Tableau Server 企业部署指南

上次更新 2025/2/10

目录

Tableau Server 企业部署指南

Tableau Server 企业部署指南(EDG)的制定旨在提供有关(在本地或在云中)部署 Tableau Server 的说明性指南。该指南在参考体系结构的上下文中提供了针对企业方案的部署指南。我们已经测试了参考体系结构,以验证是否符合安全性、规模和性能基准,这些基准符合行业标准的最佳实践。

在高层,行业标准企业部署的核心功能由分层拓扑组成,其中服务器应用程序功能的每个层(Web 网关层、应用程序层和数据层)均受访问控制的子网限定和保护。从 Internet 访问服务器应用程序的用户在 Web 层进行身份验证。身份验证完成后,该请求将被代理到受保护的子网,应用程序层将在该子网中处理业务逻辑。高价值数据受到第三个子网(数据层)的保护。应用程序层中的服务通过受保护的网络与数据层进行通信,以向后端数据源提供数据请求。

在此部署中,安全性是所有设计决策和实施的重中之重。但是,可靠性、性能和可扩展性也是优先要求。给定参考体系结构的分布式和模块化设计,通过将每个节点上的兼容服务进行策略性共置并在阻塞点处添加服务,可以通过线性可预测的方式扩展可靠性和性能。

谁应阅读本指南

EDG 是为企业 IT 管理员开发的,他们可能需要:

- IT 管理的 Tableau 部署
- 行业合规实施
- 行业部署最佳实践
- 默认安全部署

EDG 是用于部署企业参考架构的实施指南。虽然此版本的 EDG 包含一个示例 AWS/Linux 实施,但经验丰富的企业 IT 管理员可以将本指南用作资源,将规定的参考架构部署到任何行业标准数据中心环境中。

版本

此版本的 EDG 针对 Tableau Server 的 2021.2.3 版本(或更高版本)制定。虽然您可以将 EDG 用作部署较旧版本的 Tableau Server 的一般参考,但建议您将参考体系结构与 Tableau Server 2021.2.3 或更高版本一起部署。某些功能和选项在较旧版本的 Tableau Server 上不可用。

如需获得最新的功能和改进,我们建议将 EDG 与 Tableau Server 2022.1.7 及更高版本一起部署。

本指南中描述的参考架构支持以下 Tableau 客户端:使用兼容浏览器的 Web 制作、Tableau Mobile 和 Tableau Desktop 版本 2021.2.1 或更高版本。其他 Tableau 客户端(Tableau Prep、Bridge 等)尚未通过参考架构进行验证。

突出特点

Tableau Server 参考架构的第一个版本引入了以下方案和功能:

- 客户端预身份验证:在访问内部 Tableau Server 之前,Tableau 客户端(Desktop、Mobile、Web 制作)都通过 Web 层中的企业身份验证提供程序进行身份验证。此过程是通过在 Tableau Server 独立网关上配置一个 authN 插件作为反向代理服务器来管理的。请参见第 5 部分 - 配置 Web 层。
- 零信任部署:由于到 Tableau Servers 的所有流量都经过预身份验证,因此整个 Tableau 部署在不需要可信连接的私有子网中运行。
- 外部存储库:参考架构指定将 Tableau 存储库安装到外部 PostgreSQL 数据库上,允许 DBA 将存储库作为通用数据库进行管理、优化、扩展和备份。
- 初始节点恢复:EDG 引入了一个脚本,可在发生故障时自动进行初始节点恢复。
- 基于 tar 的备份和还原:在 Tableau 部署的战略里程碑处使用熟悉的 tar 备份。如果发生故障或部署错误配置,您可以通过恢复关联的 tar 备份快速恢复到之前的部署阶段。
- 性能改进:客户和实验室验证表明,与标准部署相比,运行 EDG 时性能提高了 15-20%。

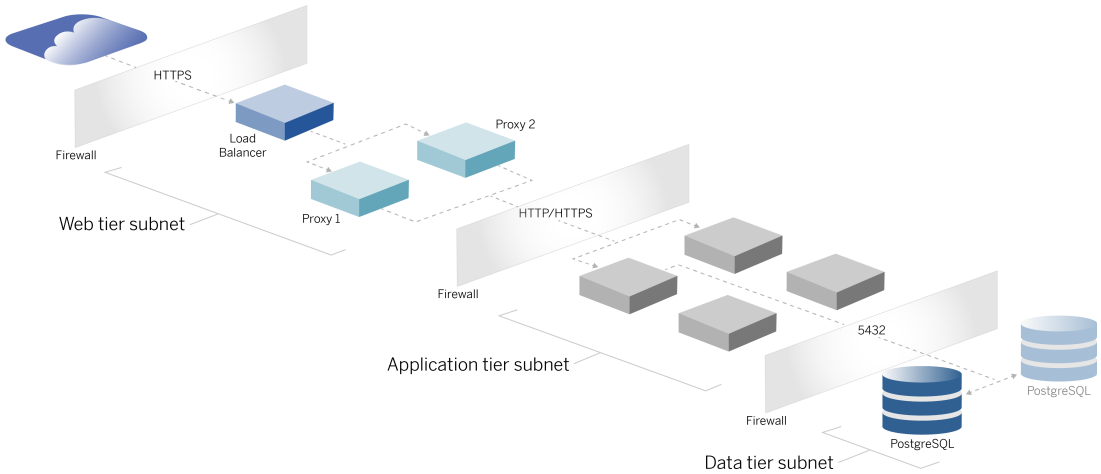
许可

本指南中规定的 Tableau Sever 参考架构需要 Tableau Advanced Management 许可证才能启用 Tableau Server 外部存储库。您还可以选择部署 Tableau Server 外部文件存储，这也需要 Tableau Advanced Management 许可证。请参见“*关于 Tableau Server 上的 Tableau Advanced Management*”(Linux)。

第 1 部分 - 了解企业部署

第 1 部分更详细地描述了行业标准企业部署的功能和要求，Tableau Server 企业部署指南就是针对这些特性和要求而设计的。

以下网络图显示了带有 Tableau Server 参考架构的通用数据中心分层部署。



行业标准和部署要求

以下是行业标准部署的功能。以下是参考体系结构的设计要求：

- 多层网络设计：网络受所保护子网的限制，以限制每一层的访问：Web 层，应用程序层和数据层。由于所有通信都在下一个子网处终止，因此无法跨子网进行任何通信。
- 默认情况下阻止的端口和协议：默认情况下，每个子网或安全组将阻止所有入站和出站端口及协议。通过在端口或协议配置中打开例外来部分启用通信。
- 现成的 Web 身份验证：来自 Internet 的用户请求通过 Web 层中反向代理上的身份验证模块进行身份验证。因此，在传入受保护的应用程序层之前，将在 Web 层上验证对应用程序层的所有请求。
- 与平台无关：解决方案可以与本地服务器应用程序一起部署，也可以在云中部署。

- 与技术无关:解决方案可以部署在虚拟机环境或容器中。也可以部署在 Windows 或 Linux 上。但是,参考架构和支持文档的初始版本是为在 AWS 中运行的 Linux 开发的。
- 高度可用:系统中的所有组件均以群集方式部署,并设计为在主动/主动或主动/被动部署中运行
- 孤立角色:每个服务器都扮演不同的角色。这种设计对所有服务器进行了分区,以便可将特定于服务的管理员的访问降至最低。例如,DBA 管理 PostgreSQL for Tableau,身份管理员管理 Web 层中的身份验证模块,网络和云管理员启用流量和连接。
- 线性可扩展:作为不同的角色,您可以根据负载配置文件独立扩展每个层服务。
- 客户端支持:参考架构支持所有 Tableau 客户端:Tableau Desktop(版本 2021.2 或更高版本)、Tableau Mobile 和 Tableau Web Authoring。

安全措施

如上所述,行业标准数据中心设计的主要功能是安全性。

- 访问:每个层受一个子网约束,该子网使用端口筛选在网络层实施访问控制。子网之间的通信访问也可以由应用程序层通过进程之间经过身份验证的服务来实施。
- 集成:体系结构设计为在 Web 层中的反向代理上插入身份验证提供程序 (IdP)。
- 隐私:客户端使用 SSL 对进入 Web 层的流量进行加密。进入内部子网的流量也可以根据需要进行加密。

Web 代理层

Web 层是 DMZ(也称为外围区域)中的一个子网,充当 Internet 与部署应用程序的内部子网之间的安全缓冲区。Web 层托管不存储任何敏感信息的反向代理服务器。反向代理服务器配置有一个 AuthN 插件,用于在将客户端请求重定向到 Tableau Server 之前,使用受信任的 IdP 对客户端会话进行预身份验证。有关详细信息,请参见使用 AuthN 模块进行预身份验证。

负载平衡器

部署设计包括在反向代理服务器之前部署企业负载平衡解决方案。

负载平衡器通过以下方式提供了重要的安全性和性能增强:

- 虚拟化应用程序层服务的前端 URL
- 实施 SSL 加密
- 卸载 SSL
- 在客户端和 Web 层服务之间实施压缩
- 抵御 DOS 攻击
- 提供高可用性

注意：Tableau Server 版本 2022.1 包括 Tableau Server 独立网关。独立网关是 Tableau 网关进程的独立实例，用作 Tableau 感知反向代理。在发布时，独立网关已经过验证，但尚未在 EDG 参考架构中进行全面测试。完整测试完成后，EDG 将使用 Tableau Server 独立网关说明性指南进行更新。

应用程序层

应用程序层位于运行服务器应用程序核心业务逻辑的子网中。应用程序层由跨群集中的分布式节点配置的服务和流程组成。仅可从 Web 层访问应用程序层，用户不能直接访问。

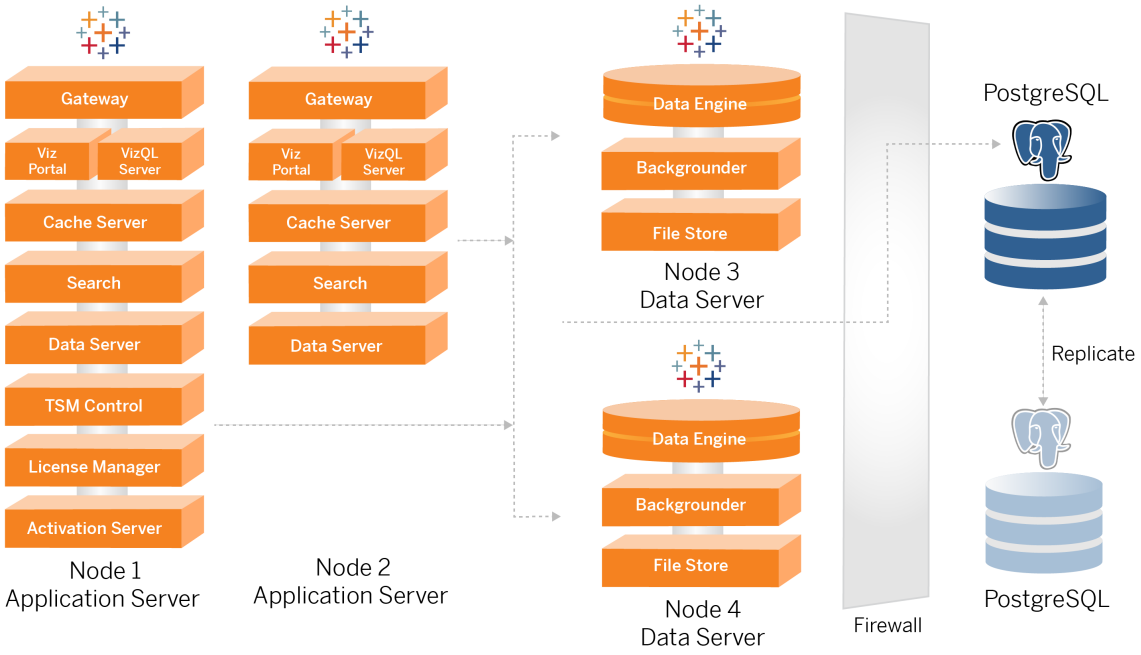
通过配置应用程序进程，使具有不同资源使用配置文件(即，CPU 密集型与内存密集型)的进程位于同一位置，可以提高性能和可靠性。

数据层

数据层是一个保存有价值数据的子网。进入此层的所有流量都来自应用程序层，因此已经过身份验证。除了具有端口配置的网络层上的访问要求外，该层还应包括经过身份验证的访问，以及可选的应用程序层加密流量。

第 2 部分 - 了解 Tableau Server 部署参考体系结构

下图显示了相关的 Tableau Server 进程以及它们在参考架构中的部署方式。此部署被认为是适合企业的最小 Tableau Server 部署。



本主题中的流程图旨在显示每个节点的主要定义流程。许多支持进程也在图中未显示的节点上运行。有关所有进程的列表，请参见本指南的配置部分，即第 4 部分 - 安装并配置 Tableau Server。

Tableau Server 进程

Tableau Server 参考体系结构是一个四节点的 Tableau Server 群集部署，在 PostgreSQL 上有外部存储库：

- Tableau Server 初始节点(节点 1):运行所需的 TSM 管理和许可服务,这些服务只能在群集中的单个节点上运行。在企业情境中,Tableau Server 初始节点是群集中的主节点。此节点还与节点 2 一起运行冗余应用程序服务。
- Tableau Server 应用程序节点(节点 1 和节点 2):这两个节点提供客户端请求、连接并查询数据源以及数据节点。
- Tableau Server 数据节点(节点 3 和节点 4):两个专用于管理数据的节点。
- 外部 PostgreSQL:此主机运行 Tableau Server 存储库进程。对于 HA 部署,您必须运行额外的 PostgreSQL 主机以实现主动/被动冗余。

  您还可以在 Amazon RDS 上运行 PostgreSQL。有关在 RDS 上运行存储库与在 EC2 实例上运行存储库之间的差异的详细信息,请参见*"Tableau Server 外部存储库"*(Linux)。

  使用外部存储库部署 Tableau Server 需要 Tableau Advanced Management 许可证。

  如果您的组织没有内部 DBA 专业知识,您可以选择在默认的内部 PostgreSQL 配置中运行 Tableau Server 存储库进程。在默认情况下,存储库在具有嵌入式 PostgreSQL 的 Tableau 节点上运行。在这种情况下,我们建议在专用 Tableau 节点上运行存储库,并在其他专用节点上运行被动存储库以支持存储库故障转移。请参见"*存储库故障转移* (Linux)"。

  例如,本指南中描述的 AWS 实现解释了如何在 EC2 实例上运行的 PostgreSQL 上部署外部存储库。

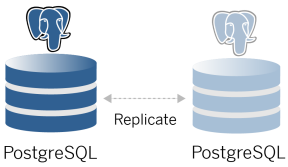
- 可选:如果您的组织使用外部存储,您可以将 Tableau 文件存储部署为外部服务。本指南不包括核心部署方案中的外部文件存储。请参见"*使用外部文件存储安装 Tableau Server*(Linux)"。

  使用外部文件存储部署 Tableau Server 需要 Tableau Advanced Management 许可证。

PostgresSQL 存储库

Tableau Server 存储库是用于存储服务器数据的 PostgresSQL 数据库。此数据包括有关 Tableau Server 用户、组和组分配、权限、项目、数据源和数据提取元数据的信息以及刷

新信息。



默认的 PostgresSQL 部署会消耗将近 50% 的系统内存资源。根据其使用情况(用于生产和大型生产部署),资源使用量可能会上升。因此,我们建议在未运行任何其他资源密集型服务器组件(比如 VizQL、后台程序或数据引擎)的计算机上运行存储库进程。与任何这些组件一起运行存储库进程将产生 IO 争用、资源约束,并降低部署的整体性能。

节点 1:初始节点

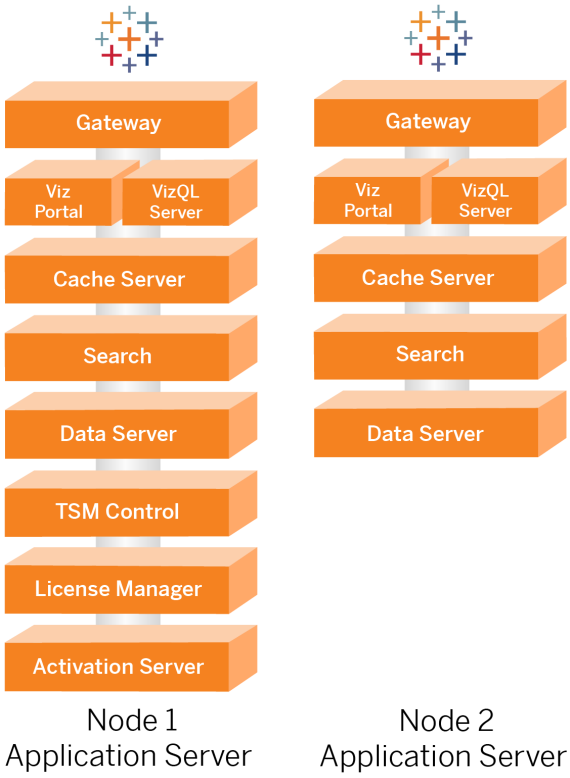
初始节点运行少量重要进程,并与节点 2 共享应用程序负载。

您在其中安装 Tableau 的第一台计算机(即“初始节点”)有一些特有的特征。三个进程只能运行在初始节点上,并且不能转移到任何其他节点(除非在出现故障的情况下):许可证服务(许可证管理器)、激活服务和 TSM 控制器(管理控制器)。

节点 1 故障转移和自动恢复

许可、激活和 TSM 控制器服务对于 Tableau Server 部署的运行状况至关重要。如果节点 1 发生故障,用户仍然可以连接到 Tableau Server 部署,因为正确配置的参考架构会将请求路由到节点 2。但是,如果没有这些核心服务,部署将处于挂起失败的临界状态。请参见初始节点自动恢复。

节点 1 和 2:应用程序服务器



节点 1 和 2 运行 Tableau Server 进程，这些进程提供客户端请求、查询数据源、生成可视化项、处理内容和管理以及其他核心 Tableau 业务逻辑。应用服务器不存储用户数据。

注意:“应用程序服务器”是一个术语，也指 TSM 中列出的 Tableau Sever 进程。“应用程序服务器”的基础进程是 VizPortal。

并行运行时，节点 1 和节点 2 可扩展以服务来自反向代理服务器上运行的负载平衡逻辑的请求。作为冗余节点，如果这些节点之一出现故障，则客户端请求和服务由其余节点处理。

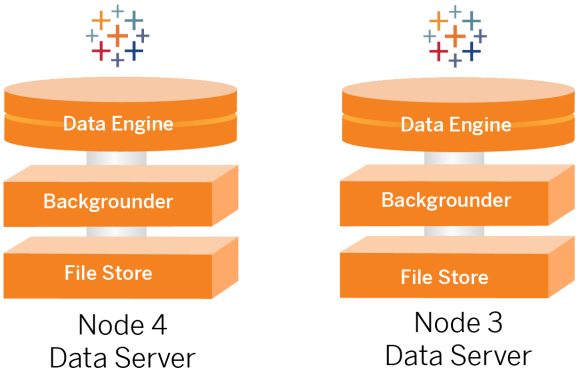
参考架构的设计使互补的应用程序进程在同一台计算机上运行。这意味着进程不会争用计算资源并产生争用。

例如，VizQL(应用程序服务器上的一个核心处理服务)受到 CPU 和内存的高度限制；VizQL 占用计算机上 CPU 和内存的近 60-70%。因此，设计参考体系结构的目的是使其他受内存或 CPU 约束的进程与 VizQL 不在同一节点上。测试表明，负载量或用户数不会影响 VizQL 节点上的内存或 CPU 使用率。例如，减少负载测试中的并发用户数只会影响仪表板或可视化项加载过程的性能，而不会降低资源利用率。因此，根据使用高峰期的可用内存和 CPU，您可以考虑添加更多 VizQL 进程。作为典型工作簿的起始位置，为每个 VizQL 进程分配 4 个内核。

扩展应用服务器

参考架构旨在依据基于使用的模型进行扩展。作为一般起点，我们建议至少使用两个应用程序服务器，每个应用程序服务器最多支持 1000 个用户。随着用户群的增加，计划每增加 1000 个用户就添加一个应用程序服务器。监控使用情况和性能，为您的组织调整每台主机的用户群。

节点 3 和 4:数据服务器



由于以下原因，文件存储、数据引擎 (Hyper) 和后台程序进程位于节点 3 和 4 上的同一位置：

- 数据提取优化：在同一节点上运行后台程序、Hyper 和文件存储可优化性能和可靠性。在提取过程中，后台程序查询目标数据库，在同一节点上创建 Hyper 文件，然后上载到文件存储。通过将这些过程放置在同一节点上，数据提取创建工作流程不需要跨网络或节点复制大量数据。

- 免费资源平衡:后台程序主要占用大量 CPU。数据引擎是一个占用大量内存的进程。将这些进程结合起来可以最大限度地利用每个节点上的资源。
- 数据进程的整合:由于这些进程都是后端数据进程,因此有必要在最安全的数据层中运行它们。在参考架构的未来版本中,应用程序和数据服务器将在不同的层中运行。但是,由于 Tableau 架构中的应用程序依赖性,此时应用程序和数据服务器必须在同一层中运行。

扩展数据服务器

与应用程序服务器一样,规划 Tableau 数据服务器所需的资源需要基于使用的建模。通常,假设每个数据服务器每天最多可以支持 2000 个数据提取刷新作业。随着数据提取作业的增加,请在没有文件存储服务的情况下添加其他数据服务器。通常,双节点数据服务器部署适用于为文件存储服务使用本地文件系统的部署。请注意,添加更多应用程序服务器不会以线性方式影响数据服务器的性能或规模。事实上,除了来自额外用户查询的一些开销外,添加更多应用程序主机和用户的影响很小。

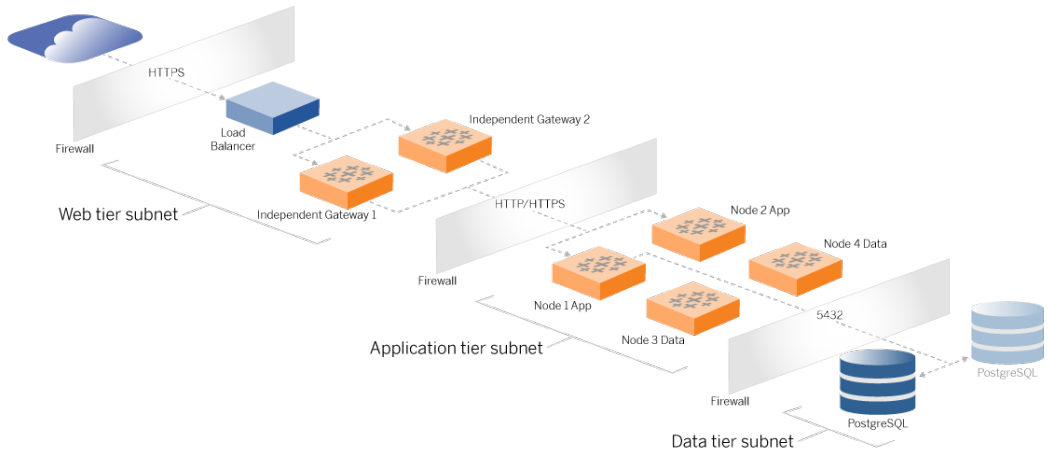
第 3 部分 - 准备 Tableau Server 企业部署

第 3 部分描述了准备基础结构以部署 Tableau Server 参考体系结构的要求。在开始之前，我们建议您先阅读第 2 部分 - 了解 Tableau Server 部署参考体系结构。

除了需求描述之外，本主题还提供了 AWS 环境中参考架构的实施示例。本指南的其余部分基于本主题中开始的 AWS 参考架构示例。

参考体系结构的核心原则是采用数据中心安全最佳实践进行标准化。具体而言，该体系结构旨在将服务隔离到受保护的网络子网中。子网间通信仅限于特定的协议和端口流量。

下图说明了本地部署或客户管理的云部署的参考体系结构子网设计。有关示例云部署，请参见下面的示例：在 AWS 中配置子网和安全组部分。



子网

创建三个子网：

- Web 层
- 应用层
- 数据子网。

防火墙/安全组规则

下面的选项卡描述了数据中心每一层的防火墙规则。有关 AWS 特定的安全组规则，请参见本主题后面的部分。

Web 层

Web 层是一个公共 DMZ 子网，它将处理入站 HTTPS 请求并将请求代理到应用程序层。此设计针对可能以您的组织为目标的恶意软件提供了一层防御。Web 层阻止对应用层/数据层的访问。

流量	类型	协议	端口范围	源
入站	SSH	TCP	22	Bastion 子网(用于云部署)
入站	HTTP	TCP	80	Internet (0.0.0.0/0)
入站	HTTPS	TCP	443	Internet (0.0.0.0/0)
出站	所有流量	所有	所有	

应用程序层

应用程序子网是 Tableau Server 部署所在的位置。应用程序子网包括 Tableau 应用程序服务器(节点 1 和节点 2)。Tableau 应用程序服务器处理用户对数据服务器的请求并运行核心业务逻辑。

应用程序子网还包括 Tableau 数据服务器(节点 3 和节点 4)。

进入应用层的所有客户端流量都在 Web 层进行身份验证。对应用程序子网的管理访问通过堡垒主机进行身份验证和路由。

流量	类型	协议	端口范围	源
入站	SSH	TCP	22	Bastion 子网(用于云部署)
入站	HTTPS	TCP	443	Web 层子网
出站	所有流量	所有	所有	

数据层

数据子网是外部 PostgreSQL 数据库服务器所在的子网。

流量	类型	协议	端口范围	源
入站	SSH	TCP	22	Bastion 子网(用于云部署)
入站	PostgreSQL	TCP	5432	应用程序层子网
出站	所有流量	所有	所有	

堡垒

大多数企业安全团队不允许从本地管理系统直接与部署在云中的节点通信。相反,所有进入云节点的管理 SSH 流量都通过堡垒主机(也称为“跳转服务器”)进行代理。对于云部署,我们建议将堡垒主机代理连接到参考架构中的所有资源。这是本地环境的可选配置。

堡垒主机对管理访问进行身份验证,并且只允许通过 SSH 协议传输的流量。

流量	类型	协议	端口范围	源	Destination
入站	SSH	TCP	22	管理计算机 IP 地址	
出站	SSH	TCP	22		Web 层子网
出站	SSH	TCP	22		应用程序层子网

示例:在 AWS 中配置子网和安全组

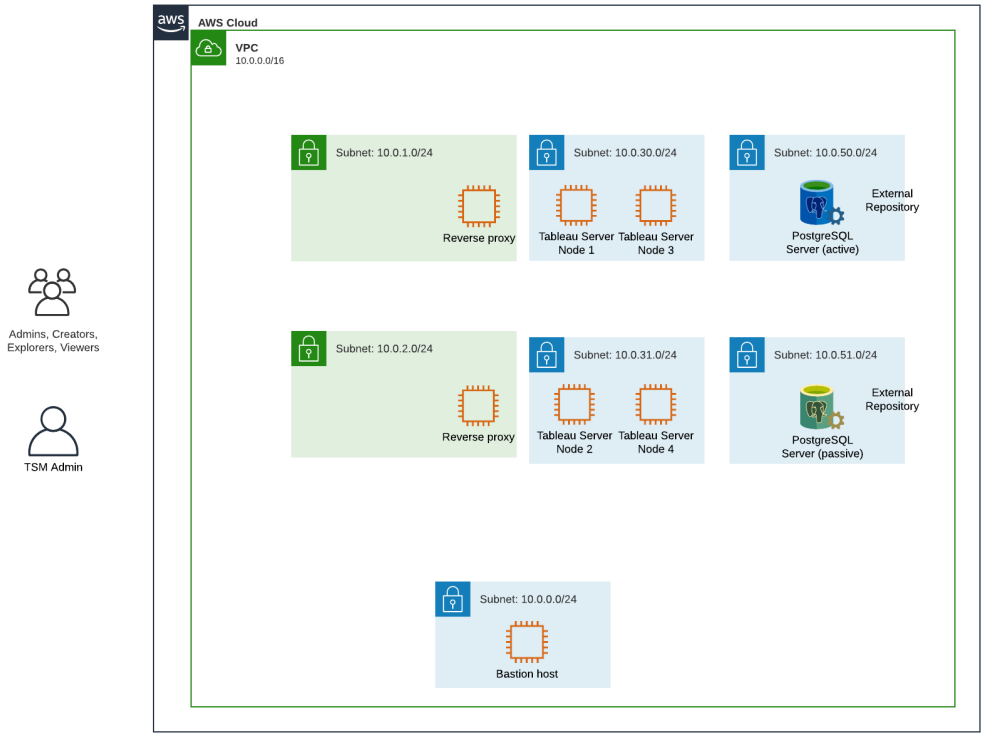
本部分提供了分步过程,用于在 AWS 中为 Tableau Server 参考架构部署创建和配置 VPC 网络环境。

