Tableau Server Enterprise
部署指南

最近更新時間 2024/4/25

目錄

Tableau Server Enterprise 部署指南

Tableau Server Enterprise 部署指南 (EDG) 的撰寫目的在於為部署 Tableau Server(內部或雲端) 提供指示。該指南在參考架構下為企業情境提供部署指導。我們已經測試了參考架構，以驗證是否符合安全性、規模和性能基準，這些基準符合業界標準的最佳作法。

在較高層級上，業界標準企業部署的核心功能由分層拓撲組成，其中伺服器應用程式功能的每一層 (Web 閘道層，應用程式層和資料層) 均受存取控制的子網路限制和保護。從網際網路存取伺服器應用程式的使用者會在 Web 層進行驗證。驗證完成後，會對該請求進行 Proxy 處理，並將其轉至受保護的子網路，應用程式層將在該子網路中處理業務邏輯。高價值資料受到第三個子網路的保護：資料層。應用程式層中的服務透過受保護的網路與資料層通訊，以服務於對後端資料來源的資料請求。

在此部署中，安全性是所有設計決策和實作的重中之重。但是，可靠性、效能和可延伸性也是優先要求。基於參考架構的分佈和模組化設計，透過讓每個節點上的相容服務進行策略性共存，並在阻塞點處新增服務，可以以線性可預測的方式擴展可靠性和效能。

誰應閱讀本指南

EDG 是為企業 IT 管理員開發的，他們可能需要：

- IT 管理的 Tableau 部署
- 業界合規執法
- 業界部署最佳做法
- 預設安全部署

EDG 是用於部署企業參考架構的實作指南。雖然此版本的 EDG 包含範例 AWS/Linux 實作，但經驗豐富的企業 IT 管理員可以將本指南用作資源，以將指定的參考架構部署到任何業界標準資料中心環境中。

版本

此版本的 EDG 是為 Tableau Server 2021.2.3 版本(或更高版本)開發的。雖然您可以使用 EDG 作為部署舊版 Tableau Server 的一般參考,但我們建議您使用 Tableau Server 2021.2.3 或更高版本部署參考架構。某些功能和選項在較舊版本的 Tableau Server 上不可用。

對於最新的功能和改進,我們建議將 EDG 部署到 Tableau Server 2022.1.7 及更高版本。

本指南中描述的參考架構支援以下 Tableau 用戶端:具有相容瀏覽器的 Web 製作版、Tableau Mobile 和 Tableau Desktop 版本 2021.2.1 或更高版本。其他 Tableau 用戶端(Tableau Prep、Bridge 等)尚未透過參考架構進行驗證。

突出顯示功能

Tableau Server 參考架構的第一個版本引入了以下案例和功能:

- 用戶端預先驗證:在存取內部 Tableau Server 之前,Tableau 用戶端(桌面版、行動版、Web 製作版)均透過 Web 層中的企業驗證提供者進行驗證。此流程透過在充當反向 Proxy 伺服器的 Tableau Server 獨立閘道上設定 authN 外掛程式來管理。請參閱第 5 部分 - 配置 Web 層。
- 零信任部署:由於到 Tableau Server 的所有流量都經過預先驗證,因此整個 Tableau 部署在不需要受信任連線的私人子網路中執行。
- 外部存放庫:參考架構指定將 Tableau 存放庫安裝到外部 PostgreSQL 資料庫上,允許 DBA 作為一般資料庫管理、最佳化、規模化和備份存放庫。
- 初始節點復原:EDG 引入了一個指令碼,可在發生故障時自動進行初始節點還原。
- 基於 tar 的備份和還原:可在到達 Tableau 部署的戰略里程碑時使用熟悉的 tar 備份。如果發生故障或部署設定錯誤,可以通過復原關聯的 tar 備份快速復原到之前的部署階段。
- 效能改進:客戶和實驗室驗證資料表明,與標準部署相比,執行 EDG 時效能可提高 15-20%。

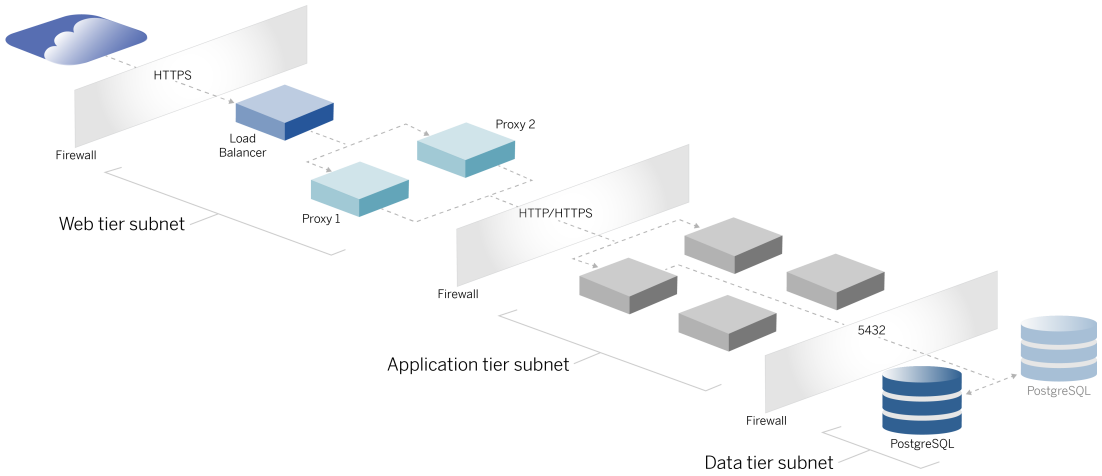
授權

本指南中所述的 Tableau Sever 參考架構需要 Tableau Advanced Management 授權才能啟用 Tableau Server 外部存放庫。還可以選擇部署 Tableau Server 外部檔案存放區，這也需要 Tableau Advanced Management 授權。請參閱*關於 Tableau Server 上的 Tableau Advanced Management* (Linux)。

