-upgrade backup

You should always create a backup before upgrading Tableau Server. If you have created a backup yourself, before the upgrade, you can choose to skip this and save time during the upgrade. You can create a backup while Tableau Server is running and minimize the amount of time the server is unavailable during upgrade. The process for creating a pre-upgrade backup is the same as for creating regular backups, with one additional consideration for distributed installations.
The Tableau backup file (.tsbak) includes configuration information as well as data. Therefore, a backup of a distributed installation of Tableau Server will include configuration information about the additional nodes, including their IP addresses.

**Note:** You should uninstall Tableau Server from any nodes that you are not including in your new installation to avoid conflicts between the older nodes and the new installation.

Script the backup process

If you back up often, you might want to create a script that performs the backup and related tasks for you. These tasks include:

- Saving log files that are older than seven days if you do want to keep them, or conversely, removing log files that you don't want to back up.
- Running the backup itself.
- Copying the backup file to a separate computer for safekeeping.
- Verifying that the server is running after the script is finished.

This section discusses `tsm` commands you can use together to perform a backup and related tasks.

Save logs before cleanup

Performing a backup removes log files that are older than seven days. If you want to preserve the logs before the backup, run the following command before you start the backup:

```
tsm maintenance ziplogs
```

Run the backup

**To create the backup, use the `tsm maintenance backup` command:**

```
tsm maintenance backup --file <backup_file> --append-date
```
Note the following about the command:

- Add **--append-date** to the command to include the date in the backup file name.
- By default the backup file is created in `/var/opt/tableau/tableau_server-/data/tabsvc/files/backsups/`. For more information, see tsm maintenance backup.

Copy the backup file to another computer

As a best practice, after the backup is created, copy the backup file to another location that is separate from Tableau Server.

**Restore from a Backup**

Use the **tsm maintenance restore** command to restore your Tableau Server data. You might do this if you had a system failure and need to restore your data, if you need to switch back to a previous version of Tableau Server (for example, if there is a problem with an upgrade), or if you are moving Tableau Server to new hardware.

Only backups created using **tsm maintenance backup**, or created by the Tableau Server uninstall process can be used to restore Tableau Server data.

When you use **tsm maintenance restore** to restore your Tableau data, the contents of the PostgreSQL database, data extracts, and configuration files are overwritten with the content in the backup file (**.tsbak**). If you are running a distributed installation of Tableau Server, perform the restore on the initial node.

You can only restore from a backup that has the same type of identity store as the running server. For example, a backup from a server using local authentication can be restored to a Tableau Server initialized with local authentication, but a backup from a server using Active Directory authentication cannot be restored to a server initialized with local authentication.
Restore Tableau Server from a backup file

1. Stop the server. At a command prompt, type:

   tsm stop

2. Restore from a backup file. At a command prompt, type:

   tsm maintenance restore --file <file_name>

   In the above line, replace <file_name> with the name of the backup file you want to restore from.

   The restore command expects a backup file in the directory defined in the TSM basefilepath.backuprestore variable. By default:

   /var/opt/tableau/tableau_server/data/tabsvc/files/backups/

   For more information about file paths and how to change them, see tsm File Paths.

3. Restart the server:

   tsm start

When you restore a .tsbak file, Tableau Server automatically creates a copy of its current data folder, names it tabsvc.bak-*, and places it in /var/opt/tableau/tableau_server/data. This folder is an emergency backup of Tableau Server that Tableau Support may be able to use in case something goes wrong during backup restoration.

When the restore is complete and you have verified that Tableau Server is running correctly with all the expected data, it’s safe to remove any tabsvc.bak-* folders from /var/-opt/tableau/tableau_server/data to free additional disk space. In Tableau Server clusters, tabsvc.bak-* folders are created on each machine running Tableau Server.
Important: Only remove the tabsvc.bak-* folders. Do not remove the tabsvc folder, which is also located under /var/opt/tableau/tableau_server/data. It contains necessary Tableau Server data.

Server Maintenance

As an administrator, you will want to check the status of the server, analyze and monitor the activity on the server, manage scheduled tasks, or perform certain maintenance activities such as clearing saved data connection passwords. In addition, there are several settings that you may want to specify to customize the user experience for people using the server. You can do some of these tasks from the General page of the Status page and others from the Settings page.

Data Clean Up

You should periodically purge log files that are generated and stored on Tableau Server. If you are running Tableau Server on a distributed deployment, then you must purge log files on each node.

The tsm maintenance cleanup command deletes log files and temporary files. You should run the tsm maintenance cleanup command regularly.

View Server Process Status

You can use the Process Status table on the Server Status page to view the state of Tableau processes on each Tableau server:
Possible status indicators are listed at the bottom of the table:

- **Active** — The process is functioning as intended. See File Store in Troubleshoot Server Processes for details on possible active states.
- **Busy** — The process is completing some task. See File Store and Repository in Troubleshoot Server Processes for more information.
- **Passive** — The repository is in passive mode
- **Unlicensed** — The process is unlicensed.
• **Down**—The process is down. The implications of this differ depending on the process.
• **Status unavailable**—Tableau Server is unable to determine the status of the process.

If there is additional information, a message appears below the status icon:

![Ready for removal](image)

For more information about troubleshooting process status, see Troubleshoot Server Processes.

Access Status Remotely

You must be a Tableau Server administrator to see the Server Status page, but you can grant remote access to other computers to allow access to a machine-readable (XML) version of the Status table by non-admin users and by computers other than the primary Tableau Server node. One reason you might do this is as part of a remote monitoring process.

To grant remote access to Tableau Server status:

1. Open a command prompt as an administrator and type the following:

   tsm configuration set -k wgserver.systeminfo.allow_referrer_ips -v <ip address>

   In the above command, `<ip address>` is the IPv4 address of the computer for which you want to enable remote access to the Tableau Server status XML.

   For example:

   `tsm configuration set -k wgserver.systeminfo.allow_referrer_ips -v 10.32.139.31`
If you are enabling remote access for more than one computer, use commas to separate each IP address.

```bash
tsm configuration set -k wgserver.systeminfo.allow_refferer_ips -v 10.32.139.31,10.32.139.35
```

2. Commit the configuration change:

```bash
tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Now, users of computers with the IP addresses that have been added can view Tableau process status by entering the URL `http://<server>:/admin/systeminfo.xml` in a browser or from a command line (for example, `curl http://jsmith/admin/systeminfo.xml`).

For details on the XML that is returned, see Get Process Status as XML.

### Get Process Status as XML

To get a machine-readable version of the server process status, that is, a version of the status formatted in XML, use the following URL:

`http://my_tableau_server/admin/systeminfo.xml`

You must be signed in to Tableau Server to view the machine-readable process status, or have enabled remote access.

The server returns a status report similar to the following:

```xml
<systeminfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <machines>
```

- 730 -
<machine name="my_tableau_server">
  <repository worker="my_tableau_server:8060" status="Active" preferred="false"/>
  <dataengine worker="my_tableau_server:27042" status="Active"/>
  <applicationserver worker="my_tableau_server:8600" status="Active"/>
  <apiserver worker="my_tableau_server:8000" status="Active"/>
  <vizqlserver worker="my_tableau_server:9100" status="Active"/>
  <dataserver worker="my_tableau_server:9700" status="Active"/>
  <backgrounder worker="my_tableau_server:8250" status="Active"/>
  <gateway worker="my_tableau_server:80" status="Active"/>
  <searchandbrowse worker="my_tableau_server:11000" status="Active"/>
  <cacheserver worker="my_tableau_server:6379" status="Active"/>
  <filestore worker="my_tableau_server:9345" status="Active" pendingTransfers="0" failedTransfers="0" syncTimestamp="2015-02-27T20:30:48.564Z"/>
  <clustercontroller worker="my_tableau_server:12012" status="Active"/>
  <coordination worker="my_tableau_server:12000" status="Active"/>
</machine>
</machines>
<service status="Active"/>
</systeminfo>
Status values in the XML

- `<process>` **worker** - The name of the node running the process and the port the process is using.

- **status** - The status of the process on the node. Possible values are: Active, Passive, Unlicensed, Busy, Down, ReadOnly, ActiveSyncing, StatusNotAvailable, StatusNotAvailableSyncing, NotAvailable, DecommissionedReadOnly, DecommissioningReadOnly, and DecommissionFailedReadOnly

- **pendingTransfers** - A count of the workbook or data source extracts the node needs to get to be fully synced. These represent items that were published to this file store node, and items that were published to other file store nodes and need to be copied to this node.

- **failedTransfers** - A count of the workbooks or data sources that did not transfer successfully to this file store node during the last automated job. The automated job normally runs about every 15 to 30 minutes, but may take longer when transferring a large number of extracts or large extracts.

  Failed transfers do not necessarily indicate a problem with Tableau Server. The recurring automated job will normally transfer files that failed during the previous sync. Reasons for failed file transfers are listed in the logs.

- **syncTimestamp** - The time in UTC of the last automated job that ran and synchronized files.

Troubleshoot Server Processes

When Tableau Server is functioning properly, processes will show as Active, Busy or Passive (Repository). If there is additional information, a message appears below the status icon:
Possible status indicators are:

<table>
<thead>
<tr>
<th>Active</th>
<th>Busy</th>
<th>Passive</th>
<th>Unlicensed</th>
<th>Down</th>
<th>Status unavailable</th>
</tr>
</thead>
</table>

The following sections provide troubleshooting recommendations for status messages that you may see.

Cluster Controller

This message will only display if you have more than two nodes.

**Status: Down; Message: "Node degraded"**

One or more of the following are true:

- Repository on the node is stopped.
- Node cannot respond to failover elsewhere on the cluster.
- If Tableau Server is configured for high availability and this is the active repository, failover to the second repository occurs.
- No status available for repository or file store on this node.

No action is necessary unless the cluster controller is regularly down or is down for an extended period of time.

If that occurs, take the following actions, in order, until the problem is resolved:

1. Check disk space. If disk space is limited, save the log files (use `tsm maintenance ziplogs`) in case you need them for Support, then remove unnecessary files.
2. Restart Tableau Server.
3. If Cluster Controller continues to show as down, save the log files (`tsm main-
tenance ziplogs`) and contact Support.
File Store

File Store status only reflects the state of the file store when the page was loaded.

An active status (✓) with no message indicates that no extracts were being synchronized when the page was loaded. It is possible that the recurring "catch-all" job is running and synchronizing extracts.

**Status: Busy; Message: "Synchronizing"**

"Synchronizing," usually indicates that extracts were being synchronized across file store nodes when the page was loaded.

However, the "synchronizing" message is also returned following installation (both single-node and multi-node). After Tableau initializes the status should disappear within 15 or 20 minutes.

**Status: Down; "Data Extracts unavailable"**

On a single-node installation: "Data Extracts unavailable" indicates that existing extracts may be available but publish/refresh will fail. On multi-node installations, this message indicates that extract synchronization will fail for this node.

No action is necessary unless the file store is regularly down or is down for an extended period of time.

If that occurs, take the following actions, in order, until the problem is resolved:

1. Check disk space. If disk space is limited, save the log files (use `tsm maintenance ziplogs`) in case you need them for Support, then remove unnecessary files.
2. Restart Tableau Server.
3. If Cluster Controller continues to show as down, save the log files (`tsm maintenance ziplogs`) and contact Support.

**Status: Busy; "Decommissioning"**
This message indicates that this File Store is in read-only mode and that any unique files on this node are being replicated to other file store nodes.

To remove this node, wait until the status message changes to "Ready for removal".

**Status: Active; "Ready for removal"**

This message indicates that the File store is in read-only mode.

You can safely stop (tsm stop) the cluster and remove File Store processes, or remove entire node.

**Status: Active; "Decommission failed"**

This message indicates that the File store is in read-only mode, and that at least one unique file failed to replicate to another file store node.

To resolve a failed decommissioning:

1. Run the `tsm topology filestore decommission` command again.
2. Check disk space on other file store nodes. Decommissioning will fail if another file store node does not have enough space to store all the extracts.
3. Check the `tsm.log` file on the initial node and additional nodes for errors.
4. Stop Tableau Server (tsm stop) and then try running the `tsm topology filestore decommission` command again.
5. Put the file store node back into read/write mode (`tsm topology filestore recommission`), collect logs, and then contact Support.
6. With Support: copy and merge extracts directory from this file store node to the same directory on another file store node.

Repository

**Status: Busy; Message: "Setting up"**

The "Setting up" message indicates one or more of the following states:

- Passive repository is being synchronized with active repository.
- Repository is not ready to handle failover.
Repository may have gotten more than two minutes behind active repository and is being setup again (this is faster than waiting for a sync).

Failover occurred and this former active repository is rejoining the cluster.

Wait until the repository status message changes to "Passive".

If this message does not appear, or if it is taking a long time:

1. Check disk space and free space if possible.
2. Check cluster controller logs for errors.
3. Restart node.

Staus: Busy; Message: "Synchronizing"

Repository is synchronizing, for example after a failover.

Staus: Down; Message: n/a

When the Repository shows a status of down and there is no message, then the Repository is in one of the following states:

- If the installation is configured for high availability, failover of the repository occurred.
- Processes are restarting with updated database connection configurations after fail-over.
- If another active repository is not available, Tableau Server is down.

Take these actions in order until a step resolves the problem:

1. Wait several minutes for cluster controller to attempt to restart.
2. Restart Tableau Server (tsm restart).
3. Check disk space. If disk space is limited, save the log files (use tsm maintenance ziplogs) in case you need them for Support, then remove unnecessary files.
4. Restart Tableau Server.
5. If repository continues to show as down, save the log files (tsm maintenance ziplogs) and contact Support.

Staus: Passive; Message: n/a
A passive status with no message indicates that the node is working as intended and that it is ready for failover if needed.

VizQL Server

⚠️ Status: Unlicensed; Message: n/a

For information about unlicensed status for a VizQL Server process, see Handle an Unlicensed VizQL Server Process.

Clear Saved Data Connection Passwords

As the administrator, if you enable users to save data source passwords, server users can save data source passwords across multiple visits and browsers so they are no prompted for their credentials each time they connect to a data source.

You can reset the data source passwords for all Tableau Server users. Doing this forces them to sign in to the data sources the next time they visit a view that requires database authentication. Server users can also clear their saved data connection passwords on an individual basis using their User Preferences page.

To clear saved data connection passwords for all server users:

1. In a site, click **Settings > General**.
2. Under Saved Credentials, click **Clear All Saved Credentials**.

![Saved Credentials](image)

**Set the Default Start Page for All Users**

By default, Tableau Server installs with the Views page as the default start page for all users. As the administrator, you can change this to another page that all users have access to, such as the Workbooks page. Individual users will be able to override your setting (search for "Access Your Profile and Account Settings" in the Tableau Server Help for details).

To set the default start page for all users:

1. Navigate to the page you want to be the default page.

2. Click your name on the upper right corner of the page.

3. Select **Make This the Start Page for All Users**.
Disable Automatic Client Authentication

After Tableau Desktop clients first successfully sign in to Tableau Server, they are automatically authenticated in the future. Client sessions are managed by OAuth access and refresh tokens.

By default, all client tokens reset after a year. If a client token has not been used in 14 days, then it will expire. You can change these values by setting the `refresh_token.absolute_expiry_in_seconds` and `refresh_token.idle_expiry_in_seconds` options. See `tsm configuration set Options`.

As a Tableau Server administrator can also disable automatic authentication.

To immediately disconnect all clients from Tableau Server and require users to sign in every time they connect:

1. In the site menu, click Manage All Sites, and then click Settings > General.
2. Under Connected Clients, clear the option Allow clients to automatically connect to Tableau Server.
3. Click Save.

Server Settings (General)

The following settings are available on the General page in Server - Settings.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedded Credentials - Allow publishers to embed data source credentials in a workbook</td>
<td>Allows publishers to attach passwords to published workbooks that will automatically authenticate web users to connect to data sources. The passwords are attached to workbooks and are only accessible on server. That is, when the workbook is opened in Tableau Desktop, users will still need to enter a user name and password to connect to the data source. When this</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Embedded Credentials - Allow publishers to schedule data extract refreshes</td>
<td>Allows publishers to assign workbooks to schedules. This option is only available if <strong>Allow publishers to embed data source credentials in a workbook</strong> is enabled. When this setting is enabled, publishers will see scheduling options in the Publish dialog box.</td>
</tr>
</tbody>
</table>
| Saved Credentials - Allow users to save passwords for data sources | Allows users to choose "Remember my password" and save data source passwords across multiple visits, browsers, and devices. (By default, users can choose to "Remember my password until I sign out," which lets them save their password only for a single browser session.)

As an administrator, you can **clear all saved passwords** at any time. In addition, users can clear their own saved passwords. |
<p>| Saved Credentials - Allow users to save OAuth access tokens for data sources | Allows users to store access tokens with their user preferences. Access tokens are provided by cloud data sources that support OAuth connections, and they are used instead of user names and passwords to grant access to the data. |
| Connected Devices - Allow devices to automatically connect to Tableau Server | Controls whether mobile users must sign in and provide their credentials every time they connect to Tableau Server, or if users can connect with their devices to Tableau Server without providing credentials after they authenticate |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>their device successfully the first time.</td>
<td>For more information, see Disable Automatic Client Authentication.</td>
</tr>
<tr>
<td>Guest Access - Enable Guest account</td>
<td>Allows users to view and interact with embedded views without having to sign in to a Tableau Server account. Permission can be assigned to the Guest User account to control the interactivity allowed for each view. This option is only available if you have a core-based server license.</td>
</tr>
<tr>
<td>Default Start Page</td>
<td>Takes you to the server’s current default start page for all users. For more information on how to change the default start page, see Set the Default Start Page for All Users. Individual users will be able to override this setting (search for &quot;Access Your Profile and Account Settings&quot; in the Tableau Server Help for details).</td>
</tr>
<tr>
<td>Language and Locale</td>
<td>Controls the language used for the server user interface and the locale used for views. Individual users can override this setting on their Account Settings page. Also, web browser settings are evaluated first to determine which language and locale should be used. For more information, see Language and Locale for Tableau Server.</td>
</tr>
<tr>
<td>Active Directory Synchronization -</td>
<td>Controls the synchronization of all Active Directory groups in Tableau Server based on a schedule that you specify after you select the option <strong>Synchronize Active Directory groups on a regular schedule</strong>. For more information, see Synchronize All Active Directory Groups on the Server.</td>
</tr>
<tr>
<td>Synchronize Active Directory groups on a</td>
<td>regular schedule</td>
</tr>
<tr>
<td>schedule</td>
<td></td>
</tr>
</tbody>
</table>
| **Recommendations Trainer** | Suggests server content, such as data sources and tables, to Tableau Desktop users. Content suggestions are based on popularity of the content or on content frequently used by other users who are similar to the current user.

Scheduling options control how often the server checks for new content and new usage information to keep the recommendations up to date.

New content includes new and updated data sources. New usage information includes information such as "Laura Rodriguez used the Food Catering data source" and "Henry Wilson used the Monthly Sales data source."

If you notice any impact on server performance, schedule this process to occur at a time when the server load is low. To track performance impact, look for the "Recommendations Trainer" task in the Background Tasks for Non Extracts view.

If you want to disable Recommendations, change the tsm configuration set Options option, "recommendations.enabled", to false. |
| **Reset to Default Settings** | Returns any server settings that have been changed since setup to their original state. |

### tsm Command Line Reference

The topics in this section include reference content for Tableau Services Manager (TSM) command line interface (CLI) to support Tableau Server for Linux.
TSM is used to manage installation and configuration of Tableau Server. To learn more about TSM, see Tableau Services Manager Overview.

**Note:** You can automate the installation and configuration tasks supported by the TSM CLI using the TSM API. To learn more about the prerelease (Alpha) TSM API, see Tableau Services Manager API.

Looking for Tableau Server on Windows? See tabadmin Commands.

**Viewing help content in the shell**

To view minimal help content from a bash shell use the `tsm help category`.

**Synopsis**

`tsm help [category] [command]`

**Commands**

`tsm help`

Help for all `tsm` commands

`tsm help <category>`

Show help for a specific command category. For example, `tsm help authentication`.

`tsm help <category> <command>`

Show help for a specific command. For example, `tsm help authentication open-id`.
**tsm help command**

List all top-level commands or categories.

**Categories**

**tsm authentication**

You can use the `tsm authentication` commands to enable, disable, and configure user authentication options for Tableau Server.

- `sspi`
- `kerberos`
- `list`
- `mutual-ssl`
- `openid`
- `saml`
- `sitesaml`
- `trusted`

**tsm authentication sspi <commands>**

This command will only work on Tableau Server on Windows. If you attempt to enable SSPI on Tableau Server on Linux, an error will be returned.

Enable or disable automatic sign-in using Microsoft SSPI.

If you use Active Directory for authentication, you can optionally enable automatic logon, which uses Microsoft SSPI to automatically sign in your users based on their Windows username and password. This creates an experience similar to single sign-on (SSO). Do not enable SSPI if you plan to configure Tableau Server for SAML, trusted authentication, or for a proxy server.
Synopsis

tsm authentication sspi enable [global options]
tsm authentication sspi disable [global options]

tsm authentication kerberos <commands>

Enable, disable, and configure Kerberos user authentication on Tableau Server. See Configure Kerberos.

Synopsis

tsm authentication kerberos enable [global options]
tsm authentication kerberos disable [global options]
tsm authentication kerberos configure --keytab-file <keytab_file.keytab> [global options]

Options for kerberos configure

-kt, --keytab-file <keytab_file.keytab>

    Required.

    Specifies the service.keytab file used for requests to the KDC.


tsm authentication list

List the server’s existing authentication-related configuration settings.

Synopsis

tsm authentication list [--verbose][global options]

Options

v, --verbose
Optional.
Show all configured parameters.

tsm authentication mutual-ssl <commands>

Enable, disable, and configure mutual SSL for user authentication on Tableau Server. To learn more about mutual SSL, see Configure Mutual SSL Authentication for Tableau Server on Linux.

Before you enable mutual SSL, you must enable and configure SSL for external communication. For information, see Configure SSL for External HTTP Traffic to and from Tableau Server (Linux).

Synopsis

tsm authentication mutual-ssl enable [global options]

tsm authentication mutual-ssl disable [global options]

tsm authentication mutual-ssl configure [options]
[global options]

Options

-cf, --ca-cert <certificate-file.crt>

Optional.
Specifies the location and file name for the certificate file. The file must be a valid, trusted certificate from a Certificate Authority (for example, Verisign).

-fb, --fallback-to-basic <true | false>

Optional.
Specifies whether Tableau Server should accept user name and password for authentication if SSL authentication fails.
Default value is false, to indicate that when configured for mutual SSL, Tableau Server does not allow a connection when SSL authentication fails.

-m, --user-name-mapping <upn | ldap | cn>

Optional.

Specifies the user name syntax (UPN, LDAP or CN) to retrieve from identity store or directory. The syntax must match the format for Subject or Subject Alternative Name on the user certificate.

-rf, --revocation-file <revoke-file.pem>

Optional.

Specifies the location and file name for the certificate revocation list file. This file can be a .pem or .der file.

**tsm authentication openid <commands>**

Enable, disable, and configure OpenID Connect (OIDC) user authentication on Tableau Server.

**Synopsis**

```bash
tsmt authentication openid enable [global options]
tsmt authentication openid disable [global options]
tsmt authentication openid configure [options] [global options]
tsmt authentication openid map-claims [options] [global options]
tsmt authentication openid get-redirect-url [global options]
```

**Options for openid configure**

```bash
-a, --client-authentication <string>
```
Optional.

Specifies custom client authentication method for OpenID Connect.

To configure Tableau Server to use the Salesforce IdP, set this value to `client_secret_post`.

`-cs, --client-secret <string>`

Optional.

Specifies the provider client secret. This is a token that is used by Tableau to verify the authenticity of the response from the IdP. This value is a secret and should be kept securely.

`-cu, --config-url <CONFIG-URL>`

Optional.

Specifies the provider configuration URL. The default value is constructed based on the name of the server (gateway.public.host), and the gateway port, if any (gateway.public.port). In addition, by default the protocol is set to `https://` if SSL is enabled for the server.

`-mf, --config-file <config-file.json>`

Optional.

Specifies the local path to the static OIDC discovery JSON document.

`-i, --client-id <CLIENT-ID>`

Optional.

Specifies the provider client ID that your IdP has assigned to your application.

`-id, --ignore-domain <true | false>`
Optional. Default: false

Set this to true if the following are true:

- You are using email addresses as usernames in Tableau Server
- You have provisioned users in the IdP with multiple domain names
- You want to ignore the domain name portion of the email claim from the IdP

Before you proceed, review the user names that will be used as a result of setting this option to true. User name conflicts may occur. In the case of a user name conflict, the risk of information disclosure is high. See Requirements for Using OpenID Connect.

-if, --iframed-idp-enabled <true | false>

Optional. Default: false

Specifies if IdP is allowed inside of an iFrame. The IdP must disable clickjack protection to allow iFrame presentation.

-ij, --ignore-jwk <true | false>

Optional. Default: false

Set this to true if your IdP does not support JWK validation. In this case, we recommend authenticating communication with your IdP using mutual TLS or another network layer security protocol.

-r, --return-url <return-url>

The URL of your server. This is typically the public name of your server, such as "http://example.tableau.com".

-sn, --custom-scope-name <string>

Optional.
Specifies a custom scope user-related value that you can use to query the IdP. See Requirements for Using OpenID Connect.

Options for openid map-claims

Use these options to change the default OIDC claims Tableau Server will use when communicating with your IdP. See Requirements for Using OpenID Connect.

-i, --id <string>

Optional. Default: sub

Change this value if your IdP does not use the sub claim to uniquely identify users in the ID token. The IdP claim that you specify should contain a single, unique string.

-un, --user-name <string>

Optional. Default: email

Change this value to the IdP claim that your organization will use to match user names as stored in Tableau Server.

tsm authentication saml <commands>

Configure Tableau Server to support single-sign on using the SAML 2.0 standard, enable or disable SAML for a site, map assertion attribute names between Tableau Server and the identity provider (IdP).

Available commands

tsm authentication saml configure [options] [global options]
tsm authentication saml disable [options] [global options]
tsm authentication saml enable [options] [global options]
tsm authentication saml export-metadata [options] [global options]
tsm authentication saml map-assertions [options]

**tsm authentication saml configure**

Configure the SAML settings for the server. Specify the SAML certificate and metadata files, provide additional required information, set additional options.

If you are configuring SAML for the first time or have previously disabled it, you must run this command with `tsm authentication saml enable`. For more information, see Configure Server-Wide SAML for Tableau Server on Linux.

**Synopsis**

```
 tsm authentication saml configure [options] [global options]
```

**Options**

```
-e, --idp-entity-id <id>
```

Required for initial SAML configuration; otherwise optional. IdP entity ID value.

Typically this is the same as the Tableau Server return URL. The entity ID that you enter is used as a base for generating site-specific entity IDs. For example, if you enter the following:

```
http://tableau-server
```

A site configured for SAML might display the following entity ID:

```
http://tableau-server/samlservice/public/sp/metadata?alias=48957410-9396-430a-967c-75bdb6e002a0
```

To find a site’s entity ID, go to the site’s Settings page, and select the **Authentication** tab. When SAML is enabled, the entity ID is shown under the first step for configuring site-specific SAML, exporting metadata.

```
-r, --idp-return-url <idp-return-url>
```

Required for initial SAML configuration; otherwise optional. The SAML return URL
configured in the IdP. This is typically the Tableau Server external URL; for example, https://tableau-server.

Notes

- http://localhost does not work for an external server.
- Adding a trailing slash to the URL (https://tableau-server/) is not supported.

-i, --idp-metadata <idp-metadata.xml>

Required for initial SAML configuration; otherwise optional. Provide the location and name of the XML metadata file you exported from the IdP’s settings.

For example, /var/opt/tableau/tableau_server/data/saml/<metadata-file.xml>

-cf, --cert-file <certificate.crt>

Required for initial SAML configuration; otherwise optional. The location and file name for the x509 certificate file for SAML. For requirements for the certificate file, see SAML Requirements.

For example, /var/opt/tableau/tableau_server/data/saml/<file.crt>

-kf, --key-file <certificate.key>

Required for initial SAML configuration; otherwise optional. Location and name of the key file that goes along with certificate.

For example, /var/opt/tableau/tableau_server/data/saml/<file.key>

-a, --max-auth-age <max-auth-age>

Optional. Default value is 7200 (30 mins).
The maximum number of seconds allowed between a user’s authentication and processing of the AuthNResponse message.

-d, --desktop-access <enable | disable>
Optional. Default value is enable.
Use SAML to sign in to the server from Tableau Desktop. To support this option, your IdP must support forms-based authentication.

-m, --mobile-access <enable | disable>
Optional. Default value is enable.
Allow using SAML to sign in from older versions of Tableau Mobile app. Devices running Tableau Mobile app version 19.225.1731 and higher ignore this option. To disable devices running Tableau Mobile app version 19.225.1731 and higher, disable SAML as a client login option on Tableau Server.

-so, --signout <enable | disable>
Optional. Enabled by default.
Enable or disable SAML sign out for Tableau Server.

-su, --signout-url <url>
Optional. Enter the URL to redirect to after users sign out of the server. By default this is the Tableau Server sign-in page. You can specify an absolute or a relative URL.

Example

tsm authentication saml configure --idp-entity-id https://tableau-server --idp-metadata /var/opt/tableau/tableau_server/data/saml/<metadata.xml> --idp-return-url https://tableau-server --cert-file /var/opt/tableau/tableau_server-
tsm authentication saml enable and saml disable

Enable or disable server-wide SAML authentication. In this context, all sites and users that you enable for SAML go through a single identity provider.

Synopsis

tsm authentication saml enable [global options]

 tsm authentication saml disable [global options]

Synopsis

tsm authentication saml export-metadata

Export the Tableau Server .xml metadata file that you will use to configure the SAML IdP.

Synopsis

tsm authentication saml export-metadata [options]
[global options]

Options

-f, --file [/path/to/file.xml]

Optional.

Specifies the location and file name in which the metadata will be written. If you don’t include this option, export-metadata saves the file to the current directory, and names it samlmetadata.xml.

-o, --overwrite

Optional.
Overwrites an existing file of the same name specified in -f, or of the default name if -f is not included. If a file specified in -f exists, and -o is not included, the command does not overwrite the existing file.

**tsm authentication saml map-assertions**

Maps attributes between the IdP and Tableau Server. Provide the name that the IdP uses for the attribute specified in each argument.

**Synopsis**

```bash
tsm authentication saml map-assertions --user-name <user-name> [global options]
```

**Options**

- `-r, --user-name <user-name-attribute>`
  
  Optional. The attribute in which the IdP stores the user name. On Tableau Server this is the display name.

- `-e, --email <email-name-attribute>`
  
  Optional. The name of the attribute in which the IdP stores email addresses.

- `-o, --domain <domain-name-attribute>`
  
  Optional. The attribute in which the IdP stores the domain name.

- `-d --display-name <display-name-attribute>`
  
  Optional. The attribute in which the IdP stores the domain name.

**Example for saml map-assertions**

```bash
tsm authentication saml map-assertions --email=Email --user-name=UserName
```
tsm authentication sitesaml enable and sitesaml disable

Set the server to allow or disallow SAML authentication at the site level. Enabling site-specific SAML gives you access to the Settings > Authentication tab in the Tableau Server web UI. The Authentication tab contains the site-specific SAML configuration settings.

Use the sitesaml enable command with saml configure if you haven't yet configured the server to allow site-specific SAML. For more information, see Configure Site-Specific SAML.

Synopsis

```
 tsm authentication sitesaml enable [global options]

 tsm authentication sitesaml disable [global options]
```

```
 tsm authentication trusted <commands>
```

Configure trusted authentication (trusted tickets) for user authentication on Tableau Server.

Synopsis

```
 tsm authentication trusted configure [options] [global options]
```

Options

```
-th, --hosts <string>
```

Optional.

Specifies the trusted host names (or IPv4 addresses) of the web servers that will be hosting pages with Tableau content.

For multiple values, enter the names in a comma-separated list where each value is encapsulated in double-quotes.

For example:
tsm authentication trusted configure -th "192.168.1.101", "192.168.1.102", "192.168.1.103"

or

// Authentication trusted configure -th "webserv1", "web-serv2", "webserv3"

-t, --token-length <integer>

Optional.

Determines the number of characters in each trusted ticket. The default setting of 24
characters provides 144 bits of randomness. The value can be set to any integer
between 9 and 255, inclusive.

Global options

-h, --help

Optional.

Show the command help.

-p, --password <password>

Required, along with --username if no session is active.

Specify the password for the user specified in --username.

--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30
minutes).

-s, --server <server address>
Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

**tsm configuration**

You can use the tsm configuration commands to get, set, and update configuration values.

- get
- set

**tsm configuration get**

View the current server configuration and topology.

Synopsis

`tsm configuration get --key [global options]`

Option

-k, --key

Required.

Get the current value of the specified configuration key.
tsm configuration set

Set or import server configuration or topology.

Quotes around the <config key> and the <config value> are optional unless there are spaces, in which case you must use quotes around the key or value.

Synopsis

tsm configuration set --key <config key> --value <config value> [global options]

Options

-k, --key <KEY>

  Required.

  Configuration key.

-v, --value <value>

  Required.

  Configuration value.

-d

  Optional.

  Reset the configuration value to its default.

Global options

-h, --help

  Optional.

  Show the command help.
-p, --password <password>

Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

tsm configuration set Options

You can use the following configuration options with the tsm configuration set command.

Essential syntax is:

```
tsm configuration set -k <config key> -v <config value>
```

After you run tsm configuration set, you must run tsm pending-changes apply. For more information, see tsm configuration and tsm pending-changes apply.
To reset a configuration key back to its default value, use the \(-d\) option:

```
tsm configuration set -k <value> -d
```

Options

api.server.enabled

**Default value:** true

Allows access to the Tableau Server REST API. By default, this functionality is enabled.

auditing.enabled

**Default value:** true

Allows access to the PostgreSQL (Tableau Server’s own database) historical auditing tables.

backgrounder.externalquerycachewarmup.enabled

**Default value:** true

Controls the caching of workbook query results after scheduled extract refresh tasks.

backgrounder.externalquerycachewarmup.view_threshold

**Default value:** 2.0

The threshold for caching workbook query results after scheduled extract refresh tasks. The threshold is equal to the number of views that a workbook has received in the past seven days divided by the number of refreshes scheduled in the next seven days.

backgrounder.extra_timeout_in_seconds

**Default value:** 1800
The number of seconds beyond the setting in `backgrounder.querylimit` before a background task is canceled. This setting makes sure that tasks do not hold up subsequent jobs if they are stalled. The setting applies to processes listed in `backgrounder.timeout_tasks`.

`backgrounder.failure_threshold_for_run_prevention`

Default value: 5

The number of consecutive failures of a subscription or extract job before that job is suspended. Suspending continuously failing jobs helps preserve background resources for other jobs. To disable suspension of failing background tasks, set this to -1.

`backgrounder.querylimit`

Default value: 7200

Longest allowable time, in seconds, for completing a single extract refresh task or subscription task. 7200 seconds = 2 hours.

**Note:** If a background task reaches this time limit, it may continue to run for an additional several minutes while being canceled.

`backgrounder.reset_schedules_on_startup`

Default value: true

Controls when to run background tasks that were scheduled to run at a time when the server was stopped. When set to `true` (the default), tasks are run at their next scheduled time. When set to `false`, all tasks that were scheduled to run when the server was stopped are run, simultaneously, at server startup, including times when the Tableau Server backup file (.tsbak) is restored.

`backgrounder.send_email_on_refresh_failure`

Default value: true
Controls whether extract refresh alerts are enabled for all sites on the server. By default alerts are enabled. To disable extract refresh alerts for all sites on a server, set this to false.

Extract alerts can be enabled or disabled on a site basis by site administrators in site settings, or at the user level in user settings.