下面的幻灯片显示了四个层中的参考架构。随着幻灯片的进行,组件元素将分层到拓扑图上:

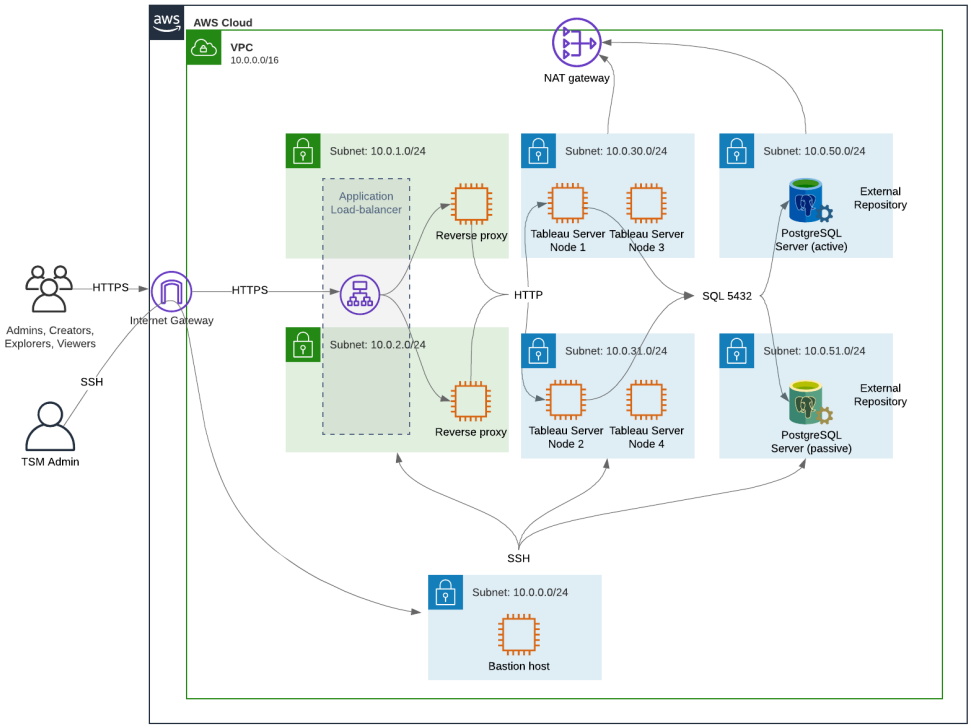
1. VPC 子网拓扑和 EC2 实例:一台堡垒主机、两个反向代理服务器、四个 Tableau 服务器和至少一个 PostgreSQL 服务器。
2. 协议流程和 Internet 连接。所有入站流量都通过 AWS Internet 网关进行管理。到 Internet 的流量通过 NAT 路由。
3. 可用区。代理、Tableau Server 和 PostgreSQL 主机均匀部署在两个可用区中。
4. 安全组。四个安全组(Public、Private、Data 和 Bastion)在协议级别保护每一层。

AWS 参考架构

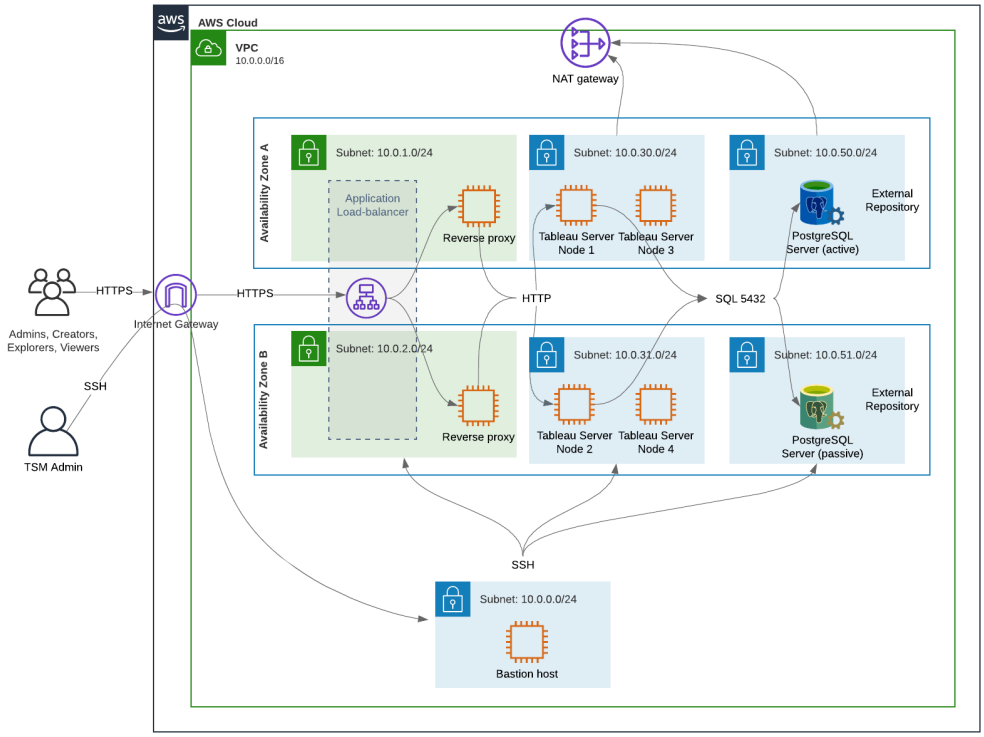
幻灯片 1:VPC 子网拓扑和 EC2 实例



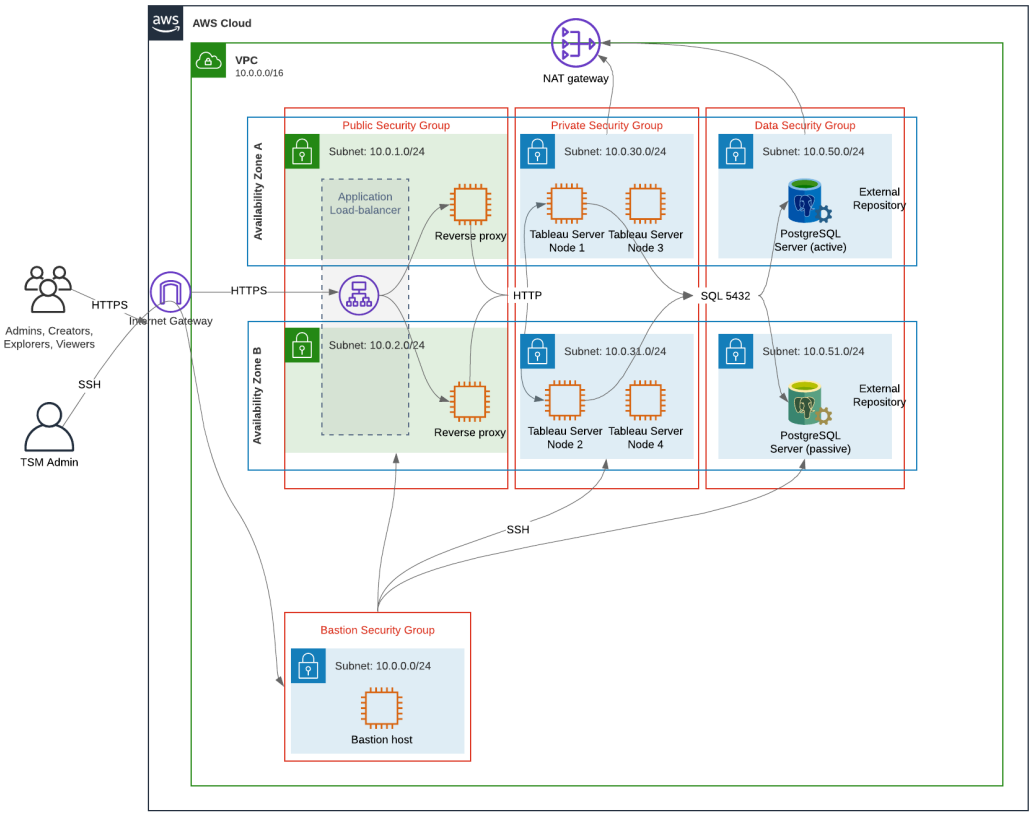
幻灯片 2:协议流程和连接



幻灯片 3:可用区


AWS Cloud
VPC
10.0.0.0/16
NAT gateway
Availability Zone A
Subnet: 10.0.1.0/24
Application
Load-balancer
Reverse proxy
Subnet: 10.0.30.0/24
Tableau Server
Node 1
Tableau Server
Node 3
Subnet: 10.0.50.0/24
PostgreSQL
Server (active)
External
Repository
Admins, Creators,
Explorers, Viewers
HTTPS
Internet Gateway
HTTPS
HTTP
SQL 5432
SSH
TSM Admin
Availability Zone B
Subnet: 10.0.2.0/24
Reverse proxy
Subnet: 10.0.31.0/24
Tableau Server
Node 2
Tableau Server
Node 4
Subnet: 10.0.51.0/24
PostgreSQL
Server (passive)
External
Repository
SSH
Subnet: 10.0.0.0/24
Bastion host

幻灯片 4:安全组



AWS 可用区和高可用性

本指南中提供的参考架构指定了一种部署，该部署可在任何单一主机出现故障时通过冗余提供可用性。但是，在 AWS 中参考架构跨两个可用区部署的情况下，可用性在可用区发生故障的极少数情况下会受到影响。

VPC 配置

本部分介绍如何：

- 安装和配置 VPC
- 配置 Internet 连接
- 配置子网
- 创建和配置安全组

配置 VPC

本节中的过程映射到“经典”VPC 体验中的 UI。您可以通过关闭 AWS VPC 仪表板左上角的新 VPC 体验来切换 UI 以显示经典视图。

运行 VPC 向导以创建默认的专用和公共子网，以及默认的路由和网络 ACL。

1. 在配置 VPC 之前，必须创建弹性 IP。使用所有默认值创建分配。
2. 运行 VPC 向导>“具有公用和专用子网的 VPC”
3. 接受大多数默认值。以下情况除外：
    - 输入 VPC 名称。
    - 指定弹性 IP 分配 ID。
    - 指定以下 CIDR 掩码：
        - 公共子网的 IPv4 CIDR：10.0.1.0/24，将此子网重命名为 `Public-a`。
        - 私有子网的 IPv4 CIDR：10.0.30.0/24，将此子网重命名为 `Private-a`。
    - 可用区：对于两个子网，为您所在的区域选择**“a”**选项。

    > **注意**：出于本示例的目的，我们使用 **a** 和 **b** 来区分给定 AWS 数据中心中的可用区。在 AWS 中，可用区名称可能与此处显示的示例不匹配。例如，一些可用区包括数据中心内的 **c** 和 **d** 区。

4. 单击**“创建 VPC”**。
5. 创建 VPC 后，创建 `Public-b`、`Private-b`、`Data` 和 `Bastion` 子网。若要创建子网，请单击**“子网”>“创建子网”**。

    - `Public-b`：对于“可用区”，为您所在的区域选择**“b”**选项。CIDR 块：10.0.2.0/24
    - `Private-b`：对于“可用区”，为您所在的区域选择**“b”**选项。CIDR 块：10.0.31.0/24
    - `Data`：对于“可用区”，为您所在的区域选择**“b”**区。CIDR 块：10.0.50.0/24可选：如果您计划跨 PostgreSQL 群集复制外部数据库，则在可用区 b 中创建一个 Data-b 子网，其 CIDR 块为 10.0.51.0/24。
    - `Bastion`：对于“可用区”，选择任一区域。CIDR 块：10.0.0.0/24
6. 创建子网后，编辑 Public 和 Bastion 子网上的路由表，以使用为关联 Internet 网关 (IGW) 配置的路由表。并编辑 Private 和 Data 子网以使用为网络地址转换器 (NAT)

配置的路由表。

- 若要确定哪个路由表配置了 IGW 或 NAT，请单击 AWS 仪表板中的**“路由表”**。选择两个路由表链接之一以打开属性页面。查看**“Routes”>“Destination”>“0.0.0.0/0”**处的 Target 值。Target 值区分路由的类型，并且将以 `igw-` 或 `nat-` 字符串开头。
- 若要更新路由表，请转到**“VPC”>“子网”>** [子网名称] **>“路由表”>“编辑路由表关联”**。

配置安全组

VPC 向导会创建一个您不会使用的安全组。创建以下安全组（**“安全组”>“创建安全组”**）。EC2 主机将跨两个可用区安装到这些组中，如上面的幻灯片图中所示。

- 创建一个新的安全组：**Private**。这是将安装 Tableau Server 的所有 4 个节点的位置。稍后在安装过程中，Private 安全组将与 10.0.30.0/24 和 10.0.31.0/24 子网关联。
- 创建一个新的安全组：**Public**（公共）。这是将安装代理服务器的位置。稍后在安装过程中，Public 安全组将与 10.0.1.0/24 和 10.0.2.0/24 子网关联。
- 创建一个新的安全组：**Data**。这是将安装 PostgreSQL 外部 Tableau 存储库的位置。稍后在安装过程中，Data 安全组将与 10.0.50.0/24（以及可选的 10.0.51.0/24）子网关联。
- 创建一个新的安全组：**Bastion**。这是将安装堡垒主机的位置。稍后在安装过程中，Bastion 安全组将与 10.0.0.0/24 子网关联。

指定入站和出站规则

在 AWS 中，安全组类似于本地环境中的防火墙。您必须指定允许传入和/或传出安全组的流量类型（例如 https、https 等）、协议（TCP 或 UDP）以及端口或端口范围（例如 80、443 等）。对于每个协议，您还必须指定目标流量或源流量。

公共安全组规则

入站规则			
类型	**协议**	**端口范围**	**源**
HTTP	TCP	80	0.0.0.0/0

HTTPS	TCP	443	0.0.0.0/0
SSH	TCP	22	Bastion 安全组

出站规则			
类型	**协议**	**端口范围**	**Destination**
所有流量	所有	所有	0.0.0.0/0

专用安全组规则

Private 安全组包含一个入站规则，以允许来自 Public 安全组的 HTTP 流量。仅在部署过程中允许 HTTP 流量来验证连接。我们建议您在完成反向代理部署并将 SSL 配置到 Tableau 后移除 HTTP 入站规则。

入站规则			
类型	**协议**	**端口范围**	**源**
HTTP	TCP	80	Public 安全组
HTTPS	TCP	443	Public 安全组
PostgreSQL	TCP	5432	Data 安全组
SSH	TCP	22	Bastion 安全组
所有流量	所有	所有	Private 安全组

出站规则			
类型	**协议**	**端口范围**	**Destination**
所有流量	所有	所有	0.0.0.0/0

PostgreSQL	TCP	5432	Data 安全组
SSH	TCP	22	Bastion 安全组

数据安全组规则

入站规则			
类型	**协议**	**端口范围**	**源**
PostgreSQL	TCP	5432	Private 安全组
SSH	TCP	22	Bastion 安全组

出站规则			
类型	**协议**	**端口范围**	**Destination**
所有流量	所有	所有	0.0.0.0/0
PostgreSQL	TCP	5432	Private 安全组
SSH	TCP	22	Bastion 安全组

堡垒主机安全组规则

入站规则			
类型	**协议**	**端口范围**	**源**
SSH	TCP	22	您将用于登录 AWS(管理计算机) 的计算机的 IP 地址和网络掩码。
SSH	TCP	22	Private 安全组
SSH	TCP	22	Public 安全组

出站规则			
类型	**协议**	**端口范围**	**Destination**
SSH	TCP	22	您将用于登录 AWS(管理计算机) 的计算机的 IP 地址和网络掩码。
SSH	TCP	22	Private 安全组
SSH	TCP	22	Public 安全组
SSH	TCP	22	Data 安全组
HTTPS	TCP	443	0.0.0.0/0(可选 : 如果需要上网下载堡垒主机上的配套软件 , 则创建此规则)

启用自动分配公共 IP

这为您提供了一个用于连接到代理服务器和堡垒主机的 IP 地址。

对于 Public 子网和 Bastion 子网 :

1. 选择子网
2. 在**“操作”**菜单下 , 选择“修改自动分配 IP 设置”。
3. 单击“启用自动分配公用 Ipv4 地址”。
4. 单击**“保存”**。

负载平衡器

注意 : 如果您要安装到 AWS 并按本指南中的示例部署进行操作 , 那么您应该稍后在部署过程安装和配置 AWS 负载平衡器 , 如第 5 部分 - 配置 Web 层中所述。

对于本地部署 , 请与您的网络管理员合作部署负载均衡器以支持参考架构的 Web 层 :

- 面向 Web 的应用程序负载平衡器，它接受来自 Tableau 客户端的 HTTPS 请求并与反向代理服务器进行通信。
- 反向代理：
  - 我们建议至少使用两个代理服务器以实现冗余和处理客户端负载。
  - 从负载平衡器接收 HTTPS 流量。
  - 支持 Tableau 主机的粘性会话。
  - 为运行网关进程的每个 Tableau Server 配置循环负载平衡代理。
  - 处理来自外部 IdP 的身份验证请求。
- 转发代理：Tableau Server 需要访问 Internet 以获得许可和地图功能。根据您的转发代理环境，您可能需要为 Tableau 服务 URL 配置转发代理安全列表。请参见 *“与 Internet 通信* (Linux)”。

配置主机计算机

最低推荐硬件

以下建议基于我们在参考体系结构中对真实数据的测试。

应用程序服务器：

- CPU：8 个物理内核（16 个 vCPU），
- RAM：128 GB（16 Gb/物理内核）
- 磁盘空间：100 GB

数据服务器

- CPU：8 个物理内核（16 个 vCPU），
- RAM：128 GB（16 Gb/物理内核）
- 磁盘空间：1 TB。如果您的部署将使用 Tableau 文件存储的外部存储，您将需要计算适当的磁盘空间。请参见“*使用外部文件存储安装 Tableau Server*(Linux)”。

代理服务器

- CPU：2 个物理内核（4 个 vCPU），
- RAM：8 GB（4 Gb/物理内核）
- 磁盘空间：100 GB

外部存储库数据库

- CPU：8 个物理内核（16 个 vCPU），
- RAM：128 GB（16 Gb/物理内核）
- 磁盘空间要求取决于您的数据负载及其对备份的影响。请参见“*磁盘空间要求 (Linux)*”中的“*备份和还原过程*”部分。

目录结构

参考架构建议将 Tableau Server 软件包和数据安装到非默认位置：

- 将软件包安装到：`/app/tableau_server`：在安装 Tableau Server 软件包之前创建此目录路径，然后在安装过程中指定此路径。
- 将 Tableau 数据安装到：`/data/tableau_data`。在安装 Tableau Server 之前不要创建此目录。相反，您必须在安装期间指定路径，然后 Tableau 安装程序将相应地创建该路径并授予其权限。

有关实现详细信息，请参见运行安装包并初始化 TSM。

示例：在 AWS 中安装和准备主机计算机

本部分说明如何为 Tableau Server 参考架构中的每种服务器类型安装 EC2 主机。

参考架构需要八个主机：

- Tableau Server 的四个实例。
- 代理服务器 (Apache) 的两个实例。
- 堡垒主机的一个实例。
- 一个或两个 EC2 PostgreSQL 数据库实例

主机实例详细信息

根据以下详细信息安装主机计算机。

Tableau Server

- Amazon Linux 2
- 实例类型：m5a.8xlarge
- 安全组 ID：Private

- 存储：EBS，150 GiB，gp2 卷类型。如果您的部署将使用 Tableau 文件存储的外部存储，您将需要计算适当的磁盘空间。请参见“*使用外部文件存储安装 Tableau Server*(Linux)”。
- 网络：在每个私有子网(10.0.30.0/24 和 10.0.31.0/24)中安装两个 EC2 主机。
- 将 Tableau Server 2021.2(或更高版本)rpm 软件包的最新维护版本从 Tableau 下载页面复制到每个 Tableau 主机。

堡垒主机

- Amazon Linux 2
- 实例类型：t3.micro
- 安全组 ID：Bastion
- 存储：EBS，50 GiB，gp2 卷类型
- 网络：Bastion 子网 10.0.0.0/24

Tableau Server 独立网关

- Amazon Linux 2
- 实例类型：t3.xlarge
- 安全组 ID：Public
- 存储：EBS，100 GiB，gp2 卷类型
- 网络：在每个公有子网(10.0.1.0/24 和 10.0.2.0/24)中安装一个 EC2 实例

PostgreSQL EC2 主机

- Amazon Linux 2
- 实例类型：r5.4xlarge
- 安全组 ID：Data
- 存储：磁盘空间要求取决于您的数据负载及其对备份的影响。请参见“*磁盘空间要求*(Linux)”中的“*备份和还原过程*”部分。
- 网络：Data 子网 10.0.50.0/24。(如果您在 HA 群集中复制 PostgreSQL，则在 10.0.51.0/24 子网中安装第二台主机)

验证：VPC 连接

安装主机计算机后，请验证网络配置。通过使用 SSH 从 Bastion 安全组中的主机连接到每个子网中的主机，验证主机之间的连接。

示例：连接到 AWS 中的堡垒主机

1. 为 ssh-agent 设置管理计算机。这使您可以连接到 AWS 中的主机，而无需将私钥文件放置在任何 EC2 实例上。

   若要在 Mac 上配置 ssh-agent，请运行以下命令：

   `ssh-add -K myPrivateKey.pem`，或者对于最新的 Mac Os，请运行 `ssh-add --apple-use-keychain myPrivateKey.pem`

   对于 Windows，请参见主题安全地连接到在专用 Amazon VPC 中运行的 Linux 实例。

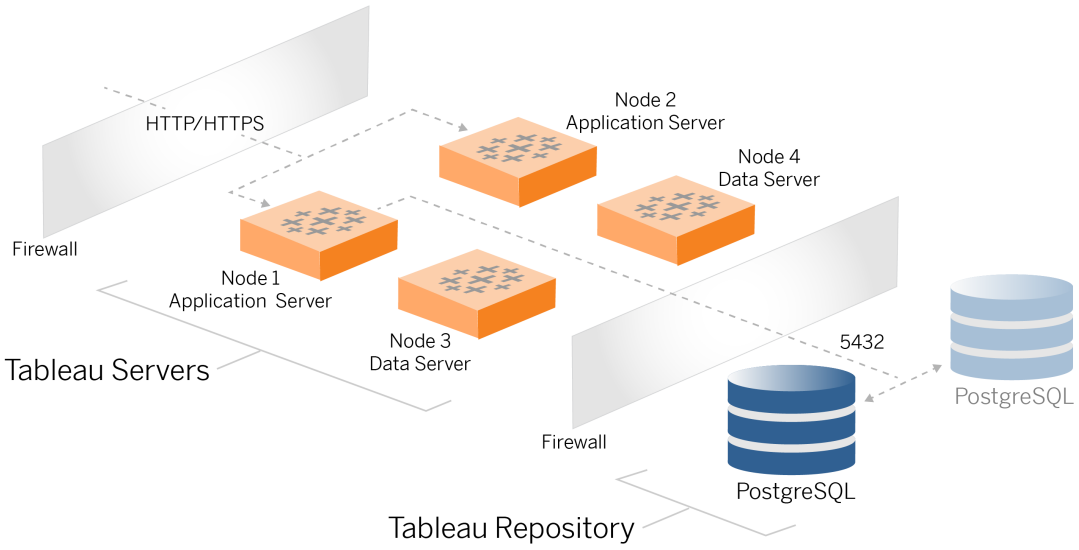
2. 通过运行以下命令连接到堡垒主机：

   ```
   ssh -A ec2-user@<public-IP>
   ```

3. 然后，您可以使用专用 IP 地址从堡垒主机连接到 VPC 中的其他主机，例如：

   ```
   ssh -A ec2-user@10.0.1.93
   ```

第 4 部分 - 安装并配置 Tableau Server



本主题描述如何完成安装和配置基准 Tableau Server 部署。此处的过程继续使用 AWS 和 Linux 参考架构示例。

整个安装过程中的 Linux 示例显示了类似 RHEL 的发行版的命令。具体来说，这里的命令是使用 Amazon Linux 2 发行版开发的。如果您运行的是 Ubuntu 发行版，请相应地编辑命令。

开始之前

您必须按照第 3 部分 - 准备 Tableau Server 企业部署中所述准备和验证环境。

安装、配置 PostgreSQL 和创建 PostgreSQL 的 tar 备份

此 PostgreSQL 实例托管 Tableau Server 部署的外部存储库。在安装 Tableau 之前，您必须安装和配置 PostgreSQL。

您可以在 Amazon RDS 或 EC2 实例上运行 PostgreSQL。有关在 RDS 上运行存储库与在 EC2 实例上运行存储库之间的差异的详细信息，请参见*"Tableau Server 外部存储库"*(Linux)。

例如，以下过程显示了如何在 Amazon EC2 实例上安装和配置 Postgres。此处显示的示例是参考架构中 PostgreSQL 的通用安装和配置。您的 DBA 应该根据您的数据规模和性能需求优化您的 PostgreSQL 部署。

要求：请注意，您必须运行 PostgreSQL 1.6，并且必须安装 uuid-ossp 模块。

PostgreSQL 版本控制

您必须为 Tableau Server 外部存储库安装兼容的 PostgreSQL 主要版本。此外，次要版本也必须满足最低要求。

Tableau Server 版本	**PostgreSQL 最低兼容版本**
2021.2.3 - 2021.2.8 2021.3.0 - 2021.3.7 2021.4.0 - 2021.4.3	12.6
2021.2.10 - 2021.2.14 2021.3.8 - 2021.3.13 2021.4.4 - 2021.4.8	12.8
2021.2.15 - 2021.2.16	12.10

2021.3.14 - 2021.3.15 2021.4.9 - 2021.4.10	
2021.2.17 - 2021.2.18 2021.3.16 - 2021.3.17 2021.4.11 - 2021.4.12	12.11
2021.3.26 2021.4.23	12.15
2022.1.0	13.3
2022.1.1 - 2022.1.3	13.4
2022.1.4 - 2022.1.6	13.6
2022.1.7 - 2022.1.16 2022.3.0 - 2022.3.7 2023.1.0 - 2023.1.4	13.7
2022.1.17 - 2022.1.19 2022.3.8 - 2022.3.19 2023.1.5 - 2023.1.15 2023.3.0 - 2023.3.8	13.11
2022.3.20 - 2022.3.x 2023.1.16 - 2023.1.x 2023.3.9 - 2023.3.x	13.14
2024.0 - 2024.x	15.6

安装 PostgreSQL

此示例安装过程描述了如何安装 PostgreSQL 版本 13.6。

登录到您在上一部分中创建的 EC2 主机。

1. 运行更新以将最新修复应用到 Linux 操作系统：

   ```
   sudo yum update
   ```

2. 在 /etc/yum.repos.d/ 路径中创建和编辑 pgdg.repo 文件。使用以下配置信息填充文件：

   ```
   [pgdg13]
   name=PostgreSQL 13 for RHEL/CentOS 7 - x86_64

   baseurl=https://download.postgresql.org/pub/repos/yum/13/redhat-
   /rhel-7-x86_64
   enabled=1
   gpgcheck=0
   ```

3. 安装 Posgres 13.6：

   ```
   sudo yum install postgresql13-server-13.6-1PGDG.rhel7.x86_64
   ```

4. 安装 uuid-ossp 模块：

   ```
   sudo yum install postgresql13-contrib-13.6-1PGDG.rhel7.x86_64
   ```

5. 初始化 Postgres：

   ```
   sudo /usr/pgsql-13/bin/postgresql-13-setup initdb
   ```

配置 Postgres

通过配置 Postgres 完成基本安装：

1. 使用以下两个条目更新 pg_hba 配置文件 `/var/lib/pgsql/13/data/pg_hba.conf`。每个条目都必须包含将在其中运行 Tableau Server 的子网掩码：

   ```
   host all all 10.0.30.0/24 password

   host all all 10.0.31.0/24 password
   ```

2. 通过以下这一行来更新 PostgreSQL 文件 `/var/lib/pgsql/13/data/postgresql.conf`：

   ```
   listen_addresses = '*'
   ```

3. 配置在重启时启动 Postgres：

   ```
   sudo systemctl enable --now postgresql-13
   ```

4. 设置超级用户密码：

   ```
   sudo su - postgres

   psql -c "alter user postgres with password 'StrongPassword'"
   ```