第 1 部分 - 了解 Enterprise 部署

第 1 部分更詳細地描述了業界標準企業部署的功能和要求，這也是《Tableau Server Enterprise 部署指南》的目的。

以下網路圖顯示了具有 Tableau Server 參考架構的通用資料中心分層部署。



業界標準和部署要求

以下是業界標準部署的功能。這些是參考架構的設計要求：

- 多層網路設計：網路受保護子網路的限制，以限制每一層的存取：Web 層、應用程式層以及資料層。由於所有通訊都在下一個子網路終止，因此無法通過子網路進行任何通訊。
- 預設封鎖的連接埠和通訊協定：預設情況下，每個子網路或安全群組都會封鎖所有入站和出站連接埠和通訊協定。透過在連接埠或通訊協定設定中開啟例外，可以部分啟用通訊。
- 開箱即用的 Web 驗證：來自網際網路的使用者請求由 Web 層中反向 Proxy 上的驗證模塊進行驗證。因此，對應用程式層的所有請求在傳遞到受保護的應用程式層之前都已在 Web 層經過驗證。
- 不倚靠平台：解決方案可以與內部部署伺服器應用程式一起部署，也可以在雲端部署。

- 不倚靠技術:解決方案可以部署在虛擬機環境或容器中。也可以部署在 Windows 或 Linux 上。但是,此參考架構和支援文件的初始版本是為在 AWS 中執行的 Linux 而開發的。
- 高度可用:系統中的所有元件均部署為叢集,並設計為在主動/主動或主動/被動部署中執行。
- 獨立角色:每個伺服器器執行的離散角色。這種設計對所有伺服器進行了分區,從而可以最大限度地減少服務特定的管理員的存取。例如,DBA 管理 Tableau 的 PostgreSQL,身分管理員管理 Web 層中的驗證模塊,網路和雲端管理員啟用流量和連線。
- 線性可延展:作為離散的角色,您可以根據負載設定檔獨立地延展每個層的服務。
- 用戶端支援:參考架構支援所有 Tableau 用戶端:Tableau Desktop(版本 2021.2 或更高版本)、Tableau Mobile 和 Tableau Web 製作。

安全措施

如上所述,業界標準資料中心設計的主要功能是安全性。

- 存取:每層受一個子網路限制,該子網路使用連接埠篩選在 Web 層強制執行存取控制。子網路之間的通訊存取也可以由應用程式層,透過程序之間的身份驗證服務來強制實施。
- 整和:架構設計旨為 Web 層中的反向 Proxy 提供身分識別提供者 (IdP) 外掛程式。
- 隱私:用戶端使用 SSL 對進入 Web 層的流量進行加密。進入內部子網路的流量也可以選擇加密。

Web proxy 層

Web 層是 DMZ 中的子網路(也稱為周邊區),充當網際網路與部署應用程式的內部子網路之間的安全緩衝區。Web 層託管不儲存任何敏感資訊的反向 Proxy 伺服器。在將用戶端請求重新導向到 Tableau Server 之前,反向 Proxy 伺服器已設定 AuthN 外掛程式,以使用受信任的 IdP 對用戶端工作階段進行預驗證。有關詳情,請參閱使用 AuthN 模組進行預先驗證。

負載平衡器

部署設計包括反向 Proxy 伺服器前端的企業負載平衡解決方案。

負載均平衡器透過以下方式提供了重要的安全性和效能強化:

- 虛擬化應用程式層服務的前端 URL
- 強制 SSL 加密
- 卸載 SSL
- 用戶端和 Web 層服務之間強制壓縮
- 抵禦 DOS 攻擊
- 提供高可用性

附註:Tableau Server 版本 2022.1 包括 Tableau Server 獨立閘道。獨立閘道是 Tableau 閘道流程的獨立執行個體,用作 Tableau 感知反向 Proxy。在發佈時,獨立閘道已經過驗證,但尚未在 EDG 參考架構中進行全面測試。全面測試完成後,EDG 將更新 Tableau Server 獨立閘道規範指南。

應用程式層

應用程式層位於執行伺服器應用程式核心業務邏輯的子網路中。應用程式層由跨叢集中分佈式節點設定的服務和流程組成。應用程式層只能從 Web 層存取,使用者不能直接存取。

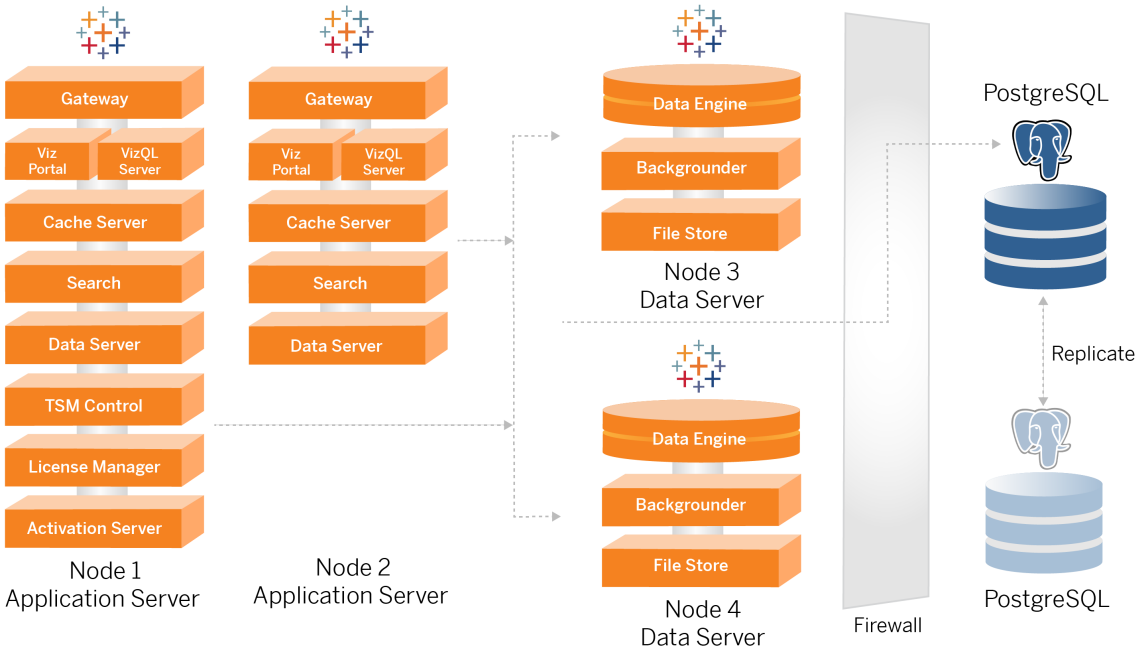
通過設定應用程式程序,使具有不同資源的設定檔(CPU 密集型與記憶體密集型)位於同一位置,可以提高性能和可靠性。