```
backgrounder.sort_jobs_by_run_time_history_observable_hours
```

Default value: -1

Controls the time window used when determining duration of the last full extract job.

Tableau Server can sort full extract refresh jobs so they are executed based on the duration of their "last run," executing the fastest full extract refresh jobs first.

The "last run" duration of a particular job is determined from a random sample of a single instance of the full extract refresh job in last <n> hours. Full extract jobs are then prioritized to run in order from shortest to longest based on their "last" run duration. By default this is sorting is disabled (-1). If enabling this, the suggested value is 36 (hours).

```
backgrounder.sort_jobs_by_type_schedule_boundary_heuristics_milliSeconds
```

Default value: 60000

Controls the time window that identifies backgrounder jobs which are determined to have the same scheduled start time.

The backgrounder process orders work that is scheduled at the same time to be executed by job type, running the fastest category of jobs first: Subscriptions, then Incremental Extracts, then Full Extracts.

Jobs are batched to determine which jobs are scheduled at the "same time". A value 60,000 milliseconds (the default) indicates jobs for schedules starting within a 1
minute window should be classified in the same batch and so are ordered by type within that batch.

backgrounder.subscription_image_caching

Default value: true

Controls whether backgrounder will cache images that are generated for subscriptions. Cached images do not have to be regenerated each time so caching improves subscription performance. By default image caching is enabled. To disable image caching for all sites on a server, set this to false.

backgrounder.timeout_tasks

Default value: refresh_extracts, increment_extracts, subscription_notify, single_subscription_notify

The list of tasks that can be canceled if they run longer than the combined values in backgrounder.querylimit and backgrounder.extra_timeout_in_seconds. The list of tasks is delimited with commas. The default list represents all the possible values for this setting.

clustercontroller.zk_session_timeout_ms

Default value: 300000

The length of time, in milliseconds, that Cluster Controller will wait for the Coordination Service (ZooKeeper), before determining that failover is required.

dataAlerts.checkIntervalInMinutes

Default value: 15
The frequency, in minutes, at which Tableau Server checks to determine if data-alert conditions are true.

(The server also checks whenever extracts related to data alerts are refreshed.)

DataServerRefreshMetadataPerSession

Default value: false

Determines whether Tableau Server will make additional queries to get updated schema data for a published data source when there have been changes in the underlying schema structure. This is disabled by default for performance reasons, and there is a delay in the display of schema changes. If you want changes in the schema of a live published data source to be reflected quickly, or if you see errors (for example, "An error occurred while communicating with the data source: Invalid column name. Statement could not be prepared.") set this to true. When set to true, Tableau Server makes additional queries to update the schema.

features.AlertOnThresholdCondition

Default value: true

Controls whether data-drive alerts data-driven alerts are enabled for users on the server.

features.DesktopReporting

Default value: false

Controls whether Desktop License Reporting is enabled on the server. When set to false (the default), no Administrative Views related to desktop licenses are available. Set this to true to enable license reporting and make license usage and expiration Administrative Views visible on the Server Status page.

gateway.http.hsts

Default value: false
The HTTP Strict Transport Security (HSTS) header forces browsers to use HTTPS on the domain where it is enabled.

gateway.http.hsts_options

Default value: "max-age=31536000"

By default, HSTS policy is set for one year (31536000 seconds). This time period specifies the amount of time in which the browser will access the server over HTTPS.

gateway.http.request_size_limit

Default value: 16380

The maximum size (bytes) of header content that is allowed to pass through the Apache gateway on HTTP requests. Headers that exceed the value set on this option will result in browser errors, such as HTTP Error 413 (Request Entity Too Large) or authentication failures.

A low value for gateway.http.request_size_limit can result in authentication errors. Single sign-on solutions that integrate with Active Directory (SAML and Kerberos) often require large authentication tokens in HTTP headers. Be sure to test HTTP authentication scenarios before deploying into production.

We recommend setting tomcat.http.maxrequestsize option to the same value that you set for this option.

gateway.http.x_content_type_nosniff

Default value: true

The X-Content-Type-Options response HTTP header specifies that the MIME type in the Content-Type header should not be changed by the browser. In some cases, where MIME type is not specified, a browser may attempt to determine the MIME type by evaluating the characteristics of the payload. The browser will then display the content accordingly. This process is referred to as "sniffing." Misinterpreting the MIME
type can lead to security vulnerabilities. The X-Content-Type-Options HTTP header is set to 'nosniff' by default with this option.

gateway.http.x_xss_protection

Default value: true

The HTTP X-XSS-Protection response header is sent to the browser to enable cross-site scripting (XSS) protection. The X-XSS-Protection response header overrides configurations in cases where users have disabled XSS protection in the browser. The X-XSS-Protection response header is enabled by default with this option.

gateway.public.host

Default value: <hostname>

The name (URL) of the server, used for external access to Tableau Server. If Tableau Server is configured to work with a proxy server or external load balancer, it is the name entered in a browser address bar to reach Tableau Server. For example, if Tableau Server is reached by entering tableau.example.com, the name for gateway.public.host is tableau.example.com.

gateway.public.port

Default value: 80 (443 if SSL)

Applies to proxy server environments only. The external port the proxy server listens on.

gateway.slow_post_protection.enabled

Default value: false

Enabling this can provide some help in protecting against slow POST (Denial-of-Service) attacks by timing out POST requests that transfer data at extremely slow rates.
Note: This will not eliminate the threat of such attacks, and could have the unintended impact of terminating slow connections.

gateway.timeout

Default value: 1800

Longest amount of time, in seconds, that the gateway will wait for certain events before failing a request (1800 seconds = 30 minutes).

gateway.trusted

Default value: IP address of proxy server machine

Applies to proxy server environments only. The IP address(es) or host name(s) of the proxy server.

gateway.trusted_hosts

Default value: Alternate names of proxy server

Applies to proxy server environments only. Any alternate host name(s) for the proxy server.

install.firewall.allowedprograms.manage

Default value: true

Controls whether Tableau Server can add firewall rules. When set to true (the default), Tableau Server will add new firewall rules to allow its processes to make connections through Windows Firewall. Change this to false if you want to manage all firewall rules yourself and do not want Tableau Server to add new rules.

java.heap.size

Default value: 128m
Size of heap for Tomcat (repository and solr). This generally does not need to change except on advice from Tableau.

monitoring.dataengine.connection_timeout

Default value: 30000

The length of time, in milliseconds, that Cluster Controller will wait for the data engine, before determining that a connection timeout occurred. The default is 30,000 milliseconds (30 seconds).

native_api.connection.limit.<connection class>

Set parallel query limit for the specified data source (connection class). This overrides the global limit for the data source.

native_api.connection.globallimit

Default value: 16

Global limit for parallel queries. Default is 16 except for Amazon Redshift which has a default of 8.

native_api.ProtocolTransitionLegacyFormat

Default value: false

Use the legacy name format for constrained delegation.

The name format was changed in version 10.1 to allow cross-domain protocol transition (S4U). If this causes problems with existing configurations and you don't need cross-domain protocol transition, configure Tableau Server to use the old behavior by setting this to true.

features.PasswordReset

Default value: false
Applies only to servers that use local authentication. Set to true to let users reset their passwords with a "Forgot password" option on the sign-in page.

pgsql.port

Default value: 8060

Port that PostgreSQL listens on.

pgsql.preferred_host

Specifies the computer name or IP address of the node with the preferred repository installed. This value is used if the --preferred or -r option is specified with the tsm topology failover-repository command.

pgsql.verify_restore.port

Default value: 8061

Port used to verify the integrity of the PostgreSQL database. See tsm maintenance backup for more information.

recommendations.enabled

Default value: true

Suggests server content, such as data sources and tables, to Tableau Desktop users. Content suggestions are based on popularity of the content or on content frequently used by other users who are similar to the current user.

refresh_token.absolute_expiry_in_seconds

Default value: 31536000

Specifies the number of seconds for absolute expiry of OAuth refresh and access tokens. The OAuth tokens are used by clients for authentication to Tableau Server.
after initial sign-in. To remove limits set to -1. To disable OAuth tokens, see Disable Automatic Client Authentication.

refresh_token.idle_expiry_in_seconds

Default value: 1209600

Specifies the number of seconds when idle OAuth tokens will expire. The OAuth tokens are used by clients for authentication to Tableau Server after initial sign-in. To remove limits set to -1.

refresh_token.max_count_per_user

Default value: 24

Specifies the maximum number of refresh tokens that can be issued for each user. If user sessions are expiring more quickly than you expect, either increase this value or set it to -1 to entirely remove token limits.

rsync.timeout

Default value: 600

Longest allowable time, in seconds, for completing file synchronization (600 seconds = 10 minutes). File synchronization occurs as part of configuring high availability, or moving the data engine and repository processes.

schedules.display_schedule_description_as_name

Default value: false

Controls whether a schedule name displays when creating a subscription or extract refresh (the default), or the "schedule frequency description" name describing the time and frequency of the schedule displays. To configure Tableau Server to display timezone-sensitive names for schedules, set this value to true.
When true, the "schedule frequency description" is also displayed after the schedule name on the schedule list page.

`schedules.display_schedules_in_client_timezone`

**Default value:** `true`

Shows the "schedule frequency description" in the timezone of the user when true (uses the client browser timezone to calculate the "schedule frequency description").

`schedules.display_schedules_in_client_timezone`

**Default value:** `true`

`schedules.display_schedules_in_client_timezone`

**Default value:** `true`

Determines whether or not Tableau Server will automatically start when operating system of the computer Tableau Server is running on is restarted.

Valid options are **start** and **pause**.

Set this to **start** if Tableau Server should start on a restart of the computer.

**Valid options**

- start
- pause

**Default value:** `pause`

**Default value:** `false`

Setting to true enables JMX ports for optional monitoring and troubleshooting.

**Default value:** `false`

**Default value:** `<number>`

Maximum number of server processes.

**Default value:** `<number>`

**Default value:** `true`

**Default value:** `true`
Determines whether or not Tableau Server will attempt to dynamically remap ports when the default or configured ports are unavailable. Setting to \texttt{false} disables dynamic port remapping.

\texttt{session.ipsticky}

**Default value:** \texttt{false}

Makes client sessions valid only for the IP address that was used to sign in. If a request is made from an IP address different from that associated with the session token, the session token is considered invalid.

In certain circumstances—for example, when Tableau Server is being accessed by computers with known and static IP addresses—this setting can yield improved security.

**Note:** Consider carefully whether this setting will help your server security. This setting requires that the client have a unique IP address and an IP address that stays the same for the duration of the session. For example, different users who are behind a proxy might look like they have the same IP address (namely, the IP address of the proxy); in that case, one user might have access to another user’s session. In other circumstances, users might have a dynamic IP address, and their address might change during the course of the session. If so, the user has to sign in again.

\texttt{sheet\_image.enabled}

**Default value:** \texttt{true}

Controls whether you can get images for views with the REST API. For more information, see \texttt{REST API Reference}.

\texttt{solr\_rebuild\_index\_timeout}

**Default value:** 3600
When Tableau Server is upgraded or when a .tsbak file is restored, the background task rebuilds the search index. This setting controls the timeout setting for that task (3600 seconds = 60 minutes).

**subscriptions.enabled**

Default value: `false`

Controls whether subscriptions are configurable system-wide. See Set Up a Server for Subscriptions.

**subscriptions.timeout**

Default value: 1800

Longest allowable time, in seconds, for a single view in a workbook subscription task to be rendered before the task times out. This value applies separately to each view in the workbook, so the total length of time to render all the views in a workbook (the full subscription task) may exceed this timeout value. 1800 seconds = 30 minutes.

**tomcat.http.maxrequestsize**

Default value: 16380

The maximum size (bytes) of header content that is allowed to pass through the Apache gateway on HTTP requests. Headers that exceed the value set on this option will result in browser errors, such as HTTP Error 413 (Request Entity Too Large) or authentication failures.

A low value for `tomcat.http.maxrequestsize` may result in authentication errors. Single sign-on solutions that integrate with Active Directory (SAML and Kerberos) often require large authentication tokens in HTTP headers. Be sure to test HTTP authentication scenarios before deploying into production.

We recommend setting `gateway.http.request_size_limit` option to the same value that you set for this option.
tomcat.https.port

    Default value: 8443

    SSL port for Tomcat (unused).

tomcat.server.port

    Default value: 8085

    Port that tomcat listens on for shutdown messages.

vizportal.adsync.update_system_user

    Default value: false

    Specifies whether email addresses and display names of users are changed (even when changed in Active Directory) when an Active Directory group is synchronized in Tableau Server. To ensure that user email addresses and display names are updated during synchronization, set vizportal.adsync.update_system_user to true, and then restart the server.

vizportal.csv_user_mgmt.index_site_users

    Default value: true

    Specifies whether indexing of site users is done user by user when importing or deleting users with a CSV file. When set to true (the default) indexing is done as each user is added or deleted. To delay the indexing of the site users until after the entire CSV file has been processed, set this to false.

vizportal.log.level

    Default value: info
The logging level for vizportal Java components. Logs are written to /var/-
op/tableau/tableau_server/data/tabsvc/logs/vizportal/*.log.

Set to debug for more information. Using the debug setting can significantly impact performance, so you should only use this setting when directed to do so by Tableau Support.

vizportal.rest_api.cors.allow_origin

Specifies the origins (sites) that are allowed access to the REST API endpoints on Tableau Server when vizportal.rest_api.cors.enabled is set to true. You can specify more than one origin by separating each entry with a comma (,).

tsm configuration set vizportal.rest_api.cors.allow_origin

If vizportal.rest_api.cors.enabled is false, the origins listed by this option are ignored. For more information, see Enabling CORS on Tableau Server.

Note: You can use an asterisk (*) as a wild card to match all sites. This is not recommended as it allows access from any origin that has access to the server and can present a security risk. Do not use an asterisk (*) unless you fully understand the implications and risks for your site.

vizportal.rest_api.cors.enabled

Default value: false

Controls whether Tableau Server allows Cross Origin Resource Sharing (CORS). When set to true, the server allows web browsers to access the Tableau REST API endpoints. You can use this option and the REST API to create custom portals. By default, this functionality is not enabled. To specify which origins (sites) have access, use the vizportal.rest_api.cors.allow_origin option. Only the origins spe-
cified with this option are allowed to make requests to the Tableau Server REST API. For more information, see Enabling CORS on Tableau Server.

`vizportal.rest_api.view_image.max_age`

Default value: 720

The amount of time, in minutes, to cache images that are generated by the Query View Image method of the REST API. For more information, see the REST API Reference in the REST API help.

`vizqlserver.allow_insecure_scripts`

Default value: false

Allows a workbook to be published to the server from Tableau Desktop, and to be opened from the server, even if the workbook contains SQL or R expressions that are potentially unsafe (for example, a SQL expression that could potentially allow SQL injection). When this setting is false (the default), publishing a workbook or opening it from the server results in an error message, and the workbook is blocked. You should set this value to true only if you want to use workbooks that contain SQL or R expressions that have been detected as potentially unsafe, and only if the workbooks come from a safe source and you have verified that they do not contain an unsafe expression.

`vizqlserver.browser.render`

Default value: true

Views under the threshold set by `vizqlserver.browser.render_threshold` or `vizqlserver.browser.render_threshold_mobile` are rendered by the client web browser instead of by the server. See About Client-Side Rendering for details.

`vizqlserver.browser.render_threshold`

Default value: 100
The default value represents a high level of complexity for a view displayed on a PC. Complexity factors include number of marks, headers, reference lines, and annotations. Views that exceed this level of complexity are rendered by the server instead of in the PC's web browser.

**vizqlserver.browser.render_threshold_mobile**

Default value: 60

The default value represents a high level of complexity for a view displayed on a tablet. Complexity factors include number of marks, headers, reference lines, and annotations. Views that exceed this level of complexity are rendered by the server instead of in the tablet's web browser.

**vizqlserver.clear_session_on_unload**

Default value: false

Determines whether or not VizQL sessions are kept in memory when a user navigates away from a view or closes their browser. The default value (false) keeps sessions in memory. To close VizQL sessions on leaving a view or closing a browser, set this to true.

**vizqlserver.extsvc.connect_timeout_ms**

Default value: 1000

Extends the timeout value, in milliseconds, for connections to Microsoft's RServer. Raise the value of this setting if Tableau is timing out before the server can respond.

**vizqlserver.extsvc.host**

Specifies an external service host.
**Note:** In versions of Tableau before version 10.1, this setting was named `vizqlserver.rserve.host`. Be sure to use this earlier setting name if your Tableau version is older than version 10.1.

This setting, and the other `vizqlserver.extsvc` settings, support external service functionality in workbooks—in particular, R servers and Python servers.

R is an open source software programming language and a software environment for statistical computing and graphics. In Tableau Desktop, you can use a set of four functions to pass R expressions to an Rserve server and obtain a result. If you upload a workbook that uses any of these functions, you should configure Tableau Server for an Rserve connection, by configuring this option and the three following. Otherwise, any worksheets that use R functionality will be unavailable.

See [Pass Expressions to External Services](#) in the Tableau Help for further details.

`vizqlserver.extsvc.port`

Default value: 6311

Specifies an external service port. This setting supports R and Python functionality in workbooks.

**Note:** In versions of Tableau before version 10.1, this setting was named `vizqlserver.rserve.port`. Be sure to use this earlier setting name if your Tableau version is older than version 10.1.

`vizqlserver.extsvc.username`

Specifies an external service username. This setting supports R and Python functionality in workbooks. Not all Rserve hosts require a username and password.
**Note:** In versions of Tableau before version 10.1, this setting was named vizqlserver.rserve.username. Be sure to use this earlier setting name if your Tableau version is older than version 10.1.

**vizqlserver.extsvc.password**

Specifies an external service password. This setting supports R and Python functionality in workbooks. Not all Rserve hosts require a username and password.

**Note:** In versions of Tableau before version 10.1, this setting was named vizqlserver.rserve.password. Be sure to use this earlier setting name if your Tableau version is older than version 10.1.

**vizqlserver.geosearch_cache_size**

Default value: 5

Sets the maximum number of different geographic search locale/language data sets that can be loaded into server memory at the same time. When the server receives a geographic search request for locale/language data set that is not in memory, it will load the set into memory. If loading the data set will exceed the specified limit, the least recently used locale/language data set is cleared from memory so the requested one can be loaded. The minimum value is 1. Each cache takes approximately 60 MB in memory (so if you set this to 10, the memory usage would be 600 MB (60 * 10)).

**vizqlserver.log.level**

Default value: info

The logging level for vizqlserver Java components. Logs are written to /var/-opt/tableau/tableau_server/data/tabsvc/logs/vizqlserver/*.log.
Set to debug for more information. Using the debug setting can significantly impact performance, so you should only use it when directed to do so by Tableau Support.

vizqlserver.port

Default value: 9100

Base port for the VizQL servers.

vizqlserver.protect_sessions

Default value: true

When set to true, prevents VizQL sessions from being reused after the original user signs out.

vizqlserver.querylimit

Default value: 1800

Longest allowable time for updating a view, in seconds.

vizqlserver.session.expiry.minimum

Default value: 5

Number of minutes of idle time after which a VizQL session is eligible to be discarded if the VizQL process starts to run out of memory.

vizqlserver.session.expiry.timeout

Default value: 30

Number of minutes of idle time after which a VizQL session is discarded.

vizqlserver.showdownload

Default value: true
Controls the display of the **Tableau Workbook** option of the Download menu in views. When set to **false**, the Tableau Workbook option is unavailable.

`vizqlserver.showshare`

**Default value:** true

Controls the display of Share options in views. To hide these options, set to false.

**Note:** Users can override the server default by setting the "showShareOptions" JavaScript or URL parameter.

`vizqlserver.url_scheme_whitelist`

Specifies one or more URL schemes to whitelist when using URL actions on views and dashboards. The schemes http, https, gopher, mailto, news, sms, tel, tsc, and tsl are whitelisted by default. This command can contain multiple comma and space-separated values, as in this example:

```
tsm configuration set vizqlserver.url_scheme_whitelist scheme1, scheme2
```

The values you specify overwrite previous settings. Therefore, you must include the full list of schemes in the set command. (You cannot amend the list of schemes by running the set command repeatedly.)

`webdataconnector.refresh.enabled`

Determines whether extract refreshes for web data connectors (WDCs) are enabled in Tableau Server. To disable refresh for all WDCs, set the value for this key to **false**, as shown below:

```
tsm configuration set --key webdataconnector.refresh.enabled --value false
```
To learn more, see Web Data Connectors in Tableau Server.

webdataconnector.whitelist.fixed

Specifies one or more web data connectors (WDCs) that can be used by to access data connections that are accessible over HTTP or HTTPS. This command is formatted as JSON data on a single line, with all double-quotes (") escaped using a back-slash (\).

For example to add a San Francisco Film Locations WDC to the safe list:

```
$ tsm configuration set --key webdataconnector.whitelist.fixed --value '"{"https://tableau.data.world:443": {"properties": {"secondary_whitelist": ["https://data.world/"]} }}"'
```

To learn more, see Web Data Connectors in Tableau Server.

webdataconnector.enabled

Default value: true

When set to true, you can use tsm commands to manage web data connectors on the server.

webdataconnector.whitelist.mode

Default value: mixed

Determines how Tableau Server can run web data connectors. Supported modes are:

- **fixed.** Users can run connectors that are on a safe list (whitelist) of URLs.
- **insecure.** Users can run any connector.
Important: Use the insecure option only for development and testing. Because connectors run custom code, running connectors that have not been vetted can pose a security threat.

wgserver.audit_history_expiration_days

Default value: 183

Specifies the number of days after which historical events records are removed from the PostgreSQL database (the Tableau Server database).

wgserver.change_owner.enabled

Default value: true

Controls whether the ownership of a workbook, data source or project can be changed. Other options include false and adminonly.

wgserver.clickjack_defense.enabled

Default value: true

When set to true, helps prevents a malicious person from "clickjacking" a Tableau Server user. In a clickjack attack, the target page is displayed transparently over a second page, and the attacker gets the user to click or enter information in the target page while the user thinks he or she is interacting with the second page.

For more information, see Clickjack Protection.

wgserver.domain.fqdn

Default value: value of %USERDOMAIN%

The fully qualified domain name of the Active Directory server to use.
wgserver.extended_trusted_ip_checking

Default value: false

Enforces IP client matching for trusted ticket requests.

wgserver.restrict_options_method

Default value: true

Controls whether Tableau Server accepts HTTP OPTIONS requests. If this option is set to true, the server returns HTTP 405 (Method Not Allowed) for HTTP OPTIONS requests.

wgserver.session.apply_lifetime_limit

Default value: false

Controls whether there is a session lifetime for server sessions. Set this to true to configure a server session lifetime.

wgserver.session.lifetime_limit

Default value: 1440

The number of minutes a server session lasts if a session lifetime is set. The default is 1440 minutes (24 hours). If wgserver.session.apply_lifetime_limit is false (the default) this is ignored.

wgserver.session.idle_limit

Default value: 240

The number of minutes of idle time before a sign-in to the web application times out.

wgserver.unrestricted_ticket
Default value: false

Specifies whether to extend access to server resources for users authenticated by trusted tickets. Default behavior allows users to access views only. Setting this to true allows users with valid trusted tickets to access server resources (projects, workbooks, and so on) as if they had signed in using their credentials.

workerX.gateway.port

Default value: 80 (443 if SSL)

External port that Apache listens on for workerX (where a “worker” is the term used for subsequent server nodes in the cluster). worker0.gateway.port is Tableau Server’s external port. In a distributed environment, worker0 is the initial Tableau Server node.

workerX.vizqlserver.procs

Default value: <number>

Number of VizQL servers.

zookeeper.config.dataLogDir

Specifies the directory and file path for ZooKeeper transaction logs. By default ZooKeeper transaction logs are written to the Tableau data directory (for example /var/opt/tableau/tableau_server/data). Use this option to specify a different location.

The drive and path apply to all nodes in a cluster. The location will be created if it does not exist. The drive must exist and be writable on all nodes. This should not be a UNC path to a share.

ZooKeeper recommends that transaction logs be written to a dedicated drive to optimize performance.

Example: tsm configuration set zookeeper.config.dataLogDir "/zookeeper"
You can use the `tsm customize` command to customize the look and feel of the Tableau Server client browser experience.

A - The Tableau logo for the browser window tab cannot be changed.

B - The server name can be changed using the `--server-name` option.

C - The header logo can be changed using the `--header-logo` option.

The image files you use should be in GIF, JPEG, or PNG format.

The header logo image can be up to 160 by 160 pixels, but not smaller than 32 by 32 pixels. For best results use an image that is 125 by 35 pixels. If the image is larger than 160 by 160 pixels, it will be clipped.

You can also add a logo to the user sign in page for Tableau Server.

The sign in logo image can be a maximum of 3000 by 3000 pixels.
The background colors on the header and sign in page are not the same. Your logo might look different depending on where it appears in the server interface.

**Note:** If you are adding image files and if you are running the Gateway process on more than one node in a Tableau Server distributed deployment, then you will need to copy the image file to each of those nodes. See Portal customization.

**Synopsis**

```bash
tsm customize [options] [global options]
```

**Options**

```
--header-logo </path/to/logo>
```

Optional.

Specify a path to the image file that will be displayed in the header only.

```
--logo </path/to/logo>
```

Optional.

Path to a single image file that will display for both the header and the sign-in window.

```
--restore-defaults
```

Optional.

Reset all customization options to default install state.

```
--server-name <server_name>
```

Optional.

Server name that appears in the browser tab, tooltips, and messages.
--signin-logo /path/to/logo

Optional.

Specify a path to the image file that will be displayed for sign-in window only.

Global options

-h, --help

Optional.

Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.
Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

**tsm data-access**

You can use the `tsm data-access` commands to enable or disable data repository access and enable SAML for single sign-on.

- `repository-access enable`
- `repository-access disable`
- `repository-access list`
- `set-saml-delegation`

**tsm data-access repository-access enable**

Enables access to the Tableau PostgreSQL database.

**Synopsis**

```
  tsm data-access repository-access enable [options]
  [global options]
```

**Options**

```
--repository-password <password>
```

Required.

The password to access the data repository.