   > **注意**：设置一个强密码。不要使用 `'StrongPassword'`，如此处的示例所示。

   ```
   exit
   ```

5. 重新启动 Postgres：

   ```
   sudo systemctl restart postgresql-13
   ```

进行 PostgreSQL 步骤 1 tar 备份

创建 PostgreSQL 配置的 tar 备份。如果您在继续部署时遇到故障，创建当前配置的 tar 快照将节省您的时间。

我们将此称为“步骤 1”备份。

在 PostgreSQL 主机上：

1. 停止 Postgres 数据库实例：

   ```
   sudo systemctl stop postgresql-13
   ```

2. 运行以下命令以创建 tar 备份：

   ```
   sudo su
   cd /var/lib/pgsql
   tar -cvf step1.13.bkp.tar 13
   exit
   ```

3. 启动 Postgres 数据库：

   ```
   sudo systemctl start postgresql-13
   ```

还原步骤 1

如果 Tableau Server 初始节点在安装过程中出现故障，则还原到“步骤 1”。

1. 在运行 Tableau 的计算机上，运行 obliterate 脚本以从主机中完全移除 Tableau Server：

   ```
   sudo /app/tableau_server/packages/scripts.<version_
   code>/./tableau-server-obliterate -a -y -y -y -l
   ```

2. 还原 PostgreSQL 第 1 阶段的 tar。在运行 Postgres 的计算机上，运行以下命令：

   ```
   sudo su
   systemctl stop postgresql-13
   cd /var/lib/pgsql
   tar  -xvf step1.13.bkp.tar
   ```

```
systemctl start postgresql-13

exit
```

继续安装 Tableau Server 的初始节点的安装过程。

安装之前

如果您根据本指南中描述的示例 AWS/Linux 实施部署 Tableau，那么您可能能够运行自动安装脚本 TabDeploy4EDG。TabDeploy4EDG 会自动执行以下过程中描述的四节点 Tableau 部署的示例安装。请参见附录 - AWS 部署工具箱。

安装 Tableau Server 的初始节点

此过程描述了如何按照参考架构的定义安装 Tableau Server 的初始节点。除了包安装和 TSM 的初始化之外，这里的过程尽可能使用 TSM 命令行。除了与平台无关之外，使用 TSM CLI 还可以更无缝地安装到虚拟化和无头环境中。

运行安装包并初始化 TSM

登录到节点 1 主机服务器。

1. 运行更新以将最新修复应用到 Linux 操作系统：

   ```
   sudo yum update
   ```

2. 将安装包从 Tableau 下载页面复制到将运行 Tableau Server 的主机计算机。

   例如，在运行类似 Linux RHEL 的操作系统的计算机上，运行：

   ```
   wget
   https://downloads.tableau.com/esdalt/2022<version>/tableau-
   server-<version>.rpm
   ```

   其中 <version> 是版本号。

3. 下载并安装依赖项：

```
sudo yum deplist tableau-server-<version>.rpm | awk
'/provider:/ {print $2}' | sort -u | xargs sudo yum -y install
```

4. 在根目录中创建 /app/tableau_server 路径：

   ```
   sudo mkdir -p /app/tableau_server
   ```

5. 运行安装程序并指定 /app/tableau_server 安装路径。例如，在类似 Linux RHEL 的操作系统上，运行：

   ```
   sudo rpm -i --prefix /app/tableau_server tableau-server-
   <version>.x86_64.rpm
   ```

6. 切换到 /app/tableau_server/packages/scripts.<version_code>/ 目录，并运行位于该处的 initialize-tsm 脚本：

   ```
   sudo ./initialize-tsm -d /data/tableau_data --accepteula
   ```

7. 初始化完成后，退出 shell：

   ```
   exit
   ```

激活并注册 Tableau Server

1. 登录到节点 1 主机服务器。

2. 在此步骤中提供 Tableau Server 产品密钥。对您购买的每个许可证密钥运行以下命令：

   ```
   tsm licenses activate -k <product key>
   ```

3. 使用如下所示的格式创建一个 json 注册文件：

   ```
   {
   "zip" : "97403",
   "country" : "USA",
   "city" : "Springfield",
   ```

```
"last_name" : "Simpson",
"industry" : "Energy",
"eula" : "yes",
"title" : "Safety Inspection Engineer",
"company_employees" : "100",
"phone" : "5558675309",
"company" : "Example",
"state" : "OR",
"opt_in" : "true",
"department" : "Engineering",
"first_name" : "Homer",
"email" : "homer@example.com"
}
```

4. 保存对文件进行的更改之后，使用 `--file` 选项传递该文件以注册 Tableau Server：

   ```
   tsm register --file path_to_registration_file.json
   ```

配置身份存储

注意：如果您的部署将使用 Tableau 文件存储的外部存储，则您需要在配置身份存储之前启用外部文件存储。请参见“*使用外部文件存储安装 Tableau Server*(Linux)”。

默认参考架构使用本地身份存储。通过使用 `tsm settings import` 命令传递 `config.json` 文件，配置包含本地身份存储的初始主机。

根据您的操作系统导入 `config.json` 文件：

`config.json`文件包含在 scripts.<版本> 目录路径(例如，`scripts.20204.21.0217.1203`)中，并经过格式设置以配置身份存储。

运行以下命令以导入 `config.json` 文件：

```
tsm settings import -f /app/tableau_
server/packages/scripts.<version_code>/config.json
```

配置外部 Postgres

1. 创建具有以下配置设置的外部数据库 json 文件：

   ```
   {
   "flavor":"generic",
   "masterUsername":"postgres",
   "host":"<instance ip address>",
   "port":5432
   }
   ```

2. 保存对文件所做的更改后，使用以下命令传递文件：

   ```
   tsm topology external-services repository enable -f
   <filename>.json --no-ssl
   ```

   将会提示您输入 Postgres 主用户名密码。

   选项 `--no-ssl`将 Tableau 配置为仅在针对 SSL/TLS 配置了 Postgres 服务器时使用 SSL/TLS。如果没有为 SSL/TLS 配置 Postgres，则连接不会加密。第 6 部分 - 安装后配置描述了在完成第一阶段部署后，如何为 Postgres 连接启用 SSL/TLS。

3. 应用更改。

   运行此命令以应用更改并重新启动 Tableau Server：

   ```
   tsm pending-changes apply
   ```

4. 删除您在步骤 1 中使用的配置文件。

完成节点 1 安装

1. 安装 Tableau Server 后，您必须初始化服务器。

   运行以下命令：

```
tsm initialize --start-server --request-timeout 1800
```

2. 初始化完成后，您必须创建 Tableau Server 管理员帐户。

   与用于安装和管理 TSM 操作系统组件的计算机帐户不同，Tableau Server 管理员帐户是用于创建 Tableau Server 用户、项目和站点的应用程序帐户。Tableau Server 管理员还将权限应用于 Tableau 资源。运行以下命令以创建初始管理员帐户。在下面的例子中，用户名为 `tableau-admin`：

   ```
   tabcmd initialuser --server http://localhost --
   username "tableau-admin"
   ```

   Tabcmd 将提示您为此用户设置密码。

验证：节点 1 配置

1. 运行以下命令以验证 TSM 服务是否正在运行：

   ```
   tsm status -v
   ```

   Tableau 应返回以下内容：

   ```
   external:
   Status: RUNNING
   'Tableau Server Repository 0' is running (Active Repository).
   node1: localhost
   Status: RUNNING
   'Tableau Server Gateway 0' is running.
   'Tableau Server Application Server 0' is running.
   'Tableau Server Interactive Microservice Container 0' is
   running.
   'MessageBus Microservice 0' is running.
   'Relationship Query Microservice 0' is running.
   'Tableau Server VizQL Server 0' is running.
   ...
   ```

   将列出所有服务。

2. 运行以下命令以验证 Tableau 管理站点是否正在运行：

```
curl localhost
```

前几行应该显示 Vizportal html，类似于此：

```
<!DOCTYPE html>
<html xmlns:ng="" xmlns:tb="">
<head ng-csp>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="initial-scale=1, maximum-
scale=2, width=device-width, height=device-height, viewport-
fit=cover">
<meta name="format-detection" content="telephone=no">
<meta name="vizportal-config ...
```

执行步骤 2 tar 备份

验证初始安装后，进行两个 tar 备份：

- PostgreSQL
- Tableau 初始节点（节点 1）

在大多数情况下，您可以通过还原这些 tar 文件来恢复初始节点的安装。还原 tar 文件比重新安装和重新初始化初始节点要快得多。

创建步骤 2 tar 文件

1. 在 Tableau 的初始节点上，停止 Tableau：

   ```
   tsm stop
   ```

   等待 Tableau 停止，然后再继续下一步。

2. 在 PostgreSQL 主机上，停止 Postgres 数据库实例：

```
sudo systemctl stop postgresql-13
```

3. 运行以下命令以创建 tar 备份：

   ```
   sudo su

   cd /var/lib/pgsql

   tar -cvf step2.13.bkp.tar 13

   exit
   ```

4. 验证 Postgres tar 文件是否是使用 root 权限创建的：

   ```
   sudo ls -al /var/lib/pgsql
   ```

5. 在 Tableau 主机上，停止 Tableau 管理服务：

   ```
   sudo /app/tableau_server/packages/scripts.<version_
   code>/./stop-administrative-services
   ```

6. 运行以下命令以创建 tar 备份：

   ```
   cd /data

   sudo tar -cvf step2.tableau_data.bkp.tar tableau_data
   ```

7. 在 Postgres 主机上，启动 Postgres 数据库：

   ```
   sudo systemctl start postgresql-13
   ```

8. 启动 Tableau 管理服务：

   ```
   sudo /app/tableau_server/packages/scripts.<version_
   code>/./start-administrative-services
   ```

9. 运行 `tsm status` 命令在重新启动之前监视 TSM 状态。

   在大多数情况下，该命令首先将返回 DEGRADED 或 ERROR 状态。稍等几分钟，然后再次运行该命令。如果返回 ERROR 或 DEGRADED 状态，请继续等待。在返回 STOPPED 状态之前不要尝试启动 TSM。然后运行以下命令：

```
tsm start
```

还原步骤 2

此过程会将 Tableau 节点 1 和 Postgres 实例还原到步骤 2。还原到此步骤后，您可以重新部署其余 Tableau 节点。

1. 在初始 Tableau 主机（节点 1）上停止 tsm 服务：

   ```
   tsm stop
   ```

2. 在 Tableau Server 部署的所有节点上停止 Tableau 管理服务。在每个节点上依次运行以下命令（节点 1、节点 2，然后是节点 3）：

   ```
   sudo /app/tableau_server/packages/scripts.<version_
   code>/./stop-administrative-services
   ```

3. 在 Tableau 服务停止后，还原 PostgreSQL 步骤 2 tar。在运行 Postgres 的计算机上，运行以下命令：

   - ```
     sudo su

     systemctl stop postgresql-13

     cd /var/lib/pgsql

     tar  -xvf step2.13.bkp.tar

     systemctl start postgresql-13

     exit
     ```

4. 还原 Tableau 步骤 2 tar。在初始 Tableau 主机上，运行以下命令：

   ```
   cd /data

   sudo rm -rf tableau_data

   sudo tar  -xvf step2.tableau_data.bkp.tar
   ```
   ```

5. 在 Tableau 节点 1 计算机上，移除以下文件：

   - `sudo rm /data/tableau_data/data/tabsvc/appzookeeper/0/version-2/currentEpoch`
   - `sudo rm /data/tableau_data/data/tabsvc/appzookeeper/0/version-2/acceptedEpoch`
   - `sudo rm /data/tableau_data/data/tabsvc/tabadminagent/0/servicestate.json`

6. 启动 Tableau 管理服务：

   ```
   sudo /app/tableau_server/packages/scripts.<version_code>/./start-administrative-services
   ```

7. 重新加载 Tableau systmctl 文件，然后再次运行 `start-administrative-services`：

   ```
   sudo su -l tableau -c "systemctl --user daemon-reload"

   sudo /app/tableau_server/packages/scripts.<version_code>/./start-administrative-services
   ```

8. 在节点 1 上，运行 `tsm status` 命令在重新启动之前监视 TSM 状态。

   在某些情况下，您会收到错误 `Cannot connect to server...`。出现此错误是因为 tabadmincontroller 服务尚未重新启动。继续定期运行 `tsm status`。如果此错误在 10 分钟后仍未消失，请再次运行 `start-administrative-services` 命令。

   刻之后，`tsm status` 命令将返回 DEGRADED 状态，然后返回 ERROR 状态。在返回 STOPPED 状态之前不要启动 TSM。然后运行以下命令：

   ```
   tsm start
   ```

在其余节点上恢复安装 Tableau Server 的安装过程

在其余节点上安装 Tableau Server

若要继续部署，请将 Tableau 安装程序复制到每个节点。

节点配置概述

本部分介绍配置节点 2-4 的过程。以下各部分提供了每个步骤的详细配置和验证过程。

Tableau Server 节点 2-4 的安装要求您在节点安装期间生成、复制和引用引导程序文件。

若要生成引导程序文件，您将在初始节点上运行 TSM 命令。然后，您会将引导程序文件复制到目标节点，并在其中运行它作为节点初始化的一部分。

以下 json 内容显示引导程序文件的示例。(证书和与密码相关值已被截断，以使示例文件更易于阅读。)

```
{
"initialBootstrapSettings" : {
       "certificate" : "-----BEGIN CERTIFICATE-----\r\...=\r\n-----END
CERTIFICATE-----",
       "port" : 8850,
       "configurationName" : "tabsvc",
       "clusterId" : "tabsvc-clusterid",
       "cryptoKeyStore" : "zs7OzgAAAAIAAAABAAAAA...w==",
       "toksCryptoKeystore" : "LS0tLS1CRUdJTiBUT00tLS0tCjM5MDBh...L",
       "sessionCookieMaxAge" : 7200,
       "nodeId" : "node1",
       "machineAddress" : "ip-10-0-1-93.us-west-1.compute.internal",
       "cryptoEnabled" : true,
       "sessionCookieUser" : "tsm-bootstrap-user",
       "sessionCookieValue" :
"eyJjdHkiOiJKV1QiLCJlbmMiOiJBMTI4Q0JDLUhQ...",
       "sessionCookieName" : "AUTH_COOKIE"
       }
}
```

引导程序文件包括基于连接的验证，以对节点 1 进行身份验证，并为引导程序进程创建加密的通道。引导程序会话是有时间限制的，并且配置和验证节点非常耗时。配置节点时，请计划创建和复制新的引导程序。

运行引导程序文件后，登录到初始 Tableau Server 节点并为新节点配置进程。完成节点配置后，必须应用更改并重新启动初始节点。新节点已配置并启动。添加节点时，部署的配置和重新启动将连续花费更长的时间才能完成。

整个安装过程中的 Linux 示例显示了类似 RHEL 的发行版的命令。如果您运行的是 Ubuntu 发行版，请相应地编辑命令。

1. 运行更新以将最新修复应用到 Linux 操作系统：

   ```
   sudo yum update
   ```

2. 下载并安装依赖项：

   ```
   sudo yum deplist tableau-server-<version>.rpm | awk
   '/provider:/ {print $2}' | sort -u | xargs sudo yum -y install
   ```

3. 在根目录中创建 /app/tableau_server 路径：

   ```
   sudo mkdir -p /app/tableau_server
   ```

4. 运行安装程序并指定 /app/tableau_server 安装路径。例如，在类似 Linux RHEL 的操作系统上，运行：

   ```
   sudo rpm -i --prefix /app/tableau_server tableau-server-
   <version>.x86_64.rpm
   ```

生成、复制并使用引导程序文件以初始化 TSM

以下程序显示了如何在另一个节点上初始化 TSM 时生成、复制和使用引导程序文件。在此示例中，引导程序文件名为 boot.json。

在此示例中，主机计算机在 AWS 中运行，其中 EC2 主机运行 Amazon Linux 2。

1. 连接到初始节点（节点 1），并运行以下命令：

   ```
   tsm topology nodes get-bootstrap-file --file boot.json
   ```

2. 将引导程序文件复制到节点 2。

   ```
   scp boot.json ec2-user@10.0.31.83:/home/ec2-user/
   ```

3. 连接到节点 2 并切换到 Tableau Server 脚本目录：

   ```
   cd /app/tableau_server/packages/scripts.<version_number>
   ```

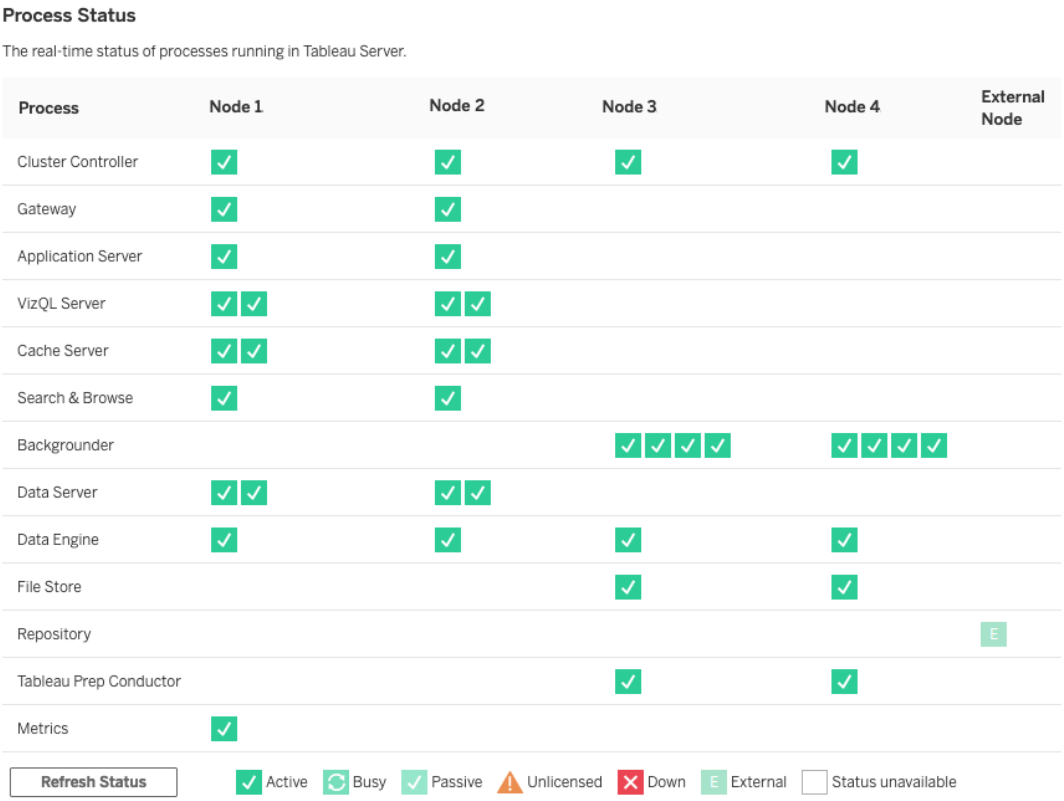
4. 运行 `initialize-tsm` 命令并引用引导程序文件：

   ```
   sudo ./initialize-tsm -d /data/tableau_data -b /home/ec2-
   user/boot.json --accepteula
   ```

5. `initialize-tsm` 完成后，删除 `boot.json`，然后退出或注销会话。

配置进程

您必须在运行 Tableau Server 管理控制器（TSM 控制器）的节点上配置 Tableau Server 群集。TSM 控制器在初始节点上运行。



配置节点 2

1. 在节点 2 上使用引导程序文件初始化 TSM 后，登录到初始节点。

2. 在初始节点 (node1) 上，运行以下命令以在节点 2 上配置进程：

```
tsm topology set-process -n node2 -pr clustercontroller -c 1
tsm topology set-process -n node2 -pr gateway -c 1
tsm topology set-process -n node2 -pr vizportal -c 1
tsm topology set-process -n node2 -pr vizqlserver -c 2
tsm topology set-process -n node2 -pr cacheserver -c 2
tsm topology set-process -n node2 -pr searchserver -c 1
tsm topology set-process -n node2 -pr dataserver -c 2
tsm topology set-process -n node2 -pr clientfileservice -c 1
tsm topology set-process -n node2 -pr tdsservice -c 1
```

```
tsm topology set-process -n node2 -pr collections -c 1
tsm topology set-process -n node2 -pr contentexploration -c 1
```

如果您安装 2022.1 或更高版本，请同时添加索引和搜索服务：

```
tsm topology set-process -n node2 -pr indexandsearchserver -c 1
```

如果您安装版本 2023.3 或更高版本，请仅包括索引和搜索服务。不要添加搜索和浏览 (searchserver) 服务

3. 在应用之前查看配置。运行以下命令：

   ```
   tsm pending-changes list
   ```

4. 在您确认您的更改在待处理列表中后(待处理列表中还有其他服务)，应用更改：

   ```
   tsm pending-changes apply
   ```

   更改将需要重新启动。配置和重新启动将花费一些时间。

5. 验证节点 2 配置。运行以下命令：

   ```
   tsm status -v
   ```

配置节点 3

在节点 3 上使用引导程序进程初始 TSM，然后运行下面的 `tsm topology set-process` 命令。

每当您设置进程时，都会显示一个协调服务警告。您可以在设置过程时忽略此警告。

1. 在节点 3 上使用引导引导程序文件初始 TSM 后，登录到初始节点 (`node1`)，并运行以下命令来配置进程：

   ```
   tsm topology set-process -n node3 -pr clustercontroller -c 1
   tsm topology set-process -n node3 -pr clientfileservice -c 1
   tsm topology set-process -n node3 -pr backgrounder -c 4
   tsm topology set-process -n node3 -pr filestore -c 1
   ```

   如果您安装 2022.1 或更高版本，请同时添加索引和搜索服务：

```
tsm topology set-process -n node3 -pr indexandsearchserver -c 1
```

2. 在应用之前查看配置。运行以下命令：

   ```
   tsm pending-changes list
   ```

3. 在您确认您的更改在待处理列表中后(该列表将包括其他自动配置的服务)，应用更改：

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

   更改将需要重新启动。配置和重新启动将花费一些时间。

4. 通过运行以下命令来验证配置：

   ```
   tsm status -v
   ```

将协调服务整体部署到节点 1-3

对于标准参考架构四节点部署，请运行以下过程：

1. 在节点 1 上运行以下命令：

   ```
   tsm stop
   tsm topology deploy-coordination-service -n node1,node2,node3
   ```

   该过程包括重新启动 TSM，这将需要一些时间。

2. 协调服务部署完成后，启动 TSM：

   ```
   tsm start
   ```

执行步骤 3 tar 备份

验证安装后，进行四个 tar 备份：

- PostgreSQL
- Tableau 初始节点(节点 1)

- Tableau 节点 2
- Tableau 节点 3

创建步骤 3 tar 文件

1. 在 Tableau 的初始节点上，停止 Tableau：

   ```
   tsm stop
   ```

2. TSM 停止后，在每个节点上停止 Tableau 管理服务。在每个节点上依次运行以下命令（节点 1、节点 2，然后是节点 3）：

   ```
   sudo /app/tableau_server/packages/scripts.<version_
   code>/./stop-administrative-services
   ```

3. 在 PostgreSQL 主机上，停止 Postgres 数据库实例：

   ```
   sudo systemctl stop postgresql-12
   ```

4. 运行以下命令以创建 tar 备份：

   ```
   sudo su
   cd /var/lib/pgsql
   tar -cvf step3.12.bkp.tar 12
   exit
   ```

5. 验证 Postgres tar 文件是否是使用 root 权限创建的：

   ```
   sudo ls -al /var/lib/pgsql
   ```

6. 在 Postgres 主机上，启动 Postgres 数据库：

   ```
   sudo systemctl start postgresql-12
   ```

7. 在节点 1、节点 2 和节点 3 上创建 tar 备份。在每个节点上运行以下命令：

- `cd /data`

  `sudo tar -cvf step3.tableau_data.bkp.tar tableau_data`

- 验证 Tableau tar 文件是否是使用 root 权限创建的：

  `ls -al`

8. 按顺序在每个节点上启动 Tableau 管理服务（节点 1、节点 2，然后是节点 3）：

   ```
   sudo /app/tableau_server/packages/scripts.<version_
   code>/./start-administrative-services
   ```

9. 运行 `tsm status` 命令在重新启动之前监视 TSM 状态。

   在大多数情况下，该命令将返回 DEGRADED 状态，然后返回 ERROR 状态。稍等片刻，然后再次运行该命令。如果返回 ERROR 或 DEGRADED 状态，请继续等待。在返回 STOPPED 状态之前不要尝试启动 TSM。然后运行以下命令：

   ```
   tsm start
   ```

还原步骤 3

此过程将还原 Tableau 节点 1、节点 2 和节点 3。它还会将 Postgres 实例还原到步骤 3。还原到此步骤后，您可以随后部署协调服务、节点 4，然后部署最终节点配置。

1. 在初始 Tableau 主机（节点 1）上停止 tsm 服务：

   ```
   tsm stop
   ```

2. TSM 停止后，停止节点 1、节点 2 和节点 3 上的 Tableau 管理服务。在每个节点上运行以下命令：

   ```
   sudo /app/tableau_server/packages/scripts.<version_
   code>/./stop-administrative-services
   ```

3. 还原 PostgreSQL 步骤 3 tar。在运行 Postgres 的计算机上，运行以下命令：

```
sudo su

systemctl stop postgresql-12

cd /var/lib/pgsql

tar -xvf step3.12.bkp.tar

systemctl start postgresql-12

exit
```

4. 在节点 1、节点 2 和节点 3 上还原 Tableau 步骤 3 tar。在每个 Tableau 节点上运行以下命令：

```
cd /data

sudo rm -rf tableau_data

sudo tar  -xvf step3.tableau_data.bkp.tar
```

5. 在 Tableau 节点 1 计算机上，移除以下文件：

   - `sudo rm /data/tableau_data/data/tabsvc/appzookeeper/1/version-2/currentEpoch`
   - `sudo rm /data/tableau_data/data/tabsvc/appzookeeper/1/version-2/acceptedEpoch`
   - `sudo rm /data/tableau_data/data/tabsvc/tabadminagent/0/servicestate.json`

   如果 shell 返回“找不到文件”错误，您可能需要更改路径名以增加路径此部分中的数字 `<n>:.../appzookeeper/<n>/version-2/...`。

6. 重新启动节点 1、节点 2 和节点 3 上的管理服务。在每个节点上运行以下命令：

```
sudo /app/tableau_server/packages/scripts.<version_
code>/./start-administrative-services

sudo su -l tableau -c "systemctl --user daemon-reload"
```

```
sudo /app/tableau_server/packages/scripts.<version_
code>/./start-administrative-services
```

7. 在节点 1 上，运行 `tsm status` 命令在重新启动之前监视 TSM 状态。

   在某些情况下，您会收到错误 `Cannot connect to server...`。出现此错误是因为 tabadmincontroller 服务尚未重新启动。继续定期运行 `tsm status`。如果此错误在 10 分钟后仍未消失，请再次运行 `start-administrative-services` 命令。

   刻之后，`tsm status` 命令将返回 DEGRADED 状态，然后返回 ERROR 状态。返回 STOPPED 状态之前不要启动 TSM。然后运行以下命令：

   ```
   tsm start
   ```

在节点 1-3 恢复部署协调服务的安装过程

配置节点 4

节点 4 的配置过程与节点 3 相同。

设置与为节点 3 设置的进程相同的进程，运行与上图所示相同的命令集，但在命令中指定 `node4`，而不是 `node3`。

与节点 3 验证一样，通过运行 `tsm status -v` 来验证节点 4 配置。

在继续之前，请等待节点 4 上的文件存储进程完成同步。在完成之前，文件存储服务状态将返回 `is synchronizing`。当文件存储服务状态返回 `is running` 时，您可以继续。

最终进程配置和验证

进程配置的最后一步是从节点 1 移除冗余进程。

1. 连接到初始节点 (node1)。
2. 停用节点 1 上的文件存储。这将导致有关从并置控制器中移除文件存储的警告。您可以忽略该警告。运行以下命令：

   ```
   tsm topology filestore decommission -n node1
   ```

3. 停用文件存储后，运行以下命令从节点 1 中移除后台程序进程：

   ```
   tsm topology set-process -n node1 -pr backgrounder -c 0
   ```

4. 在应用之前查看配置。运行以下命令：

   ```
   tsm pending-changes list
   ```

5. 确认您的更改在待处理列表中之后，应用更改：

   ```
   tsm pending-changes apply
   ```

   更改将需要重新启动。配置和重新启动将花费一些时间。

6. 验证配置：

   ```
   tsm status -v。
   ```

   在继续之前，请等待节点 4 上的文件存储进程完成同步。在完成之前，文件存储服务状态将返回 `is synchronizing`。当文件存储服务状态返回 `is running` 时，您可以继续。

执行备份

Tableau Server 的完整恢复需要包括三个组件的备份组合：

- 存储库和文件存储数据的备份文件。这个文件由 `tsm maintenance backup` 命令生成。
- 拓扑和配置导出文件。这个文件由 `tsm settings export` 命令生成。
- 身份验证证书、密钥和密钥表文件。