資料層

資料層是一個包含有價值資料的子網路。此層的所有流量都來自應用程式層,因此已經過驗證。除了具有連接埠設定的 Web 層的存取要求外,該層還應包括經過驗證的存取和與應用程式層的可選加密流量。

第 2 部分 - 瞭解 Tableau Server 部署參考架構

下圖顯示了相關的 Tableau Server 處理序以及它們在參考架構中的部署方式。此部署被認為是適合企業的最小 Tableau Server 部署。



本主題中的流程圖旨在顯示每個節點的主要定義處理序。許多支援處理序也在圖中未顯示的節點上執行。有關所有處理序的清單，請參見本指南的設定部分，即第 4 部分 - 安裝並設定 Tableau Server 。

Tableau Server 處理序

Tableau Server 參考架構是一個四節點 Tableau Server 叢集部署，在 PostgreSQL 上具有外部存放庫：

- Tableau Server 初始節點(節點 1):執行所需的 TSM 管理和授權服務,這些服務只能在叢集中的單個節點上執行。在企業內容中,Tableau Server 初始節點是叢集中的主要節點。此節點還與節點 2 一起執行冗餘應用程式服務。
- Tableau Server 應用程式節點(節點 1 和節點 2):這兩個節點服務用戶端請求,連線並查詢資料來源以及資料節點。
- Tableau Server 資料節點(節點 3 和節點 4):兩個專門管理資料的節點。
- 外部 PostgreSQL:此主機執行 Tableau Server 存放庫程序。對於 HA 部署,必須對主動/被動冗餘執行額外的 PostgreSQL 主機。

  還可以在 Amazon RDS 上執行 PostgreSQL。有關在 RDS 與 EC2 執行個體上執行存放庫之間的差異詳情,請參閱 *Tableau Server 外部存放庫* (Linux)。

  使用外部存放庫部署 Tableau Server 需要 Tableau Advanced Management 授權。

  若您的組織沒有內部 DBA 專業知識,則可以選擇在預設的內部 PostgreSQL 設定中執行 Tableau Server 存放庫流程。預設情況下,存放庫在具有內嵌 PostgreSQL 的 Tableau 節點上執行。在這種情況下,建議在專用 Tableau 節點上執行存放庫,並在其他專用節點上執行被動存放庫,以支援存放庫容錯移轉。請參閱*存放庫容錯移轉* (Linux)。

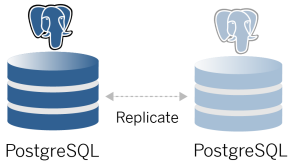
  例如,本指南中描述的 AWS 實作解釋了如何在 EC2 執行個體上執行的 PostgreSQL 上部署外部存放庫。

- 可選:若您的組織使用外部儲存體,則可以將 Tableau 檔案存放區部署為外部服務。本指南不包括核心部署情境中的外部檔案存放區。請參閱*使用外部檔案存放區安裝 Tableau Server* (Linux)。

  使用外部檔案存放區部署 Tableau Server 需要 Tableau Advanced Management 授權。

PostgresSQL 存放庫

Tableau Server 存放庫是儲存伺服器資料的 PostgresSQL 資料庫。此資料包括有關 Tableau Server 使用者、群組和群組分派、使用權限、專案、資料來源和資料擷取中繼資料的資訊以及重新整理資訊。



預設的的 PostgresSQL 部署會消耗將近 50% 的系統記憶體資源。根據其使用情況(用於生產和大型生產部署)而定,資源使用量會上升。因此,我們建議在未執行任何其他資源密集型伺服器元件(例如 VizQL、背景程式或資料引擎)的計算機上存放庫程序。與任何這些元件一起執行存放庫程序將會建立 IO 爭用、資源約束,並降低部署的整體效能。

節點 1:初始節點

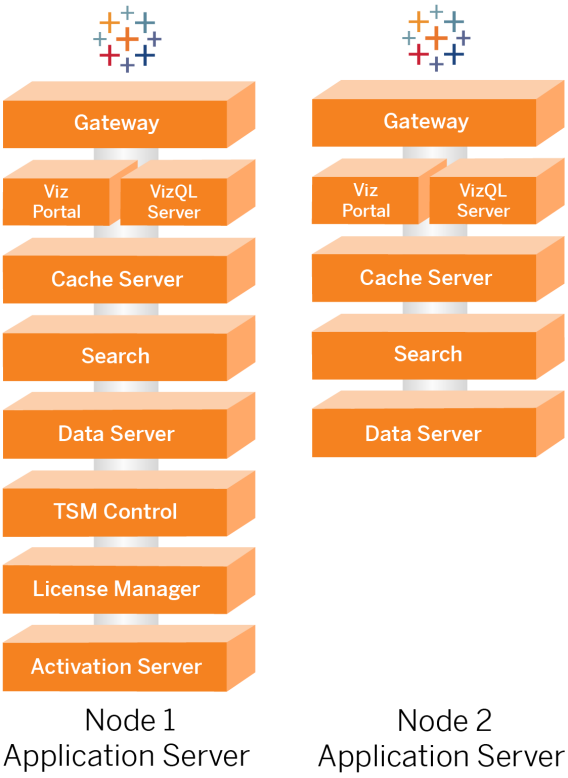
初始節點執行少量的重要流程,並與節點 2 分擔應用程式負載。

安裝 Tableau 的第一台電腦(「初始節點」)會有一些獨特的特性。有三個處理序只能在初始節點上執行,無法轉移到任何其他節點(出現故障的情況下除外),即授權服務(授權管理器)、啟用服務和 TSM 控制器(管理控制器)。

節點 1 容錯移轉和自動還原

授權、啟用和 TSM 控制器服務對 Tableau Server 部署的執行狀況至關重要。若節點 1 發生故障,使用者仍然可以連線到 Tableau Server 部署,因為正確設定的參考架構會將請求路由到節點 2。但是,若沒有這些核心服務,部署將處於擱置故障嚴重狀態。請參閱初始節點自動復原。

節點 1 和 2:應用程式伺服器



節點 1 和 2 執行 Tableau Server 程序，這些程序為用戶端請求，查詢資料來源，產生視覺效果，處理內容和管理以及其他核心 Tableau 商業邏輯提供服務。應用伺服器不儲存使用者資料。