```
--repository-username <username>
```

Required.

The username, either `tableau` or `readonly`, with access to the data repository.
-r, --restart

Optional.

Suppress the prompt for restart and restart Tableau Server.

tsm data-access repository-access disable

Disable external access to the Tableau PostgreSQL database for the default remote user.

Synopsis

tsm data-access repository-access disable [options]
[global options]

Options

--repository-username <username>

Required.

The username, either tableau or readonly, with access to the data repository.

-r, --restart

Optional.

Suppress the prompt for restart and restart Tableau Server.

tsm data-access repository-access list

Lists users who have access to the Tableau PostgreSQL database.

Synopsis

tsm data-access repository-access list [global options]
tsm data-access set-saml-delegation

Setup single sign-on for SAML SAP HANA so that Tableau Server functions as an Identity Provider (IdP) that provides single sign-on for users making SAP HANA data connections.

Synopsis

tsm data-access set-saml-delegation [options] [global options]

Options

- kf, --cert-key <cert-key>
  Optional.
  The SAML certificate key file.

- cf, --cert-file <file-path>
  Optional.
  The location of the SAML certificate file.

- uf, --username-format <username-format>
  Optional.
  Username format using valid format keys.

- e, --enabled <enabled>
  Optional.
  Enables SAML for single sign-on.

- uc, --username-case <username-case>
  Optional.
Username case using valid case keys.

Global options

-h, --help

Optional.

Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.
You can use the `tsm initialize` command to initialize Tableau Server.

**Note:** You must apply or discard pending changes before running `tsm initialize` or the initialize will fail. Apply pending changes using the `tsm pending-changes apply` command. Discard any pending changes you do not want to apply using `tsm pending-changes discard`.

### Synopsis

```
  tsm initialize [options] [global options]
```

### Options

- `-r, --start-server`
  
  Optional. Leave the server running after initialization is complete.

### Global options

- `-h,  --help`
  
  Optional.
  
  Show the command help.

- `-p,  --password <password>`
  
  Required, along with `-u or  --username` if no session is active.
  
  Specify the password for the user specified in `-u or  --username`.

- `--request-timeout <timeout in seconds>`
  
  Optional.
Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>
Optional.
Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>
Required if no session is active, along with -p or --password.
Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

ts m jobs
You can use the tsm jobs commands to list, reconnect to, and cancel jobs.

- cancel
- list
- reconnect

ts m jobs cancel
Cancel a job on the server.
Synopsis
ts m jobs cancel --id <jobID> [global options]
Options

-i,--id <jobID>
Required.
Id of the job to cancel.

tsm jobs list
List asynchronous jobs on the server.

Synopsis
tsm jobs list [--status <status>] [global options]

Options
-t,--status <status>
Optional.
Filter for jobs that match the given status.

tsm jobs reconnect
Reconnect to an asynchronous job to display its progress. If no job id is specified, it reconnects to the latest job.

Synopsis
tsm jobs reconnect[--id <jobID>] [global options]

Options
-i,--id <jobID>
Optional.
Specifies the id of the job that should be reconnected.
Global options

-h, --help

Optional.
Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.
Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.
Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.
Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.
Specify a user account. If you do not include this option, the command is run using credentials you signed in with.
You can use the `tsm licenses` commands to manage server license tasks like activating or deactivating a Tableau Server product key on- or off-line, and getting associated files for offline activation or deactivation.

- activate
- deactivate
- get-offline-activation-file
- get-offline-deactivation-file
- list
- refresh

### tsm licenses activate

Activates a Tableau Server product key.

**Synopsis**

```
tsm licenses activate --license-file <file.tlf>  
[global options]
```

**Options**

- `-f, --license-file <file.tlf>`
  
  Required if doing offline activation.

  Specifies the license file (`<file>.tlf`) used for offline activation.

- `-k, --license-key <product_key>`
  
  Required if activating a valid product key.

  Specifies the product key to use for online activation.
-t, --trial

Required if activating a trial license.

Activate a trial license.

tsm licenses deactivate

Deactivates a Tableau Server product key either online or offline.

Synopsis

tsm licenses deactivate --license-file <return_file.tlr>
[global options]

Options

-f, --license-file <return_file.tlr>

Required if doing offline deactivation.

Specifies the license file (<file>.tlf) used for offline deactivation.

-k, --license-key <product_key>

Required if deactivating a product key.

Specifies the product key to use for online deactivation.

tsm licenses get-offline-activation-file

Generate an offline activate file to use for activating Tableau Server offline.

Synopsis

tsm licenses get-offline-activation-file --license-key
<product_key> --output-dir <path> [global options]
Options

-k, --license-key <product_key>

Required.

Specifies the product key to use for offline activation.

-o, --output-dir <path>

Required.

The location where the offline activation file should be saved. This location must exist.

tsm licenses get-offline-deactivation-file

Generate an offline deactivation file to use for deactivating Tableau Server offline.

Synopsis

    tsm licenses get-offline-deactivation-file --license-key <product_key> --output-dir <path> [global options]

Options

-k, --license-key <product_key>

Required.

Specifies the product key to use for offline deactivation.

-o, --output-dir <path>

Required.

The existing location where the offline deactivation file should be saved.
tsm licenses list

List licenses on Tableau Server. The output of this command varies, depending on the license type you are using. This command provides license keys and expiration dates, but does not provide license quotas. For more information about the output of this command, see View Server Licenses.

Synopsis

tsm licenses list [global options]

tsm licenses refresh

Update the maintenance expiration date of all product keys on Tableau Server.

Synopsis

tsm licenses refresh [global options]

Global options

-h, --help

Optional.

Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.
Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

tsm login

You can use the tsm login command to log in to Tableau Server.

Synopsis

tsm login [global options]

Global options

-h, --help

Optional.

Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.
--request-timeout <timeout in seconds>

Optional.
Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.
Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.
Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

tsm logout

You can use the tsm logout command to log out of Tableau Server.

Synopsis

 tsm logout [global options]

Global options

-h, --help

Optional.
Show the command help.
-p, --password <password>

Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

tsm maintenance

You can use the tsm maintenance commands to manage server maintenance tasks like creating regular backups or restoring Tableau Server from a previously created backup.

- backup
- cleanup
tsm maintenance backup

Creates a backup of the data managed by Tableau Server. This data includes the Tableau PostgreSQL database (the repository) which contains workbook and user metadata, extract (.tde or .hyper) files, and server configuration data. See Backup and Restore.

You can use a backup of Tableau Server on Linux to restore Tableau Server on Linux but you cannot use it to restore Tableau Server on Windows.

Synopsis

tsm maintenance backup --file <backup_file> [options]
[global options]

Options

-d, --append-date

Optional.

Append the current date to the backup file name.

-f, --file <backup_file>

Required.

Write the backup to the specified file. An extension of .tsbak is automatically used. The file is written to the directory defined in the TSM basefilepath.backuprestore variable. By default:
/var/opt/tableau/tableau_server/data/tabsvc/files/backups/

For more information about file paths and how to change them, see tsm File Paths.

-i, --description <string>
  Optional.
  Include the specified description of the backup file.

-k, --skip-verification
  Optional.
  Do not verify the integrity of the database backup.

--override-disk-space-check
  Optional.
  Attempt to create a backup even when there is a low disk space warning.

Examples

This example creates a backup called ts_backup-<yyyy-mm-dd>.tsbak in the /var/-opt/tableau/tableau_server/data/tabsvc/files/backups/ directory:

```bash
tsm maintenance backup -f ts_backup -d
```

tsm maintenance cleanup

Deletes old log files and temporary files.

**Note:** This command was added in Tableau Server version 10.5.1.

Synopsis

```bash
tsm maintenance cleanup [options] [global options]
```
Options

-a, --all

Optional.

Perform all cleanup operations with default retention values. Equivalent to running the cleanup command with the following options: -l -t.

-l, --log-files

Optional.

Delete log files that are older than the retention-period. Files in the subdirectories under data/tabsvc/logs will be deleted.

--log-files-retention <# of days>

Optional.

Default: 1 (24 hours)

Delete logs older than this number of days. This command does not apply to temporary files.

-t, --temp-files

Optional.

Delete all files and subdirectories in the following directories:

- /var/opt/tableau/tableau_server/data/tabsvc/temp: Only directories that are storing files for expired (not running) sessions are deleted.
- /var/opt/tableau/tableau_server/data/tabsvc/httpd/temp
- /var/opt/tableau/tableau_server/temp
-v, --verbose

Optional.

Show details of what was cleaned.

Examples

This example cleans up all log files older than 2 days old:

```
tsm maintenance cleanup -l --log-files-retention 2
```

```
tsm maintenance reindex-search
```

Use the reindex-search command to rebuild the search index.

Synopsis

```
tsm maintenance reindex-search [global options]
```

```
tsm maintenance restore
```

Restore Tableau Server using the specified backup file. See Backup and Restore.

You can only restore from a backup that has the same type of identity store as the running server. For example, a backup from a server using local authentication can be restored to a Tableau Server initialized with local authentication, but a backup from a server using Active Directory authentication cannot be restored to a server initialized with local authentication.

Synopsis

```
tsm maintenance restore --file <file_name> [--restart-server] [global options]
```

Options

```
-f, --file <file_name>
```
Required.

Specifies the backup file to restore from.

The restore command expects a backup file in the directory defined in the TSM basefilepath.backuprestore variable. By default:

/var/opt/tableau/tableau_server/data/tabsvc/files/backups/

For more information about file paths and how to change them, see tsm File Paths.

-r, --restart-server

Optional.

Restart the server after the restore.

tsm maintenance send-logs

Upload the specified file to Tableau and associate it with a support case.

Synopsis

tsm maintenance send-logs --case <case_number> --email <contact_email> --file <filename> [global options]

Option

-c,--case <case_number>

Required.

Support case number.

-e,--email <contact_email>

Required.
Contact email.

-f, --file <filename>

Required.

Specifies the log file archive to send.

tsm maintenance validate-resources

Validate workbooks and data sources for a site.

Synopsis

tsm maintenance validate-resources --site-id <site ID>
[global options]

Option

-id, --site-id <site ID>

Required.

ID for the site whose resources you are validating.

-r, --repair

Optional.

Repair invalid resources.

tsm maintenance ziplogs

Use the ziplogs command to create an archive of Tableau Server log files.
Note: If you cannot run the ziplogs command successfully, you can manually zip the Tableau Server logs. For more information, see Manually gather logs.

Synopsis

```bash
tsm maintenance ziplogs [options] [global options]
```

Options

-`-a, --all`
  
  Optional.
  
  Include all files except PostgreSQL data.

-`-d, --with-postgresql-data`
  
  Optional.
  
  Include the PostgreSQL data folder if Tableau Server is stopped or PostgreSQL dump files if Tableau Server is running.

-`-f, --file <name>`
  
  Optional.
  
  Specify a name for the zipped archive file. If no name is provided the archive is created as `logs.zip`. The file is written to the directory defined in the TSM `basefilepath.log_archive` variable. By default:

  ```bash
  /var/opt/tableau/tableau_server/data/tabsvc/files/log-archives/
  ```
  
  For more information about file paths and how to change them, see tsm File Paths.

-`-i, --description <string>`
  
  Optional.
Include the specified description of the archive file.

-\(l\), --\(with\)-latest-dump

Optional.

Include latest dumps.

-\(m\), --\(minimum\)date <date>

Optional.

Earliest date of files to be included. If not specified, a maximum of two days of data are included. Format of date should be "mm/dd/yyyy".

-\(o\), --\(over\)write

Optional.

For an overwrite of an existing ziplog file. If a file by the same name already exists and this option is not used, the ziplogs command will fail.

By default the backup file is written to the /var/opt/tableau/tableau_server-data/tabsvc/files/log-archives/directory.

For more information about file paths and how to change them, see tsm File Paths.

-\(t\), --\(with\)-netstat-info

Optional.

Include netstat information.

Global options

-\(h\), --\(hel\)p
Optional.
Show the command help.

-p, --password <password>
Required, along with -u or --username if no session is active.
Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>
Optional.
Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>
Optional.
Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>
Required if no session is active, along with -p or --password.
Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

**tsm pending-changes**

*You can use the tsm pending-changes commands to apply, discard, or view pending configuration and topology changes to Tableau Server.*
You can use the `tsm pending-changes apply` command to apply pending configuration and topology changes to Tableau Server.

A prompt warns you that the command will restart Tableau Server. If Tableau Server is currently running it is restarted, and if it is stopped it is left in a stopped state with no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior.

**Synopsis**

`tsm pending-changes apply [global options]`

**Options**

- `-iw`, `--ignore-warnings`
  
  Optional.

  Ignore warning level constraints.

- `-r`, `--restart`
  
  Optional.

  Suppress the prompt for restart. This only suppresses the prompt. The restart behavior is unchanged.

You can use the `tsm pending-changes discard` command to discard pending configuration and topology changes to Tableau Server.
Synopsis

```
tsm pending-changes discard [options] [global options]
```

Options

```
--config-only

Optional.

Discard only pending configuration changes.
```

```
--topology-only

Optional.

Discard only pending topology changes.
```

### tsm pending-changes list

You can list pending configuration and topology changes to Tableau Server.

Synopsis

```
tsm pending-changes list [options] [global options]
```

Options

```
--config-only

Optional.

List only pending configuration changes.
```

```
--topology-only

Optional.

List only pending topology changes.
```
Global options

-h, --help

Optional.
Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.
Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.
Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.
Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.
Specify a user account. If you do not include this option, the command is run using credentials you signed in with.
**tsm register**

You can use the `tsm register` command to register Tableau Server. This command lets you either display the registration file template (using the `--template` option) or provide the path to a completed registration file (using the `--file` option). You must use one of these two options when calling the `tsm register` command.

**Synopsis**

```
tsm register --template | --file <registration-filename> [global options]
```

**Options**

`--file <registration-filename>`

Required.

Path to the file that contains the registration data.

`--template`

Required.

Display registration file template.

**Global options**

`-h, --help`

Optional.

Show the command help.

`-p, --password <password>`

Required, along with `-u` or `--username` if no session is active.
Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>
  Optional.
  Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>
  Optional.
  Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>
  Required if no session is active, along with -p or --password.
  Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

tsm restart
  You can use the tsm restart command to restart Tableau Server.

Synopsis

tsm restart [global options]

Global options

-h, --help
  Optional.
  Show the command help.
-p, --password <password>

Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

**tsm security**

Use the tsm security commands to configure Tableau Server support for external (gateway) SSL or repository (Postgres) SSL. Repository SSL configuration includes the option to enable SSL over direct connections from Tableau clients—including Tableau Desktop, Tableau Mobile, and web browsers—to the repository.
• tsm security external-ssl enable
• tsm security external-ssl disable
• tsm security external-ssl list
• tsm security repository-ssl enable
• tsm security repository-ssl disable
• tsm security repository-ssl get-certificate-file
• tsm security regenerate-internal-tokens
• tsm security repository-ssl list
• tsm security validate-asset-keys

Prerequisites

Before you configure SSL, you must acquire certificates, and then copy them to the computer that runs the Tableau Server gateway process. Additional preparation is required for enabling direct connections from clients. To learn more, see the following articles:

Configure SSL for External HTTP Traffic to and from Tableau Server (Linux)

Configure SSL for Internal Postgres Communication

Note: For information about mutual (two-way) SSL, see Configure Mutual SSL Authentication for Tableau Server on Linux and tsm authentication mutual-ssl commands.

tsm security external-ssl enable

Enable and specify certificate and key files for SSL over external HTTP communication.

Synopsis

tsm security external-ssl enable --cert-file <file.crt> --key-file <file.key> [options] [global options]
Options

--cert-file <file.crt>

Required. Specify the name of a valid PEM-encoded x509 certificate with the extension .crt.

--key-file <file.key>

Required. Specify a valid RSA or DSA private key file, with the extension .key by convention.

--chain-file <chainfile.crt>

Specify the certificate chain file (.crt)

Optional for Tableau Server, required if your deployment includes Tableau Mobile and Tableau Desktop on the Mac.

Some certificate providers issue two certificates for Apache. The second certificate is a chain file, which is a concatenation of all the certificates that form the certificate chain for the server certificate.

All certificates in the file must be x509 PEM-encoded and the file must have a .crt extension (not .pem).

--passphrase

Optional. Passphrase for the certificate file. The passphrase you enter will be encrypted while at rest.

**Note:** If you create a certificate key file with a passphrase, you cannot reuse the SSL certificate key for SAML.

--protocols <list protocols>

Optional. List the Transport Layer Security (TLS) protocol versions you want to allow
or disallow.

TLS is an improved version of SSL. Tableau Server uses TLS to authenticate and encrypt connections. Accepted values include protocol versions supported by Apache. To disallow a protocol, prepend the protocol version with a minus (-) character.

Default setting: "all, -SSLv2, -SSLv3"

This default explicitly does not allow clients to use SSL v2 or SSL v3 protocols to connect to Tableau Server. However, we recommend that you also disallow TLS v1 and TLS v1.1.

Before you deny a specific version of TLS, verify that the browsers from which your users connect to Tableau Server support TLS v1.2. You might need to preserve support for TLSv1.1 until browsers are updated.

If you do not need to support TLS v1 or v1.1, use the following command to allow TLS v1.2 (using the value all), and explicitly deny SSL v2, SSL v3, TLS v1, and TLS v1.1.

```bash
tsma security external-ssl enable --certfile file.crt --keyfile file.key --protocols "all -SSLv2 -SSLv3 -TLSv1 -TLSv1.1"
```

**tsm security external-ssl disable**

Removes the server’s existing SSL configuration settings and stops encrypting traffic between external clients and the server.

**Synopsis**

```bash
tsma security external-ssl disable [global options]
```

**tsm security external-ssl list**

Displays a list of settings related to the configuration of gateway external SSL. The list includes the names of the certificate files in use, but not their location.

**Synopsis**

```bash
tsma security external-ssl list [global options]
```
tsm security repository-ssl enable

Enables SSL and generates the server’s .crt and .key files used for encrypted traffic between the Postgres repository and other server components. Enabling this also gives you the option to enable SSL over direct connections from Tableau clients to the server.

Synopsis

tsm security repository-ssl enable [options] [global options]

Options

-i, --internal-only

Optional. When set to --internal-only, Tableau Server uses SSL between the repository and other server components, and it supports but does not require SSL for direct connections through tableau or readonly users.

If this option is not set, Tableau Server requires SSL for traffic between the repository and other server components, as well as for direct connections from Tableau clients (for connections through the tableau or readonly users).

When you specify this value, you must also complete the steps described in Configure Postgres SSL to Allow Direct Connections from Clients.

tsm security repository-ssl disable

Stop encrypting traffic between the repository and other server components, and stop support for direct connections from Tableau clients.

Synopsis

tsm security repository-ssl disable [global-options]

tsm security repository-ssl get-certificate-file

Get the public certificate file used for SSL communication with the Tableau repository. SSL must be enabled for repository communication before you can retrieve a certificate. The
certificate file is distributed automatically to internal clients of the repository in the Tableau Server cluster. To enable remote clients to connect over SSL to the repository, you must copy the public certificate file to each client.

Synopsis

tsm security repository-ssl get-certificate-file [global-options]

Options

-f, --file

Required.

Full path and file name (with .cert extension) where the certificate file should be saved. If a duplicate file exists it will be overwritten.

tsm security regenerate-internal-tokens

Stops the server, regenerates a new tokens, keys, and secrets. Also creates new security tokens that Tableau Server uses internally. Then starts the server.

The security tokens include the passwords used by Tableau Server to access the repository, and the certificates used to validate internal SSL connections between Tableau Server components and the repository.

Synopsis

tsm security regenerate-internal-tokens [options] [global options]

Options

-r, --restart

Optional.
Suppress the prompt that notifies you that running `regenerate-internal-tokens` will stop Tableau Server and then start it after regenerating the key and tokens.

Regenerate tokens and apply changes on a single server

- Run the following commands:

  ```
  tsm security regenerate-internal-tokens
  tsm pending-changes apply
  ```

  Like the `regenerate-internal-tokens` command, the `pending-changes-apply` command displays a prompt to let you know it will restart Tableau Server if the server is running. You can use the `-r` option with this command as well. Although `pending-changes-apply` generally does not start the server if it is stopped, the `regenerate-internal-tokens` always stops and starts the server as part of its process.

Regenerate tokens and apply changes in a Tableau Server cluster

1. Run the commands described in the single-server section above, to regenerate tokens and apply changes.

   The `regenerate-internal-tokens` command copies the required certificate file to the same location on each other node.

2. Stop the server, and then use a secure process to copy the master key to each node in the cluster:
   a. Run the following command:
      ```
      tsm stop
      ```
   b. Use a secure process to copy the file `tableauserver.jks` to the `keystores` directory on each node. The default location is as follows:
3. Run the following command to start the server:

   tsm start

**tsm security repository-ssl list**

Returns the existing repository (Postgres) SSL configuration.

**Synopsis**

```bash
tsm security repository-ssl list [global-options]
```

**tsm security validate-asset-keys**

Confirms whether the asset keys in use are valid and present.

**Synopsis**

```bash
tsm security external-ssl validate-asset-keys [global options]
```

**Global options**

- `-h, --help`
  
  Optional.
  
  Show the command help.

- `-p, --password <password>`
  
  Required, along with `-u or --username` if no session is active.
  
  Specify the password for the user specified in `-u or --username`.

- `--request-timeout <timeout in seconds>`
Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

tsm settings

You can use the tsm settings commands to set and update configuration values.

- export
- import

Important: The server configuration file referenced in this topic includes a copy of the master keystore file used for encrypting configuration secrets. We strongly recommend that you take additional measures to secure the node configuration file, using mechanisms as described in Securing secrets for import and export operations.

tsm settings export

Export the current server configuration and topology to a file.
Synopsis

```
tsm settings export [--output-config-file path/to/file.json]
[global options]
```

Options

```
-f, --output-config-file <file>
```

Required.

Specifies the location and name of the file created by this operation.

**tsm settings import**

Import server configuration or topology.

Synopsis

```
tsm settings import [--output-config-file path/to/file.json]
[global options]
```

Options

```
-f, --import-config-file <FILE>
```

Required.

Path to input file.

```
--config-only
```

Optional.

```
--topology-only
```

Optional.
Global options

-h, --help

Optional.
Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.
Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.
Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.
Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.
Specify a user account. If you do not include this option, the command is run using credentials you signed in with.
You can use the tsm sites commands to export an existing site for import to a new site (also referred to as site migration), and to import the new site. An unlock command is available in case an error leaves a site locked.

The tsm sites commands will use your local file store to hold the export and import data. If you are running a multinode Tableau cluster, then you must run the tsm sites commands on a Tableau Server node that is running the Data Engine process. For information about the Data Engine process and the processes that require it, see Process Reference.

For comprehensive steps for migrating a site, see Export or Import a Site.

- export
- import
- import-verified
- unlock

### tsm sites export

Export a specified Tableau Server site to a .zip file. You can export a site to archive its settings at a specific point in time, or to complete the first step of a site migration process.

**Note:** The tsm sites import and tsm sites export commands can leave a site in a locked state if an error occurs. To unlock a site, use the tsm sites unlock command.

**Synopsis**

```
  tsm sites export --site-id <source-siteID> --file <export-file> [options] [global options]
```
Options

-f, --file <export-file>

Required.

Specify the name of the file to which Tableau Server saves all of the site’s information.

This file is generated to the directory defined in the TSM basefilepath.site_export.exports variable. By default:

/var/opt/tableau/tableau_server/data/tabsvc/files/siteexports

For more information about file paths and how to change them, see tsm File Paths.

-id, --site-id <source-siteID>

Required.

The site ID for the site you are exporting. You can get the site ID from the URL when you’re signed in to the site from a web browser. For information about locating the site ID, see Prepare the source and target sites.

-ow, --overwrite

Optional.

Overwrite an export file of the same name that already exists.

tsm sites import

This command uses the .zip file you created using tsm sites export to generate a set of .csv files that show how the exported source site settings will map to the new target site.

Before you use this command, you must copy the .zip file to the directory in which Tableau will expect it. This location is defined in the TSM basefilepath.site_import.-exports variable. By default:

/var/opt/tableau/tableau_server/data/tabsvc/files/siteimports
The .zip file is generated to the exports directory. By default: /var/.../-files/siteexports.

For more information about file paths and how to change them, see tsm File Paths.

Note: The tsm sites import and tsm sites export commands can leave a site in a locked state if an error occurs. To unlock a site, use the tsm sites unlock command.

Synopsis

```
  tsm sites import --file <export-file.zip> --site-id <target-siteID> [options] [global options]
```

Options

- **-f,--file <export-file.zip>**
  
  Required.

  Name of the .zip file created by the tsm sites export process, and which you must copy to the import directory (by default, /var/opt/tableau/tableau_server-/data/tabsvc/files/siteimports).

- **-id,--site-id <target-siteID>**
  
  Required.

  The site ID for the new site you are importing to (the target site). For information about locating the site ID, see Prepare the source and target sites.

- **-k,--no-verify**
  
  Optional.

  Skip verification of mapping files.
-m, --override-schedule-mapper <mapping-file.csv>

Optional.

Schedule mapping file to override the normal mapping by name.

tsm sites import-verified

Specify the directory that contains an exported site’s .csv mapping files, to import to a new site. This is the final step of a site migration process.

Synopsis

tsm sites import-verified --import-job-dir <importjob-directory> --site-id <target-siteID> [options] [global options]

Options

-id, --site-id <target-siteID>

Required.

The site ID for the new site you are importing to (the target site). For information about locating the site ID, see Prepare the source and target sites.

-w, --import-job-dir <importjob-directory>

Required.

The parent of the mappings directory that contains the .csv files from the exported (source) site. The name of this parent directory includes the import id and date and time. For example:

/var/opt/tableau/tableau_server-
data/tabsvc/files/siteimports/working/import_ff00_20180102022014457

tsm sites unlock

Use this command to unlock a site.
Options

-id, --site-id <target-siteID>

Required.

The site ID for the site you are unlocking. For information about locating the site ID, see Prepare the source and target sites.

-d, --desired-state <state to leave unlocked site in>

Optional.

The state the site should be left in after it is unlocked. Options are "active" and "suspended". The default is "active" if not specified.

For example:

```
mtm sites unlock -id mysite -d suspended
```

Global options

-h, --help

Optional.

Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.
Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>
Optional.
Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>
Required if no session is active, along with -p or --password.
Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

**tsm start**
You can use the `tsm start` command to start Tableau Server.

**Synopsis**

```
tsm start [global options]
```

**Global options**

-h, --help
Optional.
Show the command help.

-p, --password <password>
Required, along with -u or --username if no session is active.
Specify the password for the user specified in -u or --username.
--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

**tsm status**

You can use the tsm status command to display the status of Tableau Server.

Synopsis

```sh
tsm status [global options]
```

Options

-v, --verbose

Optional.

Display status for every node in the Tableau Server cluster.

**tsm status** will return one of four potential statuses for a Tableau Server node:
- RUNNING: The node is running without error statuses for any service.
- DEGRADED: A primary service - such as the repository - is in an error state.
- ERROR: One or more services is in an error state.
- STOPPED: The node is stopped.

When running `tsm status` with the `--verbose` option, TSM will return a status for each individual service. There are four status messages:

- is running: The service is running.
- status is unavailable: The status is not returned - such as when processes are starting up.
- is in an error state: The process is running, but returning errors. This status indicates the process failed to install properly, or has not been configured.
- is stopped: The process is stopped.

Global options

- `-h, --help`
  Optional.
  Show the command help.

- `-p, --password <password>`
  Required, along with `-u` or `--username` if no session is active.
  Specify the password for the user specified in `-u` or `--username`.

- `--request-timeout <timeout in seconds>`
  Optional.
Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

tsm stop

You can use the tsm stop command to stop Tableau Server.

Synopsis

tsm stop [global options]

Global options

-h, --help

Optional.

Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.
--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

tsm topology

You can use the tsm topology commands to prepare File Store nodes for safe removal or to put them back into read-write mode. You can also initiate a repository failover, get a list of nodes or ports, get the bootstrap configuration file required to add additional nodes to your cluster, and remove nodes.

- cleanup-coordination-service
- deploy-coordination-service
- failover-repository
- filestore
  - decommission
  - recommission
tsm topology cleanup-coordination-service

Use the `tsm topology cleanup-coordination-service` command to remove the non-production Tableau Server Coordination Service ensemble after you deploy a new ensemble. This command removes the old Coordination Service instances on all nodes in the non-production Coordination Service ensemble and is required after you deploy a new Coordination Service ensemble. To learn more about Coordination Service ensembles, see Deploy a Coordination Service Ensemble.

Synopsis

```
rm topology cleanup-coordination-service [global options]
```

**tsm topology deploy-coordination-service**

You can use the `tsm topology deploy-coordination-service` command to deploy the Tableau Server Coordination Service. This command deploys a Coordination Service ensemble, which is a set of Coordination Service instances that run on specified nodes in your server cluster. To learn more about Coordination Service ensembles, including how many nodes in your cluster should have a Coordination Service instance, see Deploy a Coordination Service Ensemble.

Synopsis

```
tsm topology deploy-coordination-service --node-names <nodeID,nodeID,...> [global-options]
```
Options

-n, --node-names <nodeID,nodeID,...>

Required.

Node IDs of nodes to include in the new Coordination Service ensemble, separated by commas. You can specify 1, 3, or 5 Coordination Service nodes, depending on the total number of nodes in your cluster. For more information, see The Coordination Service Quorum.

tsm topology filestore decommission

You must use the tsm topology filestore decommission command to prepare a file store node or nodes for safe removal. This command puts the specified nodes into read-only mode and ensures there is no unique content on the specified nodes.

If decommissioning results in a single file store node, you must use the --override option or the decommission will fail.

Synopsis

```bash
tsm topology filestore decommission --node-names <nodeID,nodeID,...> [options] [global options]
```

Options

-n, --node-names <nodeID,nodeID,...>

Required.

List of one or more nodes to decommission, specified by node ID and separated by commas.

-o, --override

Optional.
Overrides warnings or failures that would normally occur if removing the target File Store node would reduce the number of remaining file store nodes to one.

--delete-filestore

Optional.

Forces the removal of the file store, even if it has not been decommissioned. You should only use this option if the node the file store is on is in an error state and decommissioning cannot be done. Any unique files on the node will be permanently deleted.

**tsm topology filestore recommission**

Use the `tsm topology filestore recommission` command to revert any decommissioned nodes back to read-write mode.

**Synopsis**

```bash
tsm topology filestore recommission --node-names <nodeID,nodeID,...> [global options]
```

**Options**

```bash
-n, --node-names <nodeID,nodeID,...>
```

Required.

List of one or more nodes to recommission, specified by node ID and separated by commas.

**tsm topology nodes get-bootstrap-file**

You can use the `tsm topology nodes get-bootstrap-file` command to get the bootstrap file that is required to add a new node to the cluster.
Important: The bootstrap file contains a copy of the master keystore file used for encrypting the configuration secrets. We strongly recommend that you take additional measures to secure the bootstrap file using mechanisms as described in Securing secrets for import and export operations.

Synopsis

tsm topology nodes get-bootstrap-file --file <file>
[global options]

Options

-f, --file <file>

Required.

Full path and file name where the configuration file will be saved. If a duplicate file exists it will be overwritten.

tsm topology failover-repository

You can use the tsm failover-repository to manually initiate a repository failover from the current active repository to the second, passive repository.

The tsm topology failover-repository command is persistent. The failover repository remains the active repository until you issue the command again, or, if Tableau Server is configured for it, until automatic failover occurs. If you have a preferred active repository configured, use the --preferred option to switch back to that repository. If Tableau Server is configured for high availability, failover of the repository is automatic when necessary. Use the failover-repository command to manually fail over the repository.

Synopsis

tsm failover-repository --preferred | --target <node_name>
[global options]
Options

-r, --preferred

Required if -t or --target is not used.
Use the configured preferred node as the target for repository failover.

-t, --target <node_name>

Required if -r or --preferred is not used.
The fully qualified name of the target node onto which failover will occur.

tsm topology list-nodes

Display the nodes in the cluster and (optionally) the services on each node.

Synopsis
tsmtopology list-nodes [options] [global options]

Options

-v, --verbose

Optional.
Shows the services on each node.

tsm topology list-ports

Display the ports in the cluster.

Synopsis
tsmtopology list-ports [options] [global options]
Options

--node-name <nodeID>

Optional.

Specify the node to list ports for.

--service-name

Optional.

Specify the service to list ports for.

tsm topology remove-nodes

Remove nodes from the cluster.

To complete removal of a node, you also must run the tsm pending-changes apply command. Some scenarios require that you move or redeploy processes before removing nodes. See Remove a Node.

**Note:** To remove a node from a cluster it must have been configured with a process at some point in the past. If you are removing a node on which you've not configured any processes, then you must add a process on it, run tsm pending-changes apply, and then remove the node.

Synopsis

tsm topology remove-nodes --node-names <nodeID,nodeID,...>
[global options]
Options

-n, --node-names <nodeID,nodeID,...>

Required.

Specify the node or nodes to remove. If specifying multiple nodes, separate node IDs with a comma.

**tsm topology set-ports**

Set the ports for a service instance.

**Synopsis**

`tsm topology set-ports --node-name <nodeID> --port-name <port_name> --port-value <port_value> [options] [global options]`

**Options**

-i, --instance <instance_id>

Optional.

Specifies the instance id of the service. Defaults to 0 (zero) if not specified.

-n, --node-name <nodeID>

Required.

Specifies the node ID of the node.

-pn, --port-name <port_name>

Required.
The name of the port to be set, in this format: service_name:port_type. If no port type is specified, the primary port is assumed.

-pv, --port-value <port_value>

Required.

The port to set.

-r, --restart

Optional.

Suppress the prompt for restart and restart Tableau Server when necessary.

tsmtopology set-process

Set the number of instances of a process on a node. If a node already has the specified process, the number is updated to match the specified count.

You can only set one process at a time. If you specify more than one process, any process after the first one will be silently ignored.

Note: For a complete list of process names, see Process Reference.

Synopsis

tsm topology set-process --count <process_count> --node <nodeID> --process <process_name> [global options]

Options

-c, --count <process_count>

Required.
The process count (number of instances) to set.

-n, --node <nodeID>

Required.

Specifies the node ID of the node on which to set the process.

-pr, --process <process_name>

Required.

The name of the process to be set.

**tsm topology toggle-coordination-service**

You can use the `tsm topology toggle-coordination-service` command to switch between coordination service ensembles. To learn more about Coordination Service ensembles, see [Deploy a Coordination Service Ensemble](#).

**Synopsis**

```
 tsm topology toggle-coordination-service [global options]
```

**Global options**

-h, --help

Optional.

Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.
Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

**tsm user-identity-store**

You can use the tsm user-identity-store commands to configure the identity store for Tableau Server. You can also connect to and authenticate with Active Directory and LDAP directory services.

- get-group-mappings
- get-user-mappings
- list
- set-connection
- set-group-mappings
- set-user-mappings
verify-group-mappings
verify-user-mappings

**tsm user-identity-store get-group-mappings [options]**
Displays identity store group mappings.

**Synopsis**

```bash
tsm user-identity-store get-group-mappings [global options]
```

**tsm user-identity-store get-user-mappings [options]**
Displays identity store user mappings.

**Synopsis**

```bash
tsm user-identity-store get-user-mappings [global options]
```

**tsm user-identity-store list [options]**
Lists user-identity configuration.

**Synopsis**

```bash
tsm user-identity-store list [options] [global options]
```

**Options**

- `-v, --verbose`
  
  Optional.
  
  Lists all configuration parameters.

**tsm user-identity-store set-connection [options]**
Sets identity store connection parameters.
Synopsis