有关备份和还原过程的完整描述，请参见 Tableau Server 主题*执行 Tableau Server 的完整备份和还原* (Linux)。

在部署的这个阶段，完全还原所需的所有相关文件和资产都是通过运行 `tsm maintenance backup` 和 `tsm settings export` 命令包括的。

1. 运行以下命令将配置和拓扑设置导出到名为 `ts_settings_backup.json` 的文件

   ```
   tsm settings export -f ts_settings_backup.json
   ```

2. 运行以下命令在名为 `ts_backup-<yyyy-mm-dd>.tsbak` 的文件中创建存储库和文件存储数据的备份。忽略有关文件存储不在控制器节点上的警告。

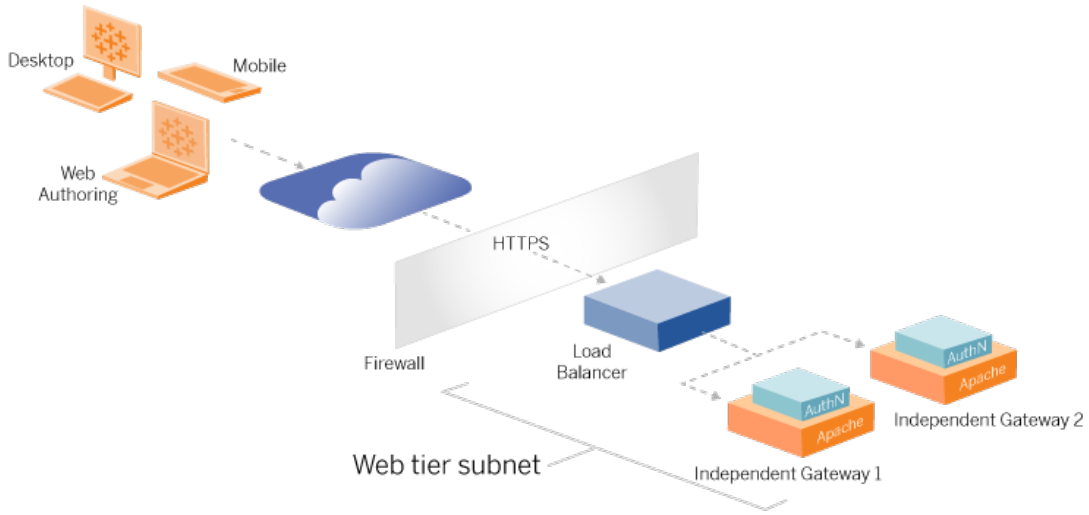
   ```
   tsm maintenance backup -f ts_backup -d --skip-compression
   ```

   备份文件的位置：

   ```
   /data/tableau_data/data/tabsvc/files/backups/
   ```

3. 复制这两个文件并将它们保存在您的 Tableau Server 部署未共享的其他存储资产上。

第 5 部分 - 配置 Web 层



参考架构的 Web 层应包括以下组件：

- 面向 Web 的应用程序负载平衡器，它接受来自 Tableau 客户端的 HTTPS 请求并与反向代理服务器进行通信。
- 反向代理：
  - 我们建议部署 Tableau Server 独立网关。
  - 我们建议至少使用两个代理服务器以实现冗余和处理客户端负载。
  - 从负载平衡器接收 HTTPS 流量。
  - 支持 Tableau 主机的粘性会话。
  - 为运行网关进程的每个 Tableau Server 配置循环负载平衡代理。
  - 处理来自外部 IdP 的身份验证请求。
- 转发代理：Tableau Server 需要访问 Internet 以获得许可和地图功能。您必须为 Tableau 服务 URL 配置转发代理安全列表。请参见“*与 Internet 通信* (Linux)”。
- 所有与客户端相关的流量都可以通过 HTTPS 加密：
  - 客户端到应用程序负载平衡器
  - 应用程序负载平衡器到反向代理服务器
  - 代理服务器到 Tableau Server
  - 反向代理上运行的身份验证处理程序到 IdP
  - Tableau Server 到 IdP

Tableau Server 独立网关

Tableau Server 版本 2022.1 引入了 Tableau Server 独立网关。独立网关是 Tableau 网关进程的独立实例，用作 Tableau 感知反向代理。

独立网关支持对后端 Tableau Server 的简单循环负载平衡。然而，独立网关并不打算作为企业应用负载平衡器。我们建议在企业级应用程序负载平衡器后运行独立网关。

独立网关需要 Advanced Management 许可证。

身份验证和授权

默认参考架构指定安装 Tableau Server 并配置本地身份验证。在此模型中，客户端必须连接到 Tableau Server 才能通过本机 Tableau Server 本地身份验证过程进行身份验证。我们不建议在参考架构中使用此身份验证方法，因为该场景要求未经身份验证的客户端与应用程序层通信，这存在安全风险。

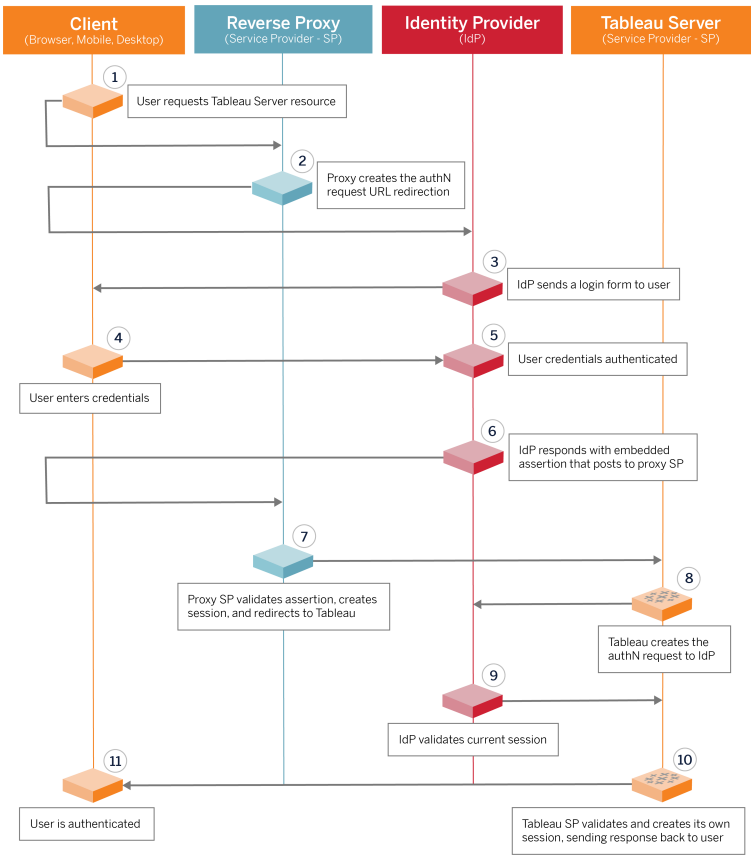
相反，我们建议配置一个企业级外部身份提供程序和一个 AuthN 模块对所有进入应用程序层的流量进行预身份验证。使用外部 IdP 配置时，不使用本机 Tableau Server 本地身份验证过程。在 IdP 对用户进行身份验证后，Tableau Server 会授权访问部署中的资源。

使用 AuthN 模块进行预身份验证

在本指南中记录的示例中，配置了 SAML SSO，但可以使用大多数外部身份提供程序和 AuthN 模块配置预身份验证过程。

在参考架构中，反向代理配置为在将这些请求通过代理传送到 Tableau Server 之前创建与 IdP 的客户端身份验证会话。我们将此过程称为*预身份验证*阶段。反向代理只会将经过身份验证的客户端会话重定向到 Tableau Server。然后，Tableau Server 将创建一个会话，使用 IdP 验证会话的身份验证，然后返回客户端请求。

下图显示了配置了 AuthN 模块的预身份验证和身份验证过程的分步详细信息。反向代理可能是通用的第三方解决方案或 Tableau Server 独立网关：



配置概览

这是配置 Web 层的过程的概述。在每个步骤之后验证连接：

1. 配置两个反向代理以提供对 Tableau Server 的 HTTP 访问。
2. 在代理服务器上使用粘性会话配置负载平衡逻辑，以连接到运行网关进程的每个 Tableau Server 实例。
3. 在 Internet 网关处使用粘性会话配置应用程序负载平衡，以将请求转发到反向代理服务器。
4. 使用外部 IdP 配置身份验证。您可以通过在反向代理服务器上安装身份验证处理程序来配置 SSO 或 SAML。AuthN 模块管理外部 IdP 和您的 Tableau 部署之间的身份验证握手。Tableau 还将充当 IdP 服务提供商并使用 IdP 对用户进行身份验证。

5. 若要在此部署中使用 Tableau Desktop 进行身份验证，您的客户端必须运行 Tableau Desktop 2021.2.1 或更高版本。

使用 Tableau Server 独立网关的示例 Web 层配置

本主题的其余部分提供了一个端到端过程，该过程描述了如何使用 Tableau Server 独立网关在示例 AWS 参考架构中实现 Web 层：有关使用 Apache 作为反向代理的示例配置，请参见附录 - 具有 Apache 示例部署的 Web 层。

示例配置由以下组件组成：

- AWS 应用程序负载平衡器
- Tableau Server 独立网关
- Mellon 身份验证模块
- Okta IdP
- SAML 身份验证

注意：本部分中提供的示例 Web 层配置包括部署第三方软件和服务的详细过程。我们已尽最大努力验证和记录启用 Web 层方案的过程。但是，第三方软件可能会发生变化，或者您的方案可能与此处描述的参考架构不同。请参考第三方文档以获得权威的配置详细信息和支持。

此部分中的 Linux 示例显示了类似 RHEL 的发行版的命令。具体来说，这里的命令是使用 Amazon Linux 2 发行版开发的。如果您运行的是 Ubuntu 发行版，请相应地编辑命令。

在此示例中部署 Web 层遵循逐步配置和验证过程。核心 Web 层配置包含以下用于在 Tableau 和 Internet 之间启用 HTTP 的步骤。独立网关运行并配置为在 AWS 应用程序负载平衡器后面进行反向代理/负载平衡：

1. 准备环境
2. 安装独立网关

3. 配置独立网关服务器
4. 配置 AWS 应用程序负载平衡器

设置 Web 层并验证与 Tableau 的连接后，使用外部提供程序配置身份验证。

准备环境

在部署独立网关之前完成以下任务。

1. AWS 安全组更改。将公共安全组配置为允许来自专用安全组的入站独立网关整理流量 (TCP 21319)。
2. 如第 4 部分 - 安装并配置 Tableau Server中所述，在四节点 Tableau Server 群集上安装版本 22.1.1(或更高版本) 。
3. 如配置主机计算机中所述，在 Public 安全组中配置两个代理 EC2 实例。

安装独立网关

Tableau Server 独立网关需要 Advanced Management 许可证。

部署 Tableau Server 独立网关包括安装和运行 .rpm 包，然后配置初始状态。本指南中包含的过程为部署到参考架构中提供了说明性指导。

如果您的部署与参考架构不同，请参见核心 Tableau Server 文档“*安装带有独立网关*的 Tableau Server (Linux)”。

> **重要信息**：配置独立网关可能是一个容易出错的过程。跨独立网关服务器的两个实例对配置问题进行故障排除非常困难。因此，我们建议一次配置一个独立网关服务器。配置第一台服务器并验证功能后，您应该配置第二台独立网关服务器。

尽管您将分别配置每个独立网关服务器，但在您安装到公共安全组中的两个 EC2 实例上运行此安装过程：

1. 运行更新以将最新修复应用到 Linux 操作系统：

   ```
   sudo yum update
   ```

2. 如果安装了 Apache，请将其移除：

   ```
   sudo yum remove httpd
   ```

3. 将版本 2022.1.1（或更高版本）的独立网关安装包从 Tableau 下载页面复制到将运行 Tableau Server 的主机计算机。

   例如，在运行类似 Linux RHEL 的操作系统的计算机上，运行：

   ```
   wget
   https://downloads.tableau.com/esdalt/2022<version>/tableau-
   server-tsig-<version>.x86_64.rpm
   ```

4. 运行安装程序。例如，在类似 Linux RHEL 的操作系统上，运行：

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

5. 切换到 /opt/tableau/tableau_tsig/packages/scripts.<version_code>/ 目录，并运行位于该处的 initialize-tsig 脚本：除了 --accepteula 标志外，您还必须包括运行 Tableau Server 部署的子网的 IP 范围。使用 -c 选项指定 IP 范围。下面的示例显示了指定了示例 AWS 子网的命令：

   ```
   sudo ./initialize-tsig --accepteula -c "ip 10.0.30.0/24
   10.0.31.0/24"
   ```

6. 初始化完成后，打开 tsighk-auth.conf 文件并复制文件中的身份验证密文。作为后端 Tableau Server 配置的一部分，您需要为每个独立网关实例提交此代码：

   ```
   sudo less /var/opt/tableau/tableau_tsig/config/tsighk-auth.conf
   ```

7. 在独立网关的两个实例上运行上述步骤后，准备 tsig.json 配置文件。配置文件由一个“independentGateways”数组组成。该数组包含配置对象，每个对象定义

独立网关实例的连接详细信息。

复制以下 JSON 并根据您的部署环境对其进行自定义。此处的示例显示了示例 AWS 参考架构的文件。

下面的示例 JSON 文件仅包含一个独立网关的连接信息。在此过程的稍后部分，您将包括第二个独立网关服务器的连接信息。

为接下来的程序将文件另存为 `tsig.json`。

```
{
"independentGateways": [
 {
       "id": "ip-10-0-1-169.ec2.internal",
       "host": "ip-10-0-1-169.ec2.internal",
       "port": "21319",
       "protocol" : "http",
       "authsecret": "13660-27118-29070-25482-9518-22453"
       }]
 }
```

- `"id"` - 运行独立网关的 AWS EC2 实例的私有 DNS 名称。
- `"host"` - 与 `"id"` 相同。
- `"port"`- 整理端口，默认情况下为 `"21319"`。
- `"protocol"` - 客户端流量协议。为初始配置将此项保留为 `http`。
- `"authsecret"` - 您在上一步中复制的密文。

独立网关：直接连接与中继连接

在继续之前，您必须决定在部署中配置哪种连接方案：直接连接或中继连接。此处简要描述了每个选项以及相关的决策数据点。

中继连接：您可以将独立网关配置为通过单个端口将客户端通信中继到 Tableau Server 上的网关进程。我们将此称为*中继*连接：

- 中继进程导致从独立网关到后端 Tableau Server 网关进程的额外跃点。与直接连接配置相比，额外的跃点会降低性能。

- 中继模式支持 TLS。中继模式下的所有通信都仅限于单一协议(HTTP 或 HTTPS),因此可以使用 TLS 进行加密和身份验证。

直接连接:独立网关可以通过多个端口直接与后端 Tableau Server 进程通信。我们将这种通信称为*直接*连接:

- 由于直接连接到后端 Tableau Server,因此与中继连接选项相比,客户端性能显着提高。
- 需要打开 16 个以上从公共子网到私有子网的端口,以便直接处理从独立网关到 Tableau Server 计算机的通信。
- 从独立网关到 Tableau Server 的进程尚不支持 TLS。

配置中继连接

若要在 Tableau Server 和独立网关之间运行 TLS,您必须配置中继连接。EDG 中的示例场景配置有中继连接。

1. 将 `tsig.json` 复制到 Tableau Server 部署的节点 1。
2. 在节点 1 上运行以下命令以启用独立网关。

```
tsm stop
tsm configuration set -k gateway.tsig.proxy_tls_optional -v
none
tsm pending-changes apply
tsm topology external-services gateway enable -c tsig.json
tsm start
```

配置直接连接

由于直接连接不支持 TLS,我们建议只有当您能够通过其他方式保护所有网络流量时,才配置直接连接。若要在 Tableau Server 和独立网关之间运行 TLS,您必须配置中继连接。EDG 中的示例场景配置有中继连接。

如果您将独立网关配置为直接连接到 Tableau Server，则必须启用配置以触发通信。Tableau Server 与独立网关通信后，将建立协议目标。然后，您必须从独立网关计算机检索 `proxy_targets.csv`，并打开 AWS 中从 Public 安全组到 Private 安全组的相应端口。

1. 将 `tsig.json` 复制到 Tableau Server 部署的节点 1。
2. 在节点 1 上运行以下命令以启用独立网关。

   ```
   tsm stop
   tsm topology external-services gateway enable -c tsig.json
   tsm start
   ```

3. 在独立网关计算机上运行以下命令以查看 Tableau Server 群集正在使用的端口：

   ```
   less /var/opt/tableau/tableau_tsig/config/httpd/proxy_
   targets.csv
   ```

4. 配置 AWS 安全组。添加 `proxy_targets.csv` 中列出的 TCP 端口，以允许从 Public 安全组到 Private 安全组的通信。

   我们建议自动化端口入口配置，因为如果 Tableau Server 部署发生更改，端口可能会更改。在 Tableau Server 部署上添加节点或重新配置进程将触发对独立网关所需的端口访问的更改。

验证：基本拓扑配置

您应该能够通过浏览到 `http://<gateway-public-IP-address>` 来访问 Tableau Server 管理员页面。

如果 Tableau Server 登录页面未加载，或者如果 Tableau Server 未启动，请遵循以下故障诊断步骤：

网络：

- 通过从 Tableau Server 节点 1 运行以下 `wget` 命令，验证 Tableau 部署和独立网关实例之间的连接：wget http://<独立网关的内部 IP 地址>:21319，例如：

  ```
  wget http://ip-10-0-1-38:21319
  ```

如果连接被拒绝或失败，请验证公共安全组是否配置为允许来自私有安全组的独立网关整理流量 (TCP 21319)。

如果安全组配置正确，则验证您在独立网关初始化期间指定了正确的 IP 地址或 IP 范围。您可以在位于 `/etc/opt/tableau/tableau_tsig/environment.bash` 的 `environment.bash` 文件中查看和更改此配置。如果您对此文件进行了更改，请按如下所述重新启动 tsig-http 服务。

在代理 1 主机上：

1. 用独立的网关存根文件覆盖 httpd.conf 文件：

   ```
   cp /var/opt/tableau/tableau_tsig/config/httpd.conf.stub
   /var/opt/tableau/tableau_tsig/config/httpd.conf
   ```

2. 作为故障排除的第一步，重新启动 tsig-httpd：

   ```
   sudo su - tableau-tsig
   systemctl --user restart tsig-httpd
   exit
   ```

在 Tableau 节点 1 上

- 仔细检查 `tsig.json` 文件。如果发现错误，请修复它们，然后运行 `tsm topology external-services gateway update -c tsig.json`。
- 如果运行直接连接，请验证 `proxy_targets.csv` 中列出的 TCP 端口配置为从 Public 安全组到 Private 安全组的入口端口。

配置 AWS 应用程序负载平衡器

将负载平衡器配置为 HTTP 侦听器。此处的过程描述了如何在 AWS 中添加负载平衡器。

步骤 1：创建目标组

标组是定义运行代理服务器的 EC2 实例的 AWS 配置。这些是来自 LBS 的流量的目标。

1. “EC2”>**“目标组”>“创建目标组”**

2. 在“创建”页面上：

   - 输入目标组名称，例如 `TG-internal-HTTP`
   - 目标类型：实例
   - 协议：HTTP
   - 端口：80
   - VPC：选择您的 VPC
   - 在**“运行状况检查”>“高级运行状况检查设置”>“成功代码”**下，将代码列表追加为：`200,303`。
   - 单击**“创建”**

3. 选择刚刚创建的目标组，然后单击**“目标”**选项卡：

   - 单击**“编辑”**。
   - 选择正在运行代理应用程序的 EC2 实例（或单个实例，如果您一次配置一个），然后单击**“添加到已注册”**。
   - 单击**“保存”**。

步骤 2：启动负载平衡器向导

1. “EC2”>**“负载平衡器”>“创建负载平衡器”**

2. 在“选择负载平衡器类型”页面上，创建一个应用程序负载平衡器。

> **注意**：为配置负载均衡器而显示的 UI 在 AWS 数据中心之间不一致。下面的过程“向导配置”映射到从**“步骤 1 配置负载均衡器”**开始的 AWS 配置向导。
>
> 如果您的数据中心在页面底部包含**“创建负载均衡器”**按钮的单个页面中显示所有配置，请按照下面的“单页面配置”过程进行操作。

向导配置

1. **“配置负载平衡器”**页面：

   - 指定名称
   - 方案：面向互联网（默认）
   - IP 地址类型：ip1v4（默认）
   - 侦听器（侦听器和路由）：
     a. 保留默认的 HTTP 侦听器
     b. 单击**“添加侦听器”**并添加 `HTTPS:443`
   - VPC：选择您已安装所有内容的 VPC
   - 可用区：
     - 为您的数据中心区域选择 **a** 和 **b**
     - 在每个相应的下拉选择器中，选择 Public 子网（您的代理服务器所在的位置）。
   - 单击：**“配置安全设置”**

2. **“配置安全设置”**页面

   - 上载您的公共 SSL 证书。
   - 单击**“下一步：配置安全组”**。

3. **“配置安全组”**页面：

   - 选择 Public 安全组。如果选择了“默认”安全组，则清除该选择。
   - 单击**“下一步：配置路由”**。

4. **“配置路由”**页面

   - 目标组：现有目标组。
   - 名称：选择您之前创建的目标组
   - 单击**“下一步：注册目标”**。

5. **“注册目标”**页面

   - 应会显示您之前配置的两个代理服务器实例。
   - 单击**“下一步：审查”**。

6. **“查看”**页面

   单击**“创建”**。

单页面配置

基本配置

- 指定名称
- 方案:面向互联网(默认)
- IP 地址类型:ip1v4(默认)

网络映射

- VPC:选择您已安装所有内容的 VPC
- 映射:
  - 为您的数据中心区域选择 **a** 和 **b**(或类似的)可用区
  - 在每个相应的下拉选择器中,选择 Public 子网(您的代理服务器所在的位置)。

安全组

选择 Public 安全组。如果选择了"默认"安全组,则清除该选择。

侦听器和路由

- 保留默认的 HTTP 侦听器。对于**"默认操作"**,指定您之前设置的目标组。
- 单击**"添加侦听器"**并添加 `HTTPS:443`。对于**"默认操作"**,指定您之前设置的目标组。

安全侦听器设置

- 上载您的公共 SSL 证书。

单击**"创建负载平衡器"**。

步骤 3:启用粘性

1. 创建负载平衡器后,您必须在目标组上启用粘性。
   - 打开 AWS 目标组页面(**"EC2">"负载平衡">"目标组"**),选择您刚刚设置的目标组实例。在**"操作"**菜单上,选择**"编辑属性"**。
   - 在**"编辑属性"**页面上,选择**"粘性"**,指定持续时间 `1 day`,然后**保存更改**。

2. 在负载平衡器上，在 HTTP 侦听器上启用粘性。选择您刚刚配置的负载平衡器，然后单击**“侦听器”**选项卡：

    - 对于**“HTTP:80”**，单击**“查看/编辑规则”**。在生成的**“规则”**页面上，单击编辑图标（一次位于页面顶部，然后再次位于规则旁边）以编辑规则。删除现有 THEN 规则，并通过单击**“添加操作”>“转发至...”**替换该规则。在生成的 THEN 配置中，指定您创建的相同目标组。在“组级粘性”下，启用粘性并将持续时间设置为 1 天。保存设置，然后单击**“更新”**。

步骤 4：在负载平衡器上设置空闲超时

在负载平衡器上，将空闲超时更新为 400 秒。

选择您为此部署配置的负载平衡器，然后单击**“操作”>“编辑属性”**。将**“空闲超时”**设置为 400 秒，然后单击**“保存”**。

步骤 5：验证 LBS 连接

打开 AWS 负载平衡器页面（**“EC2”>“负载平衡器”**），选择您刚刚设置的负载平衡器实例。

在**“描述”**下，复制 DNS 名称并将其粘贴到浏览器中以访问 Tableau Server 登录页面。

如果您收到 500 级错误，那么您可能需要重新启动代理服务器。

使用公共 Tableau URL 更新 DNS

使用 AWS 负载平衡器描述中的域 DNS 区域名称在您的 DNS 中创建 CNAME 值。流向您的 URL (tableau.example.com) 的流量应发送到 AWS 公共 DNS 名称。

验证连接

DNS 更新完成后，您应该能够通过输入您的公共 URL（例如 `https://tableau.example.com`）导航到 Tableau Server 登录页面。

示例身份验证配置：带有外部 IdP 的 SAML

以下示例介绍如何使用 Okta IdP 和 Mellon 身份验证模块为在 AWS 参考架构中运行的 Tableau 部署设置和配置 SAML。

此示例取自上一节，并假设您一次配置一个独立网关。

该示例介绍了如何通过 HTTP 配置 Tableau Server 和独立网关。Okta 将通过 HTTPS 向 AWS 负载平衡器发送请求，但所有内部流量都将通过 HTTP 传输。针对此场景进行配置时，请在设置 URL 字符串时注意 HTTP 与 HTTPS 协议。

此示例使用 Mellon 作为独立网关服务器上的预身份验证服务提供程序模块。此配置可确保只有经过身份验证的流量连接到 Tableau Server，Tableau Server 还充当 Okta IdP 的服务提供商。因此，您必须配置两个 IdP 应用程序：一个用于 Mellon 服务提供商，另一个用于 Tableau 服务提供商。

创建 Tableau 管理员帐户

配置 SAML 时的一个常见错误是在启用 SSO 之前忘记在 Tableau Server 上创建管理员帐户。

第一步是在 Tableau Server 上创建一个具有服务器管理员角色的帐户。对于示例 Okta 场景，用户名必须采用有效电子邮件地址格式，例如 user@example.com。您必须为此用户设置密码，但在配置 SAML 后将不再使用该密码。

配置 Okta 预身份验证应用程序

本部分描述的端到端场景需要两个 Okta 应用程序：

- Okta 预认证应用程序
- Okta Tableau Server 应用程序

这些应用程序中的每一个都与您需要分别在反向代理和 Tableau Server 上配置的不同元数据相关联。

此过程描述了如何创建和配置 Okta 预身份验证应用程序。在本主题的后面，您将创建 Okta Tableau Server 应用程序。有关用户受限的免费测试 Okta 帐户，请参见 Okta 开发人员网页。

为 Mellon 预身份验证服务提供商创建 SAML 应用程序集成。

1. 打开 Okta 管理仪表板 >**“应用程序”**>**“创建应用程序集成”**。

2. 在**“创建新的应用程序集成”**页面上，选择**“SAML 2.0”**，然后单击**“下一步”**。

3. 在**“常规设置”**选项卡上，输入应用程序名称（例如 `Tableau Pre-Auth`），然后单击**“下一步”**。

4. 在**“配置 SAML”**选项卡上：

   - 单点登录 (SSO) URL。单点登录 URL 中路径的最后一个元素在 `mellon.conf` 配置文件称为 `MellonEndpointPath`，该配置文件将在本过程的后面进行介绍。您可以指定您想要的任何端点。在此示例中，`sso` 是端点。最后一个元素 `postResponse` 是必须的：`https://tableau.example.com/sso/postResponse`。
   - 清除复选框：**“将此用于收件人 URL 和目标 URL”**。
   - 收件人 URL：与 SSO URL 相同，但使用 HTTP。例如，`http://tableau.example.com/sso/postResponse`。
   - 目标 URL：与 SSO URL 相同，但带有 HTTP。例如，`http://tableau.example.com/sso/postResponse`。
   - 受众 URI（SP 实体 ID）。例如，`https://tableau.example.com`。
   - 名称 ID 格式：`EmailAddress`
   - 应用程序用户名：`Email`
   - 属性声明：名称 = `mail`；名称格式 = `Unspecified`；值 = `user.email`。

   单击**“下一步”**。

5. 在**“反馈”**选项卡上，选择：

   - **我是添加内部应用程序的 Okta 客户**
   - **这是我们创建的内部应用程序**
   - 单击**“完成”**。

6. 创建预授权 IdP 元数据文件：

- 在 Okta 中：**“Applications”（应用程序）>“Applications”（应用程序）>** 您的新应用程序（例如，`Tableau Pre-Auth`）**>“Sign On”（登录）**
- 在**“SAML 签名证书”**旁边，单击**“查看 SAML 设置说明”**。
- 在**“如何针对 <预授权> 应用程序配置 SAML 2.0”**页面上，向下滚动到**“可选”**部分，**向您的 SP 提供商提供以下 IDP 元数据**。
- 复制 XML 字段的内容并将它们保存在一个名为 `pre-auth_idp_metadata.xml` 的文件中。