> **附註**:「應用程式伺服器」是一個術語，也指 TSM 中列出的 Tableau Sever 流程。「應用程式伺服器」的基礎流程為 VizPortal。

並行執行的節點 1 和節點 2 進行擴充，以對來自反向 Proxy 伺服器上執行的負載平衡邏輯的請求提供服務。作為冗餘節點，若這些節點之一出現故障，則用戶端請求和服務會由其餘節點進行處理。

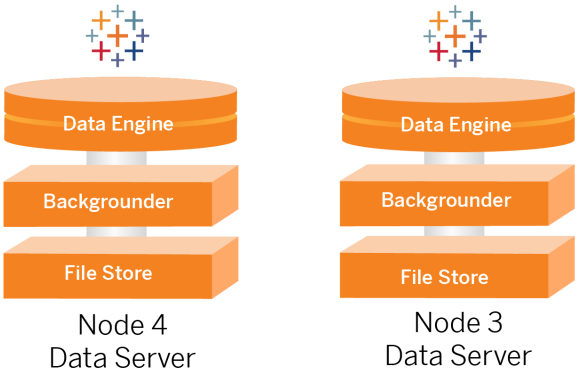
參考架構的設計使互補的應用程序在同一台電腦上執行。這代表程序不會競爭電腦資源並產生爭用。

例如，應用程式伺服器上的核心處理服務 VizQL 佔用大量 CPU 和記憶體，佔用了電腦上將近 60% 至 70% 的 CPU 和記憶體。因此，設計參考架構的目的是使其他記憶體或 CPU 綁定程序與 VizQL 不在同一節點上。測試顯示，使用者的負載或數量不會影響 VizQL 節點上的記憶體或 CPU 使用率。例如，減少負載測試中的並行使用者數量只會影響儀表板或視覺效果載入程序的效能，而不會降低資源利用率。因此，根據尖峰使用量期間的可用記憶體和 CPU，可以考慮新增更多 VizQL 程序。作為典型工作簿的起始位置，為每個 VizQL 程序指派 4 個內核。

擴充應用程式伺服器

該參考架構旨在根據基於使用情況的模型進行擴充。作為一般起點，我們建議至少使用兩個應用程式伺服器，每個最多支援 1000 個使用者。隨著使用者的增加，計劃每增加 1000 個使用者，則新增一個應用程式伺服器。監視使用情況和效能，為您的組織調整每台主機的使用者。

節點 3 和 4：資料伺服器



由於以下原因，檔案存放區、資料引擎 (Hyper)、背景程式流程共同位於節點 3 與 4 上：

- 擷取最佳化：在同一節點上執行背景程式、Hyper 和檔案存放區可最佳化效能和可靠性。在擷取程序中，背景程式查詢目標資料庫，在同樣節點上建立 Hyper 檔案，然後上傳到檔案存放區。透過在同一節點放置這些程序，擷取建立工作流程不需要跨網路或節點複製大量資料。
- 免費資源平衡：背景程式主要佔用大量 CPU。資料引擎是一個佔用大量記憶體的程序。耦合這些程序可以最大限度地利用每個節點上的資源。

- 資料程序的整合:由於這些程序都是後端資訊流程,因此有必要在最安全的資料層中執行它們。在參考架構的未來版本中,應用程式和資料伺服器將在不同層中執行。但是,由於 Tableau 架構中的應用程式有依賴性,此時應用程式和資料伺服器必須執行在同一層中。

擴充資料伺服器

與應用程式伺服器一樣,規劃 Tableau 資料伺服器所需的資源需要基於使用情況的建模。通常,假設每個資料伺服器每天最多可以支援 2000 個資料擷取重新整理作業。隨著擷取作業的增加,請在沒有檔案存放區服務的情況下新增其他資料伺服器。通常,雙節點資料伺服器部署適用於使用本機檔案系統進行檔案存放區服務的部署。請注意,新增更多應用程式伺服器不會以線性方式影響資料伺服器的效能或規模。事實上,除了來自額外使用者查詢的一些額外負荷外,新增更多應用程式主機和使用者的影響很小。

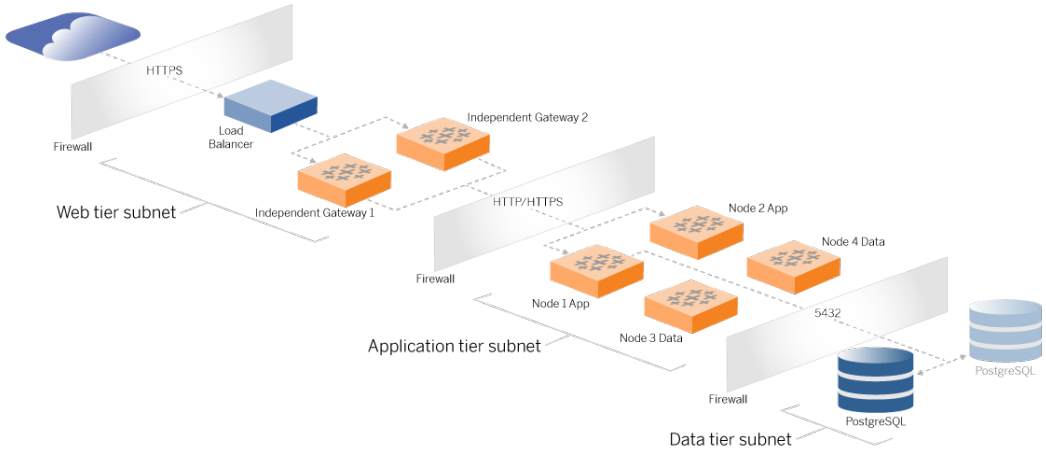
第 3 部分 - 準備 Tableau Server Enterprise 部署

第 3 部分會介紹準備基礎結構以部署 Tableau Server 參考架構的要求。在開始之前，我們建議您先閱讀第 2 部分 - 瞭解 Tableau Server 部署參考架構。