```bash
tsm user-identity-store set-connection --kerbkeytab <kerbkeytab> [options] [global options]
```

Options

- **b, --bind <username and password | Kerberos>**
  
  Optional.
  
  Set LDAP bind type.

- **d, --domain <domain>**
  
  Optional.
  
  Domain name.

- **hn, --hostname <hostname>**
  
  Optional.
  
  Hostname value.

- **kc, --kerbconfig <kerbconfig>**
  
  Optional.
  
  Kerberos configuration file path.

- **kp, --kerbprincipal <kerbprincipal>**
  
  Optional.
  
  Kerberos Principal.

- **kt, --kerbkeytab <kerbkeytab>**
Required.
Kerberos keytab file path.

-l,--port <port>
Optional.
Set LDAP Port value.

-lp,--ldappassword <ldappassword>
Optional.
LDAP Password.

-lu,--ldapusername <ldapusername>
Optional.
Set LDAP Username value.

-n,--nickname <nickname>
Optional.
NetBIOS name (nickname).

**tsm user-identity-store set-group-mappings [options]**

Sets identity store group mappings and configures LDAP directories that implement an arbitrary or custom schema.

**Synopsis**

```
 tsm user-identity-store set-group-mappings [options] [global options]
```
Options

-b, --basefilter <groupbasefilter>

Optional.

Set group BaseFilter value.

-cn, --classnames <groupclassnames>

Optional.

Override default user classname values (contains "group" string) with the values you set here. You can provide multiple classnames separated by commas.

-d, --description <description>

Optional.

Group description.

-e, --groupemail <groupemail>

Optional.

Group email value.

-m, --member <member>

Optional.

Set the group members.

-n, --groupname <groupname>

Optional.

Name of the group.
tsm user-identity-store set-user-mappings [options]

Sets identity store user mappings and configures LDAP directories that implement an arbitrary or custom schema.

Synopsis

```bash
  tsm user-identity-store set-user-mappings --certificate <certificate> [options] [global options]
```

Options

- `-c, --certificate <certificate>`
  
  Optional.
  
  Users' certificate file location.

- `-cn, --classnames <user_classnames>`
  
  Optional.
  
  Override default user classname values ("user" and "inetOrgPerson") with the values you set here. You can provide multiple classnames separated by commas.

- `-dn, --displayname <displayname>`
  
  Optional.
  
  Display name of the user.

- `-e, --email <email>`
  
  Optional.
  
  Users' email address.

- `-jp, --jpegphoto <jpegfile>`
Optional.

Users' jpeg image location.

-m, --memberof <typename>

Optional.

Group that the user is a member of.

-t, --thumbnail <thumbnail>

Optional.

Users' thumbnail location.

-ub, --basefilter <userbasefilter>

Optional.

Users' BaseFilter.

-uu, --ldapusername <ldapusername>

Optional.

User name.

ts hands-user-identity-store verify-group-mappings [options]

Validates configuration for LDAP group mapping.

Synopsis

ts hands-user-identity-store verify-group-mappings --verify <typename> [global options]
Options

-v, --verify <group_name>

Optional.

Name of group to search for.

tsm user-identity-store verify-user-mappings [options]

Validates configuration for LDAP user mapping.

Synopsis

```
tsm user-identity-store verify-user-mappings --verify <user_name> [global options]
```

Options

-v, --verify <user_name>

Optional.

Name of user to search for.

Global options

-h, --help

Optional.

Show the command help.

-p, --password <password>

Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.
--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

tsm version

You can use the tsm version command to get the version of Tableau Server.

Synopsis

tsm version [global options]

Global options

-h, --help

Optional.

Show the command help.

-p, --password <password>
Required, along with -u or --username if no session is active.

Specify the password for the user specified in -u or --username.

--request-timeout <timeout in seconds>

Optional.

Wait the specified amount of time for the command to finish. Default value is 7200 (30 minutes).

-s, --server <server address>

Optional.

Use the specified address for Tableau Services Manager. If no server is specified, https://<localhost | dnsname>:8850 is assumed.

-u, --username <user>

Required if no session is active, along with -p or --password.

Specify a user account. If you do not include this option, the command is run using credentials you signed in with.

**tsm File Paths**

Certain tsm commands read files from or write files to default locations. These default locations are determined by basefilepath variables defined for each command. You can use tsm to view the current value of the variables, and to change the locations.

**Default locations for files**

During the tsm maintenance backup, restore, send-logs, and ziplogs processes, and the tsm sites export and sites import processes, Tableau Server
uses default locations for the files created or used by these commands.

By default:

- **tsm maintenance commands**:

  - **backup**—The backup `.tsbak` file is generated in:
    
    `/var/opt/tableau/tableau_server-
     /data/tabsvc/files/backups`

  - **restore**—The restore process restores a backup file from:
    
    `/var/opt/tableau/tableau_server-
     /data/tabsvc/files/backups`

  - **send-logs**—The send-logs sends the logs file from:
    
    `/var/opt/tableau/tableau_server-
     /data/tabsvc/files/backups`

  - **ziplogs**—The ziplogs file is generated in:
    
    `/var/opt/tableau/tableau_server/data/tabsvc/files/log-
     archives`

- **tsm sites**

  - **export**—The export `.zip` file is generated to the following directory:
    
    `/var/opt/tableau/tableau_server-
     /data/tabsvc/files/siteexports`

  - **import**—During the import process, Tableau Server looks for files in:
    
    `/var/opt/tableau/tableau_server-
     /data/tabsvc/files/siteimports`
Get the current file location

You can see the current file location for a specific command using `tsm configuration get`:

- For tsm maintenance commands:
  - backup, restore, and send-logs:
    
    ```
    tsm configuration get -k basefilepath.backuprestore
    ```
  - ziplogs:
    
    ```
    tsm configuration get -k basefilepath.log_archive
    ```

- For tsm sites commands:
  - export
    
    ```
    tsm configuration get -k basefilepath.site_export.-exports
    ```
  - import
    
    ```
    tsm configuration get -k basefilepath.site_import.-exports
    ```

Change the current file location

You can change the expected file locations using the `tsm configuration set command` to update the `basefilepath` variables.

Changing a `basefilepath` variable does not move existing files from the original directory to the new directory. If you want existing backup, restore, log files, or site export or import files to reside in the new directory you specify, you must move them manually. You are responsible for creating the new location and for setting the correct permissions to allow tsm access to any files that will be placed there, and to the directory structure containing
those files. For more information about permissions and tsm, see Files and Permissions in TSM.

- For tsm maintenance commands:

  - To change the backup, restore, or send-logs directory, run the following command:

    ```bash
tsm configuration set -k basefilepath.backuprestore -v "new/directory/path"
    ```

  - To change the ziplogs directory:

    ```bash
tsm configuration set -k basefilepath.log_archive -v "new/directory/path"
    ```

- For tsm sites commands:

  - To change the sites export directory:

    ```bash
tsm configuration set -k basefilepath.site_export.-exports -v "new/directory/path"
    ```

  - To change the sites import directory:

    ```bash
tsm configuration set -k basefilepath.site_import.-exports -v "new/directory/path"
    ```

After you change a default file location you need to do the following:

1. Stop Tableau Server:

   ```bash
tsm stop
   ```

2. Restart the TSM Controller:

   ```bash
sudo systemctl restart tabadmincontroller_0.service
   ```
3. Wait several minutes for the controller to restart. You can confirm the controller has restarted with this command:

```shell
tsm status -v
```

When you can run that command and the Tableau Server Administration Controller is listed as 'running' the controller has restarted.

4. Start Tableau Server:

```shell
tsm start
```

Entity Definitions and Templates

Configuration File Example

This article provides an example of a complete JSON configuration file, with gatewaySettings and identityStore entities specified. In addition, a configuration key sets the gateway timeout to 900 seconds.

Your configuration file will look different depending on the options you need to set.

You might set multiple .json configuration files during installation. To set the values for each file in Tableau Server, you run the following command, once for each configuration file:

```shell
tsm settings import -f /path/to/file.json
```

After you set the configuration files, run tsm pending-changes apply to apply the changes from all of the .json files you’ve set.

```json
{
   "configEntities": {
      "gatewaySettings": {
         "_type": "gatewaySettingsType",
```
"port": 80,
"firewallOpeningEnabled": true,
"sslRedirectEnabled": true,
"publicHost": "localhost",
"publicPort": 80
},
"identityStore": {
  "_type": "identityStoreType",
  "type": "local",
  "domain": "example.lan",
  "nickname": "EXAMPLE"
}
},
"configKeys": {
  "gateway.timeout": "900"
}
}

Entities vs keys

As shown in the example above, there are two classes of configuration parameters: configEntities and configKeys.

configEntities

Certain types of configuration are done through entity sets that map to specific scenarios, such as the identity store and gateway configurations. When you pass a set of configEntities with the tsm settings import -f /path/to/file.json command, TSM validates the configuration. If values passed are invalid, TSM will provide an error. This enables you to make changes during the configuration process, rather than experience a configuration failure at initialization or run time.

Entities can be set only by including a configEntities block in a .json file.

configKeys
Entities cover only a small portion of the configuration values that can be set. Hundreds of keys correspond to parameters stored in .yml files. Tableau Server uses these parameters to store all of the configuration information for all services.

You can set individual keys with the tsm configuration command. But during deployment, setting them along with other configuration scenarios in JSON files, as shown above, is more convenient.

Unlike configEntities, configKeys are not validated.

**Note:** We do not recommend setting parameters that are not documented in tsm configuration set Options.

gatewaySettings Entity

You must configure the gateway settings for the Tableau Server computer.

Use the configuration file template below to create a json file. After you have filled in the options with the appropriate values, pass the json file and apply settings with the following commands:

```
tsm settings import -f /path/to/file.json

tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

Gateway settings

The gateway settings in the template below specify the HTTP settings for Tableau Server. We recommend using SSL/TLS. Tableau Server is hard-coded to use port 443 for
SSL/TLS. Therefore, if you enable SSL, you do not need to update the gatewaySettings entity.

Configuration template

Use this template to configure the gateway settings.

All entity options are case sensitive.

For more explanation about configuration files, entities, and keys see Configuration File Example.

```
{
   "configEntities": {
      "gatewaySettings": {
         "_type": "gatewaySettingsType",
         "port": 80,
         "sslRedirectEnabled": true,
         "publicHost": "localhost",
         "publicPort": 80
      }
   }
}
```

Configuration file reference

This table includes all of the options that can be included with the "gatewaySettings" entity set.

_type

Required.

Value: "gatewaySettingsType"

Do not change.

port
Specifies HTTP port. Default is port 80.

sslRedirectEnabled

Options: true or false.

publicHost

Specifies host name for http/s service.

publicPort

The externally facing port that clients use to access the gateway process on Tableau Server over HTTP/HTTPS. This value defaults to 80, which is the same as the default for the gateway port itself. However, if there is a proxy or a NAT between gateway and an external client, the public port used by the proxy can be different from the internal port the gateway uses. See Configuring Reverse Proxies for Tableau Server.

trustedIPs

Specifies trusted IP addresses that communicate with Tableau Server. Trusted IP addresses include upstream proxy servers and servers that are used for trusted authentication with Tableau Server. See Configuring Reverse Proxies for Tableau Server and Trusted Authentication.

If you are running Tableau Server in a cluster then all other nodes of the cluster will automatically be included in the corresponding configuration file that this entity updates. Therefore, if you specify a new value for trustedIPs, then you must include the IP addresses for the other nodes in the value.

Enter multiple IP addresses in a comma-separated list.

trustedHosts

Specifies trusted IP addresses that communicate with Tableau Server. Typically, this value contains a list of upstream proxy servers. The values in trustedHosts are used to determine client request targets.
If you are running Tableau Server in a cluster then all other nodes of the cluster will automatically be included in the corresponding configuration file that this entity updates. Therefore, if you specify a new value for trustedIPs, then you must include the IP addresses for the other nodes in the value.

The hostname may be entered as a name or an IP address. Format is hostname:port, for example, proxy_server1:80. In the case where a given proxy server has multiple name forms (proxy_server1, proxy_server1.example.com, etc), each name must be entered in the hostname:port format. Enter multiple hostname:port pairs in a comma-separated list.

**identityStore Entity**

Tableau Server requires an identity store to store user and group information. Review Authentication and Identity Store topics before configuring the identity store for the first time. After you have configured the identity store on Tableau Server, you cannot change it without reinstalling the server.

All entity options are case sensitive.

**Before you begin**

Review the following information:

- If you plan to enable OpenID Connect then you must configure the local identity store
- If you will not be using the local identity store, then you will be using some version of LDAP. In this case, work with your directory/LDAP administrator to configure Tableau Server for your LDAP schema and bind requirements.
- Do not connect to LDAP with simple bind over a unsecured connection. We recommend LDAPS for simple bind. See LDAP over SSL.
- We recommend validating the LDAP configuration before initializing the server, see Verify LDAP configuration (Optional).
- To use Kerberos authentication for the Tableau Server service, then you'll need a keytab file for GSSAPI bind, as described in the sections below. In the context of Kerberos, GSSAPI bind is all you need during the base installation of Tableau Server. After you install the server, you can then configure Kerberos for user authentication and Kerberos delegation to data sources.
In this topic, we make the distinction between LDAP (the protocol for connecting to directory services) and an LDAP server (an implementation of a directory service). For example, slapd is an LDAP server that is part of the OpenLDAP project.

Use one of the configuration file templates below to create a json file. After you have filled in the options with the appropriate values, pass the json file and apply settings with the following commands:

```
tsm settings import -f /path/to/file.json

tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

**Configuration templates**

Select an identity store configuration template to edit:

- Local
- LDAP - Active Directory
- OpenLDAP - GSSAPI Bind
- OpenLDAP - Simple Bind

For more explanation about configuration files, entities, and keys see Configuration File Example.

**Local**

Configure local as the identity store type if your organization does not already have an Active Directory or LDAP server for user authentication. When you select local as the identity store type, you use Tableau Server to create and manage users.

```json
{
    "configEntities":{
```
{ "identityStore": { "_type": "identityStoreType", "type": "local" } } 

Important
The LDAP configuration templates below are examples. The templates, as presented, will not configure LDAP connectivity in your organization. You must work with your directory administrator to edit the LDAP template values for a successful deployment.

LDAP - Active Directory

Configure this option to connect to Active Directory with GSSAPI (Kerberos) bind. Tableau Server uses the LDAP protocol to connect to Active Directory. Tableau Server includes support for Active Directory schema. Therefore, if you set the "directoryServiceType" option to "activedirectory" then you do not need to provide schema info in the "identityStoreSchemaType" option.

Important: If you are installing into Active Directory, and the computer where you are installing Tableau Server is already joined to the domain, then the computer may already have a configuration file and a keytab file. In this case, the Kerberos files are for Linux operating system functionality and authentication. Strictly speaking, you can use these files for GSSAPI bind, but we don't recommend using them. Instead, contact your Active Directory administrator and request a keytab specifically for the Tableau Server service.

{ "configEntities":{
"identityStore": {
  "_type": "identityStoreType",
  "type": "activedirectory",
  "domain": "your-domain.lan",
  "nickname": "YOUR-DOMAIN-NICKNAME",
  "directoryServiceType": "activedirectory",
  "bind": "gssapi",
  "kerberosKeytab": "<path to key tab file>",
  "kerberosConfig": "/etc/krb5.conf",
  "kerberosPrincipal": "your-principal@YOUR.DOMAIN"
}
}

We recommend binding to Active Directory with GSSAPI (Kerberos). However, you can connect with simple bind and LDAPS. To connect with simple bind, change bind to simple, remove the three Kerberos entities, and add the port/sslPort, username, and password options. The following example shows Active Directory with simple bind json.

{
  "configEntities":{
    "identityStore": {
      "_type": "identityStoreType",
      "type": "activedirectory",
      "domain": "your-domain.lan",
      "nickname": "YOUR-DOMAIN-NICKNAME",
      "directoryServiceType": "activedirectory",
      "hostname": "optional-ldap-server",
      "sslPort": "636",
      "bind": "simple",
      "username": "username",
      "password": "password"
    }
  }
}
OpenLDAP - GSSAPI bind

In most cases, organizations that use OpenLDAP with GSSAPI (Kerberos) will use a keytab file to store credentials. In the following example, a keytab file is used for authentication credentials.

However, you can provide credentials through the `username` and `password` entities.

You can also specify both a keytab and a username and password. In this case, Tableau Server will attempt to use the keytab, but if authentication fails for any reason it will fallback and use the username and password credentials.

```json
{
  "configEntities": {
    "identityStore": {
      "_type": "identityStoreType",
      "type": "activedirectory",
      "domain": "your-domain.lan",
      "nickname": "YOUR-DOMAIN-NICKNAME",
      "directoryServiceType": "openldap",
      "bind": "gssapi",
      "kerberosKeytab": "<path to key tab file>",
      "kerberosConfig": "/etc/krb5.conf",
      "kerberosPrincipal": "your-principal@YOUR.DOMAIN",
      "identityStoreSchemaType": {
        "userBaseFilter": "(objectClass=inetOrgPerson)",
        "userUsername": "user",
        "userDisplayName": "displayname",
        "userEmail": "email",
        "userCertificate": "certificate",
        "userThumbnail": "thumbnail",
        "userJpegPhoto": "photo",
        "groupBaseFilter": "(objectClass=groupofNames)"
      }
    }
  }
}
```
"groupName": "groupName",
"groupEmail": "groupemail",
"groupDescription": "groupdescription",
"member": "member",
"distinguishedNameAttribute": "",
"serverSideSorting": "",
"rangeRetrieval": ""
}
}
}

OpenLDAP - Simple bind

{
  "configEntities":{
    "identityStore": {
      "_type": "identityStoreType",
      "type": "activedirectory",
      "domain": "my.root",
      "nickname": "",
      "hostname": "optional-ldap-server",
      "port": "389",
      "directoryServiceType": "openldap",
      "bind": "simple",
      "username": "cn=username,dc=your,dc=domain",
      "password": "password",
      "identityStoreSchemaType": {
        "userBaseFilter": "(objectClass=inetOrgPerson)",
        "userUsername": "user",
        "userDisplayName": "displayname",
        "userEmail": "email",
        "userCertificate": "certificate",
        "userThumbnail": "thumbnail",
        "userJpegPhoto": "photo",
      }
    }
  }
}
"groupBaseFilter": "(objectClass=groupOfNames)",
"groupName": "groupname",
"groupEmail": "groupemail",
"groupDescription": "groupdescription",
"member": "member",
"distinguishedNameAttribute": "",
"serverSideSorting": "true",
"rangeRetrieval": "false"
}

}

}

Configuration template reference

Shared identity store options

type
Where you want to store user identity information. Either local or activedirectory. (If you want to connect to any LDAP server, select activedirectory.)

domain
The domain of the computer where you installed Tableau Server.

nickname
The nickname of the domain. This is also referred to as the NetBIOS name in Windows environments. The nickname option is required for all LDAP entities. If your organization does not require a nickname/NetBIOS, then pass a blank key, for example: "nickname": "".

LDAP GSSAPI bind options

directoryservicetype
The type of directory service that you want to connect to. Either activedirectory or openldap.
bind

The way that you want to secure communication to the directory service. Enter gssapi for GSSAPI/Kerberos.

kerberosConfig

The path to the Kerberos configuration file.

kerberosKeytab

The path to the Kerberos keytab file. It is recommended that you create a keytab file with keys specifically for Tableau Server and that you do not share the keytab file with other applications on the computer. For example, on Linux, you might place the keytab file in the /var/opt/tableau/keytab directory.

kerberosPrincipal

The service principal name for Tableau Server on the host machine. The keytab must have permission for this principal. Do not use the existing system keytab at /etc/krb5.keytab. We recommend that you register a new service principal name. To see principals in a given keytab, run the klist -k command.

domain

In Windows Active Directory environments, specify the domain where Tableau Server is installed, for example, "example.lan". In LDAP directories, specify the root domain name in the same format. For example, if your root is "dc=my,dc=root", specify "my.root".

LDAP simple bind options

directoryservicetype

The type of directory service that you want to connect to. Either active.directory or openldap.

bind

The way that you want to secure communication to the directory service. Enter simple for LDAP unless you are connecting to an LDAP server with Kerberos. For Kerberos, enter gssapi.
In Windows Active Directory environments, specify the domain where Tableau Server is installed, for example, "example.lan". In LDAP directories, specify the root domain name in the same format. For example, if your root is "dc=my,dc=root", specify "my.root".

The hostname of the LDAP server. You can enter a hostname or an IP address for this value.

Use this option to specify the non-secure port of the LDAP server. Plaintext is usually 389.

Use this option to specify the secure port of the LDAP server. We recommend secure LDAP for simple bind. LDAPS is usually port 636.

The user name that you want to use to connect to the directory service. For LDAP servers, enter the distinguished name (DN) of the user that you want to use to connect. For example, you might enter cn=username,dc=your-local-domain,dc=lan.

The password of the user that you want to use to connect to the LDAP server.

If you configure an LDAP connection to an LDAP server, you can enter schema information specific to your LDAP server in the identityStoreSchemaType object.

The filter that you want to use for users of Tableau Server. For example, you might specify an object class attribute and an organization unit attribute.
userUsername
    The attribute that corresponds to user names on your LDAP server.

userDisplayName
    The attribute that corresponds to user display names on your LDAP server.

userEmail
    The attribute that corresponds to user email addresses on your LDAP server.

userCertificate
    The attribute that corresponds to user certificates on your LDAP server.

userThumbnail
    The attribute that corresponds to user thumbnail images on your LDAP server.

userJpegPhoto
    The attribute that corresponds to user profile images on your LDAP server.

groupBaseFilter
    The filter that you want to use for groups of users of Tableau Server. For example, you might specify an object class attribute and an organization unit attribute.

gROUPName
    The attribute that corresponds to group names on your LDAP server.

groupEmail
    The attribute that corresponds to group email addresses on your LDAP server.

groupDescription
    The attribute that corresponds to group descriptions on your LDAP server.
The attribute that describes the list of users in a group.

distinguishedNameAttribute
The attribute that stores the distinguished names of users. This attribute is optional, but it greatly improves the performance of LDAP queries.

serverSideSorting
Whether the LDAP server is configured for server-side sorting of query results. The default value is true.

rangeRetrieval
Whether the LDAP server is configured to return a range of query results for a request. The default value is true.

groupClassNames
By default Tableau Server looks for LDAP group object classes containing the string “group”. If your LDAP group objects do not fit the default class name, override the default by setting this value. You can provide multiple classnames separated by commas. For example: “basegroup, othergroup”.

userClassNames
By default Tableau Server looks for LDAP user object classes containing the string “user” and “inetOrgPerson”. If your LDAP user objects do not use these default class names, override the default by setting this value. You can provide multiple classnames separated by commas. For example: “userclass1, userclass2”.

kerberosSettings Entity
Before you configure Kerberos authentication, review Kerberos Requirements.

Use the configuration file template below to create a json file. After you have filled in the options with the appropriate values, pass the json file and apply settings with the following commands:

tsm settings import -f /path/to/file.json
tsm pending-changes apply

The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the -r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

Configuration template

Use this template to configure Kerberos settings.

All entity options are case sensitive.

For more explanation about configuration files, entities, and keys see Configuration File Example.

After you have finished with the initial configuration of Kerberos authentication, use the tsm authentication kerberos <commands> sub-category to set additional values.

```json
{
  "configEntities": {
    "kerberosSettings": {
      "_type": "kerberosSettingsType",
      "enabled": "true",
      "keytabFile": "/path/to/keytab_file"
    }
  }
}
```

Configuration file reference

The following list includes all of the options that can be included with the "kerberosSettings" entity set.

Option
Value

enabled

Options: true or false.

Enables Kerberos authentication.

keytabFile

Required.

Path to valid Kerberos keytab file.

dBClasses

Comma-separated list of database classes for global credentials. May be required for connecting to Cloudera data sources.

**mutualSSLSettings Entity**

Before you configure mutual SSL, review Configure SSL for External HTTP Traffic to and from Tableau Server (Linux).

The **mutualSSLSettings** entity combines both SSL and mutual SSL configuration. Mutual SSL requires that external SSL has been enabled and properly configured.

Use the configuration file template below to create a .json file. After you have filled in the options with the appropriate values, pass the .json file and apply settings with the following commands:

```
tsm settings import -f </path/to/file.json>

tsm pending-changes apply
```

The **pending-changes apply** command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the **-r**
option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

**Configuration template**

Use this template to configure mutual SSL settings.

All entity options are case sensitive.

For more explanation about configuration files, entities, and keys see Configuration File Example.

```json
{
    "configEntities": {
        "mutualSSLSettings": {
            "_type": "mutualSSLSettingsType",
            "sslEnabled": true,
            "proxyLogin": false,
            "clientCertRequired": true,
            "caCertFile": "required",
            "keyFileName": "required",
            "keyPassphrase": "",
            "chainFile": "",
            "revocationFile": "",
            "redirect": false,
            "fallbackToPassword": true,
            "protocols": "",
            "cipherSuite": "",
            "forceHttpsForPublicEmbed": false
        }
    }
}
```
Configuration file reference

sslEnabled

Enable SSL. This is a prerequisite to enabling mutual SSL.

clientCertRequired (MutualSSL)

Set to true to enable mutual SSL authentication. Set to false to disable.

cacertFile (MutualSSL)

Required.

Specify the CA-issued certificate file for two-way SSL. The file path must be readable by Tableau Server.

certFileName

Specify the file that contains the concatenation of PEM encoded CA certificates that form the certificate chain for the server certificate.

Alternatively the referenced file can be the same as cacertFile when the CA certificates are directly appended to the server certificate for convenience.

keyFileName

If the key is not combined with the certificate, use this configuration key to point to the key file. If you have both an RSA and a DSA private key, you can configure both in parallel (for example, to also allow the use of DSA ciphers).

keyPassphrase

Optional. Passphrase for the certificate file. The passphrase you enter will be encrypted while at rest.
Note: If you create a certificate key file with a passphrase, you cannot reuse the SSL certificate key for SAML.

revocationFile

Specifies the file path for an SSL CA Certificate Revocation List (.crl) file.

Redirect

Default: true. Specifies whether Tableau Server should redirect http requests as https requests to the appropriate endpoint.

clientCertMapping (MutualSSL)

Specifies the method for retrieving the user name from the certificate.

Accepted values: ldap, upn, cn

- For a server using local authentication, the default setting is upn (User Principal Name).
- When Tableau Server authentication is configured for Active Directory (AD), the default is ldap (Lightweight Directory Access Protocol). This tells the server to go to AD to validate the user, and it ignores the names inside the certificate.

You can set cn for either authentication type to use the CN in the Subject DN in the certificate.

For more information, see Mapping a Client Certificate to a User During Mutual Authentication.

fallbackToPassword (MutualSSL)

Set to true to give users the option to sign in to Tableau Server through their user name and password if mutual SSL authentication fails. Set to false to disallow this fallback option.
protocols

List the Transport Layer Security (TLS) protocol versions you want to allow or disallow.

**Default value:** "all -SSLv2 -SSLv3"

However, we recommend the using the following setting:

"all -SSLv2 -SSLv3 -TLSv1 -TLSv1.1"

For more information, see tsm security external-ssl enable. For general information, see the Apache online documentation.

cipherSuite

List ciphers to allow or disallow for SSL.

**Default value:** "HIGH: MEDIUM: !aNULL: !MD5: !RC4"

However, we recommend that you add Triple-DES to the list of disabled cipher suites, and set this key to:

"HIGH: MEDIUM: !aNULL: !MD5: !RC4: !3DES"

For more information, see the --cipher-suite option in tsm security external-ssl enable CLI reference.

proxyLogin

Default: false. Indicates that Tableau Server uses a proxy for SSL on sign-in only. It controls the protocol the server reports to Tableau Desktop for sign-in APIs.

forceHTTPForPublicEmbed

Default value: false. Forces the code for embedded views to use SSL.
openIDSettings Entity

Before you configure OpenID authentication, review Requirements for Using OpenID Connect.

Use the configuration file template below to create a json file. After you have filled in the options with the appropriate values, pass the json file and apply settings with the following commands:

```bash
  tsm settings import -f /path/to/file.json

  tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

Configuration template

Use this template to configure OpenID settings.

All entity options are case sensitive.

For more explanation about configuration files, entities, and keys see Configuration File Example.

After you have finished with the initial configuration of OIDC, use the `tsm authentication openid <commands>` sub-category to set additional values.

```json
{
    "configEntities": {
        "openIDSettings": {
            "_type": "openIDSettingsType",
            "enabled": true,
            "clientId": "required",
```
"clientSecret": "required",
"configURL": "required if staticFile value is not set",
"staticFile": "required if configURL value is not set",
"externalURL": "required"
}
}

Configuration file reference

The following list includes all of the options that can be included with the "openIDSettings" entity set.

_type

Required.

Do not change.

enabled

Required.

Set to true.

clientId

Required.

Specifies the provider client ID that your IdP has assigned to your application. For example, "laakjwdlnaoioloadjkwha".

clientSecret

Required.
Specifies the provider client secret. This is a token that is used by Tableau to verify the authenticity of the response from the IdP. This value is a secret and should be kept securely.

For example, "fwahfkjaw72123=".

**configURL**

Required.

Specifies provider configuration URL. If you do not specify a configuration URL, then delete this option and specify a path and file name for `staticFile` instead.

**staticFile**

Required.

Specifies the local path to the static OIDC discovery JSON document. If you do not specify a static file, then delete this option and specify a url for `configURL` instead.

**externalURL**

Required.

The URL of your server. This is typically is the public name of your server, such as http://example.tableau.com.

**connectionTimeout**

Optional.

Specifies connection timeout span in seconds. Default is 10.

**readTimeout**

Optional.

Specifies read timeout span in seconds. Default is 30.
ignoreDomain

Set this to true if the following are true:

- You are using email addresses as usernames in Tableau Server
- You have provisioned users in the IdP with multiple domain names
- You want to ignore the domain name portion of the email claim from the IdP

Before you proceed, review the user names that will be used as a result of setting this option to true. User name conflicts may occur. In the case of a user name conflict, the risk of information disclosure is high. See Requirements for Using OpenID Connect

ignoreJWK

Set this to true if your IdP does not support JWK validation. In this case, we recommend authenticating communication with your IdP using mutual TLS or another network layer security protocol. Default is false.

customScope

Specifies a custom scope user-related value that you can use to query the IdP. See Requirements for Using OpenID Connect

idClaim

Change this value if your IdP does not use the sub claim to uniquely identify users in the ID token. The IdP claim that you specify should contain a single, unique string.

usernameClaim

Change this value to the IdP claim that your organization will use to match user names as stored in Tableau Server.

clientAuthentication

Specifies custom client authentication method for OpenID Connect.
To configure Tableau Server to use the Salesforce IdP, set this value to `client_secret_post`.

`iFramedIDPEnabled`

Set to `true` to allow IdP displayed in an iFrame. The IdP must disable clickjack protection to allow iFrame presentation.

`samlSettings Entity`

This article contains a template and reference for configuring server-wide SAML on Tableau Server, using a configuration file with keys and values for the `samlSettings` entity. This information supplements the SAML configuration steps in Configure Server-Wide SAML for Tableau Server on Linux.

To create a SAML configuration template and apply it to Tableau Server, you complete the following steps:

1. Review the following two sections that describe the template and how it’s structured (Template categories and definitions and samlSettings configuration template).
2. Paste the JSON code shown in the template into a new text file, and save it using a `.json` extension.
3. Use the SAML configuration entity reference to help you provide values where required.
4. Add optional key/value pairs specific to your environment.
5. Pass the configuration file to Tableau Server.

**Template categories and definitions**

The template uses placeholders for each key value. These placeholders are categorized as follows:
• **Required:** Attributes with the "required" value must be replaced with valid data before you run the configuration command. Review the configuration file reference for valid values.

• **Hard-coded:** Attribute names that are prepended with an underscore (_), for example "_type" hold hard-coded values. Do not change these values.

• **Default values:** Attributes that are set to a value that is not "required" are default values. These are required attributes that you can change as appropriate for your environment.

• **Empty sets:** Values that are empty (" ") can be passed as they are, or you can provide a value for your installation.

All entity options are case sensitive.

**samlSettings configuration template**

Paste this code into a text file and customize it for your environment, using the reference below.

```json
{
  "configEntities": {
    "samlSettings": {
      "_type": "samlSettingsType",
      "enabled": true,
      "returnUrl": "required",
      "entityId": "required",
      "certFile": "required",
      "keyFile": "required",
      "idpMetadataFile": "required",
      "idpDomainAttribute": "",
      "idpUsernameAttribute": "required"
    }
  }
}
```
SAML configuration entity reference

The following table includes all of the options you can include with the "samlSettings" entity set.

**idpMetadataFile**

Required. The path and file name for the XML file generated by the IdP. The XML metadata must include the user name attribute (assertion).

If you completed the steps described in Configure Server-Wide SAML for Tableau Server on Linux the value you enter here would be:

/var/opt/tableau/tableau_server/data/saml/<metadata-file.xml>

**enabled**

true | false

Required. Indicates whether SAML authentication is enabled and can be configured on the server.