7. （可选）配置多重身份验证：

   - 在 Okta 中：**“Applications”（应用程序）>“Applications”（应用程序）>** 您的新应用程序（例如，`Tableau Pre-Auth`）**>“Sign On”（登录）**
   - 在**“Sign On Policy”（登录策略）**下，单击**“Add Rule”（添加规则）**。
   - 在**“App Sign On Rule”（应用程序登录规则）**上，指定名称和不同的 MFA 选项。若要测试功能，您可以将所有选项保留为默认值。但是，在**“Actions”（操作）**下，您必须选择**“Prompt for factor”（因素提示）**，然后指定用户必须登录的频率。单击**“Save”（保存）**。

创建和分配 Okta 用户

1. 在 Okta 中，使用您在 Tableau 中创建的相同用户名 (user@example.com) 创建一个用户：**“Directory”（目录）>“People”（人员）>“Add person”（添加人员）**。
2. 创建用户后，将新的 Okta 应用程序分配给该人员：单击用户名，然后在**“Assign Application”（分配应用程序）**中分配应用程序。

安装 Mellon 进行预身份验证

此示例使用常用的开源模块 mod_auth_mellon。一些 Linux 发行版从较旧的存储库中打包了过时的 mod_auth_mellon 版本。这些过时的版本可能包含未知的安全漏洞或功能问题。如果您选择使用 mod_auth_mellon，请检查您使用的是当前版本。

mod_auth_mellon 模块是第三方软件。我们已尽最大努力验证和记录启用此方案的过程。但是，第三方软件可能会发生变化，或者您的方案可能与此处描述的参考架构不同。请参考第三方文档以获得权威的配置详细信息和支持。

1. 在运行独立网关的活动 EC2 实例上，安装 Mellon 身份验证模块的当前版本。

2. 创建 `/etc/mellon` 目录：

```
sudo mkdir /etc/mellon
```

将 Mellon 配置为预身份验证模块

在独立网关的第一个实例上运行此过程。

您必须有依据 Okta 配置创建的 pre-auth_idp_metadata.xml 文件的副本。

1. 更改目录：

   ```
   cd /etc/mellon
   ```

2. 创建服务提供程序元数据。运行 mellon_create_metadata.sh 脚本。您必须在命令中包含您组织的实体 ID 和返回 URL。

   返回 URL 在 Okta 中称为*单点登录 URL*。返回 URL 中路径的最后一个元素在 mellon.conf 配置文件中称为 MellonEndpointPath，该配置文件将在本过程的后面进行介绍。在此示例中，我们指定 sso 作为端点路径。

   例如：

   ```
   sudo /usr/libexec/mod_auth_mellon/mellon_create_metadata.sh
   https://tableau.example.com "https://tableau.example.com/sso"
   ```

   该脚本返回服务提供商证书、密钥和元数据文件。

3. 重命名 mellon 目录中的服务提供商文件以便于阅读。我们将在文档中使用以下名称来引用这些文件：

   ```
   sudo mv *.key mellon.key
   sudo mv *.cert mellon.cert
   sudo mv *.xml sp_metadata.xml
   ```

4. 将 pre-auth_idp_metadata.xml 文件复制到同一文件夹。

5. 更改 /etc/mellon 目录中所有文件的所有权和权限：

```
sudo chown tableau-tsig mellon.key
sudo chown tableau-tsig mellon.cert
sudo chown tableau-tsig sp_metadata.xml
sudo chown tableau-tsig pre-auth_idp_metadata.xml
sudo chmod +r * mellon.key
sudo chmod +r * mellon.cert
sudo chmod +r * sp_metadata.xml
sudo chmod +r * pre-auth_idp_metadata.xml
```

6. 创建 /etc/mellon/conf.d 目录：

   ```
   sudo mkdir /etc/mellon/conf.d
   ```

7. 在 /etc/mellon/conf.d 目录中创建 global.conf 文件。

   复制如下所示的文件内容，但使用您的根域名更新 MellonCookieDomain。举例来说，如果 Tableau 的域名是 tableau.example.com，请为根域输入 example.com。

   ```
   <Location "/">
   AuthType Mellon
   MellonEnable auth
   Require valid-user
   MellonCookieDomain <root domain>
   MellonSPPrivateKeyFile /etc/mellon/mellon.key
   MellonSPCertFile /etc/mellon/mellon.cert
   MellonSPMetadataFile /etc/mellon/sp_metadata.xml
   MellonIdPMetadataFile /etc/mellon/pre-auth_idp_metadata.xml
   MellonEndpointPath /sso
   </Location>

   <Location "/tsighk">
   MellonEnable Off
   </Location>
   ```

8. 在 /etc/mellon/conf.d 目录中创建 mellonmod.conf 文件。

该文件包含一个指令，该指令指定 mod_auth_mellon.so 文件的位置。此处示例中的位置是文件的默认位置。验证文件是否在此位置，或更改此指令中的路径以匹配 mod_auth_mellon.so 的实际位置：

```
LoadModule auth_mellon_module /usr/lib64/httpd/modules/mod_
auth_mellon.so
```

在 Okta 中创建 Tableau Server 应用程序

1. 在 Okta 仪表板中：**“Applications”（应用程序）>“Applications”（应用程序）>“Browse App Catalog”（浏览应用程序目录）**
2. 在**“Browse App Integration Catalog”（浏览应用程序集成目录）**中，搜索 Tableau，选择“Tableau Server”磁贴，然后单击**“Add”（添加）**。
3. 在**“Add Tableau Server”（添加 Tableau Server）>“General Settings”（常规设置）”**上，输入标签，然后单击**“Next”（下一步）**。
4. 在“Sign-On Options”（登录选项）中，选择**“SAML 2.0”**，然后向下滚动到“Advanced Sign-on Settings”（高级登录设置）：
   - **“SAML Entity ID”（SAML 实体 ID）**：输入公共 URL，例如 https://tableau.example.com。
   - **“Application user name format”（应用程序用户名格式）**：“Email”（电子邮件）
5. 单击**“Identity Provider metadata”（身份提供程序元数据）**链接以启动浏览器。复制浏览器链接。这是您在以下过程中配置 Tableau 时将使用的链接。
6. 单击**“Done”（完成）**。
7. 将新的 Tableau Server Okta 应用程序分配给您的用户 (user@example.com)：单击用户名，然后在**“Assign Application”（分配应用程序）**中分配应用程序。

在 Tableau Server 上设置身份验证模块配置

在 Tableau Server 节点 1 上运行以下命令。这些命令指定 Mellon 配置文件在远程独立网关计算机上的文件位置。仔细检查这些命令中指定的文件路径是否映射到远程独立网关计算机上的路径和文件位置。

```
tsm configuration set -k gateway.tsig.authn_module_block -v
"/etc/mellon/conf.d/mellonmod.conf" --force-keys
tsm configuration set -k gateway.tsig.authn_global_block -v
"/etc/mellon/conf.d/global.conf" --force-keys
```

为了减少停机时间，在启用 SAML 之前请不要应用更改，如下一部分所述。

在 Tableau Server 上为 IdP 启用 SAML

在 Tableau Server 节点 1 上运行此过程。

1. 从 Okta 下载 Tableau Server 应用程序元数据。使用您在上一过程中保存的链接：

   ```
   wget https://dev-66144217.okta.com/app/exk1egxgt1fhjkSeS5d7/sso/saml/metadata -O idp_metadata.xml
   ```

2. 将 TLS 证书和相关密钥文件复制到 Tableau Server。密钥文件必须是 RSA 密钥。有关 SAML 证书和 IdP 要求的详细信息，请参见“*SAML 要求* (Linux)”。

   为了简化证书管理和部署，以及作为安全最佳实践，我们建议使用由主要受信任的第三方证书颁发机构 (CA) 生成的证书。或者，您可以为 TLS 生成自签名证书或使用 PKI 中的证书。

   如果您没有 TLS 证书，则可以使用下面的嵌入式过程生成自签名证书。

生成自签名证书

在 Tableau Server 节点 1 上运行此过程。

a. 生成签名根证书颁发机构 (CA) 密钥：

   ```
   openssl genrsa -out rootCAKey-saml.pem 2048
   ```

b. 创建根 CA 证书：

```
openssl req -x509 -sha256 -new -nodes -key rootCAKey-
saml.pem -days 3650 -out rootCACert-saml.pem
```

系统将提示您输入证书字段的值。例如:

```
Country Name (2 letter code) [XX]:US
State or Province Name (full name) []:Washington
Locality Name (eg, city) [Default City]:Seattle
Organization Name (eg, company) [Default Company
Ltd]:Tableau
Organizational Unit Name (eg, section) []:Operations
Common Name (eg, your name or your server's hostname)
[]:tableau.example.com
Email Address []:example@tableau.com
```

c. 创建证书和相关密钥(在下面的示例中为 server-saml.csr 和 server-saml.key)。证书的使用者名称必须与 Tableau 主机的公共主机名称匹配。使用者名称是使用 -subj 选项以"/CN=<host-name>"格式设置的,例如:

```
openssl req -new -nodes -text -out server-saml.csr -keyout
server-saml.key -subj "/CN=tableau.example.com"
```

d. 使用您在上面创建的 CA 证书签署新证书。以下命令还将以 crt 格式输出证书:

```
openssl x509 -req -in server-saml.csr -days 3650 -CA
rootCACert-saml.pem -CAkey rootCAKey-saml.pem -
CAcreateserial -out server-saml.crt
```

e. 将密钥文件转换为 RSA。Tableau 需要用于 SAML 的 RSA 密钥文件。若要转换密钥,请运行以下命令:

```
openssl rsa -in server-saml.key -out server-saml-rsa.key
```

3. 配置 SAML。运行以下命令，指定您的实体 ID 和返回 URL，以及元数据文件、证书文件和密钥文件的路径：

   ```
   tsm authentication saml configure --idp-entity-id
   "https://tableau.example.com" --idp-return-url
   "https://tableau.example.com" --idp-metadata idp_metadata.xml -
   -cert-file "server-saml.crt" --key-file "server-saml-rsa.key"

   tsm authentication saml enable
   ```

4. 如果您的组织运行的是 Tableau Desktop 2021.4 或更高版本，则您必须运行以下命令以通过反向代理服务器启用身份验证。

   Tableau Desktop 2021.2.1 - 2021.3 的版本无需运行此命令也可工作，前提是您的预身份验证模块（例如 Mellon）配置为允许保留顶级域 Cookie。

   ```
   tsm configuration set -k features.ExternalBrowserOAuth -v false
   ```

5. 应用配置更改：

   ```
   tsm pending-changes apply
   ```

重新启动 tsig-httpd 服务

当您的 Tableau Server 部署应用更改时，重新登录到 Tableau Server 独立网关计算机并运行以下命令以重新启动 tsig-httpd 服务：

```
sudo su - tableau-tsig
systemctl --user restart tsig-httpd
exit
```

验证 SAML 功能

若要验证端到端 SAML 功能，请使用您在此过程开始时创建的 Tableau 管理员帐户通过公共 URL（例如 https://tableau.example.com）登录到 Tableau Server。

如果 TSM 不启动(“网关错误”),或者在尝试连接时出现浏览器错误,请参见Tableau Server 独立网关故障排除。

配置独立网关的第二个实例上的身份验证模块

成功配置独立网关的第一个实例后,部署第二个实例。此处的示例是安装本主题中描述的 AWS/Mellon/Okta 方案的最终过程。该过程假定您已经在第二个实例上安装了独立网关,如本主题前面所述(安装独立网关)。

部署第二个独立网关的过程需要以下步骤:

1. 在独立网关的第二个实例上:安装 Mellon 身份验证模块。

   不要按照本主题前面的描述配置 Mellon 身份验证模块,而是必须按照后续步骤中的描述克隆配置。

2. 在已配置的(第一个)独立网关实例上:

   获取现有 Mellon 配置的 tar 副本。tar 备份将保留所有目录层次结构和权限。运行以下命令:

   ```
   cd /etc

   sudo tar -cvf mellon.tar mellon
   ```

   将 `mellon.tar` 复制到独立网关的第二个实例。

3. 在独立网关的第二个实例上:

   将 tar 文件提取(“解压缩”)到 `/etc` 目录中的第二个实例中。运行以下命令:

   ```
   cd /etc

   sudo tar -xvf mellon.tar
   ```

4. 在 Tableau Server 部署的节点 1 上：使用来自第二个独立网关的连接信息更新连接文件 (tsig.json)。您将需要按照本主题前面（安装独立网关）中的描述检索身份验证密钥。

   此处显示了一个示例连接文件 (tsig.json)：

```
{
"independentGateways": [
 {
   "id": "ip-10-0-1-169.ec2.internal",
   "host": "ip-10-0-1-169.ec2.internal",
   "port": "21319",
   "protocol" : "http",
   "authsecret": "13660-27118-29070-25482-9518-22453"
 },
 {
   "id": "ip-10-0-2-230.ec2.internal",
   "host": "ip-10-0-2-230.ec2.internal",
   "port": "21319",
   "protocol" : "http",
   "authsecret": "9055-27834-16487-27455-30409-7292"
 }]
 }
```

5. 在 Tableau Server 部署的节点 1 上：运行以下命令以更新配置：

```
tsm stop

tsm topology external-services gateway update -c tsig.json

tsm start
```

6. 在独立网关的两个实例上：在 Tableau Server 启动时，重新启动 tsig-httpd 进程：

```
sudo su - tableau-tsig

systemctl --user restart tsig-httpd
```

```
exit
```

7. 在 AWS 的**“EC2”>“目标组”中**:更新目标组以包含运行第二个独立网关实例的 EC2 实例。

   选择刚刚创建的目标组,然后单击“目标”选项卡:

   - 单击**“编辑”**。
   - 选择第二台独立网关计算机的 EC2 实例,然后单击**“添加到已注册”**。单击**“保存”**。

第 6 部分 - 安装后配置

配置从负载平衡器到 Tableau Server 的 SSL/TLS

一些组织需要从客户端到后端服务的端到端加密通道。到目前为止所描述的默认参考架构指定了从客户端到在您组织的 Web 层中运行的负载均衡器的 SSL。

本部分介绍如何在示例 AWS 参考架构中为 Tableau Server 和独立网关配置 SSL/TLS。有关描述如何在 AWS 参考架构中在 Apache 上配置 SSL/TLS 的配置示例，请参见示例：在 AWS 参考架构中配置 SSL/TLS。

目前，在 8000-9000 范围内运行的后端 Tableau Server 进程不支持 TLS。若要启用 TLS，您必须为独立网关配置与 Tableau Server 的中继连接。

此过程介绍如何针对独立网关到 Tableau Server 以及 Tableau Server 到独立网关的连接启用和配置 TLS。该过程加密通过 HTTPS/443 的中继流量和通过 HTTPS/21319 的整理流量。

此示例中的 Linux 程序显示了类似 RHEL 的发行版的命令。具体来说，这里的命令是使用 Amazon Linux 2 发行版开发的。如果您运行的是 Ubuntu 发行版，请相应地编辑命令。

此处的指南是对本指南中介绍的特定 AWS 示例参考架构的规定。因此，不包括可选配置。有关完整的参考文档，请参见*在独立网关上配置 TLS*(Linux)。

配置 TLS 之前

在工作时间之外执行 TLS 配置。配置至少需要重新启动一次 Tableau Server。如果您正在运行完整的四节点参考架构部署，则重新启动可能需要一段时间。

- 验证客户端是否可以通过 HTTP 连接到 Tableau Server。使用独立网关配置 TLS 是一个多步骤过程，可能需要进行故障排除。因此，我们建议在配置 TLS 之前从完全可操作的 Tableau Server 部署开始。

- 收集 TLS/SSL 证书、密钥和相关资产。您将需要适用于独立网关和 Tableau Server 的 SSL 证书。为了简化证书管理和部署，以及作为安全最佳实践，我们建议使用由主要受信任的第三方证书颁发机构 (CA) 生成的证书。或者，您可以为 TLS 生成自签名证书或使用 PKI 中的证书。

  本主题中的示例配置使用以下资产名称作为说明：

  - `tsig-ssl.crt`：独立网关的 TLS/SSL 证书。
  - `tsig-ssl.key`：独立网关上 `tsig-ssl.crt` 的私钥。
  - `ts-ssl.crt`：Tableau Server 的 TLS/SSL 证书。
  - `ts-ssl.key`：Tableau Server 上 `tsig-ssl.crt` 的私钥。
  - `tableau-server-CA.pem`：为 Tableau Server 计算机生成证书的 CA 的根证书。如果您使用来自主要可信第三方的证书，通常不需要此证书。
  - `rootTSIG-CACert.pem`：为独立网关计算机生成证书的 CA 的根证书。如果您使用来自主要可信第三方的证书，通常不需要此证书。
  - 本指南的第 5 部分详细介绍了 SAML 所需的其他证书和密钥文件资产。
  - 如果您的实施需要使用证书链文件，请参见知识库文章使用具有证书链的证书时在独立网关上配置 TLS。

- 验证您是否有权访问 IdP。如果您使用 IdP 进行身份验证，您可能需要在配置 SSL/TLS 后更改 IdP 上的收件人和目标 URL。

为 TLS 配置独立网关计算机

配置 TLS 可能是一个容易出错的过程。由于跨独立网关的两个实例进行故障排除可能非常耗时，我们建议仅使用一个独立网关在 EDG 部署上启用和配置 TLS。在您验证 TLS 可在整个部署中工作后，然后配置第二台独立网关计算机。

步骤 1：将证书和密钥分发到独立网关计算机

只要 tsig-httpd 用户具有对文件的读取访问权限，您就可以将资产分发到任意目录。其他过程中引用了这些文件的路径。我们在整个主题中将使用 `/etc/ssl` 下的示例路径，如下所示。

1. 为私钥创建目录:

   ```
   sudo mkdir -p /etc/ssl/private
   ```

2. 将证书和密钥文件复制到 /etc/ssl 路径。例如,

   ```
   sudo cp tsig-ssl.crt /etc/ssl/certs/
   ```

   ```
   sudo cp tsig-ssl.key /etc/ssl/private/
   ```

3. (可选) 如果您在 Tableau Server 上为 SSL/TLS 使用自签名或 PKI 证书, 则您还必须将 CA 根证书文件复制到独立网关计算机。例如,

   ```
   sudo cp tableau-server-CA.pem /etc/ssl/certs/
   ```

步骤 2:更新 TLS 的环境变量

您必须更新独立网关配置的端口和协议环境变量。

通过更新文件 /etc/opt/tableau/tableau_tsig/environment.bash 来更改这些值, 如下所示:

```
TSIG_HK_PROTOCOL="https"
TSIG_PORT="443"
TSIG_PROTOCOL="https"
```

步骤 3:更新 HK 协议的存根配置文件

手动编辑存根配置文件 (/var/opt/tableau/tableau_tsig/config/httpd.conf.stub), 为整理 (HK) 协议设置与 TLS 相关的 Apache httpd 指令。

存根配置文件包含一个与 TLS 相关的指令块, 这些指令已使用 #TLS# 标记注释掉。从指令中移除标记, 如下例所示。请注意, 该示例显示借助 SSLCACertificateFile 选项, 为 Tableau Server 上使用的 SSL 证书使用根 CA 证书的情况。

```
#TLS# SSLPassPhraseDialog exec:/path/to/file
<VirtualHost *:${TSIG_HK_PORT}>
SSLEngine on
```

```
#TLS# SSLHonorCipherOrder on
#TLS# SSLCompression off
SSLCertificateFile /etc/ssl/certs/tsig-ssl.crt
SSLCertificateKeyFile /etc/ssl/private/tsig-ssl.key
SSLCACertificateFile /etc/ssl/certs/tableau-server-CA.pem
#TLS# SSLCARevocationFile /path/to/file
</VirtualHost>
```

如果您重新安装独立网关，这些更改将丢失。我们建议制作备份副本。

步骤 4：复制存根文件并重新启动服务

1. 复制您在上一步中更新的文件，以使用更改更新 httpd.conf：

   ```
   cp /var/opt/tableau/tableau_tsig/config/httpd.conf.stub
   /var/opt/tableau/tableau_tsig/config/httpd.conf
   ```

2. 重新启动独立网关服务：

   ```
   sudo su - tableau-tsig
   systemctl --user restart tsig-httpd
   exit
   ```

重新启动后，独立网关将无法运行，直到您在 Tableau Server 上运行下一组步骤。在 Tableau Server 上完成这些步骤后，独立网关将获取更改并联机。

为 TLS 配置 Tableau Server 节点 1

在 Tableau Server 部署中的节点 1 上运行这些步骤：

步骤 1：复制证书和密钥并停止 TSM

1. 确认您已将 Tableau Server“外部 SSL”证书和密钥复制到节点 1。

2. 为了最大限度地减少停机时间，我们建议停止 TSM，运行以下步骤，然后在应用更改后启动 TSM：

   ```
   tsm stop
   ```

步骤 2:设置证书资产并启用独立网关配置

1. 指定独立网关的证书和密钥文件的位置。这些路径引用独立网关计算机上的位置。请注意,此示例假定使用相同的证书和密钥对来保护 HTTPS 和整理流量:

   ```
   tsm configuration set -k gateway.tsig.ssl.cert.file_name -v
   /etc/ssl/certs/tsig-ssl.crt --force-keys
   tsm configuration set -k gateway.tsig.ssl.key.file_name -v
   /etc/ssl/private/tsig-ssl.key --force-keys
   ```

2. 为独立网关的 HTTPS 和 HK 协议启用 TLS:

   ```
   tsm configuration set -k gateway.tsig.ssl.enabled -v true --
   force-keys
   tsm configuration set -k gateway.tsig.hk.ssl.enabled -v true --
   force-keys
   ```

3. (可选)如果您在独立网关上为 SSL/TLS 使用自签名或 PKI 证书,则必须上载 CA 根证书文件。CA 根证书文件是用于为独立网关计算机生成证书的根证书。例如,

   ```
   tsm security custom-cert add -c rootTSIG-CACert.pem
   ```

4. (可选)如果您在 Tableau Server 上为 SSL/TLS 使用自签名或 PKI 证书,则您还必须将 CA 根证书文件复制到独立网关 `/etc/ssl/certs` 目录。CA 根证书文件是用于为 Tableau Server 计算机生成证书的根证书。将证书复制到独立网关后,必须使用以下 tsm 命令指定证书在节点 1 上的位置。例如,

   ```
   tsm configuration set -k gateway.tsig.ssl.proxy.gateway_relay_
   cluster.cacertificatefile -v /etc/ssl/certs/tableau-server-
   CA.pem --force-keys
   ```

5. (可选:仅用于测试目的)如果您在计算机之间使用共享自签名或 PKI 证书,因此证书上的使用者名称与计算机名称不匹配,那么您必须禁用证书验证。

   ```
   tsm configuration set -k gateway.tsig.ssl.proxy.verify -v
   optional_no_ca --force-keys
   ```

步骤 3:为 Tableau Server 启用“外部 SSL”并应用更改

1. 在 Tableau Server 上启用和配置“外部 SSL”:

   ```
   tsm security external-ssl enable --cert-file ts-ssl.crt --key-file ts-ssl.key
   ```

2. 应用更改。

   ```
   tsm pending-changes apply
   ```

步骤 4:更新网关配置 JSON 文件并启动 tsm

1. 在 Tableau Server 端更新独立网关配置文件(例如,`tsig.json`),为独立网关对象指定 `https` 协议:

   ```
   "protocol" : "https",
   ```

2. 移除(或注释掉)独立网关第二个实例的连接信息。在保存之前,请务必在外部编辑器中验证 JSON。

   为独立网关的单个实例配置和验证 TLS 后,您将使用独立网关的第二个实例的连接信息更新此 JSON 文件。

3. 运行以下命令以更新独立网关配置:

   ```
   tsm topology external-services gateway update -c tsig.json
   ```

4. 启动 TSM。

   ```
   tsm start
   ```

5. 在 TSM 启动时,登录到独立网关实例并重新启动 `tsig-httpd` 服务:

   ```
   sudo su - tableau-tsig
   ```

   ```
   systemctl --user restart tsig-httpd
   ```

   ```
   exit
   ```

将 IdP 身份验证模块 URL 更新为 HTTPS

如果您为 Tableau 配置了外部身份提供程序，则您可能需要更新 IdP 管理仪表板中的返回 URL。

举例来说，如果您使用 Okta 预身份验证应用程序，您将需要更新应用程序以对收件人 URL 和目标 URL 使用 HTTPS 协议。

为 HTTPS 配置 AWS 负载平衡器

如果您按照本指南中的说明使用 AWS 负载平衡器进行部署，那么您可以重新配置 AWS 负载平衡器以将 HTTPS 流量发送到运行独立网关的计算机：

1. 删除现有 HTTP 目标组：

   在**“目标组”**中，选择已为负载平衡器配置的 HTTP 目标组，单击**“操作”**，然后单击**“删除”**。

2. 创建 HTTPS 目标组

   “目标组”>“创建目标组”

   - 选择“实例”
   - 输入目标组名称，例如 `TG-internal-HTTPS`
   - 选择您的 VPC
   - 协议：HTTPS 443
   - 在**“运行状况检查”>“高级运行状况检查设置”>“成功代码”**下，将代码列表追加为：`200,303`。
   - 单击**“创建”**。

3. 选择刚刚创建的目标组，然后单击**“目标”**选项卡：

   - 单击**“编辑”**
   - 选择正在运行已配置的 Tableau Server 独立网关的 EC2 实例，然后单击**“添加到已注册”**。
   - 单击**“保存”**。

4. 创建目标组后，您必须启用粘性：

- 打开 AWS 目标组页面(**“EC2”>“负载平衡”>“目标组”**),选择您刚刚设置的目标组实例。在**“操作”**菜单上,选择**“编辑属性”**。
- 在**“编辑属性”**页面上,选择**“粘性”**,指定持续时间 `1 day`,然后**保存更改**。

5. 在负载平衡器上,更新侦听器规则。选择您为此部署配置的负载平衡器,然后单击**“侦听器”**选项卡。
   - 对于**“HTTP:80”**,单击**“查看/编辑规则”**。在生成的**“规则”**页面上,单击编辑图标(一次位于页面顶部,然后再次位于规则旁边)以编辑规则。删除现有 THEN 规则,并通过单击**“添加操作”>“重定向至...”**替换该规则。在生成的 THEN 配置中,指定 `HTTPS` 和端口 `443`,并将其他选项保留为默认设置。保存设置,然后单击**“更新”**。
   - 对于**“HTTPS:443”**,单击**“查看/编辑规则”**。在生成的**“规则”**页面上,单击编辑图标(一次位于页面顶部,然后再次位于规则旁边)以编辑规则。删除现有 THEN 规则,并通过单击**“添加操作”>“转发至...”**替换该规则。将目标组指定为您刚刚创建的 HTTPS 组。在**“组级粘性”**下,启用粘性并将持续时间设置为 1 天。保存设置,然后单击**“更新”**。

6. 在负载平衡器上,将空闲超时更新为 400 秒。选择您为此部署配置的负载平衡器,然后单击**“操作”>“编辑属性”**。将**“空闲超时”**设置为 `400` 秒,然后单击**“保存”**。

验证 TLS

若要验证 TLS 功能,请使用您在此过程开始时创建的 Tableau 管理员帐户通过公共 URL(例如 https://tableau.example.com)登录到 Tableau Server。

如果 TSM 没有启动或出现其他错误,请参见Tableau Server 独立网关故障排除。

为 SSL 配置独立网关的第二个实例

成功配置独立网关的第一个实例后,部署第二个实例。

部署第二个独立网关的过程需要以下步骤:

1. 在已配置的(第一个)独立网关实例上;将以下文件复制到第二个独立网关实例上的相应位置:

   - /etc/ssl/certs/tsig-ssl.crt
   - /etc/ssl/private/tsig-ssl.key(您需要在第二个实例上创建private目录)。
   - /var/opt/tableau/tableau_tsig/config/httpd.conf.stub
   - /etc/opt/tableau/tableau_tsig/environment.bash

2. 在 Tableau Server 部署的节点 1 上:使用来自第二个独立网关的连接信息更新连接文件 (tsig.json)。

   此处显示了一个示例连接文件 (tsig.json):

```
{
"independentGateways": [
 {
   "id": "ip-10-0-1-169.ec2.internal",
   "host": "ip-10-0-1-169.ec2.internal",
   "port": "21319",
   "protocol" : "https",
   "authsecret": "13660-27118-29070-25482-9518-22453"
 },
 {
   "id": "ip-10-0-2-230.ec2.internal",
   "host": "ip-10-0-2-230.ec2.internal",
   "port": "21319",
   "protocol" : "https",
   "authsecret": "9055-27834-16487-27455-30409-7292"
 }]
 }
```

3. 在 Tableau Server 部署的节点 1 上:运行以下命令以更新配置:

```
tsm stop

tsm topology external-services gateway update -c tsig.json
```

```
tsm start
```