除了介紹要求之外，本主題還會提供 AWS 環境中參考架構的實作範例。本指南的其餘部分均基於本主題中提供的 AWS 參考架構範例。

參考架構的核心原則是採用資料中心安全最佳作法進行標準化。具體而言，該架構旨在將服務隔離到受保護的網路子網路中。子網路間通訊僅限於特定的通訊協定和連接埠流量。

下圖說明了內部部署或客戶管理的雲端部署的參考架構子網路設計。有關雲部署的範例，請參閱以下章節範例：在 AWS 中設定子網路和安全性群組。



子網路

建立三個子網路：

- Web 層
- 應用層
- 資料子網路。

防火牆/安全群組規則

下方的索引標籤將介紹資料中心每一層的防火牆規則。有關 AWS 特定的安全性群組規則, 請參閱本主題後面的章節。

Web 層

Web 層是一個公用 DMZ 子網路, 它將處理輸入 HTTPS 請求並將請求代理到應用程式層。此設計對針對您的組織的惡意軟體提供一層防禦。Web 層阻止對應用 / 資料層的存取。

流量	類型	通訊協定	連接埠範圍	來源
輸入	SSH	TCP	22	Bastion 子網路(用於雲部署)
輸入	HTTP	TCP	80	網際網路 (0.0.0.0/0)
輸入	HTTPS	TCP	443	網際網路 (0.0.0.0/0)
輸出	所有流量	全部	全部	

應用層

應用子網路是 Tableau Server 部署所在的位置。應用子網路包括 Tableau 應用伺服器(節點 1 和節點 2)。Tableau 應用伺服器處理使用者對資料伺服器的請求並執行核心商業邏輯。

應用子網路還包括 Tableau 資料伺服器(節點 3 和節點 4)。

到應用層的所有用戶端流量都在 Web 層進行身份驗證。對應用子網路的管理存取是透過 Bastion 主機進行身分驗證和路由。

流量	類型	通訊協定	連接埠範圍	來源
輸入	SSH	TCP	22	Bastion 子網路(用於雲部署)
輸入	HTTPS	TCP	443	Web 層子網路
輸出	所有流量	全部	全部	

資料層

資料子網路是外部 PostgreSQL 資料庫伺服器的所在位置。

流量	類型	通訊協定	連接埠範圍	來源
輸入	SSH	TCP	22	Bastion 子網路(用於雲部署)
輸入	PostgreSQL	TCP	5432	應用程式層子網路
輸出	所有流量	全部	全部	

Bastion

大多數企業安全團隊不允許從內部部署管理系統到部署在雲端中節點的直接通訊。相反，到雲端節點的所有管理 SSH 流量都透過 Bastion 主機(也稱為「跳轉伺服器」)進行 Proxy。對於雲部署，建議將 Bastion 主機 Proxy 連線到參考架構中的所有資源。這是內部環境的可選組態。

Bastion 主機會對管理存取進行身分驗證，並且只允許透過 SSH 通訊協定的流量。

流量	類型	通訊協定	連接埠範	來源	目的地

			圍		
輸入	SSH	TCP	22	管理員電腦的 IP 位址	
輸出	SSH	TCP	22		Web 層子網路
輸出	SSH	TCP	22		應用程式層子網路

範例:在 AWS 中設定子網路和安全性群組

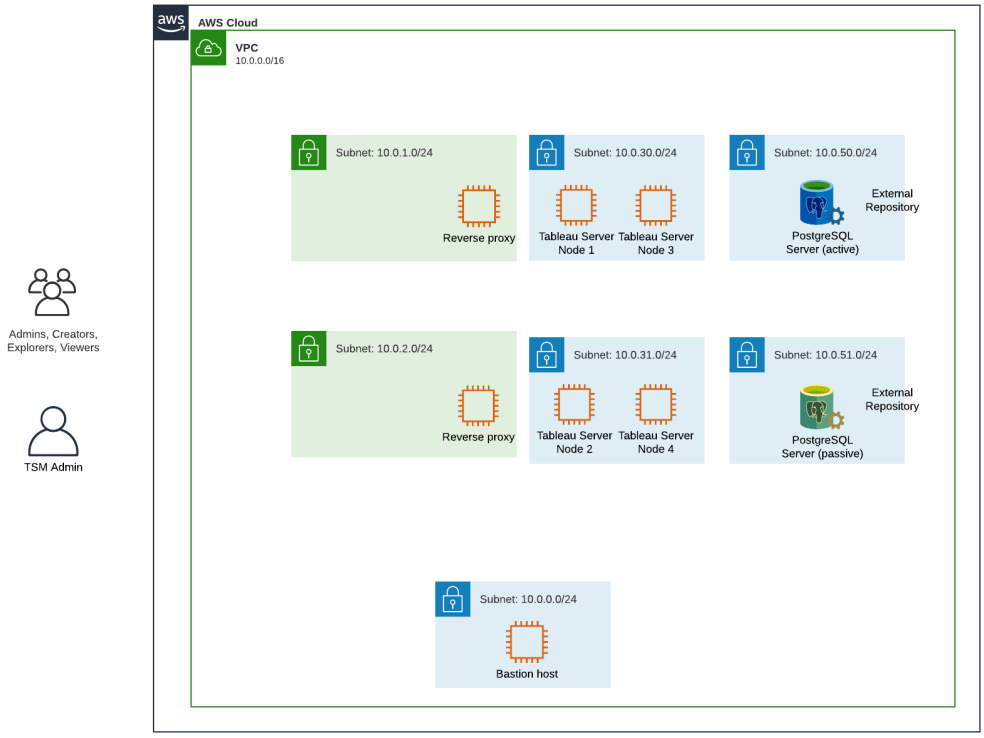
本章節提供了步驟過程,為 AWS 中的 Tableau Server 參考架構,建立和設定VPC以及網路環境。

下方的投影片會顯示四層參考架構。瀏覽投影片時,您會看到元件元素分層到拓撲圖上:

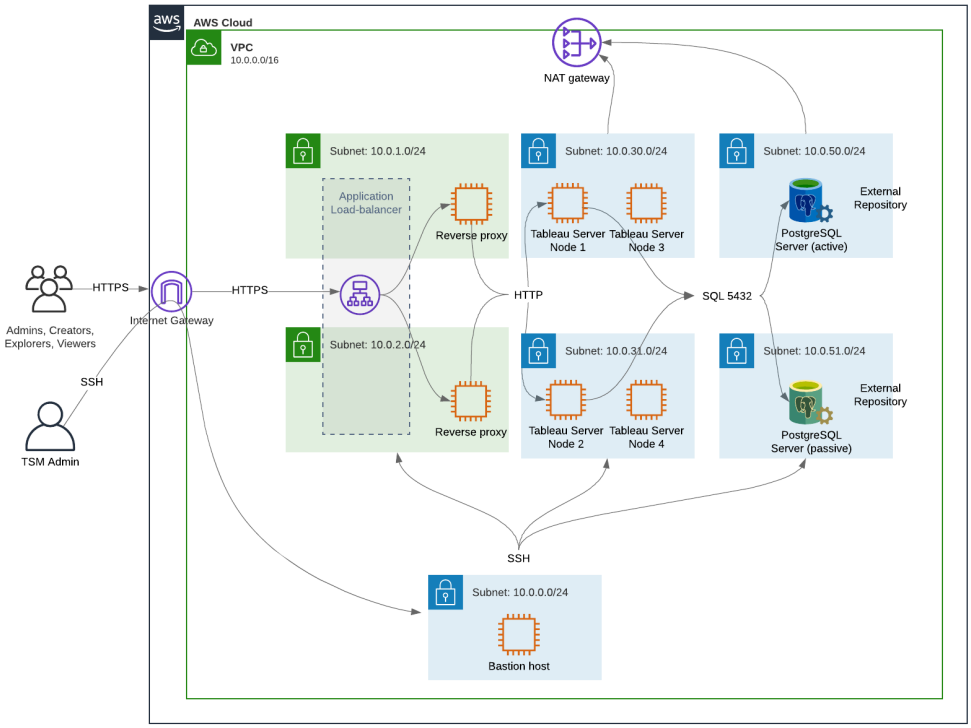
1. VPC 子網路拓撲和 EC2 執行個體:一個 Bastion 主機、兩個反向 Proxy 伺服器、四個 Tableau 伺服器和至少一個 PostgreSQL 伺服器。
2. 通訊協定流程和網際網路連線。所有輸入流量都透過 AWS 網際網路閘道進行管理。到網際網路的流量透過 NAT 路由。
3. 可用區域。Proxy、Tableau Server 和 PostgreSQL 主機均勻部署在兩個可用區域中。
4. 安全性群組。四個安全性群組(公用、私有、資料和 Bastion)在通訊協定層級保護每一層。