**returnURL**

The SAML return URL configured in the IdP. This is typically the external URL that Tableau Server users enter in their browser to access the server, such as http://tableau-server.

**Notes**

- http://localhost does not work for an external server.

- Adding a trailing slash to the URL (http://tableau-server/) is not supported.
**entityId**

Required. IdP entity ID value.

Typically the same as the Tableau Server return URL. The entity ID that you enter is used as a base for generating site-specific entity IDs. For example, if you enter "http://tableau-server", an individual site configured for SAML might display an entity ID something like the following:

http://tableau-server-/samlservice/public/sp/metadata?alias=48957410-9396-430a-967c-75b-db6e002a0

**certFile**

Enter the location and file name of the x509 certificate (.crt) file for SAML. For example:

/var/opt/tableau/tableau_server/data/saml/<file.crt>

For more information, see SAML Requirements and Configure Server-Wide SAML for Tableau Server on Linux.

**keyFile**

Specify the location of the RSA or DSA private key (.key) file that accompanies the certificate file and is not password protected. For example:

/var/opt/tableau/tableau_server/data/saml/<file.key>

For more information, see SAML Requirements and Configure Server-Wide SAML for Tableau Server on Linux.

**idpDomainAttribute**

Optional. This value is used when Tableau Server includes user accounts from multiple domains. Specify the fully qualified domain name (FQDN) of the domain you want to be the default. For example tableauserver.mycompany.com.
If you do not provide a value for this key, the value used depends on the Tableau Server user authentication setting:

- For local authentication, the value `local` is used.
- For Active Directory, Tableau uses the FQDN from the configuration setting `wgserver.domain.default`.

To get the value for `wgserver.domain.default`, you can run the following command:

```
tsm configuration get --key wgserver.domain.default
```

**idpUsernameAttribute**

Required. In the IdP metadata, find the attribute that is used to specify user name values, and enter the name of that attribute. For example, `username`.

**idpEmailAttribute**

Optional. In the IdP metadata, the name of the attribute the IdP uses to specify users’ email addresses.

**idpDisplayNameAttribute**

Optional. In the IdP metadata, the name of the attribute the IdP uses to specify users’ display names.

**idpAttributeUseDefaultDomain**

`true` | `false`

Optional. The default value is `true` for a single domain or for multiple domains when you do not specify a value for `idpDomainAttribute`. The default value is `false` if you specify a non-default value for `idpDomainAttribute`.

**desktopNoSAML**
Allow users to use SAML authentication when they sign in from Tableau Desktop. To use this option, your IdP must support forms-based authentication.

By default this is not set, so the effective behavior is equivalent to setting it to false. Set this to true to disable SAML authentication through Tableau Desktop.

**appNoSAML**

true | false

Allow using SAML to sign in from older versions of Tableau Mobile app. Devices running Tableau Mobile app version 19.225.1731 and higher ignore this option. To disable devices running Tableau Mobile app version 19.225.1731 and higher, disable SAML as a client login option on Tableau Server.

**logoutEnabled**

true | false

Specifies whether SAML logout is enabled for Tableau Server. The default is true. This setting applies only if SAML authentication is enabled for the server.

**logoutUrl**

Enter the URL to redirect to after users sign out of the server. By default this is the Tableau Server sign-in page. You can specify an absolute or a relative URL.

**maxAuthenticationAge**

Specifies the maximum number of seconds allowed between a user’s authentication and processing of the AuthNResponse message. The default value is 7200 (30 mins).

**maxAssertionTime**

Specifies the maximum number of seconds, from creation, that a SAML assertion is
usable. Default value is 3000.

**sha256Enabled**

true | false

Indicates whether the SAML certificate is an SHA-2 (256 or 512 bit) certificate. Default value is false. If you use the same certificate for SAML as you do for SSL, you will most likely set this to true.

**signRequests**

true | false

Default value is true.

**iFramedIdpEnabled**

true | false

Default value is false, meaning that when users select the sign-in button on an embedded view, the IdP’s sign-in form opens in a pop-up window.

When you set it to true, and a server SAML user who is already signed in navigates to a web page with an embedded view, the user will not need to sign in to see the view.

You can set this to true only if the IdP supports signing in within an iframe. The iframe option is less secure than using a pop-up, so not all IdPs support it. If the IdP sign-in page implements clickjack protection, as most do, the sign-in page cannot display in an iframe, and the user cannot sign in.

If your IdP does support signing in via an iframe, you might need to enable it explicitly. However, even if you can use this option, it disables Tableau Server clickjack protection for SAML, so it still presents a security risk.

**port**

The port that handles SAML authentication traffic.
Default value: 80

If your server is configured for SSL (recommended), set this to 443.

singleLogoutSupported

true | false

Enable or disable SAML single logout (SLO). This value is not set by default, so effectively it behaves as if the value were set to false. To set it to true, the IdP must support SLO.

siteSpecific

enabled | disabled

 Specifies whether the server is configured to support site-specific SAML authentication, in which you can configure sites to use different IdPs or IdP applications. Set to disabled by default. If you set to enabled, you then configure a site’s SAML settings in the Tableau Server web UI. Sign in to the site on which you want to use SAML, and go to the Authentication tab on the Settings page.

Pass the configuration file to Tableau Server

After you have provided an appropriate value for each entity you include in the configuration template, use the following commands to pass the .json file and apply settings to Tableau Server.

```
tsm settings import -f /path/to/file.json

 tsm pending-changes apply
```

See also

After you complete the initial SAML configuration, use tsm authentication mutual-ssl <commands> to set additional values.
For the command-line reference for configuring SAML, see tsm authentication saml <commands>.

**sapHanaSettings Entity**

Use the sapHanaSettings entity to configure SAML delegation for SAP HANA.

Review Configure SAP HANA SAML delegation before you continue.

Use the configuration file template below to create a json file. After you have filled in the options with the appropriate values, pass the json file and apply settings with the following commands:

```
tsm settings import -f /path/to/file.json

tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

**SAP HANA SAML settings**

The settings in the template below specify the settings for Tableau Server in a SAML delegation scenario with SAP HANA.

If you are running Tableau Server in a distributed deployment, then you may need to copy the certificate and key files to other nodes in the cluster. See Copying Files in a Distributed Deployment.

**Configuration template**

Use this template to configure trusted authentication settings.
All entity options are case sensitive.

For more explanation about configuration files, entities, and keys see Configuration File Example.

```
{
  "configEntity": {
    "_type": "sapHanaSettingsType",
    "enabled": "true",
    "usernameFormat": "username",
    "usernameCase": "preserve",
    "certFile": "/path/to/cert_file",
    "keyFile": "/path/to/key_file"
  }
}
```

Configuration file reference

This table includes all of the options that can be included with the "gatewaySettings" entity set.

enabled

- **Required.**
- **Values:** true or false

usernameFormat

- **Values:** username, domain_and_username, or email
  - Specifies user name credential format.

usernameCase

- **Values:** lower, upper, or preserve
  - Specifies user input name case.
certFile

Specifies file path and name for the certificate file. For example, "/var/-opt/tableau/tableau_server/data/saml/saml_cert.crt".

keyFile

Specifies file path and name for the certificate key. For example, "/var/-opt/tableau/tableau_server/data/saml/saml_key.der".

trustedAuthenticationSettings Entity

Before you configure trusted authentication, review How Trusted Authentication Works.

Use the configuration file template below to create a json file. After you have filled in the options with the appropriate values, pass the json file and apply settings with the following commands:

tsm settings import -f /path/to/file.json

tsm pending-changes apply

The pending-changes apply command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the -r option, but this does not change the restart behavior. For more information, see tsm pending-changes apply.

Configuration template

Use this template to configure trusted authentication settings.

All entity options are case sensitive.

For more explanation about configuration files, entities, and keys see Configuration File Example.
After you have finished with the initial configuration of trusted authentication, use the tsm authentication trusted <commands> sub-category to set additional values.

```json
{
    "configEntities": {
        "trustedAuthenticationSettings": {
            "_type": "trustedAuthenticationSettingsType",
            "trustedHosts": "required"
        }
    }
}
```

**Configuration file reference**

The following list includes all of the options that can be included with the "trustedAuthenticationSettings" entity set.

**trustedHosts**

Required.

IP address or host names of web servers that request trusted tickets from Tableau Server.

For multiple values, enter the names in a comma-separated list where each value is encapsulated in double-quotes.

For example:

```bash
tsm authentication trusted configure -th "192.168.1.101", "192.168.1.102", "192.168.1.103"
```

or

```bash
tsm authentication trusted configure -th "webserv1", "webserv2", "webserv3"
```

The values you specify overwrite previous settings. Therefore, you must include the full list of hosts when you configure this value.
tokenLength

Optional.

The value can be set to any integer between 9 and 255, inclusive.

Determines the number of characters in each trusted ticket. The default setting of 24 characters provides 144 bits of randomness. This option is ignored unless `useV2Tickets` is set to `true`, which is not a recommended best practice.

logLevel

Optional.

all | debug | info | warn | error | fatal | off

Default: info

Specifies logging level for processes related to creating and redeeming trusted tickets. See Change Logging Levels.

timeoutInSeconds

Optional.

Default: 180

Specifies the length of time (in seconds) to invalidate trusted tickets after they are created.

tryCount

Optional.

Integer.

Default: 10

Specifies the number of times to attempt to create a trusted ticket entry.

use9DigitToken
Optional.

true | false

Default: false

When set to true, tickets are 9 digits long (as in version 8.0 and earlier) and the option tokenLength is ignored. This option is intended for temporary support of legacy code.

Warning: Setting this option to true severely and negatively impacts the security strength of trusted ticket authentication.

useV2Tickets

Optional.

true | false

Default: false

Specifies whether Tableau Server should return a legacy URL format for trusted ticket requests. The legacy URL format includes a 24 character, Base64-encoded string. Beginning with Tableau Server 10.3, the URL that is returned has been updated and includes a Base64-encoded UUID and a 24 character secure random string. Only set this option this to true if you have deployed trusted tickets with custom code that requires the legacy URL format. We recommend instead, updating your custom code to accept the new URL format.

tabcmd

Tableau provides the tabcmd command-line utility which you can use to automate site administration tasks on your Tableau Server site. For example, creating or deleting users, projects, and groups.
This utility is included with Tableau Server, and is automatically installed on the initial server node. You can also run it from other computers, even computers that are not part of your Tableau Server installation, but to do so you need to download the tabcmd installer from the Tableau website. For more information, see Install tabcmd below.

In this article

- Install tabcmd
- How to use tabcmd
- Examples

Install tabcmd

When Tableau Server or Tableau Online is upgraded to a new version, if an updated version of tabcmd is required, you can download it from the Tableau Releases page on the Tableau website, to avoid issues caused by version incompatibilities. Using an out of date version of tabcmd can cause errors and unpredictable results.

1. Open a web browser and go to the Tableau Server Releases page (do this even if you use Tableau Online).

2. **For Tableau Online**: select the latest release.

   **For Tableau Server (Windows or Linux)**: select the release that matches your server version.

   If the expanded information shows maintenance releases, select the latest maintenance release or the one that matches your server version.
This takes you to the release notes page, where you can read about security improvements and resolved issues.

3. Scroll to the **Download Files** section under the resolved issues, select the tabcmd download link that is compatible with the computer on which you'll run the tabcmd commands.

   The remaining steps refer to this computer as "the tabcmd computer."

4. Save the installer to the tabcmd computer, or a location accessible from that computer.
(a mounted drive, for example).

5. Complete the installation steps as appropriate for the operating system of the tabcmd computer:

**Windows**

Run the installer.

We recommend that you install tabcmd to a folder named **tabcmd** at the root of the C:\ drive (C:\**tabcmd**), for convenience when using tabcmd, and to accommodate some limitations with the Windows operating system.

**Note** The tabcmd Setup program does not add the tabcmd directory to the Windows PATH variable. You can add it manually, or you can include the full path to tabcmd each time you call it.

**Linux**

a. Log on as a user with sudo access to the tabcmd computer.

b. Navigate to the directory where you copied the .rpm or .deb package that you downloaded.

   - On RHEL-like distributions, including CentOS, run the following command:

     ```bash
     sudo yum install tableau-tabcmd-<version>.noarch.rpm
     ```

   - On Ubuntu, run the following command:

     ```bash
     sudo gdebi -i tableau-tabcmd-<version>_all.deb
     ```
How to use tabcmd

The basic steps for using tabcmd are as follows:

1. Open the Command Prompt as an administrator.

2. On a Windows computer, change to the directory where tabcmd is installed.

   If you are using tabcmd on the Tableau Server on Windows primary node, change to the Tableau Server bin folder. For example:

   ```
cd C:\Program Files\Tableau\Tableau Server\10.5\bin
   ```

   If you installed tabcmd on a computer other than the primary node, change to the directory where you installed tabcmd.

   On a Linux computer, you do not need to change to the install directory.

3. Run the tabcmd command.

When you use tabcmd, you must establish an authenticated server session. The session identifies the server or Tableau Online site and the user running the session. You can start a session first, and then specify your command next, or you can start a session and execute a command all at once. If you are using tabcmd to perform more than one task, you must run tasks one after another (serially), rather than in parallel.

Commands (such as `login`) and the options (such as `-s`, `-u`, etc.) are not case sensitive, but the values you provide (such as `User@Example.com`) are case sensitive.

Examples

The following command demonstrates starting a session with the Tableau Server named `tabserver.mycompany.com`:

```
tabcmd login -s http://tabserver.mycompany.com -u admin -p mypassword
```
The next example shows a command that deletes a workbook named *Sales_Workbook*:

```bash
tabcmd delete "Sales_Workbook"
```

Here’s how to accomplish all of the above with one command—note that you do not need login here:

```bash
tabcmd delete "Sales_Workbook" -s http://tabserver.mycompany.com -u admin -p mypassword
```

A Tableau Server can run multiple sites. When a workbook is on the Default site of a multi-site server you don't need to specify Default, the above command is sufficient. However, if the command applies to something on a site other than Default, you need to specify the site ID for that site (see login). Here’s the same command for a workbook that’s on the West Coast Sales site (site ID wsales):

```bash
tabcmd delete "Sales_Workbook" -s http://tabserver.mycompany.com -t wsales -u admin -p mypassword
```

The options `-s`, `-t`, `-u`, and `-p` are among the `tabcmd` global variables, which can be used with any command.

For more information, see `tabcmd` Commands.

**tabcmd Commands**

Looking for Tableau Server on Windows? See `tabcmd` Commands.

You can use the following commands with the `tabcmd` command line tool:

- `addusers` (to group)
- `creategroup`
- `createproject`
- `createsite`
- `createsiteusers`
- `createusers`
- `delete workbook-name or datasource-name`
addusers group-name

Adds users to the specified group.

Example

tabcmd addusers "Development" --users "users.csv"

Options

--users

Add the users in the given .csv file to the specified group. The file should be a simple
list with one user name per line. User names are not case sensitive. The users should already be created on Tableau Server.

For more information, see CSV Import File Guidelines.

--[no-]complete

When set to `complete` this option requires that all rows be valid for any change to succeed. If not specified, `--complete` is used.

Global options

The following options are used by all `tabcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.
-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided, the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck
When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/Scheet1 is a required value for the export command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/Scheet1
```

Top

creategroup group-name

Creates a group. Use addusers (for local groups) and syncgroup (for Active Directory groups) commands to add users after the group has been created.

Example

```
tabcmd creategroup "Development"
```

Global options
The following options are used by all `tabcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

- `h`, `--help`
  Displays the help for the command.

- `c`, `--use-certificate`
  Use client certificate to sign in. Required when mutual SSL is enabled.
  For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:
  - **Windows**: Configure Mutual SSL
  - **Linux**: Configure Mutual SSL

- `s`, `--server`
  The Tableau Server URL, which is required at least once to begin session.

- `u`, `--user`
  The Tableau Server username, which is required at least once to begin session.

- `p`, `--password`
  The Tableau Server password, which is required at least once to begin session.

- `--password-file`
  Allows the password to be stored in the given `.txt` file rather than the command line for increased security.
-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID.
If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.
--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/Sheet1 is a required value for the export command.

```bash
tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1
```

Top

createproject project-name

Creates a project.

Using tabcmd, you can specify only a top-level project in a project hierarchy. To automate tasks you want to perform on a project within a parent project, use the equivalent Tableau REST API call.

Example

```bash
tabcmd createproject -n "Quarterly_Reports" -d "Workbooks showing quarterly sales reports."
```

Options

- **-n, --name**

  Specifies the name of the project that you want to create.

- **-d, --description**

  Specifies a description for the project.

Global options
The following options are used by all `tabcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

- **-h, --help**

  Displays the help for the command.

- **-c, --use-certificate**

  Use client certificate to sign in. Required when mutual SSL is enabled.

  For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

  - **Windows**: Configure Mutual SSL
  - **Linux**: Configure Mutual SSL

- **-s, --server**

  The Tableau Server URL, which is required at least once to begin session.

- **-u, --user**

  The Tableau Server username, which is required at least once to begin session.

- **-p, --password**

  The Tableau Server password, which is required at least once to begin session.

- **--password-file**

  Allows the password to be stored in the given `.txt` file rather than the command line for increased security.
-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.
--

Specifies the end of options on the command line. You can use -- to indicate to `tabcmd` that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a `tabcmd` command, where `-430105/SHEET1` is a required value for the `export` command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1
```
Options

-r, --url

Used in URLs to specify the site. Different from the site name.

--user-quota

Maximum number of users that can be added to the site.

--[no-]site-mode

Allows or denies site administrators the ability to add users to or remove users from the site.

--storage-quota

In MB, the amount of workbooks, extracts, and data sources that can be stored on the site.

Global options

The following options are used by all tabcmd commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate
Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port
Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1
createsiteusers filename.csv

Adds users to a site, based on information supplied in a comma-separated values (CSV) file. If the user is not already created on the server, the command creates the user before adding that user to the site.

The CSV file must contain one or more user names and can also include (for each user) a password, full name, role, administrator level, publisher (yes/no), and email address. For information about the format of the CSV file, see CSV Import File Guidelines. As an alternative to including role, administrator level, and publisher permissions in the CSV file, you can pass role information to the command using the --role option.

By default, users are added to the site that you are logged in to. To add users to a different site, include the global --site option and specify that site. (You must have permissions to create users on the site you specify.)

If the server contains multiple sites, you cannot assign the ServerAdministrator role to a user by using the createsiteusers command. (Use createusers instead.) If you specify the ServerAdministrator role for the role option, the command returns an error. If the CSV file includes System as value for administrator, the value is ignored and the user is assigned the Unlicensed role. However, if the server contains only one site (the default site), you can assign the ServerAdministrator role or specify system for the administrator value; in that case, the createsiteusers command works like the createusers command.

By default, this command creates users using a synchronous operation (it waits for all operations to complete before proceeding). You can use the --no-wait option to specify an asynchronous operation.

Local authentication

If the server is configured to use local authentication, the information in the CSV file is used to create users.
Active Directory authentication

If the server is configured to use Active Directory authentication, user information is imported from Active Directory to the server. In that case, any password and friendly name information in the CSV file is ignored. Further, if a user is specified in the CSV file but there is no corresponding user in Active Directory, the user is not added to Tableau Server. For Active Directory users, the user name is not guaranteed to be unique across domains, therefore you must include the domain as part of the user name (for example, example\Adam or adam@example.com.

While these can be sent either as domain/username or username@domain.com, we recommend using the domain/username format. See User Management in Active Directory Deployments for more information.

Example

tabcmd createsiteusers "users.csv" --role "Interactor"

Options

--admin-type

Deprecated. Use the --role option instead.

Use Site to assign or None to remove the site administrator right for any user who does not already have an administrator setting in the .csv file. The default is None for new users and unchanged for existing users. If the server contains multiple sites; system administrators cannot be created or demoted using createsiteusers. (Use createusers instead.)

--complete

Requires that all rows be valid for any change to succeed. This is the default setting.

--license

Deprecated. Use the --role option instead.
Specifies the license level (Interactor, Viewer, or Unlicensed) for any user who
does not already have a license level setting in the .csv file. The default is Unli-
censed for new users and unchanged for existing users.

**Note:** License levels were used in earlier versions of Tableau. They were
replaced by site roles starting in Tableau Server 9.0.

```
--no-complete
```

Specifies that the command should make changes on the server even if not all rows
contain valid information. Rows that contain invalid information are skipped.

```
--no-publisher
```

Deprecated. Use the **--role** option instead.

Disallows publishing rights for any users who do not already have a publisher setting in
the .csv file. This is a default value for new users.

```
--nowait
```

Do not wait for asynchronous jobs to complete.

```
--publisher
```

Deprecated. Use the **--role** option instead.

Assigns publishing rights for any user who does not already have a publisher setting in
the .csv file. The default is no publishing rights (equivalent to **--no-publish**) for new
users and unchanged for existing users.

```
-r, --role
```

Specifies a site role (ServerAdministrator, SiteAdministrator, Publisher,
Interactor, ViewerWithPublish, Viewer, UnlicensedWithPublish, or Unlicensed) for any user who does not already have a role specified in the .csv file. The default is Unlicensed for new users and unchanged for existing users.

If you have a user-based server installation, and if the command creates a new user but you have already reached the limit on the number of licenses for your users, the user is added as an unlicensed user.

**Note:** On Tableau Server, you cannot assign the ServerAdministrator role if the server has more than one site. In that case, use the `createuser` command.

**Note:** If you specify a role option, you cannot also include license, publisher, no-publisher, or administrator options.

--silent-progress

Do not display progress messages for the command.

Global options

The following options are used by all `tabcmd` commands. The `--server, --user, and --password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate
Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

- `-s, --server`

  The Tableau Server URL, which is required at least once to begin session.

- `-u, --user`

  The Tableau Server username, which is required at least once to begin session.

- `-p, --password`

  The Tableau Server password, which is required at least once to begin session.

- `--password-file`

  Allows the password to be stored in the given `.txt` file rather than the command line for increased security.

- `-t, --site`

  Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

- `-x, --proxy`

  Host:Port
Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1
Top

createusers filename.csv

Create users in Tableau Server, based on information supplied in a comma-separated values (CSV) file.

The CSV file must contain one or more user names and can also include (for each user) a password, full name, role, administrator level, publisher (yes/no), and email address. For information about the format of the CSV file, see CSV Import File Guidelines. As an alternative to including role, administrator level, and publisher permissions in the CSV file, you can pass role information to the command using the --role option.

If the server has only one site (the default site), the user is created and added to the site. If the server has multiple sites, the user is created but is not added to any site. To add users to a site, use createsiteusers.

If you have a user-based server installation, and if the command creates a new user but you have already reached the limit on the number of licenses for your users, the user is added as an unlicensed user.

Local authentication

If the server is configured to use local authentication, the information in the CSV file is used to create users.

Active Directory authentication

If the server is configured to use Active Directory authentication, user information is imported from Active Directory to the server. In that case, any password and friendly name information in the CSV file is ignored. Further, if a user is specified in the CSV file but there is no corresponding user in Active Directory, the user is not added to Tableau Server. For Active Directory users, the user name is not guaranteed to be unique across domains, therefore you must include the domain as part of the user name (for example, example\Adam or adam@example.com).
While these can be sent either as `domain/username` or `username@domain.com`, we recommend using the `domain/username` format. See User Management in Active Directory Deployments for more information.

**Example**

```bash
tabcmd createusers "users.csv" --role "ServerAdministrator"
```

```bash
tabcmd createusers "users.csv"
```

**Options**

**--admin-type**

Deprecated. Use the `--role` option instead.

Assigns or removes the site administrator right (`Site` or `None`) for any user who does not already have an administrator setting in the `.csv` file. The default is `None` for new users and unchanged for existing users.

**--complete**

Requires that all rows be valid for any change to succeed. This is the default setting.

**--license**

Deprecated. Use the `--role` option instead.

Specifies the license level (`Interactor`, `Viewer`, or `Unlicensed`) for any user who does not already have a license level setting in the `.csv` file. The default is `Unlicensed` for new users and unchanged for existing users.

**Note:** License levels were used in earlier versions of Tableau Server, but have been replaced by site roles starting with Tableau Server 9.0.
--no-complete

Specifies that the command should make changes on the server even if not all rows contain valid information. Rows that contain invalid information are skipped.

--no-publisher

Deprecated. Use the --role option instead.

Disallows publishing rights for any users who do not already have a publisher setting in the .csv file. This is a default value for new users.

--nowait

Do not wait for asynchronous jobs to complete.

--publisher

Deprecated. Use the --role option instead.

Assigns publishing rights for any user who does not already have a publisher setting in the .csv file. The default is no publishing rights (equivalent to --no-publish) for new users and unchanged for existing users.

-r, --role

Specifies a role (ServerAdministrator, SiteAdministrator, Publisher, Interactor, ViewerWithPublish, Viewer, UnlicensedWithPublish, or Unlicensed) for any user who does not already have a role specified in the .csv file. The default is Unlicensed for new users and unchanged for existing users.

On a multi-site server, the command does not assign the user to a site. Therefore, the only roles that the command will assign are ServerAdministrator and Unlicensed. In that case, if you specify a different role (like Publisher or Viewer), the command assigns the Unlicensed role.
On a single-site server, the user is created and added to the default site using the role that you specify.

If you have a user-based server installation, and if the command creates a new user but you have already reached the limit on the number of licenses for your users, the user is added as an unlicensed user.

**Note:** If you specify a role option, you cannot also include license, publisher, no-publisher, or administrator options.

```bash
--silent-progress
```

Do not display progress messages for the command.

**Global options**

The following options are used by all `tabcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

```bash
-h, --help
```

Displays the help for the command.

```bash
-c, --use-certificate
```

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:
- **Windows:** Configure Mutual SSL

- **Linux:** Configure Mutual SSL

- `s, --server`
  
  The Tableau Server URL, which is required at least once to begin session.

- `u, --user`
  
  The Tableau Server username, which is required at least once to begin session.

- `p, --password`
  
  The Tableau Server password, which is required at least once to begin session.

- `--password-file`
  
  Allows the password to be stored in the given .txt file rather than the command line for increased security.

- `t, --site`
  
  Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

- `x, --proxy`
  
  Host:Port
  
  Uses the specified HTTP proxy.

- `--no-prompt`
  
  When specified, the command will not prompt for a password. If no valid password is
provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server’s SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1
```

### Top

delete **workbook-name** or **datasource-name**

Deletes the specified workbook or data source from the server.
This command takes the name of the workbook or data source as it is on the server, not the file name when it was published.

Example

tabcmd delete "Sales_Analysis"

Options

-\texttt{r, --project}

\texttt{The name of the project containing the workbook or data source you want to delete. If not specified, the “Default” project is assumed.}

\texttt{Using tabcmd, you can specify only a top-level project in a project hierarchy. To automate tasks you want to perform on a project within a parent project, use the equivalent Tableau REST API call.}

-\texttt{--workbook}

\texttt{The name of the workbook you want to delete.}

-\texttt{--datasource}

\texttt{The name of the data source you want to delete.}

Global options

The following options are used by all tabcmd commands. The \texttt{--server, --user, and --password} options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-\texttt{h, --help}
Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.
-x, --proxy
    Host:Port

    Uses the specified HTTP proxy.

--no-prompt

    When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

    When specified, an HTTP proxy will not be used.

--no-certcheck

    When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

    When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

    Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

    Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows
how you might use -- in a `tabcmd` command, where `-430105/Sheet1` is a required value for the `export` command.

    tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1

**Top**

deletegroup *group-name*

Deletes the specified group from the server.

**Example**

    tabcmd deletegroup "Development"

**Global options**

The following options are used by all `tabcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

- `-h, --help`

  Displays the help for the command.

- `-c, --use-certificate`

  Use client certificate to sign in. Required when mutual SSL is enabled.

  For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:
- **Windows:** Configure Mutual SSL
- **Linux:** Configure Mutual SSL

- `s, --server`
  The Tableau Server URL, which is required at least once to begin session.

- `u, --user`
  The Tableau Server username, which is required at least once to begin session.

- `p, --password`
  The Tableau Server password, which is required at least once to begin session.

- `--password-file`
  Allows the password to be stored in the given .txt file rather than the command line for increased security.

- `t, --site`
  Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

- `x, --proxy`
  Host:Port
  Uses the specified HTTP proxy.

- `--no-prompt`
  When specified, the command will not prompt for a password. If no valid password is
provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/Sheet1 is a required value for the export command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1
```

Top

deleteproject project-name

Deletes the specified project from the server.
Using `tabcmd`, you can specify only a top-level project in a project hierarchy. To automate tasks you want to perform on a project within a parent project, use the equivalent Tableau REST API call.

**Example**

`tabcmd deleteproject "Designs"`

**Global options**

The following options are used by all `tabcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

- `h`, `--help`
  
  Displays the help for the command.

- `c`, `--use-certificate`
  
  Use client certificate to sign in. Required when mutual SSL is enabled.

  For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

  - **Windows**: Configure Mutual SSL
  - **Linux**: Configure Mutual SSL

- `s`, `--server`
  
  The Tableau Server URL, which is required at least once to begin session.

- `u`, `--user`
The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port
Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server’s SSL certificate.
--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1

Top

deletesite site-name

Deletes the specified site from the server.

Example

tabcmd deletesite "Development"

Global options

The following options are used by all tabcmd commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token
is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

- `h, --help`
  Displays the help for the command.

- `c, --use-certificate`
  Use client certificate to sign in. Required when mutual SSL is enabled.
  For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:
  
  - **Windows**: Configure Mutual SSL
  - **Linux**: Configure Mutual SSL

- `s, --server`
  The Tableau Server URL, which is required at least once to begin session.

- `u, --user`
  The Tableau Server username, which is required at least once to begin session.

- `p, --password`
  The Tableau Server password, which is required at least once to begin session.

- `--password-file`
  Allows the password to be stored in the given .txt file rather than the command line for increased security.

- `t, --site`
Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server’s SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.
Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

```plaintext
tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1
```

**Top**

**deletesiteusers filename.csv**

Removes users from from the site that you are logged in to. The users to be removed are specified in a file that contains a simple list of one user name per line. (No additional information is required beyond the user name.)

By default, if the server has only one site, or if the user belongs to only one site, the user is also removed from the server. On a Tableau Server Enterprise installation, if the server contains multiple sites, users who are assigned the role of **Server Administrator** are removed from the site but are not removed from the server.

If the user owns content, the user’s role is change to **Unlicensed**, but the user is not removed from the server or the site. The content is still owned by that user. To remove the user completely, you must change the owner of the content and then try removing the user again.

If the user was imported from Active Directory, the user is removed from the site and possibly from the server. However, the user is not deleted from Active Directory.

**Example**

```plaintext
tabcmd deletesiteusers "users.csv"
```

**Global options**

The following options are used by all tabcmd commands. The **--server**, **--user**, and --
-password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.
-t, --site
Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy
Host:Port
Uses the specified HTTP proxy.

--no-prompt
When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy
When specified, an HTTP proxy will not be used.

--no-certcheck
When specified, tabcmd (the client) does not validate the server's SSL certificate.

[--no-]cookie
When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout
Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.
Specifies the end of options on the command line. You can use `--` to indicate to `tabcmd` that anything that follows `--` should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use `--` in a `tabcmd` command, where `-430105/SHEET1` is a required value for the `export` command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1
```

**Top**

**deleteusers filename.csv**

Deletes the users listed in the specified comma-separated values (.csv) file.

The .csv file should contain a simple list of one user name per line.

**Example**

```
tabcmd deleteusers "users.csv"
```

**Options**

```
--[no-]complete
```

When set to `--complete` this option requires that all rows be valid for any change to succeed. If not specified, `--complete` is used.

**Global options**

The following options are used by all `tabcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.
-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with
multiple sites.

-\(x\), \(--\text{proxy}\)

**Host:Port**

Uses the specified HTTP proxy.

\(--\text{no-prompt}\)

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

\(--\text{no-proxy}\)

When specified, an HTTP proxy will not be used.

\(--\text{no-certcheck}\)

When specified, tabcmd (the client) does not validate the server’s SSL certificate.

\(--[\text{no-}]\text{cookie}\)

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

\(--\text{timeout}\)

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.


delimiter

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and
can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1
```

**Top**

**editdomain**

Changes the nickname or full domain name of an Active Directory domain on the server. A domain “nickname” is the Windows NetBIOS domain name.

You can modify the nickname for any domain the server is using. In general, you can modify the full domain name for any domain except the one that you used to sign in. However, if the user name that you are currently signed in with exists in both the current domain and the new domain, you can modify the full name for the current domain.

Review **User Management in Active Directory Deployments** to understand how multiple domains, domain name mapping, and user names interact with Tableau Server.

To see a list of domains, use **listdomains**.

**Examples**

```
tabcmd editdomain --id 2 --nickname "new-nickname"
```

```
tabcmd editdomain --id 3 --name "new-name"
```

**Options**

```
--id

The ID of domain to change. To get a list of domain IDs, use use **listdomains**.