4. 在独立网关的两个实例上：在 Tableau Server 启动时，在独立网关的两个实例上重新启动 tsig-httpd 进程：

   ```
   sudo su - tableau-tsig

   systemctl --user restart tsig-httpd

   exit
   ```

5. 在 AWS 的**“EC2”>“目标组”中**：更新目标组以包含运行第二个独立网关实例的 EC2 实例。

   选择刚刚创建的目标组，然后单击“目标”选项卡：

   - 单击**“编辑”**。
   - 选择第二台独立网关计算机的 EC2 实例，然后单击**“添加到已注册”**。单击**“保存”**。

针对 Postgres 配置 SSL

您可以选择为 Tableau Server 上外部存储库连接的 Postgres 连接配置 SSL (TLS)。

为了简化证书管理和部署，以及作为安全最佳实践，我们建议使用由主要受信任的第三方证书颁发机构 (CA) 生成的证书。或者，您可以为 TLS 生成自签名证书或使用 PKI 中的证书。

此过程描述了如何使用 OpenSSL，在示例 AWS 参考架构中类似于 RHEL 的 Linux 发行版上的 Postgres 主机上生成自签名证书。

生成并签署 SSL 证书后，您必须将 CA 证书复制到 Tableau 主机。

在运行 Postgress 的主机上：

1. 生成签名根证书颁发机构 (CA) 密钥：

   ```
   openssl genrsa -out pgsql-rootCAKey.pem 2048
   ```

2. 创建根 CA 证书：

   ```
   openssl req -x509 -sha256 -new -nodes -key pgsql-rootCAKey.pem
   -days 3650 -out pgsql-rootCACert.pem
   ```

   系统将提示您输入证书字段的值。例如：

   ```
   Country Name (2 letter code) [XX]:US
   State or Province Name (full name) []:Washington
   Locality Name (eg, city) [Default City]:Seattle
   Organization Name (eg, company) [Default Company Ltd]:Tableau
   Organizational Unit Name (eg, section) []:Operations
   Common Name (eg, Postgres server's hostname) []:ip-10-0-1-
   189.us-west-1.compute.internal
   Email Address []:example@tableau.com
   ```

3. 为 Postgres 计算机创建证书和相关密钥(在下面的示例中为 server.csr 和 server.key)。证书的使用者名称必须与 Postgres 主机的 EC2 私有 DNS 名称匹配。使用者名称是使用 -subj 选项以"/CN=<private DNS name>"格式设置的，例如：

   ```
   openssl req -new -nodes -text -out server.csr -keyout
   server.key -subj "/CN=ip-10-0-1-189.us-west-1.compute.internal"
   ```

4. 使用您在步骤 2 中创建的 CA 证书签署新证书。以下命令还将以 crt 格式输出证书：

   ```
   openssl x509 -req -in server.csr -days 3650 -CA pgsql-
   rootCACert.pem -CAkey pgsql-rootCAKey.pem -CAcreateserial -out
   server.crt
   ```

5. 将 crt 和 key 文件复制到 Postgres /var/lib/pgsql/13/data/ 路径：

   ```
   sudo cp server.crt /var/lib/pgsql/13/data/
   ```

   ```
   sudo cp server.key /var/lib/pgsql/13/data/
   ```

6. 切换到根用户：

```
sudo su
```

7. 设置证书和密钥文件的权限。运行以下命令：

```
cd /var/lib/pgsql/13/data
chown postgres.postgres server.crt
chown postgres.postgres server.key
chmod 0600 server.crt
chmod 0600 server.key
```

8. 更新 pg_haba 配置文件 `/var/lib/pgsql/13/data/pg_hba.conf` 以指定 md5 信任：

   将现有的连接语句从

   ```
   host all all 10.0.30.0/24 password,和

   host all all 10.0.31.0/24 password
   ```

   更改为

   ```
   host all all 10.0.30.0/24 md5,和

   host all all 10.0.31.0/24 md5。
   ```

9. 通过以下这一行来更新 postgresql 文件 `/var/lib/pgsql/13/data/postgresql.conf`：

   ```
   ssl = on
   ```

10. 退出根用户模式：

    ```
    exit
    ```

11. 重新启动 Postgres：

    ```
    sudo systemctl restart postgresql-13
    ```

可选：在 Tableau Server 上针对 Postgres SSL 启用证书信任验证

如果您遵循了第 4 部分 - 安装并配置 Tableau Server中的安装过程，那么 Tableau Server 将为 Postgres 连接配置可选的 SSL。这意味着在 Postgres 上配置 SSL（如上所述）将导致加密连接。

如果希望对连接进行证书信任验证，则必须在 Tableau Server 上运行以下命令来重新配置 Postgres 主机连接：

```
tsm topology external-services repository replace-host -f
<filename>.json -c CACert.pem
```

其中 `<filename>.json` 是配置外部 Postgres中描述的连接文件。`CACert.pem` 是 Postgres 使用的 SSL/TLS 证书的 CA 证书文件。

可选：验证 SSL 连接

若要验证 SSL 连接，您必须：

- 在 Tableau Server 节点 1 上安装 Postgres 客户端。
- 将您在前面的过程中创建的根证书复制到 Tableau 主机。
- 从节点 1 连接到 Postgres 服务器

在节点 1 上安装 Postgres 客户端

此示例演示如何安装版本 Postgres 13.4。安装与您为外部存储库运行的版本相同的版本。

1. 在节点 1 上，在 `/etc/yum.repos.d` 路径中创建和编辑 pgdg.repo 文件，使用以下配置信息填充该文件。

   ```
   [pgdg13]
   name=PostgreSQL 13 for RHEL/CentOS 7 - x86_64
   ```

```
baseurl=https://download.postgresql.org/pub/repos/yum/13/redha-
t/rhel-7-x86_64
enabled=1
gpgcheck=0
```

2. 安装 Postgres 客户端：

```
sudo yum install postgresql13-13.4-1PGDG.rhel7.x86_64
```

将根证书复制到节点 1

将 CA 证书 (pgsql-rootCACert.pem) 复制到 Tableau 主机：

```
scp ec2-user@<private-DNS-name-of-Postgress-host>:/home/ec2-
user/pgsql-rootCACert.pem /home/ec2-user
```

通过 SSL 从节点 1 连接到 Postgres 主机：

从节点 1 运行以下命令，指定 Postgres 服务器主机 IP 地址和根 CA 证书：

```
psql "postgresql://postgres@<IP-
address>:5432/postgres?sslmode=verify-ca&sslrootcert=pgsql-
rootCACert.pem"
```

例如：

```
 psql
"postgresql://postgres@10.0.1.189:5432/postgres?sslmode=verify-
ca&sslrootcert=pgsql-rootCACert.pem"
```

Postgres 将提示您输入密码。登录成功后，shell 会返回：

```
psql (13.4)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-
SHA384, bits: 256, compression: off)
Type "help" for help.
postgres=#
```

配置 SMTP 和事件通知

Tableau Server 向管理员和用户发送电子邮件通知。为了实现这一点，您必须配置 Tableau Server 以将邮件发送到您的电子邮件服务器。您还必须指定要发送的事件类型、阈值和订阅信息。

对于 SMTP 和通知的初始配置，我们建议您使用下面的配置文件模板创建一个 json 文件。您还可以使用*"tsm configuration set"*(Linux) 中描述的语法设置下面列出的任何单一配置键。

在 Tableau Server 部署中的节点 1 上运行此过程：

1. 将以下 json 模板复制到文件。使用您的 SMTP 配置选项以及组织的订阅和警报通知自定义文件。

    - 若要查看所有 SMTP 选项的列表和描述，请参见*"SMTP CLI 配置参考"*(Linux)。
    - 若要查看所有通知事件选项的列表和描述，请参见*"配置服务器事件通知"*(Linux) 的"CLI"部分。

```
{
"configKeys": {
       "svcmonitor.notification.smtp.server": "SMTP server host
name",
       "svcmonitor.notification.smtp.send_account": "SMTP user name",
       "svcmonitor.notification.smtp.port": 443,
       "svcmonitor.notification.smtp.password": "SMTP user account
password",
       "svcmonitor.notification.smtp.ssl_enabled": true,
       "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",
       "backgrounder.notifications_enabled": true,
```

```
        "subscriptions.enabled": true,
        "subscriptions.attachments_enabled": true,
        "subscriptions.max_attachment_size_megabytes": 150,
        "svcmonitor.notification.smtp.enabled": true,
        "features.DesktopReporting": true,
        "storage.monitoring.email_enabled": true,
        "storage.monitoring.warning_percent": 20,
        "storage.monitoring.critical_percent": 15,
        "storage.monitoring.email_interval_min": 25,
        "storage.monitoring.record_history_enabled": true
        }
}
```

2. 运行 `tsm settings import -f file.json` 以将 json 文件传递至 Tableau 服务管理器。
3. 运行 `tsm pending-changes apply` 命令以应用更改。
4. 运行 `tsm email test-smtp-connection` 以查看和验证连接配置。

安装 PostgreSQL 驱动程序

若要在 Tableau Server 上查看管理员视图，必须在 Tableau Server 部署的节点 1 上安装 PostgreSQL 驱动程序。

1. 转到 Tableau 驱动程序下载页面，并复制 PostgreSQL jar 文件的 URL。
2. 在 Tableau 部署的每个节点上运行以下过程：
    - 创建以下文件路径：

      ```
      sudo mkdir -p /opt/tableau/tableau_driver/jdbc
      ```

    - 从新路径中，下载最新版本的 PostgreSQL jar 文件：例如：

```
sudo wget
https://downloads.tableau.com/drivers/linux/postgresql/pos-
tgresql-42.2.22.jar
```

3. 在初始节点上，重新启动 Tableau Server：

```
tsm restart
```

配置强密码策略

如果您不使用 IdP 身份验证解决方案部署 Tableau Server，我们建议对默认 Tableau 密码策略进行安全强化。

如果您使用 IdP 部署 Tableau Server，则必须使用 IdP 管理密码策略。

以下过程包括用于在 Tableau Server 上设置密码策略的 json 配置。有关以下选项的详细信息，请参见“*本地身份验证* (Linux)”。

1. 将以下 json 模板复制到文件。使用您的密码策略配置填入键值。

```
{
 "configKeys": {
       "wgserver.localauth.policies.mustcontainletters.enabled":
true,
       "wgserver.localauth.policies.mustcontainuppercase.enabled":
true,
       "wgserver.localauth.policies.mustcontainnumbers.enabled":
true,
       "wgserver.localauth.policies.mustcontainsymbols.enabled":
true,
       "wgserver.localauth.policies.minimumpasswordlength.enabled":
true,
       "wgserver.localauth.policies.minimumpasswordlength.value": 12,
       "wgserver.localauth.policies.maximumpasswordlength.enabled":
false,
       "wgserver.localauth.policies.maximumpasswordlength.value":
```

```
255,
       "wgserver.localauth.passwordexpiration.enabled": true,
       "wgserver.localauth.passwordexpiration.days": 90,
       "wgserver.localauth.ratelimiting.maxbackoff.minutes": 60,
       "wgserver.localauth.ratelimiting.maxattempts.enabled": false,
       "wgserver.localauth.ratelimiting.maxattempts.value": 5,
       "vizportal.password_reset": true
 }
}
```

2. 运行 `tsm settings import -f file.json`，以将 json 文件传递给 Tableau 服务管理器，从而配置 Tableau Server。

3. 运行 `tsm pending-changes apply` 命令以应用更改。

第 7 部分 - 验证、工具和故障排除

此部分包括安装后验证步骤和故障排除指南。

故障转移系统验证

配置部署后，我们建议运行简单的故障转移测试来验证系统冗余。

我们建议运行以下步骤来验证故障转移功能：

1. 关闭独立网关的第一个实例 (TSIG1)。所有入站流量都应通过独立网关的第二个实例 (TSIG2) 进行路由。
2. 重新启动 TSIG1，然后关闭 TSIG2。所有入站流量都应通过 TSIG1 路由。
3. 重新启动 TSIG2。
4. 关闭 Tableau Server 节点 1。所有 Vizportal/应用服务流量将故障转移到节点 2。

   注意：截至 2022 年 9 月，Tableau Server 2021.4 及更高版本的某些版本上的节点 1 高可用性受到影响。如果节点 1 关闭，客户端连接将失败。此问题已在这些维护版本中得到修复：

   - 2021.4.15 及更高版本
   - 2022.1.11 及更高版本
   - 2023.1.3 及更高版本

   为了确保使用 ATR 激活的 Tableau Server 安装在初始节点故障后有 72 小时的宽限期，请安装或升级到这些版本之一。有关更多详细信息，请参见 Tableau 知识库中的使用 ATR 的 Tableau Server HA 在初始节点故障后没有宽限期。
5. 重新启动节点 1 并关闭节点 2。所有 Vizportal/应用服务流量将故障转移到节点 1。
6. 重新启动节点 2。

在此上下文中，“关闭”或“重新启动”是通过关闭操作系统或虚拟机来完成的，而无需事先尝试应用程序的正常关闭。目标是模拟硬件或虚拟机故障。

每个故障转移测试的最小验证步骤是验证用户身份并执行基本的视图操作。

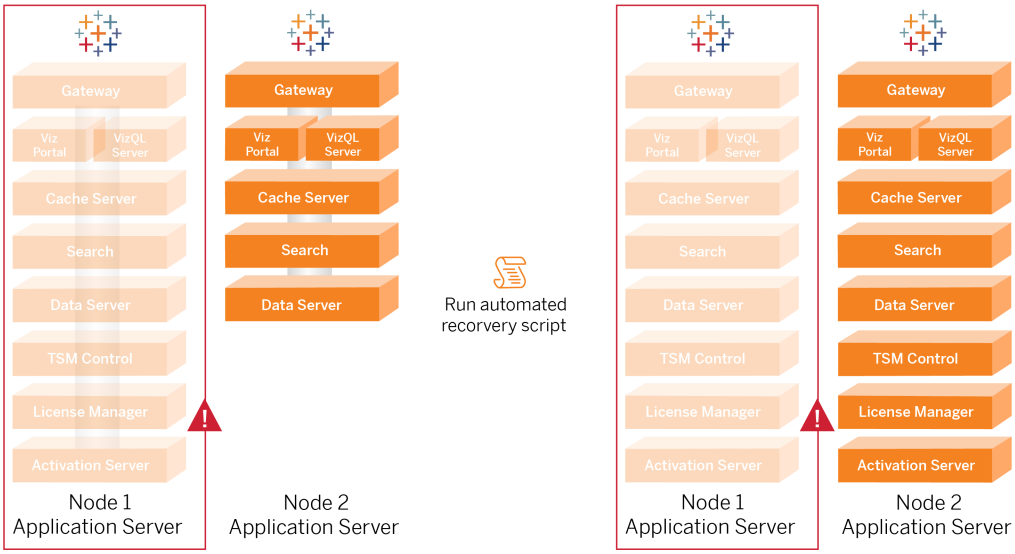
当您在模拟失败后尝试登录时，可能会收到“错误请求”浏览器错误。即使您清除了浏览器中的缓存，也可能会看到此错误。当浏览器缓存来自先前 IdP 会话的数据时，通常会发生此问题。如果即使在您清除本地浏览器缓存后此错误仍然存在，请通过连接其他浏览器来验证 Tableau 方案。

初始节点自动恢复

Tableau Server 版本 2021.2.4 及更高版本在脚本目录 (`/app/tableau_server/packages/scripts.<version>`) 中包括自动初始节点恢复脚本 `auto-node-recovery`。

如果初始节点存在问题，并且节点 2 上有冗余进程，则无法保证 Tableau Server 可以继续运行。在初始节点出现故障后，Tableau Server 最多可以继续运行 72 小时，然后才会因为缺少许可服务而影响其他进程。如果是这样，在初始节点出现故障后，您的用户或许能够继续登录并且查看和使用其内容，但您将无法重新配置 Tableau Server，原因是您没有管理控制器的访问权限。

即使配置了冗余进程，在初始节点出现故障后，Tableau Server 也可能无法继续运行。



恢复初始节点(节点 1)故障:

1. 登录 Tableau Server 节点 2。
2. 切换到脚本目录:

   ```
   cd /app/tableau_server/packages/scripts.<version>
   ```

3. 运行以下命令以启动脚本:

   ```
   sudo ./auto-node-recovery -p node1 -n node2 -k <license keys>
   ```

   其中 `<license keys>` 是用于您的部署的许可证密钥的逗号分隔(无空格)列表。如果您无权访问许可证密钥,请访问 Tableau 客户门户以检索它们。例如:

   ```
   sudo ./auto-node-recovery -p node1 -n node2 -k TSB4-8675-309F-TW50-9RUS,TSNM-559N-ULL6-22VE-SIEN
   ```

自动节点恢复脚本将执行大约 20 个步骤以将服务恢复到节点 2。随着脚本的进行,每个步骤都显示在终端中。更详细的状态会记录到 `/data/tableau_data/logs/app-controller-move.log`。在大多数环境中,脚本需要 35 到 45 分钟才能完成。

对初始节点恢复进行故障排除

如果节点恢复失败，您可能会发现以交互方式运行脚本以允许或禁止过程中的离散步骤很有用。举例来说，如果脚本在整个过程中失败，您可以查看日志文件，更改配置，然后再次运行脚本。通过以交互模式运行，您可以随后跳过所有步骤，直到到达失败的步骤。

若要以交互模式运行，请向 script 参数中添加 `-i` 开关。

重建故障节点

运行脚本后，节点 2 将运行以前在出现故障的节点 1 主机上的所有服务。若要在 4 节点中添加，您需要使用引导程序文件部署新的 Tableau Server 主机，并按照第 4 部分中指定的原始节点 2 的方式对其进行配置。请参见配置节点 2。

switchto

Switchto 是 Tim 的一个脚本，它可以轻松地完成窗口之间的切换。

1. 将以下代码复制到堡垒主机上主目录中名为 `switchto` 的文件中。

```
#!/bin/bash
#-----------------------------------------------------------------
-----
# switchto
#
# Helper function to simplify SSH into the various AWS hosts
when
# following the Tableau Server Enterprise Deployment Guide
(EDG).
#
# Place this file on your bastion host and provide your AWS
hosts'
# internal ip addresses or machine names here.
# Example: readonly NODE1="10.0.3.187"
```

```
#
readonly NODE1=""
readonly NODE2=""
readonly NODE3=""
readonly NODE4=""
readonly PGSQL=""
readonly PROXY1=""
readonly PROXY2=""

usage() {
echo "Usage: switchto.sh [ node1 | node2 | node3 | node4 |
pgsql | proxy1 | proxy2 ]"
}


ip=""

case $1 in
       node1)
               ip="$NODE1"
               ;;
       node2)
               ip="$NODE2"
               ;;
       node3)
               ip="$NODE3"
               ;;
       node4)
               ip="$NODE4"
               ;;
       pgsql)
               ip="$PGSQL"
               ;;
       proxy1)
               ip="$PROXY1"
```

```
                ;;
        proxy2)
                ip="$PROXY2"
                ;;
        ?)
                usage
                exit 0
                ;;
        *)
                echo "Unkown option $1."
                usage
                exit 1
                ;;
esac

if [[ -z $ip ]]; then
echo "You must first edit this file to provide the ip addresses
of your AWS hosts."
exit 1
fi

ssh -A ec2-user@$ip
```

2. 更新脚本中的 IP 地址以映射到您的 EC2 实例，然后保存文件。
3. 对脚本文件应用权限：

   ```
   sudo chmod +x switchto
   ```

用法：

若要切换到主机，请运行以下命令：

```
./switchto <target>
```

例如，若要切换到节点 1，请运行以下命令：

```
./switchto node1
```

Tableau Server 独立网关故障排除

在 Tableau Server 上配置独立网关、Okta、Mellon 和 SAML 可能是一个容易出错的过程。最常见的失败根本原因是字符串错误。例如，在配置期间指定的 Okta URL 上的尾部斜杠 (/) 可能会导致与 SAML 断言相关的不匹配错误。这只是一个例子。在配置期间有很多机会在任何应用程序中输入不正确的字符串。

重新启动 tableau-tsig 服务

始终通过重新启动独立网关计算机上的 tableau-tsig 服务来开始（和完成）故障排除。重新启动此服务很快，并且通常会触发更新的配置从 Tableau Server 加载。

在独立网关计算机上运行以下命令：

```
sudo su - tableau-tsig

systemctl --user restart tsig-httpd

exit
```

查找不正确的字符串

如果您犯了字符串错误（复制/粘贴错误、字符串被截断等），请花时间浏览您配置的每个设置：

- Okta 预身份验证配置。仔细查看您设置的 URL。查找尾部斜杠。验证 HTTP 与 HTTPS。
- 节点 1 上 SAML 配置的 Shell 历史记录。审查您运行的 `tsm authentication saml configure` 命令。验证所有 URL 是否与您在 Okta 中配置的 URL 匹配。当您从节点 1 查看 shell 历史记录时，请验证指定 Mellon 配置文件路径的 `tsm configuration set` 命令是否准确映射到您在独立网关上复制文件的文件路径。
- 独立网关上的 Mellon 配置。查看 shell 历史记录以验证您创建的元数据是否使用您在 Okta 和 Tableau SAML 中配置的相同 URL 字符串。验证 `/etc/mellon/conf.d/global.conf` 中指定的所有路径是否正确，并且 `MellonCookieDomain` 设置为您的根域，而不是您的 Tableau 子域。

搜索相关日志

如果所有字符串似乎都设置正确，那么您应该检查日志是否有错误。

Tableau Server 将错误和事件记录到数十个不同的日志文件中。独立网关也记录到一组本地文件。我们建议按以下顺序检查这些日志。

独立网关日志文件

这些独立网关日志文件的默认位置为 `/var/opt/tableau/tableau_tsig/logs`。

- `access.log`：此日志非常有用，因为它具有显示来自 Tableau Server 节点的连接的条目。如果您在尝试启动 TSM 时遇到网关错误（不会启动），并且 `access.log` 文件中没有条目，则存在核心连接问题。始终将验证 AWS 安全组配置作为第一步。另一个常见问题是 `tsig.json` 中的拼写错误。如果您对 tsig.json 进行更新，请在运行 `tsm topology external-services gateway update -c tsig.json` 之前运行 `tsm stop`。tsig.json 更新后，运行 `tsm start`。
- `error.log`：除其他条目外，此日志还包括 SAML 和 Mellon 错误。

Tableau Server tabadminagent 日志文件

`tabadminagent`（而不是 `tabadmincontroller`）文件集是解决独立网关相关错误的唯一相关日志文件。

您必须找到已将独立网关错误记录到 `tabdminagent`的位置。这些错误可以在任何节点上，但它们只在一个节点上。在 Tableau Server 群集中的每个节点上执行以下步骤，直到找到“independent”字符串：

1. 在 EDG 设置中找到 Tableau Server 节点 1-4 上的 tabadminagent 日志文件位置：

   ```
   cd /data/tableau_data/data/tabsvc/logs/tabadminagent
   ```

2. 打开要读取的最新日志：

   ```
   less tabadminagent_nodeN.log
   ```

   （将 N 替换为节点编号）

3. 搜索“Independent”和“independent”的所有实例 - 使用以下搜索字符串：

   ```
   /ndependent
   ```

   如果没有匹配，则转到下一个节点并重复步骤 1-3。

4. 当您得到一个匹配：`Shift + G` 移动到底部以获取最后的错误消息。

重新加载 httpd 存根文件

独立网关管理 Apache 的 httpd 配置。通常可以修复暂时问题的一个通用操作是重新加载 httpd 存根文件，该文件是底层 Apache 配置的种子。在独立网关的两个实例上运行以下命令。

1. 将存根文件复制到 httpd.conf：

   ```
   cp /var/opt/tableau/tableau_tsig/config/httpd.conf.stub
   /var/opt/tableau/tableau_tsig/config/httpd.conf
   ```

2. 重新启动独立网关服务：

   ```
   sudo su - tableau-tsig
   systemctl --user restart tsig-httpd
   exit
   ```

删除或移动日志文件

独立网关记录所有访问事件。您需要管理日志文件存储以避免填满磁盘空间。如果您的磁盘已满，独立网关将无法写入访问事件，服务将失败。以下消息将被记录到独立网关上的 error.log 中：

```
(28)No space left on device: [client 10.0.2.209:54332] AH00646:
Error writing to /var/opt/tableau/tableau_
tsig/logs/access.%Y_%m_%d_%H_%M_%S.log
```

当您在 Tableau 节点 1 上运行 `tsm status -v` 时，此故障将导致 `external` 节点处于 `DEGRADED` 状态。状态输出中的 `external` 节点是指独立网关。

为了解决此问题，请删除或移动磁盘上的 access.log 文件。访问日志文件的存储位置为 `/var/opt/tableau/tableau_tsig/logs`。清除磁盘后，请重新启动 tableau-tsig 服务。

浏览器错误

错误请求：这种情况下的一个常见错误是来自 Okta 的“错误请求”错误。当浏览器缓存来自先前 Okta 会话的数据时，通常会发生此问题。举例来说，如果您以 Okta 管理员身份管理 Okta 应用程序，然后尝试使用不同的启用 Okta 的帐户访问 Tableau，则来自管理员数据的会话数据可能会导致“错误请求”错误。如果即使在您清除本地浏览器缓存后此错误仍然存在，请尝试通过连接其他浏览器来验证 Tableau 方案。

“错误请求”错误的另一个原因是您在 Okta、Mellon 和 SAML 配置过程中输入的许多 URL 之一中的拼写错误。检查您输入的所有这些内容都没有错误。

通常独立网关服务器上的 `error.log`文件将指定导致错误的 URL。

未找到 - 在此服务器上找不到请求的 URL：此错误表示许多配置错误之一。

如果用户使用 Okta 进行身份验证，然后收到此错误，则很可能是您在配置 SAML 时将 Okta 预身份验证应用程序上载到 Tableau Server。验证您是否在 Tableau Server 上配置了 Okta Tableau Server 应用程序元数据，而不是 Okta 预身份验证应用程序元数据

其他疑难解答步骤：

- 查看 Okta 预身份验证应用程序设置。确保按照本主题中指定的方式设置 HTTP 与 HTTPS 协议。
- 在两个独立网关服务器上重新启动 tsig-httpd。
- 验证 `sudo apachectl configtest` 在两个独立网关上是否都返回“Syntax OK”。
- 验证测试用户是否已分配给 Okta 中的两个应用程序。
- 验证是否在负载平衡器和关联的目标组上设置了粘性。

验证从 Tableau Server 到独立网关的 TLS 连接

使用 wget 命令来验证从 Tableau Server 到独立网关的连接和访问。此命令的变体可以帮助您了解证书问题是否导致连接问题。

例如，运行此 wget 命令以从 Tableau Server 中验证整理 (HK) 协议：

```
wget https://ip-10-0-1-38.us-west-1.compute.internal:21319
```

使用您为 tsig.json 文件的 host 选项包含的相同主机地址构造 URL。指定 https 协议，并在 URL 后面加上 HK 端口 21319。

检查连接并忽略证书验证：

```
wget https://ip-10-0-1-38.us-west-1.compute.internal:21319 --no-check-certificate
```

验证 TSIG 的根 CA 证书是否有效：

```
wget https://ip-10-0-1-38.us-west-1.compute.internal:21319 --ca-certificate=tsigRootCA.pem
```

如果 Tableau 能够通信，那么您可能仍会收到与内容相关的错误，但不会收到与连接相关的错误。如果 Tableau 根本无法连接，则首先验证防火墙/安全组中的协议配置。例如，独立网关所在的安全组的入站规则必须允许 TCP 21319。

附录 - AWS 部署工具箱

本主题包括在 AWS 中部署时参考架构的工具和备用部署选项。具体而言，本主题描述了如何自动完成整个 EDG 中描述的示例 AWS 部署。

TabDeploy4EDG 自动安装脚本

TabDeploy4EDG 脚本自动执行第 4 部分 - 安装并配置 Tableau Server中描述的四节点 Tableau 部署。如果您遵循本指南中描述的示例 AWS/Linux 实施，那么您可能能够运行 TabDeploy4EDG。

要求。若要运行该脚本，您必须根据第 3 部分 - 准备 Tableau Server 企业部署中的示例实现来准备和配置 AWS 环境：

- VPC、子网和安全组已按所述方式进行配置。IP 地址不必与示例实现中显示的地址相匹配。
- 运行最新、更新的 AWS Linux 2 版本的四个 EC2 实例
- PostgreSQL 已安装并已按照安装、配置 PostgreSQL 和创建 PostgreSQL 的 tar 备份 中所述方式进行配置。
- 步骤 1 tar 备份文件位于安装了 PostgreSQL 的 EC2 实例上，如进行 PostgreSQL 步骤 1 tar 备份中所述。
- 将运行 Tableau Server 部署的节点 1 的 EC2 实例已配置为与 PostgreSQL 通信，如第 4 部分 - 安装并配置 Tableau Server中所述。
- 您已使用来自堡垒主机的 SSH 会话登录到每个 EC2 实例。