AWS 參考架構

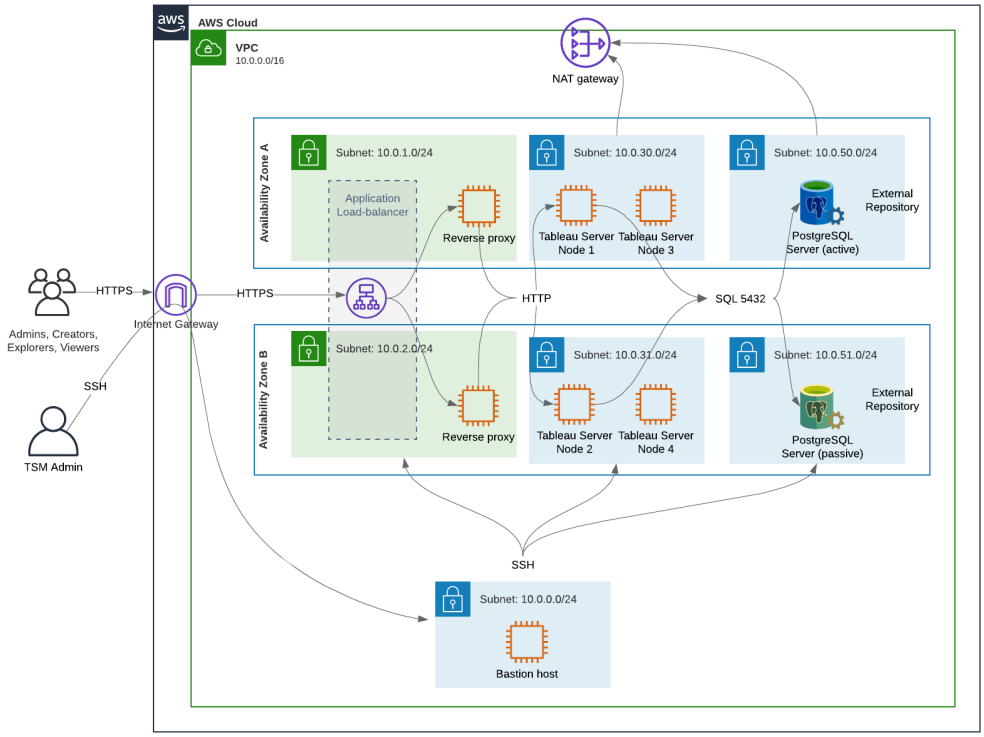
幻燈片 1:VPC 子網路拓撲和 EC2 執行個體



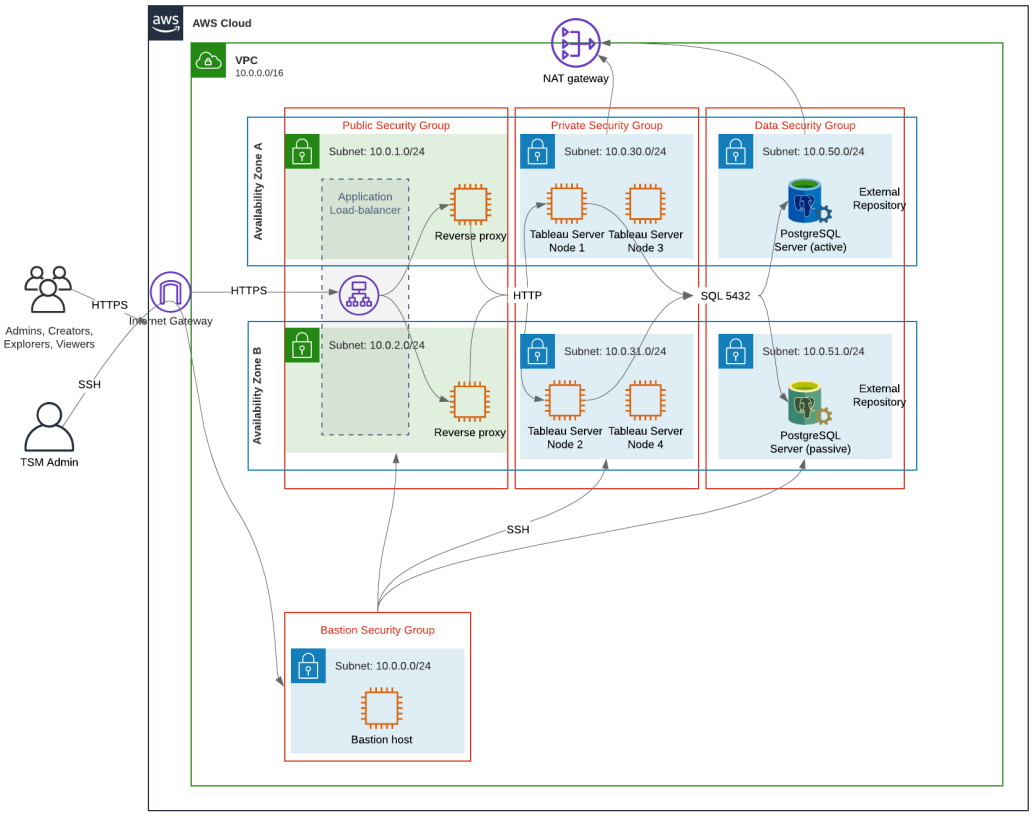
幻燈片 2:協定流程和連線



幻燈片 3:可用區域


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
External
Repository
PostgreSQL
Server (active)
HTTPS
Admins, Creators,
Explorers, Viewers
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
External
Repository
PostgreSQL
Server (passive)
SSH
Subnet: 10.0.0.0/24
Bastion host

幻燈片 4:安全性群組



AWS 可用區域和高可用性

本指南中提供的參考架構指定了一種部署，該部署可在任何單個主機出現故障時透過冗餘提供可用性。但是，在參考架構跨兩個可用區域部署的 AWS 情況下，可用性在極少數情況下會受到影響，即可用區域發生故障。

VPC 組態

本節介紹如何：

- 安裝和設定 VPC
- 設定網路連線
- 設定子網路
- 建立和設定安全性群組

設定 VPC

本小節中的程序映射到「經典」VPC 體驗中的 UI。可以透過關閉 AWS VPC 儀表板左上角的新 VPC 體驗來切換 UI 以顯示經典檢視。

執行 VPC 精靈以建立預設的私有和公用子網路以及預設的路由和網路 ACL。

1. 在設定 VPC 之前，必須建立彈性 IP。使用所有預設值建立分配。
2. 執行 VPC 精靈>「具有公用和專用子網路的 VPC」
3. 接受大部分的預設值。除了以下情況：
    - 輸入 VPC 名稱。
    - 指定彈性 IP 分配 ID。
    - 指定以下 CIDR 遮罩：
        - 公用子網路的 IPv4 CIDR：10.0.1.0/24，將此子網路重新命名為 `Public-a`。
        - 私有子網路的 IPv4 CIDR：10.0.30.0/24，將此子網路重新命名為 `Private-a`。
    - 可用區域：若為兩個子網路，為所在區域選取 **a** 選項。

    > **附註**：出於本範例的目的，我們使用 **a** 和 **b** 來區分指定 AWS 資料中心中的可用區域。在 AWS 中，可用區域名稱可能與此處顯示的範例不相符。例如，一些可用區域包括資料中心中的 **c** 和 **d** 區域。

4. 按一下**建立 VPC**。
5. 建立 VPC 後，建立 `Public-b`、`Private-b`、`Data`，和`Bastion`子網路。要建立子網路，請按一下**子網路>建立子網路**。

    - `Public-b`：若為可用區域，為所在區域選取 **b** 選項。CIDR 區塊：10.0.2.0/24
    - `Private-b`：若為可用區域，為所在區域選取 **b** 選項。CIDR 區塊：10.0.31.0/24
    - `Data`：若為可用區域，為所在區域選取 **a** 區域。CIDR 區塊：10.0.50.0/24。可選：若計劃跨 PostgreSQL 叢集複製外部資料庫，請在可用區域 b 中建立一個 CIDR 區塊為 10.0.51.0/24 的資料 b 子網路。
    - `Bastion`：若為可用區域，選取任一區域。CIDR 區塊：10.0.0.0/24
6. 建立子網路後，編輯公用和 Bastion 子網路上的路由表，以使用為相關的網際網路閘道 (IGW) 設定的路由表。並編輯私有和資料子網路以使用為網路位址轉譯器

(NAT) 設定的路由表。

- 要確定哪個路由表設定了 IGW 或 NAT，請按一下 AWS 儀表板中的**路由表**。選取兩個路由表連結其中的一個開啟屬性頁面。查看**路由 > 目的地 > 0.0.0.0/0**處的目標值。目標值區分路由的類型，並且將以`igw-`或者`nat-`字串為開頭。
- 為了更新路由表，**VPC>子網路>** [子網路_名稱]**>路由表>編輯路由表關聯**。

設定安全性群組

VPC 精靈建立一個您不會使用到的安全性群組。建立以下安全性群組(**安全性群組 > 建立安全性群組**)。EC2 主機將跨兩個可用區域安裝到這些群組中，如上方的投影片圖表中所示。

- 建立新的安全性群組：**Private**。這是安裝 Tableau Server 所有 4 個節點的位置。在安裝過程的最後，「私有」安全性群組將與 10.0.30.0/24 和 10.0.31.0/24 子網路關聯。
- 建立一個新的安全性群組：**Public**。這是安裝 proxy 伺服器的位置。在安裝過程的最後，「公用」安全性群組將與 10.0.1.0/24 和 10.0.2.0/24 子網路關聯。
- 建立新的安全性群組：**Data**。這裡是安裝 PostgreSQL 外部 Tableau 存放庫的位置。在安裝過程的最後，資料安全性群組將與 10.0.50.0/24(以及可選的 10.0.51.0/24) 子網路關聯。
- 建立一個新的安全性群組：**Bastion**。這是將安裝 Bastion 主機的位置。在安裝過程的最後，Bastion 安全性群組將與 10.0.0.0/24 子網路關聯。

指定輸入和輸出規則

在 AWS 中，安全性群組類似於內部環境中的防火牆。必須指定允許傳入和/或傳出安全性群組的流量類型(例如 https、https 等)、通訊協定(TCP 或 UDP) 以及連接埠或連接埠範圍(例如 80、443 等)。針對每個通訊協定，您還必須指定目的地或來源流量。