--name
```
The new name for the domain.

--nickname

The new nickname for the domain.

Global options

The following options are used by all tabcmd commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- Windows: Configure Mutual SSL
- Linux: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.
-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server’s SSL certificate.

--[no-]cookie
When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

`--timeout`

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

`--`

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/Series1 is a required value for the export command.

```
tabcmd export --csv -f "D:\export10.csv" -430105/Series1
```

**Top**

**editsite site-name**

Changes the name of a site or its web folder name. You can also use this command to allow or deny site administrators the ability to add and remove users. If site administrators have user management rights, you can specify how many users they can add to a site.

**Examples**

```
tabcmd editsite wc_sales --site-name "West Coast Sales"
tabcmd editsite wc_sales --site-id "wsales"
tabcmd editsite wsales --status ACTIVE
tabcmd editsite wsales --user-quota 50
```
Options

--site-name

The name of the site that's displayed.

--site-id

Used in the URL to uniquely identify the site.

--user-quota

Maximum number of users who can be members of the site.

--[no-]site-mode

Allow or prevent site administrators from adding users to the site.

--status

Set to ACTIVE to activate a site, or to SUSPENDED to suspend a site.

--storage-quota

In MB, the amount of workbooks, extracts, and data sources that can be stored on the site.

Global options

The following options are used by all tabcmd commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.
-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with
multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server’s SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and
can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1

Top export

Exports a view or workbook from Tableau Server and saves it to a file. This command can also export just the data used for a view.

Note the following when you use this command:

- **Permissions:** To export, you must have the Export Image permission. By default, this permission is Allowed or Inherited for all roles, although permissions can be set per workbook or view.

- **Exporting data:** To export just the data for a view, use the --csv option. This exports the summary data used in a view to a .csv file.

- **Specifying the view, workbook, or data to export:**
  
  - Use part of the URL to identify what to export, specifically the "workbook/view" string as it appears in the URL for the workbook or view. Do not use the “friendly name,” and exclude the :iid=<n> session ID at the end of the URL.

  For example, the Tableau sample view Global Temperatures in the Regional workbook has a URL similar to this: <server_name>/#/views/Regional/GlobalTemperatures?:iid=3

  To export the Global Temperatures view, use the string Regional/GlobalTemperatures.
Do not use `Regional/Global Temperatures, or Regional/GlobalTemperatures?:iid=3`.

- If the server is running multiple sites and the view or workbook is on a site other than Default, Use `-t <site_id>`.

- To export a workbook, get the URL string by opening a view in the workbook, and include the view in the string you use.

  In the above example, to export the `Regional` workbook, use the string `Regional/GlobalTemperatures`.

- To export a workbook, it must have been published with Show Sheets as Tabs selected in the Tableau Desktop Publish dialog box.

- **The saved file's format:** Your format options depend on what's being exported. A workbook can only be exported as a PDF using the `--fullpdf` argument. A view can be exported as a PDF (`--pdf`) or a PNG (`--png`).

- **The saved file's name and location** (optional): If you don't provide a name, it will be derived from the view or workbook name. If you don't provide a location, the file will be saved to your current working directory. Otherwise, you can specify a full path or one that's relative to your current working directory.

  **Note:** You must include a file name extension such as `.csv` or `.pdf`. The command does not automatically add an extension to the file name that you provide.

- **Dashboard web page objects not included in PDF exports:** A dashboard can optionally include a web page object. If you are performing an export to PDF of a dashboard that includes a web page object, the web page object won't be included in the PDF.

- **Non-ASCII characters and PDF exports:** If you are exporting a view or workbook
with a name that includes a character outside the ASCII character set, you need to URL encode (or percent-encode) the character.

For example if your command includes the city Zürich, you need to URL encode it as Z%C3%BCrich:

```
tabcmd export "//Cities/Sheet1?locationCity=Z%C3%BCrich" -fullpdf
```

### Clearing the Cache to Use Real-Time Data

You can optionally add the URL parameter `?:refresh=yes` to force a fresh data query instead of pulling the results from the cache. If you are using `tabcmd` with your own scripting and the `refresh` URL parameter is being used a great deal, this can have a negative impact on performance. It's recommended that you use `refresh` only when real-time data is required—for example, on a single dashboard instead of on an entire workbook.

### Examples

#### Views

```
tabcmd export "Q1Sales/Sales_Report" --csv -f "Weekly-Report.csv"
```

```
tabcmd export -t Sales "Sales/Sales_Analysis" --pdf -f "C:\Tableau_Workbooks\Weekly-Reports.pdf"
```

```
tabcmd export "Finance/InvestmentGrowth" --png
```

```
tabcmd export "Finance/InvestmentGrowth?:refresh=yes" --png
```

#### Workbooks

```
tabcmd export "Q1Sales/Sales_Report" --fullpdf
```

```
tabcmd export "Sales/Sales_Analysis" --fullpdf --pagesize tabloid -f "C:\Tableau_Workbooks\Weekly-Reports.pdf"
```
Options

-f, --filename

Saves the file with the given filename and extension.

--csv

View only. Export the view's data (summary data) in .csv format.

--pdf

View only. Export as a PDF.

--png

View only. Export as an image in .png format.

--fullpdf

Workbook only. Export as a PDF. The workbook must have been published with Show Sheets as Tabs enabled.

--pagelayout

Sets the page orientation (landscape or portrait) of the exported PDF. If not specified, its Tableau Desktop setting will be used.

--pagesize

Sets the page size of the exported PDF as one of the following: unspecified, letter, legal, note folio, tabloid, ledger, statement, executive, a3, a4, a5, b4, b5, or quarto. Default is letter.

--width
Sets the width in pixels. Default is 800 px.

--height

Sets the height in pixels. Default is 600 px.

Global options

The following options are used by all tabcmd commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL

- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.
-p, --password

   The Tableau Server password, which is required at least once to begin session.

--password-file

   Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

   Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

   Host:Port

   Uses the specified HTTP proxy.

--no-prompt

   When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

   When specified, an HTTP proxy will not be used.

--no-certcheck

   When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie
When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

`--timeout`

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

`--`

Specifies the end of options on the command line. You can use -- to indicate to `tabcmd` that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a `tabcmd` command, where `-430105/Sheet1` is a required value for the `export` command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1
```

**Top**

**get url**

 Gets the resource from Tableau Server that's represented by the specified (partial) URL. The result is returned as a file.

Note the following when you use this command:

- **Permissions**: To get a file, you must have the **Download/Web Save As** permission. By default, this permission is allowed or inherited for all roles, although permissions can be set per workbook or view.

- **Specifying a view or workbook to get**: You specify a view to get using the `"/views/<workbookname>/<viewname>.<extension>"` string, and specify a workbook to get using the
"/workbooks/<workbookname>..<extension>" string. Replace <workbookname> and <viewname> with the names of the workbook and view as they appear in the URL when you open the view in a browser and replace <extension> with the type of file you want to save. Do not use the session ID at the end of the URL (?:iid=<n>) or the "friendly" name of the workbook or view.

For example, when you open a view *Regional Totals* in a workbook named *Metrics Summary*, the URL will look similar to this:

/views/MetricsSummary_1/RegionalTotals?:iid=1

Use the string /views/MetricsSummary_1/RegionalTotals.<extension> to get the view.

Use the string /workbooks/MetricsSummary_1.<extension> to get the workbook.

- **File extension**: The URL must include a file extension. The extension determines what's returned. A view can be returned in PDF, PNG, or CSV (summary data only) format. A Tableau workbook is returned as a TWB if it connects to a published data source or uses a live connection, or a TWBX if it connects to a data extract.

  **Note**: If you are downloading a view to a PDF or PNG file, and if you include a --filename parameter that includes the .pdf or .png extension, you do not have to include a .pdf or .png extension in the URL.

- **The saved file’s name and location** (optional): The name you use for --filename should include the file extension. If you don’t provide a name and file extension, both will be derived from the URL string. If you don’t provide a location, the file is saved to your current working directory. Otherwise, you can specify a full path or one that's relative to your current working directory.

- **PNG size** (optional): If the saved file is a PNG, you can specify the size, in pixels, in
Clearing the cache to use real-time data

You can optionally add the URL parameter 
\( \text{refresh=yes} \) to force a fresh data query instead of pulling the results from the cache. If you are using tabcmd with your own scripting, using the \text{refresh} parameter a great deal can have a negative impact on performance. It's recommended that you use \text{refresh} only when real-time data is required—for example, on a single dashboard instead of on an entire workbook.

Examples

Views

\[
\text{tabcmd get } \text{"/views/Sales_Analysis/Sales_Report.png" --filename "Weekly-Report.png"}
\]

\[
\text{tabcmd get } \text{"/views/Finance/InvestmentGrowth.pdf" -f "Q1Growth.pdf"}
\]

\[
\text{tabcmd get } \text{"/views/Finance/InvestmentGrowth" -f "Q1Growth.pdf"}
\]

\[
\text{tabcmd get } \text{"/views/Finance/InvestmentGrowth.csv"}
\]

\[
\text{tabcmd get } \text{"/views/Finance/InvestmentGrowth.png?:size=640,480" -f growth.png}
\]

\[
\text{tabcmd get } \text{"/views/Finance/InvestmentGrowth.png?:refresh=yes" -f growth.png}
\]

Workbooks

\[
\text{tabcmd get } \text{"/workbooks/Sales_Analysis.twb" -f "C:\Tableau_Workbooks\Weekly-Reports.twb"
\]

Global options
The following options are used by all `tabcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given `.txt` file rather than the command line for increased security.
-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

[--no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.
--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1
```

Top

initialuser

Create the initial administrative user on a server that does not have an initial administrative user defined.

Enclose values in single quotes.

**Note:** The `tabcmd initialuser` command does not require authentication to Tableau Server, but you must run the command on the initial (primary) server node.

Examples

```
tabcmd initialuser --username 'admin' --password 'password' --server http://localhost
```

```
tabcmd initialuser --username 'admin' --password 'password' --friendly 'Tableau Admin' --server http://localhost
```

Options

- **-f, --friendly**

  Creates the initial administrative user with the display name.
Global options

The following options are used by all `tabcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

- `h`, `--help`
  
  Displays the help for the command.

- `c`, `--use-certificate`
  
  Use client certificate to sign in. Required when mutual SSL is enabled.
  
  For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:
  
  - **Windows**: Configure Mutual SSL
  
  - **Linux**: Configure Mutual SSL

- `s`, `--server`
  
  The Tableau Server URL, which is required at least once to begin session.

- `u`, `--user`
  
  The Tableau Server username, which is required at least once to begin session.

- `p`, `--password`
  
  The Tableau Server password, which is required at least once to begin session.

- `--password-file`
Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout
Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/Sheet1 is a required value for the export command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1
```

Top

listdomains

Displays a list of the Active Directory domains that are in use on the server, along with their nicknames and IDs. If the server is configured to use local authentication, the command returns only the domain name local.

Example

```
tabcmd listdomains
```

Global options

The following options are used by all tabcmd commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

```
-h, --help
```
Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.
-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server’s SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows
how you might use -- in a `tabcmd` command, where `-430105/Sheet1` is a required value for the `export` command.

```bash
tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1
```

**Top**

**listsites**

Returns a list of sites to which the logged in user belongs.

**Example**

```bash
tabcmd listsites --username adam --password mypassword
```

**Global options**

The following options are used by all `tabcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

- `-h`, `--help`

  Displays the help for the command.

- `-c`, `--use-certificate`

  Use client certificate to sign in. Required when mutual SSL is enabled.

  For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:
- **Windows:** Configure Mutual SSL

- **Linux:** Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is
provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-].cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/Sheet1 is a required value for the export command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1
```

Top

login

Logs in a Tableau Server user.
Use the `--server,--site,--username,--password` global options to create a session.

**Note:** When you use the `tabcmd login` command, you cannot use SAML single sign-on (SSO), even if the server is configured to use SAML. To log in, you must pass the user name and password of a user who has been created on the server. You will have the permissions of the Tableau Server user that you’re signed in as. For more information, see Set Users’ Site Roles and Content Permissions and Ownership.

If you want to log in using the same information you’ve already used to create a session, just specify the `--password` option. The server and user name stored in the cookie will be used.

If the server is using a port other than 80 (the default), you will need to specify the port.

You need the `--site (-t)` option only if the server is running multiple sites and you are logging in to a site other than the Default site. If you do not provide a password you will be prompted for one. If the `--no-prompt` option is specified and no password is provided the command will fail.

Once you log in, the session will continue until it expires on the server or the `logout` command is run.

**Example**

Logs user jsmith in to the Tableau Server running on their local machine:

```
tabcmd login -s http://localhost -u jsmith -p password
```

Logs administrator in to the Sales site on sales-server:

```
tabcmd login -s http://sales-server -t Sales -u administrator -p password
```
tabcmd login -s http://sales-server:8000 -t Sales -u administrator -p password

Logs administrator in to the Sales site on sales-server using SSL, but does not validate the server’s SSL certificate:

tabcmd login --no-certcheck -s https://sales-server -t Sales -u administrator -p password

Establishes a forward proxy and port for localhost:

tabcmd login --proxy myfwdproxyserver:8888 -s http://localhost -u jsmith -p password

Logs user jsmith in to the reverse proxy using SSL:

tabcmd login -s https://myreverseproxy -u jsmith -p password

Options

-s, --server

If you are running the command from a Tableau Server computer that's on your network, you can use http://localhost. Otherwise, specify the computer’s URL, such as http://bigbox.myco.com or http://bigbox.

For Tableau Online specify the URL https://online.tableau.com.

-t, --site

Include this option if the server has multiple sites, and you are logging in to a site other than the default site.

The site ID is used in the URL to uniquely identify the site. For example, a site named West Coast Sales might have a site ID of west-coast-sales.

-u, --username
The user name of the user logging in. For Tableau Online, the user name is the user’s email address.

-p, --password

Password for the user specified for --username. If you do not provide a password you will be prompted for one.

--password-file

Allows the password to be stored in the given filename.txt file rather than the command line, for increased security.

-x, --proxy

Use to specify the HTTP proxy server and port (Host:Port) for the tabcmd request.

--no-prompt

Do not prompt for a password. If no password is specified, the login command will fail.

--no-proxy

Do not use an HTTP proxy server.

--cookie

Saves the session ID on login. Subsequent commands will not require a login. This value is the default for the command.

--no-cookie

Do not save the session ID information after a successful login. Subsequent commands will require a login.
--timeout SECONDS

The number of seconds the server should wait before processing the login command. Default: 30 seconds.

Global options

The following options are used by all tabcmd commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.
-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server’s SSL certificate.

--[no-]cookie
When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the **no-** prefix to not save the session ID. By default the session is saved.

```
--timeout
```

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

```
--
```

Specifies the end of options on the command line. You can use **--** to indicate to `tabcmd` that anything that follows **--** should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use **--** in a `tabcmd` command, where `-430105/Sheet1` is a required value for the `export` command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1
```

**Top**

**logout**

Logs out of the server.

**Example**

```
tabcmd logout
```

**Top**

**publish** *filename.twb(x), filename.tds(x)*, or *filename.hyper*

Publishes the specified workbook (*.twb(x)*) data source (*.tds(x)*), or extract (*.hyper*) to Tableau Server.
If you are publishing a workbook, by default, all sheets in the workbook are published without database user names or passwords.

The permissions initially assigned to the workbook or data source are copied from the project that the file is published to. Permissions for the published resource can be changed after the file has been published.

If the workbook contains user filters, one of the thumbnail options must be specified.

**Example**

```
tabcmd publish "analysis.twbx" -n "Sales_Analysis" --db-username "jsmith" --db-password "secret-password"
```

```
tabcmd publish "analysis_sfdc.hyper" -n "Sales Analysis" --oauth-username "user-name" --save-oauth
```

If the file is not in the same directory as tabcmd, include the full path to the file.

**Example**

```
tabcmd publish "\\computer\volume\Tableau Workbooks\analysis.twbx" -n "Sales_Analysis" --db-username "jsmith" --db-password "secret-password"
```

```
tabcmd publish "\\computer\volume\Tableau Workbooks\analysis_sfdc.hyper" -n "Sales Analysis" --oauth-username "username" --save-oauth
```

**Options**

- `-n, --name`

  Name of the workbook or data source on the server. If omitted, the workbook, data source, or data extract will be named after filename.

- `-o, --overwrite`
Overwrites the workbook, data source, or data extract if it already exists on the server.

-r, --project

Publishes the workbook, data source, or data extract into the specified project. Publishes to the “Default” project if not specified.

Using tabcmd, you can specify only a top-level project in a project hierarchy. To automate tasks you want to perform on a project within a parent project, use the equivalent Tableau REST API call.

--db-username

Use this option to publish a database user name with the workbook, data source, or data extract.

--db-password

Use this option to publish a database password with the workbook, data source, or extract.

--save-db-password

Stores the provided database password on the server.

--oauth-username

The email address of the user account. Connects the user through a preconfigured OAuth connection, if the user already has a saved access token for the cloud data source specified in --name. Access tokens are managed in user preferences.

For existing OAuth connections to the data source, use this option instead of --db-username and --db-password.

--save-oauth
Saves the credential specified by --oauth-username as an embedded credential with the published workbook or data source.

Subsequently, when the publisher or server administrator signs in to the server and edits the connection for that workbook or data source, the connection settings will show this OAuth credential as embedded in the content.

If you want to schedule extract refreshes after publishing, you must include this option with --oauth-username. This is analogous to using --save-db-password with a traditional database connection.

--thumbnail-username

If the workbook contains user filters, the thumbnails will be generated based on what the specified user can see. Cannot be specified when --thumbnail-group option is set.

--thumbnail-group

If the workbook contains user filters, the thumbnails will be generated based on what the specified group can see. Cannot be specified when --thumbnail-username option is set.

--tabbed

When a workbook with tabbed views is published, each sheet becomes a tab that viewers can use to navigate through the workbook. Note that this setting will override any sheet-level security.

--append

Append the extract file to the existing data source.

--replace

Use the extract file to replace the existing data source.
--disable-uploader

Disable the incremental file uploader.

--restart

Restart the file upload.

Global options

The following options are used by all tabcmd commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user
The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server’s SSL certificate.
--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1

Top

publishsamples

Description

Publishes Tableau Sample workbooks to the specified project. Any existing samples will be overwritten.

Syntax

tabcmd publishsamples -n [project name] [Global options]

Example

Publish samples to the Inside Sales project on the Default site, as user jsmith.
tabcmd publishesamples -n "Inside Sales" -t "" -s localhost --username "jsmith" --password "secret-password"

Options

-n, --name

Required. Publishes the Tableau samples into the specified project. If the project name includes spaces, enclose the entire name in quotes.

Using tabcmd, you can specify only a top-level project in a project hierarchy. To automate tasks you want to perform on a project within a parent project, use the equivalent Tableau REST API call.

Global options

The following options are used by all tabcmd commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:
- **Windows**: Configure Mutual SSL

- **Linux**: Configure Mutual SSL

- *s*, **--server**
  
  The Tableau Server URL, which is required at least once to begin session.

- *u*, **--user**
  
  The Tableau Server username, which is required at least once to begin session.

- *p*, **--password**
  
  The Tableau Server password, which is required at least once to begin session.

- **--password-file**
  
  Allows the password to be stored in the given .txt file rather than the command line for increased security.

- *t*, **--site**
  
  Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

- *x*, **--proxy**
  
  Host:Port
  
  Uses the specified HTTP proxy.

- **--no-prompt**
  
  When specified, the command will not prompt for a password. If no valid password is
provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server’s SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1
```

Top
refreshextracts **workbook-name** or **datasource-name**

Performs a full or incremental refresh of extracts belonging to the specified workbook or data source.

This command takes the name of the workbook or data source as it appears on the server, not the file name when it was published. Only an administrator or the owner of the workbook or data source is allowed to perform this operation.

**Examples**

```bash
tabcmd refreshextracts --datasource sales_ds

tabcmd refreshextracts --project "Sales External" --datasource sales_ds

tabcmd refreshextracts --workbook "My Workbook"

tabcmd refreshextracts --url SalesAnalysis
```

**Options**

**--incremental**

Runs the incremental refresh operation.

**--synchronous**

Adds the full refresh operation to the queue used by the Backgrounder process, to be run as soon as a Backgrounder process is available. If a Backgrounder process is available, the operation is run immediately. The refresh operation appears on the Background Tasks report.

During a synchronous refresh, *tabcmd* maintains a live connection to the server while the refresh operation is underway, polling every second until the background job is done.
--workbook

The name of the workbook containing extracts to refresh. If the workbook has spaces in its name, enclose it in quotes.

--datasource

The name of the data source containing extracts to refresh.

--project

Use with --workbook or --datasource to identify a workbook or data source in a project other than Default. If not specified, the Default project is assumed.

Using tabcmd, you can specify only a top-level project in a project hierarchy. To automate tasks you want to perform on a project within a parent project, use the equivalent Tableau REST API call.

--url

The name of the workbook as it appears in the URL. A workbook published as “Sales Analysis” has a URL name of “SalesAnalysis”.

Global options

The following options are used by all tabcmd commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.
-c, --use-certificate
  Use client certificate to sign in. Required when mutual SSL is enabled.
  For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:
  
  - Windows: Configure Mutual SSL
  - Linux: Configure Mutual SSL

-s, --server
  The Tableau Server URL, which is required at least once to begin session.

-u, --user
  The Tableau Server username, which is required at least once to begin session.

-p, --password
  The Tableau Server password, which is required at least once to begin session.

--password-file
  Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site
  Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy
  Host:Port
Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/Sheet1 is a required value for the export command.

tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1
removeusers *group-name*

Removes users from the specified group.

**Example**

```
tabcmd removeusers "Development" --users "users.csv"
```

**Options**

--users

Remove the users in the given .csv file from the specified group. The file should be a simple list with one user name per line.

--[no-]complete

Requires that all rows be valid for any change to succeed. If not specified --complete is used.

**Global options**

The following options are used by all *tabcmd* commands. The --server, --user, and --password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate
Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

`-s, --server`

The Tableau Server URL, which is required at least once to begin session.

`-u, --user`

The Tableau Server username, which is required at least once to begin session.

`-p, --password`

The Tableau Server password, which is required at least once to begin session.

`--password-file`

Allows the password to be stored in the given `.txt` file rather than the command line for increased security.

`-t, --site`

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

`-x, --proxy`

Host:Port
Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

---

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/SHEET1 is a required value for the export command.

```
tabcmd export --csv -f "D:export10.csv" -- -430105/SHEET1
```
Top

`runschedule schedule-name`

Runs the specified schedule.

This command takes the name of the schedule as it is on the server.

For Tableau Online, the command can be run within the scope of a single site, using site administrator permissions.

Example

tabcmd runschedule "5AM Sales Refresh"

Global options

The following options are used by all `tbcmd` commands. The `--server`, `--user`, and `--password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help

Displays the help for the command.

-c, --use-certificate

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:
- **Windows**: Configure Mutual SSL
  - **Linux**: Configure Mutual SSL

- `-s`, `--server`
  The Tableau Server URL, which is required at least once to begin session.

- `-u`, `--user`
  The Tableau Server username, which is required at least once to begin session.

- `-p`, `--password`
  The Tableau Server password, which is required at least once to begin session.

- `--password-file`
  Allows the password to be stored in the given `.txt` file rather than the command line for increased security.

- `-t`, `--site`
  Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

- `-x`, `--proxy`
  Host:Port
  Uses the specified HTTP proxy.

- `--no-prompt`
  When specified, the command will not prompt for a password. If no valid password is
provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/Sheet1 is a required value for the export command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1
```

Top
**set setting**

Enables the specified setting on the server. Details about each setting can be seen on the Maintenance page on the server.

Use an exclamation mark in front of the setting name to disable the setting. You can enable or disable the following settings:

- allow_scheduling
- embedded_credentials
- remember_passwords_forever

**Example**

tabcmd set embedded_credentials

**Global options**

The following options are used by all `tabcmd` commands. The `-server`, `-user`, and `-password` options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

- `h`, `--help`
  
  Displays the help for the command.

- `c`, `--use-certificate`
  
  Use client certificate to sign in. Required when mutual SSL is enabled.
For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows**: Configure Mutual SSL
- **Linux**: Configure Mutual SSL

-s, --server

The Tableau Server URL, which is required at least once to begin session.

-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.
--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where --430105/SHEET1 is a required value for the export command.

tabcmd export --csv -f "D:\export10.csv" --430105/SHEET1

Top
syncgroup group-name

Synchronizes a Tableau Server group with an Active Directory group. If the Tableau Server group does not already exist, it is created and synchronized with the specified Active Directory group.

If the group name itself includes an "@" (other than as the domain separator) you need to refer to the symbol using the hex format "\0x40".

Example

tabcmd syncgroup "Development"

tabcmd syncgroup "Dev\0x40West"

Note: If you synchronize a group that you are a member of, changes that you make using this command do not apply to your user. For example, if you use this command to remove the administrator right from users in a group that you are a member of, you are still an administrator when the command finishes.

Options

--administrator

Deprecated. Use the --role option instead.

Assigns or removes the System or Site administrator right for users in the group. The None option removes the administrator right from all users in the group (except you, if you are a member of the group that you are synchronizing). If you do not include this option, users who are added to the group after you run the command are not assigned the administrator right.

--license

Deprecated. Use the --role option instead.
Specifies the license level (Interactor, Viewer, or Unlicensed) for users in the group.

**Note:** License levels were used in earlier versions of Tableau Server, but have been replaced by site roles starting in Tableau Server 9.0.

```
--no-publisher
```

Deprecated. Use the **--role** option instead.

Disallows publishing rights for users in the group.

```
--overwritesiterole
```

Allows a user’s site role to be overwritten with a less privileged one when using **--role**. By default, a user site role can be promoted when using **--role**, but cannot be demoted. Because the **--overwritesiterole** option will demote user site roles, use it with caution.

```
--publisher
```

Deprecated. Use the **--role** option instead.

Assigns publishing rights to users in the group.

```
-r, --role
```

Specifies a role (ServerAdministrator, SiteAdministrator, Publisher, Interactor, ViewerWithPublish, Viewer, UnlicensedWithPublish, or Unlicensed) for users in the group. The default is Unlicensed.
**Note:** If you specify a **role** option, you cannot also include **license**, **publisher**, **no-publisher**, or **administrator** options.

```
--silent-progress
```

Do not display progress messages for the command.

**Global options**

The following options are used by all `tabcmd` commands. The **--server**, **--user**, and **--password** options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

```
-h, --help
```

Displays the help for the command.

```
-c, --use-certificate
```

Use client certificate to sign in. Required when mutual SSL is enabled.

For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- **Windows:** Configure Mutual SSL
- **Linux:** Configure Mutual SSL

```
-s, --server
```

The Tableau Server URL, which is required at least once to begin session.
-u, --user

The Tableau Server username, which is required at least once to begin session.

-p, --password

The Tableau Server password, which is required at least once to begin session.

--password-file

Allows the password to be stored in the given .txt file rather than the command line for increased security.

-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck
When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.

--

Specifies the end of options on the command line. You can use -- to indicate to tabcmd that anything that follows -- should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use -- in a tabcmd command, where -430105/Sheet1 is a required value for the export command.

    tabcmd export --csv -f "D:\export10.csv" -- -430105/Sheet1

Top

version

Displays the version information for the current installation of the tabcmd utility.

Example

tabcmd version

Global options

The following options are used by all tabcmd commands. The --server, --user, and --
-password options are required at least once to begin a session. An authentication token is stored so subsequent commands can be run without including these options. This token remains valid for five minutes after the last command that used it.

-h, --help
Displays the help for the command.

-c, --use-certificate
Use client certificate to sign in. Required when mutual SSL is enabled.
For information about configuring the certificate, start with the following topic appropriate for your Tableau Server OS:

- Windows: Configure Mutual SSL
- Linux: Configure Mutual SSL

-s, --server
The Tableau Server URL, which is required at least once to begin session.

-u, --user
The Tableau Server username, which is required at least once to begin session.

-p, --password
The Tableau Server password, which is required at least once to begin session.

--password-file
Allows the password to be stored in the given .txt file rather than the command line for increased security.
-t, --site

Indicates that the command applies to the site specified by the Tableau Server site ID. If you do not specify a site, the Default site is assumed. Applies only to servers with multiple sites.

-x, --proxy

Host:Port

Uses the specified HTTP proxy.

--no-prompt

When specified, the command will not prompt for a password. If no valid password is provided the command will fail.

--no-proxy

When specified, an HTTP proxy will not be used.

--no-certcheck

When specified, tabcmd (the client) does not validate the server's SSL certificate.

--[no-]cookie

When specified, the session ID is saved on login so subsequent commands will not need to log in. Use the no- prefix to not save the session ID. By default the session is saved.

--timeout

Waits the specified number of seconds for the server to complete processing the command. By default the process will timeout in 30 seconds.
Specifies the end of options on the command line. You can use `--` to indicate to `tabcmd` that anything that follows `--` should not be interpreted as an option setting and can instead be interpreted as a value for the command. This is useful if you need to specify a value in the command that includes a hyphen. The following example shows how you might use `--` in a `tabcmd` command, where `-430105/SHEET1` is a required value for the `export` command.

```
tabcmd export --csv -f "D:\export10.csv" -- -430105/SHEET1
```

### Troubleshooting

**Troubleshoot Tableau Server on Linux**

Follow the suggestions in this topic to resolve common issues with Tableau Server.

- **General Troubleshooting Steps**
  - Clean install
  - Disk space
  - Remove old log files
  - Manually gather logs
  - Check systemd logs
  - Restart server

- **Installing**
  - Hardware requirements
  - Timeouts
  - Temporary database
  - Parallels
  - Tableau doesn't start - restart
  - Tableau doesn't start - hostname changed
  - Initial admin user

- **Initializing**
  - Existing tableau user
  - Unsupported locale
The following table displays the default locations of the installation, data, logs, and scripts directories:

<table>
<thead>
<tr>
<th>Directory</th>
<th>Default location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installation:</td>
<td>/opt/tableau/tableau_server</td>
</tr>
<tr>
<td>Data:</td>
<td>/var/opt/tableau/tableau_server/data</td>
</tr>
<tr>
<td>Logs:</td>
<td>/var/opt/tableau/tableau_server/data/tabsvc/logs</td>
</tr>
<tr>
<td>Scripts:</td>
<td>/opt/tableau/tableau_server-/packages/scripts.&lt;version&gt;/</td>
</tr>
</tbody>
</table>

**General Troubleshooting Steps**

Many Tableau Server issues can be addressed or tested with one or more of these basic steps:

- **Clean install**—Install Tableau Server on Linux on a computer that has never had Tableau installed on it. If you are reusing a computer or VM that has had a previous
version of Tableau Server installed, follow the steps in Remove Tableau Server from Your Computer to clean Tableau off your computer before you install the new version.

If you run into problems installing Tableau Server you may need to entirely remove Tableau from your computer and do a clean install. See Remove Tableau Server from Your Computer for details.

- **Disk space**—Make sure there is enough disk space on each computer running Tableau Server. Limited disk space can cause a failure to install, a failure to upgrade, or problems running Tableau Server.

- **Remove old log files**—If you are running out of disk space you can clean up old Tableau Server log files. These can take up space and as a best practice you should remove them regularly.

**Version 10.5.x**

If you have version 10.5.1 and higher, run this command at a terminal prompt to clean up log files you do not need:

```
tsm maintenance cleanup
```

**Version 10.5.0**

If you are running version 10.5.0 of Tableau Server on Linux, the cleanup command is not available and you need to run these commands at a terminal prompt:

```
sudo find /var/opt/tableau/tableau_server-/data/tabsvc/temp/* -mtime +2 -type f -delete
```

```
sudo find /var/opt/tableau/tableau_server-/data/tabsvc/logs/* -mtime +2 -type f -delete
```

**Important:** The Linux file system makes it possible to delete files that are open and if you do this the Tableau processes may not be able to recreate the files. This will result
in empty log files. To fix this you can stop Tableau Server, restart the TSM Controller, and restart Tableau again:

1. Stop Tableau Server:

   tsm stop

2. Restart the TSM Controller:

   sudo systemctl restart tabadmincontroller_0.service

3. Wait several minutes for the controller to restart. You can confirm the controller has restarted with this command:

   tsm status -v

   When you can run that command and the Tableau Server Administration Controller is listed as 'running' the controller has restarted.