该脚本大约需要 1.5-2 小时来安装和配置四个 Tableau 服务器。该脚本根据参考架构的规定设置配置 Tableau。该脚本执行以下操作：

- 如果指定 PostgreSQL 主机的 tar 文件的路径，则还原 PostgreSQL 主机的第 1 阶段备份。
- 清除所有节点上的现有 Tableau 安装。
- 在所有节点上运行 `sudo yum update`。
- 将 Tableau rpm 包下载并复制到每个节点。
- 下载并安装每个节点的依赖项。
- 创建 `/app/tableau_server` 并在所有节点上安装包。

- 使用本地身份存储安装节点 1，并使用 PostgreSQL 配置外部存储库。
- 执行节点 2-节点 4 的引导程序安装和初始配置。
- 删除引导程序文件和 TabDeploy4EDG 的配置文件。
- 根据参考架构规范跨 Tableau 集群配置服务。
- 验证安装并返回每个节点的状态。

下载脚本并复制到堡垒主机

1. 从 TabDeploy4EDG 示例页面复制脚本，并将代码粘贴到名为 `TabDeploy4EDG` 的文件中。
2. 将文件保存到充当堡垒主机的 EC2 主机上的主目录。
3. 运行以下命令以更改文件模式以使其可执行：

```
sudo chmod +x TabDeploy4EDG
```

运行 TabDeploy4EDG

TabDeploy4EDG 必须从堡垒主机运行。该脚本编写时假设您正在 ssh 转发代理的上下文中运行，如示例：连接到 AWS 中的堡垒主机中所述。如果您没有使用 ssh 转发代理上下文运行，那么在整个安装过程中将提示您输入密码。

1. 创建、编辑和保存注册文件 (`registration.json`)。该文件必须是格式正确的 json 文件。复制并自定义以下模板：

```
{
        "zip" : "97403",
        "country" : "USA",
        "city" : "Springfield",
        "last_name" : "Simpson",
        "industry" : "Energy",
        "eula" : "yes",
        "title" : "Safety Inspection Engineer",
        "phone" : "5558675309",
        "company" : "Example",
        "state" : "OR",
        "department" : "Engineering",
        "first_name" : "Homer",
```

```
        "email" : "homer@example.com"
}
```

2. 运行以下命令以生成模板配置文件：

   ```
   ./TabDeploy4EDG -g edg.config
   ```

3. 打开配置文件进行编辑：

   ```
   sudo nano edg.config
   ```

   至少，您必须添加每个 EC2 主机的 IP 地址、注册文件的文件路径，以及有效的许可证密钥。

4. 完成配置文件的编辑后，保存该文件并将其关闭。

5. 若要运行 TabDeploy4EDG，请运行以下命令：

   ```
   ./TabDeploy4EDG -f edg.config
   ```

示例：使用 Terraform 自动完成 AWS 基础设施部署

本部分介绍如何配置和运行 Terraform 以在 AWS 中部署 EDG 参考架构。此处提供的示例 Terraform 配置部署了一个 AWS VPC，其中包含第 3 部分 - 准备 Tableau Server 企业部署中描述的子网、安全组和 EC2 实例。

Tableau 示例网站上提供了示例 Terraform 模板，网址为 https://help.tableau.com/samples/en-us/edg/edg-terraform.zip 。必须为您的组织配置和自定义这些模板。本部分提供的配置内容介绍了您必须自定义才能部署的最低所需模板更改。

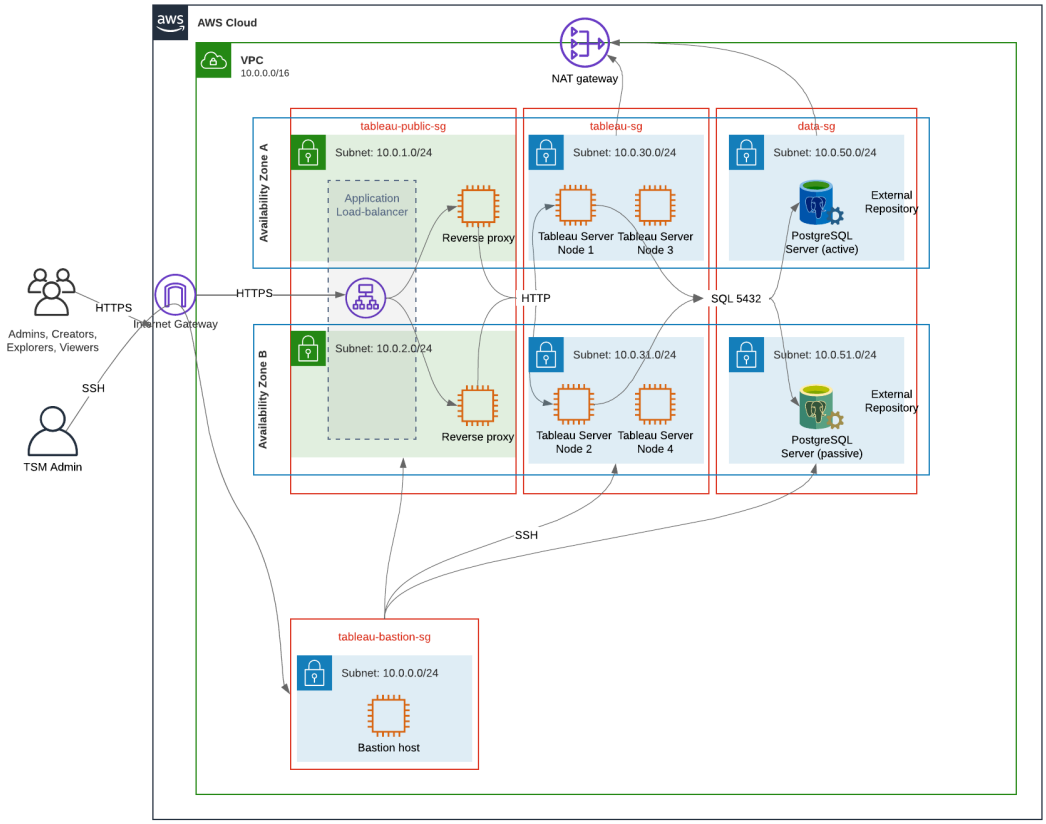
目标

此处提供的 Terraform 模板和内容旨在提供一个工作示例，使您能够在开发测试环境中快速部署 EDG。

我们已尽最大努力测试和记录示例 Terraform 部署。但是，使用 Terraform 在生产环境中部署和维护 EDG 需要 Terraform 专业知识，这超出了本示例的范围。Tableau 不为此处记录的示例 Terraform 解决方案提供支持。

结束状态

按照本部分的过程在 AWS 中设置 VPC，该 VPC 的功能等同于第 3 部分 - 准备 Tableau Server 企业部署中指定的 VPC。



本部分中的示例 Terraform 模板和支持内容：

- 按上文所示方式创建具有弹性 IP 地址、两个可用区和子网组织的 VPC(IP 地址不同)
- 创建 Bastion、Public、Private 和 Data 安全组。

- 在安全组上设置大多数入口和出口规则。在 Terraform 运行后，您将需要编辑安全组。
- 创建以下 EC2 主机（每个都运行 AWS Linux2）：堡垒、代理 1 代理 2、Tableau 节点 1、Tableau 节点 2、Tableau 节点 3、Tableau 节点 4。
- 未创建 PostgreSQL 的 EC2 主机。您必须在 Data 安全组中手动创建 EC2，然后按照安装、配置 PostgreSQL 和创建 PostgreSQL 的 tar 备份中所述安装和配置 PostgreSQL。

要求

- AWS 帐户 - 您必须有权访问允许您创建 VPC 的 AWS 账户。
- 如果您从 Windows 计算机运行 Terraform，则需要安装 AWS CLI。
- 您的 AWS 帐户中可用的弹性 IP 地址。
- 在 AWS Route 53 中注册的域。Terraform 将在 Route 53 中创建一个 DNS 区域和相关的 SSL 证书。因此，运行 Terraform 的配置文件也必须在 Route 53 中具有适当的权限。

开始之前

- 此过程中的命令行示例适用于带有 Apple OS 的终端。如果您在 Windows 上运行 Terraform，您可能需要根据需要调整带有文件路径的命令。
- Terraform 项目由许多文本配置文件（.tf 文件扩展名）组成。您可以通过自定义这些文件来配置 Terraform。如果您没有强大的文本编辑器，请安装 Atom 或 Text++。
- 如果您要与他人共享 Terraform 项目，我们建议将项目存储在 Git 中以进行变更管理。

步骤 1 - 准备环境

A. 下载并安装 Terraform

https://www.terraform.io/downloads

B. 生成公私钥对

这是您将用于访问 AWS 和生成的 VPC 环境的密钥。当您运行 Terraform 时，您将包括公钥。

打开终端并运行以下命令：

1. Create a private key. For example, `my-key.pem`:

   ```
   openssl genrsa -out my-key.pem 1024
   ```

2. 创建公钥。此密钥格式不用于 Terraform。您稍后将在此过程中将其转换为 ssh 密钥：

   ```
   openssl rsa -in my-key.pem -pubout > my-key.pub
   ```

3. 设置私钥的权限：

   ```
   sudo chmod 0600 my-key.pem
   ```

   在 Windows 上设置权限：

   - 在 Windows 资源管理器中找到该文件，右键单击它，然后选择**“属性”s**。导航到**“安全”**选项卡，然后单击**“高级”**。
   - 将所有者更改为您，禁用继承并删除所有权限。授予自己**完全控制权**，然后单击**“保存”**。将文件标记为只读。

4. 创建 ssh 公钥。这是稍后您将复制到 Terraform 中的密钥。

   ```
   ssh-keygen -y -f my-key.pem >my-key-ssh.pub
   ```

C. 下载项目并添加 state 目录

1. 下载并解压缩 EDG Terraform 项目，并将它们保存到本地计算机。解压缩下载的项目后，您将拥有一个顶级目录 edg-terraform 和一系列子目录。

2. 创建一个名为 `state` 的目录，作为顶级 `edg-terraform` 目录的对等目录。

步骤 2: 自定义 Terraform 模板

您必须自定义 Terraform 模板以适合您的 AWS 和 EDG 环境。此处的示例提供了大多数组织需要进行的最少模板自定义。您的特定环境可能需要其他定制。

本部分按模板名称组织。

在继续执行*"步骤 3 - 运行 Terraform"*之前，请务必保存所有更改。

versions.tf

There are three instances of `versions.tf` files where the `required_version` field must match the version of terraform.exe you're using. Check the version of terraform (`terraform.exe -version`) and update each of the following instances:

- `edg-terraform\versions.tf`
- `edg-terraform\modules\proxy\versions.tf`
- `edg-terraform\modules\tableau_instance\versions.tf`

key-pair.tf

1. 打开您在步骤 1B 中生成的公钥并复制该密钥：

   ```
   less my-key-ssh.pub
   ```

   Windows：复制公钥的内容。

2. 将公钥字符串复制到 public_key 参数中，例如：

   ```
   resource "aws_key_pair" "tableau" {
   key_name = "my-key"
   public_key = "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQ (truncated
   example) dZVHambOCw=="
   ```

Ensure that the `key_name` value is unique in the datacenter or `terraform apply` will fail.

locals.tf

Update `user.owner` to your name or alias. The value you enter here will be used for the "Name" tag in AWS on the resources that Terraform creates.

providers.tf

1. 根据组织的要求添加标签。例如：

```
default_tags {
  tags = {

   "Application" = "tableau",
   "Creator" = "alias@example.com",
   "DeptCode" = "8675309",
   "Description" = "EDG",
   "Environment" = "test",
   "Group" = "itcloud@example.com"
  }
}
```

2. If using `provider`, comment out the `assume_role` lines:

```
/* assume_role {
role_arn     = "arn:aws:iam::310946706895:role/terraform-
backend"
session_name = "terraform"
}*/
```

elb.tf

Under `'resource "aws_lb" "tableau" {'` choose a unique value to use for `name` and `tags.Name`.

If another AWS load balancer has the same name in the datacenter, then `terraform apply` will fail.

Add `idle_timeout`:

```
resource "aws_lb" "tableau" {
name                       = "edg-again-alb"
load_balancer_type         = "application"
subnets                    = [for subnet in aws_subnet.public :
```

```
subnet.id]
security_groups            = [aws_security_group.public.id]
drop_invalid_header_fields = true
idle_timeout               = 400
tags = {
Name = "edg-again-alb"
}
                       }
```

variables.tf

更新根域名。此名称必须与您在 Route 53 中注册的域相匹配。

```
variable "root_domain_name" {
 default = "example.com"
 }
```

默认情况下，将为 VPC DNS 域名指定子域 `tableau`。若要更改此设置，请更新 `subdomain`：

```
variable "subdomain" {
 default = "tableau"
}
```

modules/tableau_instance/ec2.tf

There are two ec2.tf files in the project. This customization is for the Tableau instance of the ec2.tf in the directory: `modules/tableau_instance/ec2.tf`.

- 如果需要，添加标签 blob：

  ```
  tags = {
   "Name" : var.ec2_name,
   "user.owner" = "ALIAS",
   "Application" = "tableau",
   "Creator" = "ALIAS@example.com",
   "DeptCode" = "8675309",
   "Description" = "EDG",
  ```

```
 "Environment" = "test",
 "Group" = "itcloud@example.com"
 }
}
```

- 根据需要，可选择更新您的存储以处理您的数据要求：

  根卷：

  ```
  root_block_device {
   volume_size = 100
   volume_type = "gp3"
  }
  ```

  应用程序卷：

  ```
  resource "aws_ebs_volume" "tableau" {
   availability_zone = data.aws_subnet.tableau.availability_zone
   size              = 500
   type              = "gp3"
  }
  ```

步骤 3 - 运行 Terraform

A. 初始化 Terraform

在终端中，切换到 `edg-terraform` 目录并运行以下命令：

```
terraform init
```

如果初始化成功，继续执行下一步。如果初始化失败，请按照 Terraform 输出中的说明进行操作。

B. 规划 Terraform

从同一目录中，运行 plan 命令：

```
terraform plan
```

该命令可以多次运行。根据需要运行多次以修复错误。当此命令运行无错误时，继续执行下一步。

C. 应用 Terraform

从同一目录中，运行 apply 命令：

```
terraform apply
```

Terraform will prompt you to verify deployment, type `Yes`.

可选：销毁 Terraform

您可以通过运行 destroy 命令销毁整个 VPC：

```
terraform destroy
```

destroy 命令只会销毁它创建的内容。如果您对 AWS 中的某些对象（即安全组、子网等）进行了手动更改，则 `destroy` 将失败。若要退出失败/挂起的销毁操作，请键入 `Control + C`。然后，您必须手动将 VPC 清理到 Terraform 最初创建它时的状态。然后，您可以运行 `destroy` 命令。

步骤 4 - 连接到堡垒

所有命令行连接都通过 TCP 22（SSH 协议）上的堡垒主机进行。

1. 在 AWS 中，在 bastion 安全组中创建一个入站规则（**“AWS”>“Security Groups”（安全组）>“Bastion SG”（堡垒 SG）>“Edit inbound rules”（编辑入站规则）**），并创建一个规则以允许来自将运行终端命令的 IP 地址或子网掩码的 SSH (TCP 22) 连接.

   可选：您可能会发现，在部署期间允许在 Private 组和 Public 组中的 EC2 实例之间复制文件很有帮助。创建入站 SSH 规则：

   - Private：创建入站规则以允许来自 Public 的 SSH
   - Public：创建入站规则以允许来自 Private 和 Public 的 SSH

2. 使用您在步骤 1.B 中创建的 pem 密钥连接到堡垒主机：

在 Mac 终端上：

从存储 pem 密钥的目录运行以下命令：

```
ssh-add —apple-use-keychain <keyName>.pem
```

If you get a warning about private key being accessible by others, then run this command: `chmod 600 <keyName.pem>` and then run the `ssh-add` command again.

Connect to the bastion host with this command: `ssh -A ec2-user@IPaddress`

For example: `ssh -A ec2-user@3.15.12.112.`

在 Windows 上，使用 PuTTY 和 Pageant：

a. 从 pem 密钥创建 ppk：使用 PuTTY 密钥生成器。打开您在步骤 1.B 中创建的 pem 密钥。密钥导入后，点击**“Save private key”（保存私钥）**。这将创建一个 ppk 文件。

b. 在 PuTTY 中 - 打开配置并进行以下更改：

- “Sessions”（会话）>“Host Name”（主机名）：添加堡垒机的 IP 地址。
- “Sessions”（会话）>“Port”（端口）：22
- “Connection”（连接）>“Data”（数据）>“Auto-login username”（自动登录用户名）：ec2-user
- “Connection”（连接）>”SSH”>“Auth”（授权）>“Allow agent forwarding”（允许代理转发）
- “Connection”（连接）>“SSH”>“Auth”（授权）> 对于私钥，单击“Browse”（浏览）并选择您刚刚创建的 .ppk 文件。

c. 安装 Pageant 并将 ppk 加载到应用程序中。

步骤 5：安装 PostgreSQL

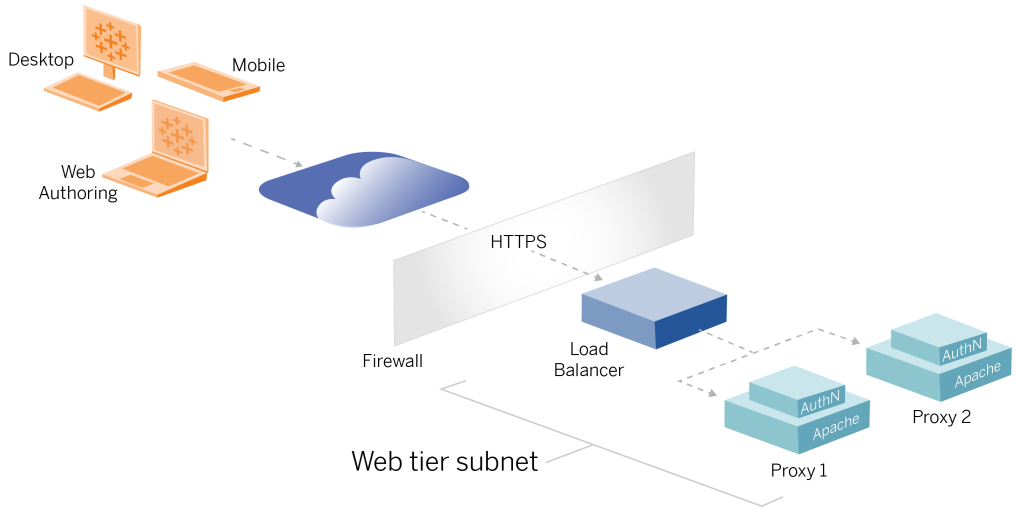
Terraform 模板不安装 PostgreSQL 用作外部存储库。但是，会创建关联的安全组和子网。如果您要在运行 PostgreSQL 的 EC2 实例上安装外部存储库，则必须按照第 3 部分 - 准备 Tableau Server 企业部署中所述部署 EC2 实例。

然后安装、配置 PostgreSQL 并创建 PostgreSQL 的 tar 备份，如第 4 部分 - 安装并配置 Tableau Server中所述。

步骤 6 -(可选) 运行 DeployTab4EDG

TabDeploy4EDG 会自动执行第 4 部分中描述的四节点 Tableau 部署的实施。请参见TabDeploy4EDG 自动安装脚本。

附录 - 具有 Apache 示例部署的 Web 层



本主题提供了一个端到端过程，该过程描述了如何在示例 AWS 参考架构中实现 Web 层：示例配置由以下组件组成：

- AWS 应用程序负载平衡器
- Apache 代理服务器
- Mellon 身份验证模块
- Okta IdP
- SAML 身份验证

注意：本部分中提供的示例 Web 层配置包括部署第三方软件和服务的详细过程。我们已尽最大努力验证和记录启用 Web 层方案的过程。但是，第三方软件可能会发生变化，或者您的方案可能与此处描述的参考架构不同。请参考第三方文档以获得权威的配置详细信息和支持。

此部分中的 Linux 示例显示了类似 RHEL 的发行版的命令。具体来说，这里的命令是使用 Amazon Linux 2 发行版开发的。如果您运行的是 Ubuntu 发行版，请相应地编辑命令。

在此示例中部署 Web 层遵循逐步配置和验证过程。核心 Web 层配置包含以下用于在 Tableau 和 Internet 之间启用 HTTP 的步骤。Apache 运行并配置为在 AWS 应用程序负载平衡器后面进行反向代理/负载平衡：

1. 安装 Apache
2. 配置反向代理以测试与 Tableau Server 的连接
3. 在代理上配置负载平衡
4. 配置 AWS 应用程序负载平衡器

设置 Web 层并验证与 Tableau 的连接后，使用外部提供程序配置身份验证。

安装 Apache

在两个 EC2 主机(代理 1 和代理 2)上运行以下过程。如果您根据参考架构示例在 AWS 中进行部署，那么您应该有两个可用区并在每个可用区中运行单个代理服务器。。

1. 安装 Apache：

   ```
   sudo yum update -y
   sudo yum install -y httpd
   ```

2. 配置在重启时启动 Apache：

   ```
   sudo systemctl enable --now httpd
   ```

3. 验证您安装的 httpd 版本是否包括 `proxy_hcheck_module`：

   ```
   sudo httpd -M
   ```

   `proxy_hcheck_module` 为必需。如果您的 httpd 版本不包含此模块，则更新到包含它的 httpd 版本。

配置代理以测试与 Tableau Server 的连接

在其中一台代理主机(代理 1)上运行此过程。此步骤的目的是验证 Internet 到您的代理服务器与私有安全组中的 Tableau Server 之间的连接。

1. 创建一个名为 tableau.conf 的文件，并将其添加到 /etc/httpd/conf.d 目录。

   复制以下代码，并使用 Tableau Server 节点 1 的专用 IP 地址指定 ProxyPass 和 ProxyPassReverse 键。

   > **重要信息**：下面显示的配置不安全，不应在生产中使用。此配置仅应在安装过程中用于验证端到端连接。

   举例来说，如果节点 1 的专用 IP 地址是 10.0.30.32，则 tableau.conf 文件的内容将为：

   ```
   <VirtualHost *:80>
   ProxyPreserveHost On
   ProxyPass "/" "http://10.0.30.32:80/"
   ProxyPassReverse "/" "http://10.0.30.32:80/"
   </VirtualHost>
   ```

2. 重启 httpd：

   ```
   sudo systemctl restart httpd
   ```

验证：基本拓扑配置

您应该能够通过浏览到 http://<proxy-public-IP-address> 来访问 Tableau Server 管理员页面。

如果您的浏览器中未加载 Tableau Server 登录页面，请在代理 1 主机上执行以下故障排除步骤：

- 作为故障排除的第一步，停止然后启动 httpd。
- 仔细检查 tableau.conf 文件。验证节点 1 专用 IP 是否正确。验证双引号并仔细检查语法。
- 使用节点 1 专用 IP 地址在反向代理服务器上运行 curl 命令，例如 curl 10.0.1.90。如果 shell 不返回 html，或者如果返回 Apache 测试网页的 html，请验证“公共”和“专用”安全组之间的协议/端口配置。

- 使用代理 1 专用 IP 地址运行 `curl` 命令，例如 `curl 10.0.0.163`。如果 shell 返回 Apache 测试网页的 html 代码，则说明代理文件配置不正确。
- 在对代理文件或安全组进行任何配置更改之后，请始终停止/启动 httpd (`sudo systemctl restart httpd`)。
- 确保 TSM 在节点 1 上运行。

在代理上配置负载平衡

1. 在您创建 `tableau.conf` 文件的同一代理主机(代理 1)上，移除现有的虚拟主机配置，并编辑文件以包含负载平衡逻辑。

   例如：

   ```
   <VirtualHost *:80>
   ServerAdmin admin@example.com
   #Load balancing logic.
   ProxyHCExpr ok234 {%{REQUEST_STATUS} =~ /^[234]/}
   Header add Set-Cookie "ROUTEID=.%{BALANCER_WORKER_ROUTE}e;
   path=/" env=BALANCER_ROUTE_CHANGED
   <Proxy balancer://tableau>
   #Replace IP addresses below with the IP addresses to the
   Tableau Servers running the Gateway service.
   BalancerMember http://10.0.3.40/ route=1 hcmethod=GET
   hcexpr=ok234 hcuri=/favicon.ico
   BalancerMember http://10.0.4.151/ route=2 hcmethod=GET
   hcexpr=ok234 hcuri=/favicon.ico
   ProxySet stickysession=ROUTEID
   </Proxy>
   ProxyPreserveHost On
   ProxyPass / balancer://tableau/
   ProxyPassReverse / balancer://tableau/
   </VirtualHost>
   ```

2. 停止然后启动 httpd：

```
sudo systemctl stop httpd
sudo systemctl start httpd
```

3. 通过浏览到代理 1 的公共 IP 地址来验证配置。

将配置复制到第二个代理服务器

1. 从代理 1 中复制 `tableau.conf` 文件，并将其保存到代理 2 主机上的 `/etc/httpd/conf.d` 目录。
2. 停止然后启动 httpd：

```
sudo systemctl stop httpd
sudo systemctl start httpd
```

3. 通过浏览到代理 2 的公共 IP 地址来验证配置。

配置 AWS 应用程序负载平衡器

将负载平衡器配置为 HTTP 侦听器。此处的过程描述了如何在 AWS 中添加负载平衡器。

步骤 1：创建目标组

标组是定义运行代理服务器的 EC2 实例的 AWS 配置。这些是来自 LBS 的流量的目标。

1. **“EC2”>“目标组”>“创建目标组”**
2. 在“创建”页面上：
   - 输入目标组名称，例如 `TG-internal-HTTP`
   - 目标类型：实例
   - 协议：HTTP
   - 端口：80
   - VPC：选择您的 VPC
   - 在**“运行状况检查”>“高级运行状况检查设置”>“成功代码”**下，将代码列表

追加为:200,303。
- 单击**“创建”**

3. 选择刚刚创建的目标组,然后单击**“目标”**选项卡:
    - 单击**“编辑”**。
    - 选择正在运行代理应用程序的 EC2 实例(或单个实例,如果您一次配置一个),然后单击**“添加到已注册”**。
    - 单击**“保存”**。

步骤 2:启动负载平衡器向导

1. “EC2”>**“负载平衡器”>“创建负载平衡器”**

2. 在“选择负载平衡器类型”页面上,创建一个应用程序负载平衡器。

注意:为配置负载均衡器而显示的 UI 在 AWS 数据中心之间不一致。下面的过程“向导配置”映射到从**“步骤 1 配置负载均衡器”**开始的 AWS 配置向导。

如果您的数据中心在页面底部包含**“创建负载均衡器”**按钮的单个页面中显示所有配置,请按照下面的“单页面配置”过程进行操作。

向导配置

1. **“配置负载平衡器”**页面:
    - 指定名称
    - 方案:面向互联网(默认)
    - IP 地址类型:ip1v4(默认)
    - 侦听器(侦听器和路由):
        a. 保留默认的 HTTP 侦听器
        b. 单击**“添加侦听器”**并添加 HTTPS:443

- VPC：选择您已安装所有内容的 VPC
- 可用区：
  - 为您的数据中心区域选择 **a** 和 **b**
  - 在每个相应的下拉选择器中，选择 Public 子网（您的代理服务器所在的位置）。
- 单击：**“配置安全设置”**

2. **“配置安全设置”**页面
   - 上载您的公共 SSL 证书。
   - 单击**“下一步：配置安全组”**。
3. **“配置安全组”**页面：
   - 选择 Public 安全组。如果选择了“默认”安全组，则清除该选择。
   - 单击**“下一步：配置路由”**。
4. **“配置路由”**页面
   - 目标组：现有目标组。
   - 名称：选择您之前创建的目标组
   - 单击**“下一步：注册目标”**。
5. **“注册目标”**页面
   - 应会显示您之前配置的两个代理服务器实例。
   - 单击**“下一步：审查”**。
6. **“查看”**页面

   单击**“创建”**。

单页面配置

基本配置

- 指定名称
- 方案：面向互联网（默认）
- IP 地址类型：ip1v4（默认）

网络映射

- VPC:选择您已安装所有内容的 VPC
- 映射:
  - 为您的数据中心区域选择 **a** 和 **b**(或类似的) 可用区
  - 在每个相应的下拉选择器中, 选择 Public 子网(您的代理服务器所在的位置) 。