4. Start Tableau Server:

   tsm start

• Manually gather logs

   If you cannot run tsm maintenance ziplogs for any reason (for example, if you have a critical failure before you run tsm initialize), you can manually collect and zip the logs by running these commands in a terminal window:

   cd /var/opt/tableau/tableau_server/data/tabsvc/

   cp /var/opt/tableau/tableau_server/logs/app-install.log logs

   cp ~/.tableau/tsm/tsm.log logs
tar -czvf ~/logs.tar.gz logs

This creates a file called logs.tar.gz in your home directory. You can upload or send this file to Tableau.

- **Restart server**—Restart Tableau Server. Issues related to indexing and processes not fully started can be resolved by restarting Tableau Server in a controlled way. To restart Tableau Server, use the `tsm restart` command. This will stop all the processes associated with Tableau Server and then restart them.

- **Edit installation and configuration files using Linux**

  You should edit or create any files used to install or configure Tableau Server on Linux using a Linux operating system. Files created using Microsoft Windows will cause errors in Tableau Server on Linux installation and configuration because Linux operating systems end files with a line-feed (LF) character, whereas Windows ends files with a carriage-return character and a line-feed character (CR LF). Non-Linux (CR LF) file endings can cause errors during Automated Installation of Tableau Server if they appear in the `config.json`, `reg_templ.json` or `secrets` files used by the automated installer. Non-Linux (CR LF) file endings can also cause errors during registration or when configuring identity store settings or gateway settings.

- **Check systemd logs**

  If Tableau Server will not start, and you do not find anything useful in the Tableau logs (see Work with Log Files for more information) you can check the logs at `/var/-logs/messages` for messages related to the TSM Service starting and stopping.

**Installing Tableau Server**

Install fails due to hardware requirements

Tableau Server cannot install if the computer you are installing on does not meet the minimum hardware requirements. For details on requirements, see Confirm Requirements.
Install fails due to timeouts

If you install Tableau Server on a computer with limited resources, for example, a computer that just meets the minimum hardware requirements, you can run into problems where tsm commands timeout due to slow response. You can specify a longer timeout by using the `global --request-timeout` option on all tsm commands. For more information on the `--request-timeout` option, see for example, tsm initialize.

Install fails with "Failed to initialize the instance of the temporary database"

Tableau Server on Linux only supports UTF-8 character encoding. If your Linux locale is missing the UTF-8 encoding, your installation can fail with an error similar to this one:

Failed to initialize the instance of the temporary database

To check if your locale is using UTF-8 encoding, run the `localectl` command at a command prompt. The resulting output should look something like this (your locale may be different):

```
[ttableauserver-centos1a ~]$ localectl
System Locale: LANG=en_US.UTF-8
[ttableauserver-centos1a ~]$ localectl
```

If the `LANG` value does not include `.UTF-8` then you need to run `localectl` to add it:

```
sudo localectl set-locale LANG=<your_locale>.UTF-8
```

**Note:** In some cases `localectl` may not complete (timeout) if your version of systemd is old. Updating systemd may fix this problem and allow you to set the UTF-8 encoding. On RHEL-like systems, use this command to update systemd:

```
sudo yum update systemd
```

Installation fails on a virtual machine in Parallels

Parallels is currently not supported. If you install Tableau Server on a Linux virtual machine in Parallels, the install might fail.
Tableau Server doesn’t start

If Tableau Server does not start or is running in a degraded state, run the `tsm restart` command. This will shut down any processes that are running, and restart Tableau Server.

Cannot start Tableau Server after installation

Tableau Server might not start if your computer’s hostname changes after installation. One of the main reasons why the hostname might change is if you use the cloud-init package on CentOS 7. If you use the cloud-init package, `reboot the computer` where you want to install Tableau Server before you begin the installation process. Alternatively, you can fix the hostname without rebooting by running the following command:

```
sudo hostnamectl set-hostname `hostnamectl --static`
```

The cloud-init package is commonly used to initialize new virtual machines, configure SSH public key authentication, and more. For example, the CentOS 7 image in the Amazon Web Services Marketplace uses cloud-init, and cloud-init is commonly used in OpenStack deployments. However, the version of cloud-init included by default in the CentOS 7 repositories (cloud-init 0.7.5-10.el7.centos.1) has a known issue that prevents your computer from displaying its Fully Qualified Domain Name (FQDN) along with its hostname until after it restarts.

Because the Tableau Server installation process uses your computer’s hostname to configure server processes and generate TLS certificates, Tableau Server might not start if it is configured to use a hostname without the FQDN.

To determine if your computer is displaying the correct hostname, run the `hostnamectl` command. In the following example, the command displays a transient hostname which indicates that it will not return the FQDN and must be restarted.

```
$ hostnamectl
    Static hostname: server01.example.com
    Transient hostname: server01
    [...] 
```
Alternatively, in the following example, the command displays the correct hostname and FQDN:

```bash
$ hostnamectl
   Static hostname: server01.example.com

[...]
```

Cannot create initial administrator account with multiple Active Directory (AD) domains

When you create the initial administrator account on Tableau Server, you might see the following error if you selected AD as the authentication type:

Failed to authenticate username and password

This occurs when Tableau Server attempts to connect to multiple AD domains. For example, you might see this error if you install Tableau Server on a computer that is part of one domain and you attempt to authenticate AD users that are part of another domain.

Initializing Tableau Server

TSM initialization fails because the `tableau` user account exists but is not a member of the group `tableau`

When you install and initialize Tableau Services Manager (TSM) and Tableau Server, the initialization script (`initialize-tsm`) creates the users and groups needed to run, or confirms that the existing ones are configured with the required characteristics. By default the script creates a user called `tableau` and adds it to a group called `tableau`. If a `tableau` user already exists but is not part of the `tableau` group, the script fails with a warning.

If this happens you can fix the conflict by using a `--unprivileged-user` flag to specify a different user, and the user will be created and added to the `tableau` group.

For example, to specify a user named `tableauserver`, you would run the script from the `/opt/tableau/tableau_server/packages/scripts.<version>` directory using this command:
sudo ./initialize-tsm --unpriviledged-user="tableauserver" --accepteula

For a complete list of options that can be used with the initialize-tsm script, use the -h option:

sudo ./initialize-tsm -h

Error initializing Tableau Server on unsupported system locale

If you attempt to install Tableau Server on a computer with a locale that is not one of the eight supported locales, you will get an error during installation.

Tableau Server will run on a system using one of the following locales:


Any other locale will generate the error.

Error initializing Tableau Server when en_US.utf8 is not included in locale list

If you attempt to install Tableau Server on a computer that does not have en_US.utf8 in the locale list, the initialization will fail with an error. To see if en_US.utf8 is included, type locale -a at a shell prompt.

If en_US.utf8 is not listed, you can en_us to the locale list by typing sudo locale-gen en_US.UTF-8 at a shell prompt on Ubuntu, or sudo localedef -i en_US -f UTF-8 at a shell prompt on RHEL-like distributions.

Error: status 10 - initializing Tableau Server when data directory path includes a period

If you attempt to install Tableau Server and specify a data directory with a path that includes a period ("."), initialization will fail with errors including:

Connection timed out

and

ERROR: TSM services returned status 10
To avoid this issue, choose a data directory that does not include a period in its path.

Error initializing Tableau Server after reinstallation

If you uninstall and reinstall Tableau Server, you can encounter an error initializing Tableau Server. For example, you might see the following error:

```
ERROR com.t-ableausoftware.tabadmin.webapp.asyncjobs.JobStepRunner - Running step WaitForConfigure failed
com.t-ableau-software.tabadmin.webapp.exceptions.ServiceFailedStateException
```

This error occurs when artifacts remain from a previous installation that cause services to fail to start. To prevent this error, use the `tableau-server-oblitrate` script in the `/opt/tableau/tableau_server/packages/scripts.<version>` folder. For more information about completely removing Tableau Server, see Remove Tableau Server from Your Computer.

Activating Tableau Server

Tableau Server license activation fails

In certain cases activation of the Tableau product key using the `tsm licenses activate -k <product_key>` command fails with an error:

```
License Server not available
```

This can happen if your computer is unable to connect through TCP port 443 to the Tableau licensing server at `licensing.tableau.com`.

To resolve this you need to configure your network and/or host-based firewalls to allow access to that address and port, or activate Tableau offline. For more information, see Activate Tableau Offline.
Reindexing Tableau Server Search & Browse

Problems that can be solved by reindexing Search & Browse

Symptoms of an index that needs to be rebuilt include:

- A blank list of sites when a user attempts to log in
- A blank list of projects when a user tries to select a project
- Missing content (workbooks, views, dashboards)
- Unexpected or inaccurate alerts (for example, an "refresh failed" alert on a workbook that does not include an extract)

If you see any of these behaviors, rebuild the Search & Browse index using the `tsm maintenance reindex-search` command.

Restarting Tableau Server

Restarting Tableau Server or applying changes fails

If one of the Tableau Server services fails, you might see an error when you attempt to restart the server or apply configuration changes.

To see if a failed service is causing the error, type the following command:

```
 tsm status -v
```

To find out why a service failed, view the `tabadminagent` and `tabadmincontroller` log files in the data directory. For example, a service might fail because of concurrency issues or port configuration issues. Please include any issues you encounter in your feedback.

As a workaround, you can attempt to resolve the failure by removing and re-adding the service in TSM. Once the service has started, you can retry your previous configuration change or retry restarting the server with the `tsm restart` command.

Error restarting Tableau Server after adding or configuring a node

If you add a or configure the node without a Gateway process, Tableau Server might fail to restart and you could see errors like these:
ERROR: com.tableau-software.tabadmin.configuration.PortConfigurationExtractor
Unable to find port config key worker1.gateway.port

and

Message: Missing port configuration value for key 'worker1.gateway.port'

These errors appear in the gateway.log file and occur when a Tableau Server node is configured with either an Application Server or VizQL Server but without a Gateway. A Gateway process is required if either Application Server or VizQL Server is running on a node.

Backup/Restore

Problems related to restoring a backup created by Tableau Server can be the result of permissions issues. Proper permissions are necessary for both the file that TSM is restoring, and the location of the file. When TSM handles the backup, it puts the file in a default location and sets permissions appropriately. You can run into permission problems if you are restoring a backup that was copied to your Linux server, or a backup from a non-default location on your server. For details about using a non-default location, including how to change the location, see tsm File Paths.

Errors may include:

Server Was Denied Access to File

and

Restoring the backup '<backup>.tsbak' was unsuccessful

The Tableau Server backup and restore processes need:

- Read permission—The processes need to access the backup file directly.
- Execute permission—The processes also require execute permissions to the
directory structure in which the file is placed.

When TSM creates a backup in the default location, it sets the permissions it needs. If you copy a file to the Linux server, or move it to a non-default directory, the permissions may not allow the TSM processes proper access. You need to verify that both the file, and the directory tree that contains it, allow access by the TSM user tableau. The file permissions must give the tableau user read access. You can do this by setting the group on the file to the tableau group, and giving that group read access. The directory permissions must give the tableau user read access. You can do this by setting the group on the directory to the tableau group, and giving that group read and execute access on the directories.

For detailed information about TSM and file permissions, see Files and Permissions in TSM.

File locations

Changing basefilepath does not change the location of an existing file

Several tsm commands write files to default locations. You can change these default locations for each command using a tsm set command, but doing so does not move any existing files from the original location to the new one, and it does not create the new location. You are responsible for creating the new location, and for making sure it has the correct permissions to allow tsm access to any files in the location, and the entire directory structure that contains the files.

For more information about changing default locations for backup, restore, site import and export, and ziplogs files, see tsm File Paths.

For information about tsm permissions, see Files and Permissions in TSM.

TSM commands

TSM command line does not show progress for long-running tasks

If you run a tsm command such as restore or ziplogs that takes more than 2 hours to complete, the command will continue to run until completion on the server. To display the progress of the job, use the tsm jobs reconnect command.
Opening Firewall ports

Manually opening firewall ports on Ubuntu

The current version of Tableau Server does not support the `ufw` firewall that is used on Ubuntu. For customers that do not want to install `firewalld` on Ubuntu, another option is to manually open those ports. The following steps will confirm that `ufw` is running, and open TCP ports 8850 and 80 to connections from any source address:

1. Run the following command to confirm `ufw` is running:

   ```bash
   sudo ufw status
   ```

   If the result is **Status: inactive**, you will need to enable `ufw` and ensure that you can continue to connect via `ssh`, which is outside the scope of these release notes.

2. Run the following command to allow access to port 8850:

   ```bash
   sudo ufw allow 8850
   ```

3. Run the following command to allow access to port 80:

   ```bash
   sudo ufw allow 80
   ```

OpenID fails on first sign-in attempt

If you have configured Open ID Connect authentication for Tableau Server, the first sign-in attempt fails. To successfully log in, users must retry authentication after the initial failure.

Administrative views do not display

The Status tab of Tableau Server includes links to visualizations that display server metrics. These visualizations require the PostgreSQL driver to access the appropriate data from the Tableau Server repository. The PostgreSQL driver is not installed automatically, so if you have not installed the driver, the views will not display. For more information, see [Install PostgreSQL Drivers](#).
Note: To use administrative views, the PostgreSQL driver must be installed on any node that runs the VizQL Server process.

Work with Log Files

Tableau Server creates log files as a normal part of its activities. You may need to use the server log files when you are troubleshooting issues with Tableau Server or if Tableau Support requests logs from you to help you resolve an issue.

You can create a zipped log file archive using the `tsm maintenance ziplogs` command. The zipped archive contains copies of the logs you can unzip and look at, or send to Tableau Support. Once you have a copy of the archive, you can delete the archive from your server. For more information on log file archives, see Archive Logs.

This collection of topics provides information about how to create log file archives, the contents of specific log files, and details about when and how you might want to look at a log.

Investigating Tableau Server Issues

The range and complexity of possible issues with Tableau Server means that there is no simple process you can use to investigate all problems, but a general approach would include these steps:

1. **Clean up** existing log files to reduce their size. For more information, see Remove old log files.
2. **Set the appropriate logging level**. This is something that Tableau Support will instruct you on. For more information, see Change Logging Levels.
3. **Reproduce the issue** you are troubleshooting so the logs capture the events related to the problem.
4. **Create an archive** of the logs. For more information see Archive Logs.
Important: Use this archive when looking at the log files. You should not edit, move or delete any files directly on the server.

5. **Review the tsm.log file** (~/.tableau/tsm) to get a understanding of TSM commands that have been run.

6. **Review the TSM admin log** (/tabadminagent/tabadminagent_node<n>-<n>.log) to understand any configuration or deployment that has been done on the server.

7. **Review the TSM controller log** (/tabadmincontroller/tabadmincontroller_node<n>-<n>.log) to understand any configuration, deployment that has been done from the command line, or any maintenance jobs.

8. **Review the Apache logs** (/httpd/access.####_##_##_##_.log and /httpd/error.log) for requests that may be related to the issue you are investigating.

The Apache logs will contain a fair amount of "noise" that does not apply to issues you are experiencing.

- If you find a request that seems to be related to your issue, search /vizqlserver for entries that include the unique request ID from the Apache logs.
- Look for the response code and message associated with the request ID.
- Search for the name of the workbook, view, dashboard, or data source that is related to your issue. Make sure to look for a relevant timestamp.
- If you find a request that seems to be related to your issue, look at the response code associated with the request. (200s are good, 500s indicate problems.)
- Locate the unique request ID associated with the request you've identified (the unique request ID is a 24 character alphanumeric string at the very end of the request).

9. **Review the log archive** further to search for other messages and possible errors.
• Use the request ID from the Apache logs to search the /vizqlserver folder of the log archive for files containing related log entries. Look for indications of a problem (for example, error messages or long-running queries).

10. **Contact support**

If you are not able to solve the issue yourself, or if requested by Tableau Support, send the zipped archive to Tableau.

See the following topics for more information:

**Server Log File Archive File Locations**

By default, Tableau Server on Linux log file archives are gathered in a zip file called log-s.zip, but you can specify a different filename when you create the archive with the tsm maintenance ziplogs command. You can copy the archive from the server to a local computer and open it there, or send it to Tableau Support.

Looking for Tableau Server on Windows? See [Server Log File Locations](#).

When you unzip the archive, a series of zip files are created that start with the directory names shown below, and that end with a version number. These zip files contain the log files from the corresponding directories. The table in this topic explains the possible contents of these zip files, along with the original location the files came from on Tableau Server, the process that created the log files, and details about the files.

**Tableau Server Log and Configuration File Locations**

The Tableau Server log files are found in the following directory: `/var/opt/tableau/tableau_server/data/tabsvc/logs`

The Tableau Server configuration file, which is not included in the log file archive, is sometimes needed by Tableau Support to investigate support cases. This configuration file is found in the following directory: `/var/opt/tableau/tableau_server/data/tabsvc/config/tabsvc<build_number>/tabsvc.yml`
<table>
<thead>
<tr>
<th>Log Archives in logs.zip</th>
<th>Details</th>
<th>Files</th>
<th>Generated by</th>
<th>Location on Tableau Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>app-zookeeper</td>
<td>Logs related to the Tableau Server Coordination Service</td>
<td>appzookeeper-#.log&lt;br&gt;control_app-zookeeper-#.log&lt;br&gt;spawn.#####.log</td>
<td>zookeeper.exe</td>
<td>/logs/appzookeeper</td>
</tr>
<tr>
<td>backgrounder</td>
<td>Logs related to subscriptions and scheduled activities like extract refreshes, &quot;Run Now&quot; tasks, and tabcmd tasks</td>
<td>backgrounder-#.log&lt;br&gt;spawn.#####.log&lt;br&gt;tomcat-#.#####-##-##.log</td>
<td>backgrounder.exe</td>
<td>/logs/backgrounder</td>
</tr>
<tr>
<td>backuprestore</td>
<td>Logs related to backup and restore scenarios</td>
<td>control-backuprestore-#.log</td>
<td></td>
<td>/logs/backuprestore</td>
</tr>
<tr>
<td>cacheserver</td>
<td>Logs related to the Cache Server process</td>
<td>control-cacheserver-#.log&lt;br&gt;redis_.#.log</td>
<td>redis-server.exe</td>
<td>/logs/cacheserver</td>
</tr>
<tr>
<td>cluster-</td>
<td>Logs related</td>
<td>clustercontroller.log</td>
<td>cluster-</td>
<td>/log-</td>
</tr>
<tr>
<td><strong>controller</strong></td>
<td>to the Cluster Controller process</td>
<td><strong>cluster-controller.log-####-##-##</strong></td>
<td><strong>controller.exe</strong></td>
<td>s/clustercontroller</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------</td>
<td>--------------------------------------</td>
<td>--------------------</td>
<td>---------------------</td>
</tr>
</tbody>
</table>
| **data-base-maintenance** | Logs related to database maintenance | control-data-basemaintenance-#.log  
data-basemaintenance-#.log  
db-migrate.log  
migration.log  
spawn.#####.log | | /log-s/data-basemaintenance |
<p>| <strong>data-server</strong> | Logs related to connections to Tableau Server data sources | dataserver-#.log | <strong>dataserver.exe</strong> | /logs/dataserver |
| <strong>filestore</strong> | Logs related to the Tableau Server File Store process that controls the storage of extracts and syncs extracts | filestore.log | <strong>filestore.exe</strong> | /logs/filestore |</p>
<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Files</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>httpd</td>
<td>Apache logs. Look here for authentication entries. Each request in the Apache log will have a request ID associated with it. This request ID is used throughout the server logs and you can use it to associate log entries with a request.</td>
<td>access.#####-##-##.log, error.log, startup.log</td>
<td>/logs/httpd</td>
</tr>
<tr>
<td>hyper</td>
<td>Logs related to Tableau data engine. A log file is generated each day with information about data extracts</td>
<td>checklicense.log, control-hyper-0.log, hyper_#####<em>##</em>##_##.log</td>
<td>/logs/hyper</td>
</tr>
<tr>
<td>Service</td>
<td>Logs related to</td>
<td>Files related to</td>
<td>Path</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------</td>
<td>------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>licenseservice</td>
<td>licensing processes</td>
<td>control-licenseservice.log</td>
<td>/logs/licenseservice</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tabclicsrv.log</td>
<td></td>
</tr>
<tr>
<td>postgresql</td>
<td>Logs related to</td>
<td>control-pgsql-#.log</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PostgresSQL database</td>
<td>shut-downCommand.####.log</td>
<td></td>
</tr>
<tr>
<td></td>
<td>including files related</td>
<td>tabspawn</td>
<td></td>
</tr>
<tr>
<td></td>
<td>to launching server</td>
<td></td>
<td>/logs/postgresql</td>
</tr>
<tr>
<td></td>
<td>processes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>samlservice</td>
<td>Logs related to</td>
<td>control-samlservice-#.log</td>
<td></td>
</tr>
<tr>
<td></td>
<td>the Security Assertion</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Markup Language (SAML)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>service in Tableau</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Server</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>searchserver</td>
<td>Logs related to search</td>
<td>control-searchserver-#.log</td>
<td></td>
</tr>
<tr>
<td></td>
<td>indexing</td>
<td>tomcat_.#####-##-##.log</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/logs/searchserver</td>
</tr>
<tr>
<td><strong>Files</strong></td>
<td>Logs related to...</td>
<td>Control-...</td>
<td>Path</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------------------------------------------</td>
<td>----------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>siteimportexport</td>
<td>site import and export operations</td>
<td>siteimportexport-0.log</td>
<td>/logs/siteimportexport</td>
</tr>
<tr>
<td>tabadminagent</td>
<td>configuration and topology changes on each server node</td>
<td>tabadminagent.log</td>
<td>/logs/tabadminagent</td>
</tr>
<tr>
<td>tabadmincontroller</td>
<td>the Tableau Services Manager (TSM) CLI and TSM API</td>
<td>tabadmincontroller-#.log</td>
<td>/logs/tabadmincontroller</td>
</tr>
<tr>
<td>tabsvc</td>
<td>the startup and shutdown of other Tableau Server service processes.</td>
<td>tabsvc.log</td>
<td>/logs/tabsvc</td>
</tr>
<tr>
<td>vizportal</td>
<td>administrative tasks, work-vizportal-0.log</td>
<td>vizportal.exe</td>
<td>/logs/vizportal</td>
</tr>
</tbody>
</table>
book and permissions management, authentication, sign-ins, initial view requests, and publishing requests.

These log files also include entries related to REST API requests. REST API log entries start with the receipt of the HTTP request, followed by information related to the processing of that request. If
the request is completed, these log entries end with the HTTP response code for that request.

<table>
<thead>
<tr>
<th>VizQLServer</th>
<th>Logs related to displaying and interacting with views. When running multiple instances of VizQL Server, the instances are distinguished by port number.</th>
<th>control_vizqlserver_node#-0.log</th>
<th>/logs/vizqlserver</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>notify-producers logs contain exceptional events.</td>
<td>tomcat_.#####-##-##.log</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>vizql-client-0.log.#####-##-##</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>vizqlserver_node#-0.log</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>spawn.#####.log</td>
<td></td>
</tr>
</tbody>
</table>
Archive Logs

You create an archive of Tableau Server log files using the `tsm maintenance ziplogs` command.

This command creates a zip file containing all of the log files and is useful when you're working with Tableau Support. If you are running a distributed installation of Tableau Server, perform this step from the initial node. Logs from all nodes will be included in the zip file.

**Note:** If you cannot run the `ziplogs` command successfully, you can manually zip the Tableau Server logs. For more information, see Manually gather logs.

To create a log file archive:

- At a command prompt, type the following command:

  ```
  tsm maintenance ziplogs -l -f <filename>
  ```

  where `<filename>` is name of the zipped archive file you want to create. Choose a unique name with no spaces. Tableau will not overwrite an existing file.

Uploading log archives for Tableau Support

You can send log files to Tableau as a part of a customer support case (a customer support case number is required). Before sending a log file, use `tsm maintenance ziplogs` command to combine the log files into a single zip file archive. If you are creating the archive to send to Tableau Support, see the Knowledge Base for information about how to upload large files.

- At a command prompt, type the following command:

  ```
  tsm maintenance send-logs -f <zip file name> -c <case number> -e <email address>
  ```
where <case number> is your support case number, <email address> is your contact email for this support case, and <zip file name> is the file name of your archive with .zip file extension.

Change Logging Levels

By default, Tableau Server logs events at the Info level. You can change this if you need to gather more information (if you are working with Tableau Support, for example). As a best practice you should not increase logging levels except when troubleshooting an issue.

Logging Levels

The following logging levels are listed in order of increasing amount of information logged:

- off
- fatal
- error
- warn
- info (the default)
- debug
- trace

**Note:** Increasing the log level to debug or trace increases the amount of information being logged and can have a significant impact to performance. You should only set a logging level to debug when investigating a specific issue. Reproduce the issue and then reset the logging level back to info.

Change Logging Levels

Set logging levels for Tableau Server using one of several tsm configuration set commands. The command you use depends on which component of Tableau Server you want to change the logging level for.

<table>
<thead>
<tr>
<th>Configuration key</th>
<th>Location of affected logs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(path begins with /var/opt/tableau/tableau_server-</td>
</tr>
</tbody>
</table>
For more information, see tsm configuration set Options.

You need to stop Tableau Server before changing the logging levels, and restart it afterward. If you are running a distributed installation of Tableau Server, set logging levels from the initial node.

To change the logging level:

1. Stop Tableau Server by typing:

   tsm stop

2. Set the logging level to by typing tsm configuration set -k <config_key> -v <value>

   where <config_key> is <process>.native_api.log.level, vizportal.log.level, or vizqlserver.log.level

   and <value> is a valid logging level.

   Examples:
3. Restart Tableau Server by typing:

```
tsm restart
```

Reset Logging Levels

After you gather the information related to the issue you are investigating, reset the logging levels so there is no lingering performance impact.

Reset the logging level back to its default (info) using the appropriate command with a `-d` option.

Examples:

```
• tsm configuration set backgrounder.native_api.log.level -v debug
• tsm configuration set vizqlserver.log.level -v warn
• tsm configuration set vizportal.log.level -v debug
```

Handle an Unlicensed Server

Tableau offers two licensing models: user-based and core-based. User-based licensing requires each active user account to be covered by a license. User-based licenses have a defined capacity, or number of users that it allows. Each user is assigned a unique user name on the server and is required to identify himself when connecting to the server.

Core-based licensing has no constraints on the number of user accounts in the system, but it does restrict the maximum number of processor cores that Tableau Server can use. You can install Tableau Server on one or more machines to create a cluster, with the restriction that the total number of cores in all the machines does not exceed the number of cores you have licensed and that all of the cores on a particular machine are covered by the license.
Unlicensed User-Based Server

The most common reason for a server that has user-based licensing to be unlicensed is an expired product key or an expired maintenance contract.

Unlicensed Core-Based Server

A core-based server can become unlicensed for a variety of reasons. A common problem is that any Tableau Server node has more cores than the license allows. When the server is unlicensed you may not be able to start or administer the server.

Handle an Unlicensed VizQL Server Process

There are several status indicators on the Tableau Server Status page that help you understand the state of Tableau Server processes. An orange-color status box, "Unlicensed", indicates that one of the VizQL server processes is unable to retrieve the Tableau Server license information.
There may be several reasons why the process is unable to access this information. For example, there may be network issues preventing a VizQL process, which is running on a worker machine, from communicating with the primary machine. Or, the process may be getting sent more requests than it can accept at that time and can’t handle the licensing request. As a result, some of your users may be able to access views while others cannot.

To resolve the problem, stop, then start Tableau Server.

Troubleshoot Performance of Workbooks with Calculations

In most cases, you can expect to see similar or better performance after upgrading your Tableau Server from versions 10.4 or earlier to 10.5 or later. However, for workbooks that use complex calculations, you may see performance issues after you upgrade. This article describes how to test and troubleshoot performance issues for workbooks that have calculations and use extract as the data source after you have upgraded your Tableau Server from versions 10.4 or earlier to 10.5 or later.

If you have not yet upgraded to Tableau Server 10.5 or later, we strongly recommend that you set up a test environment and test your workbook performance before you upgrade your production Tableau Server. For more details, see Test Workbook Performance.

If you have already upgraded to Tableau Server 10.5 or later, and you are experiencing slow response times, use the guidance provided in this topic to help troubleshoot performance issues.

Since Linux was first released in 10.5, this only applies when you are migrating from Tableau Server versions 10.4 or earlier on Windows to Linux 10.5 or later.

Here are the scenarios that most likely describe your environment after upgrading to Tableau Server 10.5 or later:

- Recently upgraded, and majority of the extracts are still in .tde format
- Recently upgraded and majority of extracts are in .hyper format
- Upgraded a while ago, workbooks have changed since upgrade
Scenario: You upgraded your Tableau Server recently. You are experiencing slow response times and the majority of your extracts have not been refreshed, so they are still in .tde format.

1. Disable any scheduled or automated extract refreshes.

2. Enable the following setting on Tableau Server which checks the number of query nodes required for Tableau to generate the workbook and displays an error when the set limit is exceeded. The error message might say something like: "Logical Query tree has 2348182 nodes, maximum number allowed is 1000000....".

Use TSM CLI to run the following commands:

```bash
 tsm configuration set -k native_api.node_limit_checker_pre_rewrite_disable -v false
 tsm pending-changes apply
```

**Note:** This setting is only available in Tableau Server 10.5.3 or later. While enabling this setting improves the reliability and stability of Tableau Server, you may see errors on a very small percentage of workbooks with when you enable this setting. For example, the Tableau team saw errors on 0.01% of the workbooks with this setting enabled on Tableau Online.

3. Test the workbooks by viewing or interacting with them. You should see an error message informing you about exceeding the node limits on workbooks that have performance issues. The error message might say something like: "Logical Query tree has 2348182 nodes, maximum number allowed is 1000000....".

4. Use Tableau Desktop to troubleshoot your workbook performance. Run this command to enable the node limit check in your Tableau Desktop:

```bash
 tableau.exe -DNodeLimitCheckerPreRewriteDisable=false
```
In Tableau Desktop, download and open the workbooks that displayed the error message about high node limits in step 2. Try optimizing the performance of workbooks using methods described in the Best Practices for Creating Calculations in Tableau topic. If successful, then publish them to Tableau Server. If you are still experiencing performance issues, continue to the next step.

5. For workbooks that load successfully, but the overall response times is slower after upgrade, try using the Compute Calculations Now option. If you have a large number of workbooks that fall under this category, consider enabling the server wide setting to retain materialized calculations for all extracts when they are upgraded from .tde to .hyper:

Use tabadmin command line tool to run the following commands:

Use TSM CLI to run the following commands:

```
tsm configuration set -k native_api.preserve_calculations_ on_hyper_refresh_conversion -v true

 tsm pending-changes apply
```

**Considerations for turning on this setting:**

Turn this setting on only if majority of your workbooks require calculations to be materialized. There are resource and time costs associated with this setting, which are as follows:

- When this setting is set to true, the extract file size will increase, affecting the overall disk space.
• Extracts with materialized calculations take a longer time to refresh than extracts without materialized calculations.

This setting only affects extracts that have not yet upgraded to .hyper. Extracts already upgraded to .hyper before turning on this setting, will not have the calculations materialized. This setting will also not affect workbooks and extracts published from Tableau Desktop, or any new extract refreshes published to Tableau Server that do not use the **Compute Calculations Now** option.

**Note:** This setting is only available in Tableau Server 10.5.3 or later.

6. Enable extract refresh schedules.

Scenario: You upgraded your Tableau Server recently. You are experiencing slow response times and a majority of your extracts have been refreshed and upgraded from .tde to .hyper.

1. Restore the backup you made before you upgraded your production Tableau Server.

   **Note:** This method assumes that your workbooks have not changed significantly since the upgrade and, or it is ok to go back to the state just before upgrade.

2. Disable any scheduled extract refreshes.

3. Enable the following setting on Tableau Server which checks the number of query nodes required for Tableau to generate the workbook and displays an error when the set limit is exceeded. The error message may something like: **"Logical Query tree has 2348182 nodes, maximum number allowed is 1000000...."**: 

   Use TSM CLI to run the following commands:

   ```bash
   tsm configuration set -k native_api.node_limit_checker_pre_rewrite_disable -v false
   ```
tsm pending-changes apply

**Note:** This setting is only available in 10.5.3 or later. While enabling this setting improves the reliability and stability of Tableau Server, you may see errors on a very small percentage of workbooks with when you enable this setting. For example, the Tableau team saw errors on 0.01% of the workbooks with this setting enabled on Tableau Online.

4. Test the workbooks by viewing or interacting with them. You should see an error message informing you about exceeding the node limits on workbooks that have performance issues. The error message might say something like: "**Logical Query tree has 2348182 nodes, maximum number allowed is 1000000....**".

5. Use Tableau Desktop to troubleshoot your workbook performance. Run this command to enable the node limit check in your Tableau Desktop:

    `tableau.exe -DNodeLimitCheckerPreRewriteDisable=false`

Download and open the workbooks that displayed the error message using Tableau Desktop. Try optimizing the performance of workbooks using methods described in the [Best Practices for Creating Calculations in Tableau](#) topic. If successful, then publish them to Tableau Server. If you are still experiencing performance issues, continue to the next step.

6. For workbooks that load successfully, but the overall response times is slower after upgrade, try using the **Compute Calculations Now** option. If you have a large number of workbooks that fall under this category, consider enabling the server wide setting to retain materialized calculations for all extracts when they are upgraded from .tde to .hyper:

    Use TSM CLI to run the following commands:
tsm configuration set -k native_api.preserve_calculations_on_hyper_refresh_conversion -v true

tsmand pending-changes apply

**Considerations for turning on this setting:**

Turn this setting on only if majority of your workbooks require calculations to be materialized. There are resource and time costs associated with this setting, which are as follows:

- When this setting is set to true, the extract file size will increase, affecting the overall disk space.

- Extracts with materialized calculations take a longer time to refresh than extracts without materialized calculations.

This setting only affects extracts that have not yet upgraded to .hyper. Extracts already upgraded to .hyper before turning on this setting, will not have the calculations materialized. This setting will also not affect workbooks and extracts published from Tableau Desktop, or any new extract refreshes published to Tableau Server that do not use the Compute Calculations Now option.

**Note:** This setting is only available in 10.5.3 or later.

7. Enable extract refresh schedules.

Scenario: You upgraded your Tableau Server a while ago and you are experiencing slow response times. It is not practical for you to roll back using the backup you made prior to upgrade because your workbooks have changed since the upgrade, and the extracts have been upgraded to .hyper format.

If you fall under this scenario, and you find that your workbooks require materialized calculations, turning on the optimization setting on refresh will not work for you as this setting only works when upgrading from .tde to .hyper. Your will need to test your workbooks
individually for performance issues, and optimize them manually and republish to Tableau Server.

1. If you see high memory and CPU consumption by VizQL Server when viewing workbooks with extracts, in addition to slower workbook response times, then you can enable the following setting on Tableau Server to help prevent VizQL process from consuming the memory resources on the machine that might result in critical Server issues. This setting checks the number of query nodes required for Tableau to generate the workbook and displays an error when the set limit is exceeded:

Use TSM CLI to run the following commands:

```
tsm configuration set -k native_api.node_limit_checker_pre_rewrite_disable -v false
tsm pending-changes apply
```

**Note:** While enabling this setting improves the reliability and stability of Tableau Server, you may see errors on a very small percentage of workbooks with when you enable this setting. For example, the Tableau team saw errors on 0.01% of the workbooks with this setting enabled on Tableau Online.

2. Test the workbooks by viewing or interacting with them. Look for workbooks where you might see an error message that says something like: "*Logical Query tree has 2348182 nodes, maximum number allowed is 1000000....*"

3. Use Tableau Desktop to troubleshoot the workbooks that you identified in step 2. Run this command to enable the node limit check in your Tableau Desktop:

```
tableau.exe -DNodeLimitCheckerPreRewriteDisable=false
```

Download and open the workbooks that displayed the error message using Tableau Desktop. Try optimizing the performance of workbooks using methods described in
the Best Practices for Creating Calculations in Tableau topic and publish them to Tableau Server. If you are still seeing performance issues, continue to the next step.

4. Use the **Compute Now** option to materialize calculations in your extract.

### Server Administrator Reference

#### Process Reference

This topic describes the options for setting the process configuration. To configure Tableau Server processes, you need to specify which processes and how many instances should run on each node. You do this using the `tsm topology set-process` command. For more information, see Changing the number of processes on a node.

Looking for Tableau Server on Windows? See [Tableau Server Processes](#).

Except where explicitly noted in the table below, applying changes in processes will stop Tableau Server if it is running when you apply those changes. After changes are applied, Tableau Server is returned to the state it was in before process configuration, so if the server was running, it will be restarted.

**Important:** Your process topology will depend on your organizational needs.

#### Licensed processes

Some of the processes that are installed as a part of Tableau Server are "licensed" processes. Licensed processes need a valid Tableau Server license in order to
run. Other processes that are installed as a part of Tableau Server are not tied to a valid license. This has the following impact:

- Every licensed process needs to regularly contact the Tableau Server License Manager service that runs on the primary Tableau Server computer to verify they are licensed. If they cannot confirm there is a valid license, for example, if the primary node is not available, the process will not run and Tableau Server may not function properly or reliably.

- If you have a core-based Tableau Server license, the cores on any node with a licensed process will count against the total count of licensed cores.

The "Licensed" column in the table below identifies those processes that require a valid license, and which impact the count of cores in core-based licenses.

### Processes

<table>
<thead>
<tr>
<th>Tableau Server Processes</th>
<th>Name shown in <code>tsm status -v</code></th>
<th>Name used with <code>tsm topology set-process</code></th>
<th>Purpose</th>
<th>Notes</th>
<th>Licensed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backgrounder</td>
<td>backgrounder</td>
<td>The Backgrounder runs server tasks, including extract refreshes, subscriptions, 'Run Now' tasks, and tasks initiated from <code>tabcmd</code>.</td>
<td>When Backgrounder is installed, Data Engine is also installed, unless the node already has an instance of Data Engine. In most situations you can</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Name shown in tsm status -v</td>
<td>Name used with tsm topology set-process</td>
<td>Purpose</td>
<td>Notes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------</td>
<td>---------</td>
<td>-------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>change the number of backgrounder instances on an existing node of a running server without causing a stop and restart the server. An exception is if you are adding backgrounder to an existing node that did not previously have backgrounder or any other process that also installs data engine.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cache Server</td>
<td>cacheserver</td>
<td>The Cache</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name shown in tsm status -v</td>
<td>Name used with tsm topology set-process</td>
<td>Purpose</td>
<td>Notes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Server is a query cache distributed and shared across the server cluster. This in-memory cache speeds user experience across many scenarios. VizQL server, backgrounder, and data server (and application server to a lesser extent) make cache requests to the cache server on behalf of users or jobs. The cache is single-threaded, so if you need better performance you should run additional instances of cache server.</td>
<td>Licensed</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tableau Server Processes
<table>
<thead>
<tr>
<th>Name shown in tsm status -v</th>
<th>Name used with tsm topology set-process</th>
<th>Purpose</th>
<th>Notes</th>
<th>Licensed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster Controller</td>
<td>cluster-controller</td>
<td>The Cluster Controller is responsible for monitoring various components, detecting failures, and running failover when needed.</td>
<td>Required on each node. Not automatically installed.</td>
<td>No</td>
</tr>
<tr>
<td>Coordination Service</td>
<td>Cannot be set with tsm topology set-process</td>
<td>The Coordination Service serves as the single source of truth.</td>
<td>Automatically installed on the initial node. No other instances are installed unless you explicitly deploy a new Coordination Service ensemble. For details, see Deploy a Coordination Service Ensemble.</td>
<td>No</td>
</tr>
<tr>
<td>Name shown in tsm status -v</td>
<td>Name used with tsm topology set-process</td>
<td>Purpose</td>
<td>Notes</td>
<td>Licensed</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------------</td>
<td>---------</td>
<td>-------</td>
<td>----------</td>
</tr>
<tr>
<td>Data Engine</td>
<td>Cannot be configured manually.</td>
<td>The Data Engine creates data extracts and processes queries.</td>
<td>Automatically installed when you install File Store, VizQL Server, Application Server (VizPortal), Data Server, or Backgrounder.</td>
<td>Yes</td>
</tr>
<tr>
<td>Data Server</td>
<td>dataserver</td>
<td>The Data Server manages connections to Tableau Server data sources.</td>
<td>When Data Server is installed, Data Engine is also installed, unless the node already has an instance of Data Engine.</td>
<td>Yes</td>
</tr>
<tr>
<td>File Store</td>
<td>filestore</td>
<td>The File Store automatically replicates extracts across Data Engine nodes.</td>
<td>When File Store is installed, Data Engine is also installed, unless the node already has an instance of</td>
<td>No</td>
</tr>
<tr>
<td>Name shown in tsm status -v</td>
<td>Name used with tsm topology set-process</td>
<td>Purpose</td>
<td>Notes</td>
<td>Licensed</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------</td>
<td>---------</td>
<td>-------</td>
<td>----------</td>
</tr>
<tr>
<td>Gateway</td>
<td>gateway</td>
<td>The Gateway is a web server that handles all requests to Tableau Server from browsers, Tableau Desktop, and other clients.</td>
<td>Required on any node with an instance of VizQL Server or Vizportal.</td>
<td>No</td>
</tr>
<tr>
<td>Repository</td>
<td>postgres</td>
<td>The PostgreSQL repository is the main database for Tableau Server. It stores workbook and user metadata.</td>
<td>You are limited to a maximum of two instances of the repository in a cluster, and must have at least three nodes in the cluster to add a second repository instance.</td>
<td>No</td>
</tr>
<tr>
<td>Search And Browse</td>
<td>searchserver</td>
<td>The Search Service handles fast search, filter,</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>Name shown in <code>tsm status -v</code></td>
<td>Name used with <code>tsm topology set-process</code></td>
<td>Purpose</td>
<td>Notes</td>
<td>Licensed</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------</td>
<td>---------</td>
<td>-------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>retrieval, and display of content metadata on the server.</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Application Server</td>
<td>vizportal</td>
<td>The Application Server (VizPortal) handles the web application, REST API calls, and supports browsing and searching.</td>
<td>When Application Server is installed, Data Engine is also installed, unless the node already has an instance of Data Engine.</td>
<td>Yes</td>
</tr>
<tr>
<td>VizQL Server</td>
<td>vizqlserver</td>
<td>The VizQL Server loads and renders views, computes and executes queries.</td>
<td>When VizQL Server is installed, Data Engine is also installed, unless the node already has an instance of Data Engine. In most situations you can change the</td>
<td>Yes</td>
</tr>
<tr>
<td>Tableau Server Processes</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name shown in <code>tsm status -v</code></td>
<td>Name used with <code>tsm topology set-process</code></td>
<td>Purpose</td>
<td>Notes</td>
<td>Licensed</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>number of VizQL instances on an existing node of a running server without causing a stop and restart the server. An exception is if you are adding VizQL to an existing node that did not previously have VizQL or any other process that also installs data engine.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tableau Services Manager Processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administration Agent</td>
</tr>
<tr>
<td>Name shown in <code>tsm status -v</code></td>
</tr>
<tr>
<td>-------------------------------</td>
</tr>
<tr>
<td>Administration Controller</td>
</tr>
<tr>
<td>Name shown in <code>tsm status -v</code></td>
</tr>
<tr>
<td>-------------------------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Service Manager</td>
</tr>
<tr>
<td>License Manager</td>
</tr>
</tbody>
</table>
## Tableau Server Processes

<table>
<thead>
<tr>
<th>Name shown in <code>tsm status -v</code></th>
<th>Name used with <code>tsm topology set-process</code></th>
<th>Purpose</th>
<th>Notes</th>
<th>Licensed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Maintenance</td>
<td>Cannot be configured manually.</td>
<td>The Database Maintenance service is responsible for performing maintenance operations on the Tableau Server repository.</td>
<td>Automatically installed on each node where you install Tableau Server. Shows a status of <code>stopped</code> in output of <code>tsm status -v</code> unless it is actively per-</td>
<td></td>
</tr>
</tbody>
</table>

License Manager process should only be manually configured if the initial node fails. For more information, see Recover from an Initial Node Failure.
<table>
<thead>
<tr>
<th>Name shown in <code>tsm status -v</code></th>
<th>Name used with <code>tsm topology set-process</code></th>
<th>Purpose</th>
<th>Notes</th>
<th>Licensed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>forming database maintenance. Maintenance can include updates related to enabling remote access to the repository and changing passwords used to access the repository. You cannot configure the Database Maintenance service manually.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backup/Restore</td>
<td>Cannot be configured manually.</td>
<td>The Backup and Restore service is responsible</td>
<td>Automatically installed on each node where you</td>
<td></td>
</tr>
<tr>
<td>Name shown in <code>tsm status -v</code></td>
<td>Name used with <code>tsm topology set-process</code></td>
<td>Purpose</td>
<td>Notes</td>
<td>Licensed</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------</td>
<td>---------</td>
<td>-------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for performing backup and restore operations on the data stored in the Tableau Server repository and file store.</td>
<td>install Tableau Server. Shows a status of stopped in output of <code>tsm status -v</code> unless it is performing a backup or restore operation. You cannot configure the Backup and Restore service manually.</td>
<td></td>
</tr>
<tr>
<td>Site Import/Export</td>
<td>Cannot be configured manually.</td>
<td>The Site Import and Export service is responsible for migrating Tableau Server sites between server clusters.</td>
<td>Automatically installed on each node where you install Tableau Server.</td>
<td></td>
</tr>
<tr>
<td>Name shown in <code>tsm status -v</code></td>
<td>Name used with <code>tsm topology set-process</code></td>
<td>Purpose</td>
<td>Notes</td>
<td>Licensed</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------</td>
<td>---------</td>
<td>-------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shows a status of stopped in output of <code>tsm status -v</code> unless it is performing an import or export. You cannot configure the Site Import and Export service manually.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAML Service</td>
<td>Cannot be configured manually.</td>
<td>The SAML Service acts as a proxy between Tableau Server and SAML Identity Providers (IdPs).</td>
<td>Automatically installed on each node where you install Tableau Server. Shows a status of stopped in</td>
<td></td>
</tr>
<tr>
<td>Name shown in <code>tsm status -v</code></td>
<td>Name used with <code>tsm topology set-process</code></td>
<td>Purpose</td>
<td>Notes</td>
<td>Licensed</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------</td>
<td>---------</td>
<td>-------</td>
<td>----------</td>
</tr>
<tr>
<td>output of <code>tsm status -v</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Server Process Limits**

When you reconfigure processes for Tableau Server, there is a limit to the amount that you can increase the number of process instances. By default, the limit is set to eight. If your machine has enough RAM and CPU cores, and you want to go above this limit, you can change the limit using the `service.max_procs` configuration option. For each process instance, Tableau recommends that the machine running the process have at least 1 GB of RAM and 1 logical CPU core.

To change the maximum number of processes allowed:

1. Type the following command, where `number` is the maximum number of process instances you want to allow:

   `tsm configuration set -k service.max_procs -v <number>`

   For example:

   `tsm configuration set -k service.max_procs -v 10`
2. Next type:

```
 tsm pending-changes apply
```

The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `–r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

### Tableau Services Manager Ports

The processes and services that make up the components of Tableau Services Manager (TSM) and Tableau Server on Linux use various ports to communicate. By default these ports are assigned (mapped) dynamically from a predefined range of ports. The port assignments are made for each service or process when it is installed. You can control what ports are used in a variety of ways.

**Note:** A small subset of processes do not use dynamic port mapping and behave uniquely. For more information, see Ports that are not dynamically mapped, below.

#### Port assignment

There are two approaches you can use for port assignment or mapping in TSM:

- **Dynamic port assignment.** This is the default and requires the least intervention by the administrator.
- **Manual port assignment.** This option requires an administrator to individually assign each port.

#### Dynamic port assignment

You can control which ports are used by changing the range of ports available to the dynamic assignment process. Doing this leaves dynamic port assignment in place but
restricts the ports that can be chosen. With dynamic port assignment in place, you can still choose to assign ports for certain processes manually. This approach enables you to assign specific ports to specific processes, while leaving the others to be mapped dynamically. Using dynamic mapping, with or without some individually assigned ports is the easiest approach, and should satisfy the requirements of most customers. Dynamically assigned ports are preserved if you export your Tableau Server configuration.

By default, ports are assigned for each service or process from available ports between 8000 to 9000. This assignment takes place when services are installed for the first time on a node. After Tableau Server is initialized, you can see which ports are being used by which services or processes by running this command:

```
 tsm topology list-ports
```

If you have a multi-node cluster, ports on all nodes are listed.

Changing the port range

For organizations that have specific requirements for ports being used, the easiest way to control this is to change the range from which ports are dynamically selected. You need to do this at installation, by specifying a minimum and maximum port for the range in your configuration file.

```
"configKeys": {
   "ports.range.min": "8300",
```

**Note:** The minimum allowable size of your port range will depend on your server installation and how many services or processes you are running. As a general best practice you should not restrict the range too tightly because port assignment is done by selecting random ports within the range, and if you do not allow a large enough range, selection may fail to find an available port.

To limit the range from which available ports are chosen to those between 8300 and 8600, your configuration file would include an entry similar to this:

```
"configKeys": {
   "ports.range.min": "8300",
```

- 1064 -
"ports.range.max": "8600"
}

Disabling dynamic port assignment

If you need more control of port assignment than you can get through a combination of restricting port range and individually assigning ports, you can disable dynamic port mapping at initial server configuration. Disabling dynamic port mapping requires you to manually assign every port for every process, so we don't recommend this unless you need to control every single port assignment.

To disable dynamic mapping, your configuration file would include an entry similar to this:

"configKeys": {
    "service.port_remapping.enabled": true
}

**Important:** If you disable dynamic port mapping, you must specify every port for every process on every node of your installation.

Manual port assignment

You can disable automatic port assignment entirely and assign a port for each process individually. If you do this, you must assign a port for every process on every node. You can assign ports either in a configuration file, when the processes are first installed, or after installation, using a TSM command. Only ports assigned at process installation are preserved if you export your Tableau Server configuration.

You can specify individual ports for specific processes, whether or not dynamic mapping is enabled. You might do this if you want a process to use a particular port, or if you've disabled dynamic mapping. There are two ways to specify ports for processes:

- At install, using a configuration file. This is similar to defining a non-default port range, but instead you specify a particular port for a specific service or process. If you are
going to assign specific ports, this approach is the most robust way to do so because the port mapping is preserved if you export the server configuration and topology settings using the `tsm settings export` command.

To define ports at installation, add information to your configuration file to specify the node (`workerN`), process (`servicename`) and instance ID (`instanceid`), port type (`porttype`), and the port to be used. The format looks like this:

`workerN.{servicename}_{instanceid}.{porttype}.port:X`

Where:

- `workerN` is an optional parameter and identifies the node for which the remapping applies. Node numbers start with zero (0). We recommend you do not include this parameter unless you need to map different ports for the same service on different nodes. If you leave this parameter off, you can map a service port on the initial node, or map the same service port on multiple nodes.
- `servicename` is the name of the process or service that will use the port.
- `instanceid` is the instance of the process. If you are going to be configuring multiple instances of a process on one node, you would need to increment this value for each instance. Start the instanceid at zero (0) and increment it by one (1) for each instance of the process. For services that only install a single instance on any given node, this must be left off.
- `porttype` If setting the primary port, do not include this option.
- `port` is the port the process or service should use.

For example, to set the port for the first instance of the file store process on the initial node to 8500, you would include a configuration file entry similar to this:

```
"configKeys": {
    "filestore_0.port": "8500"
}
```
The example above does not include the optional workerN parameter, so sets the port on all workers in the cluster. It also leaves off the porttype option because it is setting the primary filestore port.

- After install, using the `tsm topology set-ports` command. This approach allows you to specify a port for a specific process after that process has been installed. You are restricted by these limits:
  - You must set ports individually, on each node.
  - The port assignments are not preserved if you need to import a Tableau Server configuration using `tsm settings import`.

To set second instance of the file store on the initial node to use port 8500:

```
tsm topology set-ports --node-name node1 --port-name filestore --port-value 8500
```

### Ports that are not dynamically mapped

The Tableau Server repository uses two ports that are not dynamically mapped. These each have a default port that you can override using the `tsm topology set command` and a process-specific parameter.

<table>
<thead>
<tr>
<th>Port names</th>
<th>Port (default)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pgsql.port</code></td>
<td>8060</td>
<td>Port for the Tableau Repository (PostgreSQL database).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To override this port:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>tsm topology set -k pgsql.port &lt;port&gt;</code></td>
</tr>
<tr>
<td><code>pgsql.verify_restore.port</code></td>
<td>8061</td>
<td>Port for verifying the integrity of a repository backup.</td>
</tr>
</tbody>
</table>
Because these ports do not use the dynamic port mapping system, they do not show up in the output of the `tsm topology list-ports` command. To see the value of these you need to use the `tsm topology get -k <config.value>` command. For example:

```
  tsm topology get -k pgsql.port
```

Controlling port remapping with `initialize-tsm`

Port assignments are made when services are installed. This means that in order to manually map ports for the TSM-specific processes, you need to assign the ports when you run the `initialize-tsm` script. The script includes options to specify ports for individual TSM services, as well as options for defining the minimum and maximum of the port range used with dynamic mapping, and you can disable dynamic mapping.

The table below lists the options for ports when running the `initialize-tsm` script.

**Table: `initialize-tsm` script port options**

<table>
<thead>
<tr>
<th>Script option</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i</td>
<td>&lt;port&gt;</td>
<td>Sets the Coordination Service client port.</td>
</tr>
<tr>
<td>-e</td>
<td>&lt;port&gt;</td>
<td>Sets the Coordination Service peer</td>
</tr>
</tbody>
</table>
### Script option

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;port&gt;</td>
<td>port.</td>
</tr>
<tr>
<td>-m</td>
<td>&lt;port&gt;</td>
</tr>
<tr>
<td>-n</td>
<td>&lt;port&gt;</td>
</tr>
<tr>
<td>-o</td>
<td>&lt;port&gt;</td>
</tr>
<tr>
<td>-l</td>
<td>&lt;min-port&gt;</td>
</tr>
<tr>
<td>-r</td>
<td>&lt;max-port&gt;</td>
</tr>
<tr>
<td>--disable-port-remapping</td>
<td>Disables dynamic port mapping. If you do this you must assign ports for every service or process used by TSM and Tableau Server. For more information, see Manual port assignment above.</td>
</tr>
</tbody>
</table>

---

The below table lists the processes or services that use dynamically mapped ports.

<table>
<thead>
<tr>
<th>Port names</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>You must define each of these for every node if you disable dynamic port mapping.</td>
<td></td>
</tr>
<tr>
<td>appzookeeper_0.client.port</td>
<td>Coordination Service client port.</td>
</tr>
<tr>
<td>appzookeeper_0.peer.port</td>
<td>Coordination Service client port.</td>
</tr>
<tr>
<td>Port names</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>appzookeeper_0.leader.port</td>
<td>Coordination Service client port.</td>
</tr>
<tr>
<td>backgrounder_0.port</td>
<td>Backgrounder primary port.</td>
</tr>
<tr>
<td>backgrounder_0.debug.port</td>
<td>Backgrounder debug port.</td>
</tr>
<tr>
<td>backgrounder_0.jmx.port</td>
<td>Backgrounder jmx port.</td>
</tr>
<tr>
<td>backgrounder_0.jmx.rmi.port</td>
<td>Backgrounder jmx rmi port.</td>
</tr>
<tr>
<td>backgrounder_0.recommendations.trainer.port</td>
<td>Backgrounder recommendations port.</td>
</tr>
<tr>
<td>backuprestore.port</td>
<td>Backup/Restore service port.</td>
</tr>
<tr>
<td>cacheserver_0.port</td>
<td>Cache server port.</td>
</tr>
<tr>
<td>clustercontroller.status.port</td>
<td>Cluster Controller status port.</td>
</tr>
<tr>
<td>clustercontroller.storage.port</td>
<td>Cluster Controller storage port.</td>
</tr>
<tr>
<td>databasemaintenance.port</td>
<td>Database Maintenance port.</td>
</tr>
<tr>
<td>dataserver_0.port</td>
<td>Data server primary port.</td>
</tr>
<tr>
<td>dataserver_0.debug.port</td>
<td>Data server debug port.</td>
</tr>
<tr>
<td>dataserver_0.jmx.port</td>
<td>Data server jmx port.</td>
</tr>
<tr>
<td>dataserver_0.jmx.rmi.port</td>
<td>Data server jmx rmi port.</td>
</tr>
<tr>
<td>filestore.port</td>
<td>File store primary port.</td>
</tr>
<tr>
<td>filestore.status.port</td>
<td>File Store status port.</td>
</tr>
<tr>
<td>gateway.port</td>
<td>Gateway port. This defaults to 80, and if that is not available, to 8080. If that is not available, it tries 8000. That sequence is followed whether or not dynamic port assignment is enabled or not. If none</td>
</tr>
</tbody>
</table>
Port names | Description
---|---
hyper_0.port | Data engine primary port.
hyper_0.connection.port | Data engine connection port.
samlservice.port | SAML service port.
searchserver.port | Search server primary port.
searchserver.debug.port | Search server debug port.
searchserver.jmx.port | Search server jmx port.
searchserver.jmx.rmi.port | Search server jmx rmi port.
searchserver.startup.port | Search server startup port.
siteimportexport.port | Site Import/Export port.
tabadmincontroller.port | TSM Controller port.
tabadminagent.filetransfer.port | TSM Agent file transfer port.
vizportal_0.authentication.port | Application server authentication port.
vizportal_0.authorization.port | Application server authorization port.
vizportal_0.maintenance.port | Application server maintenance port.
vizportal_0.publishing.port | Application server publishing port.
vizportal_0.recommendations.port | Application server recommendations port.

of those ports are available and dynamic mapping is enabled, it takes an available port within the defined range. The gateway port must be the same on all nodes in a multi-node cluster, so if port 80 is selected on the initial node this is the port that will be used on all nodes and if it is unavailable on one of the other nodes, gateway port selection will fail.
<table>
<thead>
<tr>
<th>Port names</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vizportal_0.port</td>
<td>Application server primary port.</td>
</tr>
<tr>
<td>vizportal_0.debug.port</td>
<td>Application server debug port.</td>
</tr>
<tr>
<td>vizportal_0.jmx.port</td>
<td>Application server jmx port.</td>
</tr>
<tr>
<td>vizportal_0.jmx.rmi.port</td>
<td>Application server jmx rmi port.</td>
</tr>
<tr>
<td>vizqlserver_0.port</td>
<td>VizQL server primary port.</td>
</tr>
<tr>
<td>vizqlserver_0.debug.port</td>
<td>VizQL server debug port.</td>
</tr>
<tr>
<td>vizqlserver_0.jmx.port</td>
<td>VizQL server jmx port.</td>
</tr>
<tr>
<td>vizqlserver_0.jmx.rmi.port</td>
<td>VizQL server jmx rmi port.</td>
</tr>
</tbody>
</table>

Enable the JMX Ports

To help you work through a problem with Tableau Server, Tableau Support may ask you to enable the server's JMX ports. These ports can be useful for monitoring and troubleshooting, usually with a tool like JConsole.

To enable the JMX ports on Tableau Server:

1. Stop the server.
2. Enter the following command:
   
   ```bash
   tsm configuration set -k service.jmx_enabled -v true
   ```
3. Apply pending changes:
The `pending-changes apply` command displays a prompt to let you know this will restart Tableau Server if the server is running. The prompt displays even if the server is stopped, but in that case there is no restart. You can suppress the prompt using the `-r` option, but this does not change the restart behavior. For more information, see `tsm pending-changes apply`.

4. **Start the server.**

**Important** Enabling JMX ports can introduce some security risk. To mitigate this risk, it is important to limit access to the JMX ports to the fewest number of clients that’s practical for your scenario. You typically limit access using the host's firewall rules, an external security device, or routing rules.

How the JMX Ports Are Determined

By default, the JMX ports assigned dynamically, from within a range of available ports. For details on how port assignment is done, and how to override dynamic mapping, see Tableau Services Manager Ports.

Set Licensing Daemon Port

If you are running Tableau Server in a multinode deployment (cluster) and you have enabled a firewall on the computers running Tableau Server, then you must manually configure the licensing daemon port on each node in the cluster.

If you do not set this port, licensed processes will not start on the cluster. See Process Reference to learn more about licensed processes.

Configuring licensing daemon port during installation (recommended)

We recommend manually updating the port configuration before you initialize TSM. If you have already initialized TSM then you must follow the procedure in the next section, Configuring licensing daemon port after TSM has been initialized.
Follow this procedure on each node in the cluster as part of the node installation process.

**To configure licensing daemon port during installation process**

1. Install the Tableau Server package.

2. After the package is installed, before you start TSM, you must edit these two files:

   ```
   /opt/tableau/tableau_server-
   /packages/templates.<version>/license/tableau.lic.templ
   
   /opt/tableau/tableau_server-
   /packages/bin.<version>/tableau.lic
   ```

   Open the files with a text editor and append the line, `VENDOR tableau, with port=t=27010`, as follows:

   ```
   VENDOR tableau port=27010
   ```

3. Verify that there are no other files to update. Run the following command:

   ```
   sudo find /opt /var -name "tableau.lic*"
   ```

   If the `find` command returns files that are in the `/var` directory, then TSM has already been initialized and you must follow the procedure below.

4. Save the files and then continue with the installation procedure by starting TSM. See Start TSM.

**Configuring licensing daemon port after TSM has been initialized**

If you have already initialized TSM then you must follow the procedure below to update all instances of the port configuration on each node in the cluster. If you add more nodes to the cluster, you can save time by following the procedure above during the subsequent installations.

**To configure licensing daemon port after TSM has been initialized**
1. Log in to TSM and enter the following command:

   tsm stop

2. Run the find command to identify all instances of `tableau.lic` and `tableau-lictempl` files in the /opt and /var directories. For example, run the following command:

   sudo find /opt /var -name "tableau.lic*"

3. Open the files with a text editor and append the line, `VENDOR tableau, with port-t=27010`, as follows:

   VENDOR tableau port=27010

4. Save the files and then run the following command:

   tsm start

**Help Output for initialize-tsm Script**

The following help content is the output when you run the following command:

   sudo ./initialize-tsm -h

The `initialize-tsm` script is installed to `/opt/tableau/tableau_server-/packages/scripts.<version>/`

**Output**

**REQUIRED**

--accepteula

Indicate that you have accepted the End User License Agreement (EULA).

You can find the EULA in `/opt/tableau/tableau_server-/packages/docs.<version>`

**OPTIONAL**

-c config-name

Set the service configuration name.
If not set, the default is "tabsvc".

-d data-directory
Set a custom location for the data directory if it's not already set. If not set, the default is "/var/opt/tableau/tableau_server".

-b bootstrap-file
Optional. Location of the bootstrap file downloaded from the Tableau Server Manager on existing node. Must be provided to join existing Tableau Server cluster.

-u username
Name of the user with admin privileges on existing Tableau Server Manager.
Required if -b option specified.

-p password
Password for the Tableau Server Manager admin user.

-f
Bypass warning messages.

-g
Do NOT add the current user to the "tsmadmin" administrative group, used for default access to Tableau Server Manager, or to the "tableau" group, used for easier access to log files.

-a username
The provided username will be used as the user to be added to the appropriate groups, instead of the user running this script. Providing both -a and -g is not allowed.

-q
Quiet, suppress output except for errors and warnings.
-i coordinationservice-client-port  Client port for the coordination service

-e coordinationservice-peer-port  Peer port for the coordination service

-m coordinationservice-leader-port  Leader port for the coordination service

-n agent-filetransfer-port  Filetransfer port for the agent service

-o controller-port  Https port for the controller service

-l port-range-min  Lower port bound for automatic selection

-r port-range-max  Upper port bound for automatic selection

--disable-port-remapping  Disable automatic port selection

--unprivileged-user=<value>  Name of the unprivileged account to run Tableau Server. Default: "tableau".

--privileged-user=<value>  Name of the privileged account to run Tableau Server Manager. Default: "tsmagent".

--server-authorized-group=<value>  Name of the group that allows authorization to Tableau Server.
Default: "tableau".

--tsm-authorized-group=<value> Name of the group(s) that allows authorization to access Tableau Server Manager. Default: "tsmadmin".

--disable-account-creation Do not create groups or user accounts for Server and TSM authorization. However, the values in: unprivileged-username, authorized-groupname, privileged-username, and tsm-authorized-groupname still be used in TSM configuration.

--debug Print each command as it is run for debugging purposes. Produces extensive output.

Related topics

- Controlling port remapping with initialize-tsm
- Install Tableau Server Package

View Server Version

Server users can view the version of Tableau Server from the Help menu in the server web UI.

Tableau Services Manager (TSM) administrators can view the versions of TSM and Tableau Server from the TSM command line (CLI).
Viewing the server version from the Tableau Server web UI

- While logged into Tableau Server, click the information icon (i) and About Tableau Server.

The version of Tableau Server is listed in the About Tableau Server dialog box:

Viewing the server version and TSM version from the TSM command line

1. Open a command prompt as administrator on the initial node (the node where TSM is installed).

2. Run the following command:

    tsm version

The output displays the versions of Tableau Services Manager (TSM) and Tableau Server:
Tableau Services Manager command line version 20191.18.1023.1646.
Tableau Server version 20191.18.1023.1646.