安全组

选择 Public 安全组。如果选择了“默认”安全组, 则清除该选择。

侦听器和路由

- 保留默认的 HTTP 侦听器。对于**“默认操作”**, 指定您之前设置的目标组。
- 单击**“添加侦听器”**并添加 `HTTPS:443`。对于**“默认操作”**, 指定您之前设置的目标组。

安全侦听器设置

- 上载您的公共 SSL 证书。

单击**“创建负载平衡器”**。

步骤 3:启用粘性

1. 创建负载平衡器后, 您必须在目标组上启用粘性。

   - 打开 AWS 目标组页面(**“EC2”>“负载平衡”>“目标组”**) , 选择您刚刚设置的目标组实例。在**“操作”**菜单上, 选择**“编辑属性”**。
   - 在**“编辑属性”**页面上, 选择**“粘性”** , 指定持续时间 `1 day`, 然后**保存更改**。

2. 在负载平衡器上, 在 HTTP 侦听器上启用粘性。选择您刚刚配置的负载平衡器, 然后单击**“侦听器”**选项卡:

   - 对于**“HTTP:80”**, 单击**“查看/编辑规则”**。在生成的**“规则”**页面上, 单击编辑图标(一次位于页面顶部, 然后再次位于规则旁边) 以编辑规则。删除现有 THEN 规则, 并通过单击**“添加操作”>“转发至...”**替换该规则。在生成的 THEN 配置中, 指定您创建的相同目标组。在“组级粘性”下, 启用粘性并将持续时间设置为 1 天。保存设置, 然后单击**“更新”**。

步骤 4:在负载平衡器上设置空闲超时

在负载平衡器上,将空闲超时更新为 400 秒。

选择您为此部署配置的负载平衡器,然后单击**"操作">"编辑属性"**。将**"空闲超时"**设置为 400 秒,然后单击**"保存"**。

步骤 5:验证 LBS 连接

打开 AWS 负载平衡器页面(**"EC2">"负载平衡器"**),选择您刚刚设置的负载平衡器实例。

在**"描述"**下,复制 DNS 名称并将其粘贴到浏览器中以访问 Tableau Server 登录页面。

如果您收到 500 级错误,那么您可能需要重新启动代理服务器。

使用公共 Tableau URL 更新 DNS

使用 AWS 负载平衡器描述中的域 DNS 区域名称在您的 DNS 中创建 CNAME 值。流向您的 URL (tableau.example.com) 的流量应发送到 AWS 公共 DNS 名称。

验证连接

DNS 更新完成后,您应该能够通过输入您的公共 URL(例如 `https://tableau.example.com`)导航到 Tableau Server 登录页面。

示例身份验证配置:带有外部 IdP 的 SAML

以下示例介绍如何使用 Okta IdP 和 Mellon 身份验证模块为在 AWS 参考架构中运行的 Tableau 部署设置和配置 SAML。该示例介绍了如何配置 Tableau Server 和 Apache 代理服务器以使用 HTTP。Okta 将通过 HTTPS 向 AWS 负载平衡器发送请求,但所有内部流量都将通过 HTTP 传输。针对此场景进行配置时,请在设置 URL 字符串时注意 HTTP 与 HTTPS 协议。

此示例使用 Mellon 作为反向代理服务器上的预身份验证服务提供程序模块。此配置可确保只有经过身份验证的流量连接到 Tableau Server，Tableau Server 还充当 Okta IdP 的服务提供商。因此，您必须配置两个 IdP 应用程序：一个用于 Mellon 服务提供商，另一个用于 Tableau 服务提供商。

创建 Tableau 管理员帐户

配置 SAML 时的一个常见错误是在启用 SSO 之前忘记在 Tableau Server 上创建管理员帐户。

第一步是在 Tableau Server 上创建一个具有服务器管理员角色的帐户。对于示例 Okta 场景，用户名必须采用有效电子邮件地址格式，例如 user@example.com。您必须为此用户设置密码，但在配置 SAML 后将不再使用该密码。

配置 Okta 预身份验证应用程序

本部分描述的端到端场景需要两个 Okta 应用程序：

- Okta 预认证应用程序
- Okta Tableau Server 应用程序

这些应用程序中的每一个都与您需要分别在反向代理和 Tableau Server 上配置的不同元数据相关联。

此过程描述了如何创建和配置 Okta 预身份验证应用程序。在本主题的后面，您将创建 Okta Tableau Server 应用程序。有关用户受限的免费测试 Okta 帐户，请参见 Okta 开发人员网页。

为 Mellon 预身份验证服务提供商创建 SAML 应用程序集成。

1. 打开 Okta 管理仪表板 >**“应用程序”**>**“创建应用程序集成”**。
2. 在**“创建新的应用程序集成”**页面上，选择**“SAML 2.0”**，然后单击**“下一步”**。

3. 在**“常规设置”**选项卡上，输入应用程序名称（例如 Tableau Pre-Auth），然后单击**“下一步”**。

4. 在**“配置 SAML”**选项卡上：

   - 单点登录 (SSO) URL。单点登录 URL 中路径的最后一个元素在 mellon.conf 配置文件称为 MellonEndpointPath，该配置文件将在本过程的后面进行介绍。您可以指定您想要的任何端点。在此示例中，sso 是端点。最后一个元素 postResponse 是必须的：https://tableau.example.com/sso/postResponse。
   - 清除复选框：**“将此用于收件人 URL 和目标 URL”**。
   - 收件人 URL：与 SSO URL 相同，但使用 HTTP。例如，http://tableau.example.com/sso/postResponse。
   - 目标 URL：与 SSO URL 相同，但带有 HTTP。例如，http://tableau.example.com/sso/postResponse。
   - 受众 URI（SP 实体 ID）。例如，https://tableau.example.com。
   - 名称 ID 格式：EmailAddress
   - 应用程序用户名：Email
   - 属性声明：名称 = mail；名称格式 = Unspecified；值 = user.email。

   单击**“下一步”**。

5. 在**“反馈”**选项卡上，选择：

   - **我是添加内部应用程序的 Okta 客户**
   - **这是我们创建的内部应用程序**
   - 单击**“完成”**。

6. 创建预授权 IdP 元数据文件：

   - 在 Okta 中：**“Applications”（应用程序）>“Applications”（应用程序）>** 您的新应用程序（例如，Tableau Pre-Auth）**>“Sign On”（登录）**
   - 在**“SAML 签名证书”**旁边，单击**“查看 SAML 设置说明”**。
   - 在**“如何针对 <预授权> 应用程序配置 SAML 2.0”**页面上，向下滚动到**“可选”**部分，**向您的 SP 提供商提供以下 IDP 元数据**。
   - 复制 XML 字段的内容并将它们保存在一个名为 pre-auth_idp_metadata.xml 的文件中。

7. （可选）配置多重身份验证：

- 在 Okta 中：**“Applications”（应用程序）>“Applications”（应用程序）>** 您的新应用程序（例如，`Tableau Pre-Auth`）**>“Sign On”（登录）**
- 在**“Sign On Policy”（登录策略）**下，单击**“Add Rule”（添加规则）**。
- 在**“App Sign On Rule”（应用程序登录规则）**上，指定名称和不同的 MFA 选项。若要测试功能，您可以将所有选项保留为默认值。但是，在**“Actions”（操作）**下，您必须选择**“Prompt for factor”（因素提示）**，然后指定用户必须登录的频率。单击**“Save”（保存）**。

创建和分配 Okta 用户

1. 在 Okta 中，使用您在 Tableau 中创建的相同用户名 (user@example.com) 创建一个用户：**“Directory”（目录）>“People”（人员）>“Add person”（添加人员）**。
2. 创建用户后，将新的 Okta 应用程序分配给该人员：单击用户名，然后在**“Assign Application”（分配应用程序）**中分配应用程序。

安装 Mellon 进行预身份验证

1. 在运行 Apache 代理服务器的 EC2 实例上，运行以下命令以安装 PHP 和 Mellon 模块：

   ```
   sudo yum install httpd php mod_auth_mellon
   ```

2. 创建 `/etc/httpd/mellon` 目录

将 Mellon 配置为预身份验证模块

在两个代理服务器上运行此过程。

您必须有依据 Okta 配置创建的 `pre-auth_idp_metadata.xml` 文件的副本。

1. 更改目录：

   ```
   cd /etc/httpd/mellon
   ```

2. 创建服务提供程序元数据。运行 mellon_create_metadata.sh 脚本。您必须在命令中包含您组织的实体 ID 和返回 URL。

   返回 URL 在 Okta 中称为*单点登录 URL*。返回 URL 中路径的最后一个元素在 mellon.conf 配置文件中称为 MellonEndpointPath，该配置文件将在本过程的后面进行介绍。在此示例中，我们指定 sso 作为端点路径。

   例如：

   ```
   sudo /usr/libexec/mod_auth_mellon/mellon_create_metadata.sh
   https://tableau.example.com "https://tableau.example.com/sso"
   ```

   该脚本返回服务提供商证书、密钥和元数据文件。

3. 重命名 mellon 目录中的服务提供商文件以便于阅读。我们将在文档中使用以下名称来引用这些文件：

   ```
   sudo mv *.key mellon.key
   sudo mv *.cert mellon.cert
   sudo mv *.xml sp_metadata.xml
   ```

4. 将 pre-auth_idp_metadata.xml 文件复制到同一文件夹。

5. 在 /etc/httpd/conf.d 目录中创建 mellon.conf 文件：

   ```
   sudo nano /etc/httpd/conf.d/mellon.conf
   ```

6. 将以下内容复制到 mellon.conf 中。

   ```
   <Location />
   MellonSPPrivateKeyFile /etc/httpd/mellon/mellon.key
   MellonSPCertFile /etc/httpd/mellon/mellon.cert
   MellonSPMetadataFile /etc/httpd/mellon/sp_metadata.xml
   MellonIdPMetadataFile /etc/httpd/mellon/pre-auth_idp_
   metadata.xml
   MellonEndpointPath /sso
   ```

```
MellonEnable "info"
</Location>
```

7. 将以下内容添加到现有 tableau.conf 文件中：

   在 <VirtualHost *:80> 块内部，添加以下内容。使用实体 ID 中的公共主机名更新 ServerName ：

   ```
   DocumentRoot /var/www/html
   ServerName tableau.example.com
   ServerSignature Off
   ErrorLog logs/error_sp.log
   CustomLog logs/access_sp.log combined
   LogLevel info
   ```

   在 <VirtualHost *:80> 块外部添加 Location 块。使用顶级域更新 MellonCookieDomain 以保存 Cookie 信息，如下所示：

   ```
   <Location />
   AuthType Mellon
   MellonEnable auth
   Require valid-user
   MellonCookieDomain example.com
   </Location>
   ```

   完整的 tableau.conf 文件看起来应类似于以下示例：

   ```
   <VirtualHost *:80>
   ServerAdmin admin@example.com
   ProxyHCExpr ok234 {%{REQUEST_STATUS} =~ /^[234]/}
   Header add Set-Cookie "ROUTEID=.%{BALANCER_WORKER_ROUTE}e;
   path=/" env=BALANCER_ROUTE_CHANGED
   <Proxy balancer://tableau>
   BalancerMember http://10.0.3.36/ route=1 hcmethod=GET
   hcexpr=ok234 hcuri=/favicon.ico
   BalancerMember http://10.0.4.15/ route=2 hcmethod=GET
   hcexpr=ok234 hcuri=/favicon.ico
   ```

```
ProxySet stickysession=ROUTEID
</Proxy>
ProxyPreserveHost On
ProxyPass / balancer://tableau/
ProxyPassReverse / balancer://tableau/
DocumentRoot /var/www/html
ServerName tableau.example.com
ServerSignature Off
ErrorLog logs/error_sp.log
CustomLog logs/access_sp.log combined
LogLevel info
</VirtualHost>
<Location />
AuthType Mellon
MellonEnable auth
Require valid-user
MellonCookieDomain example.com
</Location>
```

8. 验证配置。运行以下命令：

   ```
   sudo apachectl configtest
   ```

   如果配置测试返回错误，请修复所有错误并再次运行 configtest。一个成功的配置将返回，`Syntax OK`。

9. 重启 httpd：

   ```
   sudo systemctl restart httpd
   ```

在 Okta 中创建 Tableau Server 应用程序

1. 在 Okta 仪表板中：**“Applications”（应用程序）>“Applications”（应用程序）>“Browse App Catalog”（浏览应用程序目录）**
2. 在**“Browse App Integration Catalog”（浏览应用程序集成目录）**中，搜索 `Tableau`，选择“Tableau Server”磁贴，然后单击**“Add”（添加）**。

3. 在**“Add Tableau Server”(添加 Tableau Server) >“General Settings”(常规设置)”**上，输入标签，然后单击**“Next”(下一步)**。
4. 在“Sign-On Options”(登录选项)中，选择**“SAML 2.0”**，然后向下滚动到“Advanced Sign-on Settings”(高级登录设置)：
   - **“SAML Entity ID”(SAML 实体 ID)**：输入公共 URL，例如 https://tableau.example.com。
   - **“Application user name format”(应用程序用户名格式)**：“Email”(电子邮件)
5. 单击**“Identity Provider metadata”(身份提供程序元数据)**链接以启动浏览器。复制浏览器链接。这是您在以下过程中配置 Tableau 时将使用的链接。
6. 单击**“Done”(完成)**。
7. 将新的 Tableau Server Okta 应用程序分配给您的用户 (user@example.com)：单击用户名，然后在**“Assign Application”(分配应用程序)**中分配应用程序。

在 Tableau Server 上为 IdP 启用 SAML

在 Tableau Server 节点 1 上运行此过程。

1. 从 Okta 下载 Tableau Server 应用程序元数据。使用您在上一过程中保存的链接：

   ```
   wget https://dev-66144217.okta.com/app/exk1egxgt1fhjkSeS5d7/sso/saml/metadata -O idp_metadata.xml
   ```

2. 将 TLS 证书和相关密钥文件复制到 Tableau Server。密钥文件必须是 RSA 密钥。有关 SAML 证书和 IdP 要求的详细信息，请参见“*SAML 要求* (Linux)”。

   为了简化证书管理和部署，以及作为安全最佳实践，我们建议使用由主要受信任的第三方证书颁发机构 (CA) 生成的证书。或者，您可以为 TLS 生成自签名证书或使用 PKI 中的证书。

   如果您没有 TLS 证书，则可以使用下面的嵌入式过程生成自签名证书。

生成自签名证书

在 Tableau Server 节点 1 上运行此过程。

a. 生成签名根证书颁发机构 (CA) 密钥：

```
openssl genrsa -out rootCAKey-saml.pem 2048
```

b. 创建根 CA 证书：

```
openssl req -x509 -sha256 -new -nodes -key rootCAKey-
saml.pem -days 3650 -out rootCACert-saml.pem
```

系统将提示您输入证书字段的值。例如：

```
Country Name (2 letter code) [XX]:US
State or Province Name (full name) []:Washington
Locality Name (eg, city) [Default City]:Seattle
Organization Name (eg, company) [Default Company
Ltd]:Tableau
Organizational Unit Name (eg, section) []:Operations
Common Name (eg, your name or your server's hostname)
[]:tableau.example.com
Email Address []:example@tableau.com
```

c. 创建证书和相关密钥(在下面的示例中为 server-saml.csr 和 server-saml.key)。证书的使用者名称必须与 Tableau 主机的公共主机名称匹配。使用者名称是使用 -subj 选项以"/CN=<host-name>"格式设置的，例如：

```
openssl req -new -nodes -text -out server-saml.csr -keyout
server-saml.key -subj "/CN=tableau.example.com"
```

d. 使用您在上面创建的 CA 证书签署新证书。以下命令还将以 crt 格式输出证书：

```
openssl x509 -req -in server-saml.csr -days 3650 -CA
rootCACert-saml.pem -CAkey rootCAKey-saml.pem -
```

```
CAcreateserial -out server-saml.crt
```

e. 将密钥文件转换为 RSA。Tableau 需要用于 SAML 的 RSA 密钥文件。若要转换密钥，请运行以下命令：

```
openssl rsa -in server-saml.key -out server-saml-rsa.key
```

3. 配置 SAML。运行以下命令，指定您的实体 ID 和返回 URL，以及元数据文件、证书文件和密钥文件的路径：

```
tsm authentication saml configure --idp-entity-id
"https://tableau.example.com" --idp-return-url
"https://tableau.example.com" --idp-metadata idp_metadata.xml -
-cert-file "server-saml.crt" --key-file "server-saml-rsa.key"

tsm authentication saml enable
```

4. 如果您的组织运行的是 Tableau Desktop 2021.4 或更高版本，则您必须运行以下命令以通过反向代理服务器启用身份验证。

   Tableau Desktop 2021.2.1 - 2021.3 的版本无需运行此命令也可工作，前提是您的预身份验证模块（例如 Mellon）配置为允许保留顶级域 Cookie。

```
tsm configuration set -k features.ExternalBrowserOAuth -v false
```

5. 应用配置更改：

```
tsm pending-changes apply
```

验证 SAML 功能

若要验证端到端 SAML 功能，请使用您在此过程开始时创建的 Tableau 管理员帐户通过公共 URL（例如 https://tableau.example.com）登录到 Tableau Server。

验证故障排除

错误请求：这种情况下的一个常见错误是来自 Okta 的“错误请求”错误。当浏览器缓存来自先前 Okta 会话的数据时，通常会发生此问题。举例来说，如果您以 Okta 管理员身份管理 Okta 应用程序，然后尝试使用不同的启用 Okta 的帐户访问 Tableau，则来自管理员数据的会话数据可能会导致“错误请求”错误。如果即使在您清除本地浏览器缓存后此错误仍然存在，请尝试通过连接其他浏览器来验证 Tableau 方案。

“错误请求”错误的另一个原因是您在 Okta、Mellon 和 SAML 配置过程中输入的许多 URL 之一中的拼写错误。仔细检查所有这些 URL。

通常是 Apache 服务器上的 httpd `error.log` 文件将指定导致错误的 URL。

未找到 - 在此服务器上找不到请求的 URL：此错误表示许多配置错误之一。

如果用户使用 Okta 进行身份验证，然后收到此错误，则很可能是您在配置 SAML 时将 Okta 预身份验证应用程序上载到 Tableau Server。验证您是否在 Tableau Server 上配置了 Okta Tableau Server 应用程序元数据，而不是 Okta 预身份验证应用程序元数据

其他疑难解答步骤：

- 仔细检查 `tableau.conf` 中是否有拼写错误或配置错误
- 查看 Okta 预身份验证应用程序设置。确保按照本主题中指定的方式设置 HTTP 与 HTTPS 协议。
- 在两个代理服务器上重新启动 httpd。
- 验证 `sudo apachectl configtest` 在两个代理服务器上是否都返回“Syntax OK”。
- 验证测试用户是否已分配给 Okta 中的两个应用程序。
- 验证是否在负载平衡器和关联的目标组上设置了粘性

配置从负载平衡器到 Tableau Server 的 SSL/TLS

一些组织需要从客户端到后端服务的端到端加密通道。到目前为止所描述的默认参考架构指定了从客户端到在您组织的 Web 层中运行的负载均衡器的 SSL。

若要配置从从负载平衡器到 Tableau Server 的 SSL，您必须：

- 在 Tableau 和代理服务器上安装有效的 SSL 证书。
- 配置从负载平衡器到反向代理服务器的 SSL。
- 配置从代理服务器到 Tableau Server 的 SSL。
- 您还可以配置从 Tableau Server 到 PostgreSQL 实例的 SSL。

本主题的其余部分在示例 AWS 示例参考架构的上下文中描述了此实现。

示例：在 AWS 参考架构中配置 SSL/TLS

本节介绍如何在 Tableau 上配置 SSL 以及在 Apache 代理服务器上配置 SSL这些都在示例 AWS 参考架构中运行。

此示例中的 Linux 程序显示了类似 RHEL 的发行版的命令。具体来说，这里的命令是使用 Amazon Linux 2 发行版开发的。如果您运行的是 Ubuntu 发行版，请相应地编辑命令。

步骤 1：收集证书和相关密钥

为了简化证书管理和部署，以及作为安全最佳实践，我们建议使用由主要受信任的第三方证书颁发机构 (CA) 生成的证书。

或者，您可以为 TLS 生成自签名证书或使用 PKI 中的证书。

以下过程描述如何生成自签名证书。如果您按照我们的建议使用第三方证书，则可以跳过此过程。

在代理主机之一上运行此过程。生成证书和关联的密钥后，您将与其他代理主机和 Tableau Server 节点 1 共享它。

1. 生成签名根证书颁发机构 (CA) 密钥：

   ```
   openssl genrsa -out rootCAKey.pem 2048
   ```

2. 创建根 CA 证书：

   ```
   openssl req -x509 -sha256 -new -nodes -key rootCAKey.pem -days
   3650 -out rootCACert.pem
   ```

   系统将提示您输入证书字段的值。例如：

   ```
   Country Name (2 letter code) [XX]:US
   State or Province Name (full name) []:Washington
   Locality Name (eg, city) [Default City]:Seattle
   Organization Name (eg, company) [Default Company Ltd]:Tableau
   Organizational Unit Name (eg, section) []:Operations
   Common Name (eg, your name or your server's hostname)
   []:tableau.example.com
   Email Address []:example@tableau.com
   ```

3. 创建证书和相关密钥(在下面的示例中为 `serverssl.csr` 和 `serverssl.key`)。证书的使用者名称必须与 Tableau 主机的公共主机名称匹配。使用者名称是使用 `-subj` 选项以"/CN=<host-name>" 格式设置的，例如：

   ```
   openssl req -new -nodes -text -out serverssl.csr -keyout
   serverssl.key -subj "/CN=tableau.example.com"
   ```

4. 使用您在步骤 2 中创建的 CA 证书签署新证书。以下命令还将以 `crt` 格式输出证书：

   ```
   openssl x509 -req -in serverssl.csr -days 3650 -CA
   rootCACert.pem -CAkey rootCAKey.pem -CAcreateserial -out
   serverssl.crt
   ```

步骤 2:为 SSL 配置代理服务器

在两个代理服务器上运行此过程。

1. 安装 Apache ssl 模块:

   ```
   sudo yum install mod_ssl
   ```

2. 创建 /etc/ssl/private 目录:

   ```
   sudo mkdir -p /etc/ssl/private
   ```

3. 将 crt 和 key 文件复制到以下 /etc/ssl/ 路径:

   ```
   sudo cp serverssl.crt /etc/ssl/certs/

   sudo cp serverssl.key /etc/ssl/private/
   ```

4. 使用以下更新对现有 tableau.conf 进行更新:

   - 添加 SSL 重写块:

     ```
     RewriteEngine on
     RewriteCond %{SERVER_NAME} =tableau.example.com
     RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI}
     [END,NE,R=permanent]
     ```

   - 在 SSL 重写块中,更新 RewriteCond 服务器名称:添加您的公共主机名,例如,tableau.example.com
   - 将 <VirtualHost *:80> 更改为 <VirtualHost *:443>。
   - 使用以下内容封装 <VirtualHost *:443> 和 <Location /> 块:<IfModule mod_ssl.c> ...</IfModule>。
   - BalancerMember: 将协议从 http 更改为 https。
   - 在 <VirtualHost *:443> 块内添加 SSL* 元素:

     ```
     SSLEngine on
     SSLCertificateFile /etc/ssl/certs/serverssl.crt
     SSLCertificateKeyFile /etc/ssl/private/serverssl.key
     ```

```
SSLProxyEngine on
SSLProxyVerify none
SSLProxyCheckPeerName off
SSLProxyCheckPeerExpire off
```

- 在 LogLevel 元素中：添加 ssl:warn。
- 可选：如果您已安装并配置了身份验证模块，则 tableau.conf 文件中可能包含其他元素。例如，<Location /> </Location> 块将包含元素。

此处显示了针对 SSL 配置的示例 tableau.conf 文件：

```
RewriteEngine on
RewriteCond %{SERVER_NAME} =tableau.example.com
RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI}
[END,NE,R=permanent]

<IfModule mod_ssl.c>
<VirtualHost *:443>
ServerAdmin admin@example.com
ProxyHCExpr ok234 {%{REQUEST_STATUS} =~ /^[234]/}
Header add Set-Cookie "ROUTEID=.%{BALANCER_WORKER_ROUTE}e;
path=/" env=BALANCER_ROUTE_CHANGED
<Proxy balancer://tableau>
BalancerMember https://10.0.3.36/ route=1 hcmethod=GET
hcexpr=ok234 hcuri=/favicon.ico
BalancerMember https://10.0.4.15/ route=2 hcmethod=GET
hcexpr=ok234 hcuri=/favicon.ico
ProxySet stickysession=ROUTEID
</Proxy>
ProxyPreserveHost On
ProxyPass / balancer://tableau/
ProxyPassReverse / balancer://tableau/
DocumentRoot /var/www/html
ServerName tableau.example.com
ServerSignature Off
ErrorLog logs/error_sp.log
```

```
CustomLog logs/access_sp.log combined
LogLevel info ssl:warn
SSLEngine on
SSLCertificateFile /etc/ssl/certs/serverssl.crt
SSLCertificateKeyFile /etc/ssl/private/serverssl.key
SSLProxyEngine on
SSLProxyVerify none
SSLProxyCheckPeerName off
SSLProxyCheckPeerExpire off
</VirtualHost>
<Location />
#If you have configured a pre-auth module (e.g. Mellon) include
those elements here.
</Location>
</IfModule>
```

5. 添加 index.html 文件以抑制 403 错误：

   ```
   sudo touch /var/www/html/index.html
   ```

6. 重启 httpd：

   ```
   sudo systemctl restart httpd
   ```

步骤 3：为外部 SSL 配置 Tableau Server

将 serverssl.crt 和 serverssl.key 文件从代理 1 主机复制到初始 Tableau Server（节点 1）。

在节点 1 上运行以下命令：

```
tsm security external-ssl enable --cert-file serverssl.crt --key-
file serverssl.key
tsm pending-changes apply
```

步骤 4：可选身份验证配置

如果您为 Tableau 配置了外部身份提供程序，则您可能需要更新 IdP 管理仪表板中的返回 URL。

举例来说，如果您使用 Okta 预身份验证应用程序，您将需要更新应用程序以对收件人 URL 和目标 URL 使用 HTTPS 协议。

步骤 5：为 HTTPS 配置 AWS 负载平衡器

如果您按照本指南中的说明使用 AWS 负载平衡器进行部署，那么您可以重新配置 AWS 负载平衡器以将 HTTPS 流量发送到代理服务器：

1. 取消注册现有 HTTP 目标组：

   在**“目标组”**中，选择已为负载平衡器配置的 HTTP 目标组，单击**“操作”**，然后单击**“注册和注销实例”**。

   在**“注册和注销目标”**页面上，选择当前配置的实例，单击**“注销”**，然后单击**“保存”**。

2. 创建 HTTPS 目标组

   “目标组”>“创建目标组”

   - 选择“实例”
   - 输入目标组名称，例如 `TG-internal-HTTPS`
   - 选择您的 VPC
   - 协议：HTTPS 443
   - 在**“运行状况检查”>“高级运行状况检查设置”>“成功代码”**下，将代码列表追加为：`200,303`。
   - 单击**“创建”**。

3. 选择刚刚创建的目标组，然后单击**“目标”**选项卡：

   - 单击**“编辑”**
   - 选择正在运行代理应用程序的 EC2 实例，然后单击**“添加到已注册”**。
   - 单击**“保存”**。

4. 创建目标组后，您必须启用粘性：

   - 打开 AWS 目标组页面（**“EC2”>“负载平衡”>“目标组”**），选择您刚刚设置的目标组实例。在**“操作”**菜单上，选择**“编辑属性”**。
   - 在**“编辑属性”**页面上，选择**“粘性”**，指定持续时间 `1 day`，然后**保存更改**。

5. 在负载平衡器上，更新侦听器规则。选择您为此部署配置的负载平衡器，然后单击**“侦听器”**选项卡。

    - 对于**“HTTP:80”**，单击**“查看/编辑规则”**。在生成的**“规则”**页面上，单击编辑图标(一次位于页面顶部，然后再次位于规则旁边)以编辑规则。删除现有 THEN 规则，并通过单击**“添加操作”>“重定向至...”**替换该规则。在生成的 THEN 配置中，指定 `HTTPS` 和端口 `443`，并将其他选项保留为默认设置。保存设置，然后单击**“更新”**。
    - 对于**“HTTP:443”**，单击**“查看/编辑规则”**。在生成的**“规则”**页面上，单击编辑图标(一次位于页面顶部，然后再次位于规则旁边)以编辑规则。在**“THEN”**配置中的**“转发到...”**下，将目标组更改为您刚刚创建的 HTTPS 组。在**“组级粘性”**下，启用粘性并将持续时间设置为 1 天。保存设置，然后单击**“更新”**。

步骤 6:验证 SSL

通过浏览到 https://tableau.example.com 来验证配置。