公用安全群組規則

輸入規則			
類型	**通訊協定**	**連接埠範圍**	**來源**
HTTP	TCP	80	0.0.0.0/0

HTTPS	TCP	443	0.0.0.0/0
SSH	TCP	22	Bastion 安全性群組

輸出規則			
類型	**通訊協定**	**連接埠範圍**	**目的地**
所有流量	全部	全部	0.0.0.0/0

私有安全群組規則

私有安全性群組包含一個輸入規則，以允許來自「公共」安全性群組的 HTTP 流量。只有在部署過程中允許 HTTP 流量來驗證連線。建議在完成反向 proxy 部署並將 SSL 設定到 Tableau 後移除 HTTP 輸入規則。

輸入規則			
類型	**通訊協定**	**連接埠範圍**	**來源**
HTTP	TCP	80	公用安全性群組
HTTPS	TCP	443	公用安全性群組
PostgreSQL	TCP	5432	資料安全性群組
SSH	TCP	22	Bastion 安全性群組
所有流量	全部	全部	私有安全性群組

輸出規則			
類型	**通訊協定**	**連接埠範圍**	**目的地**
所有流量	全部	全部	0.0.0.0/0

PostgreSQL	TCP	5432	資料安全性群組
SSH	TCP	22	Bastion 安全性群組

資料安全性群組規則

輸入規則			
類型	**通訊協定**	**連接埠範圍**	**來源**
PostgreSQL	TCP	5432	私有安全性群組
SSH	TCP	22	Bastion 安全性群組

輸出規則			
類型	**通訊協定**	**連接埠範圍**	**目的地**
所有流量	全部	全部	0.0.0.0/0
PostgreSQL	TCP	5432	私有安全性群組
SSH	TCP	22	Bastion 安全性群組

Bastion 主機安全性群組規則

輸入規則			
類型	**通訊協定**	**連接埠範圍**	**來源**
SSH	TCP	22	用來登入 AWS(管理員電腦)的電腦 IP 位址和網絡遮罩。
SSH	TCP	22	私有安全性群組
SSH	TCP	22	公用安全性群組

輸出規則			
類型	**通訊協定**	**連接埠範圍**	**目的地**
SSH	TCP	22	用來登入 AWS(管理員電腦)的電腦 IP 位址和網絡遮罩。
SSH	TCP	22	私有安全性群組
SSH	TCP	22	公用安全性群組
SSH	TCP	22	資料安全性群組
HTTPS	TCP	443	0.0.0.0/0(可選:若需存取網際網路以下載 Bastion 主機上的支援軟體,請建立此規則)

啟用自動指定公用 IP

提供連線到 proxy 伺服器和 Bastion 主機的 IP 位址。

針對公用和 Bastion 子網路:

1. 選擇子網路
2. 在**「動作」**功能表下,選取「修改自動分配 IP 設定」。
3. 按一下「啟用自動分配公用 Ipv4 位址」。
4. 按一下**「儲存」**。

負載平衡器

> **附註**:如果要安裝到 AWS 並遵循本指南中的範例部署,那麼應該在部署過程的最後安裝和設定 AWS 負載平衡器,如第 5 部分 - 配置 Web 層中所述。

對於內部部署,請與網路管理員合作部署負載平衡器,以支援參考架構的 Web 層:

- 面向 Web 的應用程式負載平衡器,它接受來自 Tableau 用戶端的 HTTPS 請求並與反向 proxy 伺服器進行通訊。

- 反向 proxy:
  - 建議至少使用兩個 proxy 伺服器為冗餘和處理用戶端加載。
  - 從負載平衡器接收 HTTPS 流量。
  - 支援 Tableau 主機的相黏工作階段。
  - 為執行閘道過程的每個 Tableau Server 設定循環負載平衡 proxy。
  - 處理來自外部 IdP 的身份驗證請求。
- 正向 Proxy:Tableau Server 需要存取網際網路獲得授權與對應功能。根據正向 Proxy 環境,可能需要為 Tableau 服務 URL 設定正向 Proxy 允許清單。請參閱 *與網際網路通訊*(Linux)。

設定主機電腦

最低推薦硬件

以下建議基於我們在參考架構中對真實資料的測試。

應用程式伺服器:

- CPU:8 個實體核心 (16vCPU),
- RAM:128 GB(8 GB/實體內核)
- 磁碟空間:100 GB

資料伺服器

- CPU:8 個實體核心 (16vCPU),
- RAM:128 GB(8 GB/實體內核)
- 磁碟空間:1 TB。若部署將使用 Tableau 檔案存放區的外部儲存體,將需要計算適當的磁碟空間。請參閱*使用外部檔案存放區安裝 Tableau Server* (Linux)。

Proxy 伺服器

- CPU:2 個實體核心 (4vCPU),
- RAM:8 GB(4 GB/實體核心)
- 磁碟空間:100 GB

外部存放庫資料庫

- CPU:8 個實體核心 (16vCPU),
- RAM:128 GB(8 GB/實體內核)

- 磁碟空間要求取決於資料負載以及將會如何影響備份。請參閱主題*磁碟空間要求*中的*備份和還原程序*一節(Linux)。

目錄結構

參考架構建議將 Tableau Server 套件和資料安裝到非預設位置:

- 將套件安裝到: `/app/tableau_server`:在安裝 Tableau Server 套件之前建立此目錄路徑,然後在安裝期間指定此路徑。
- 將 Tableau 資料安裝到:`/data/tableau_data`。在安裝 Tableau Server 之前不要建立此目錄。相反,必須在安裝期間指定路徑,然後 Tableau 安裝程式將相應地建立該路徑並授予其權限。

有關實作詳細資訊,請參閱執行安裝套件並初始化 TSM。

範例:在 AWS 中安裝和準備主機電腦

本節說明如何為 Tableau Server 參考架構中的每個伺服器類型安裝 EC2 主機。

參考架構需要八台主機:

- Tableau Server 的四個執行個體。
- Proxy 伺服器 (Apache) 的兩個執行個體。
- Bastion 主機的一個執行個體。
- 一個或兩個 EC2 PostgreSQL 資料庫執行個體

主機執行個體詳情

根據以下詳細資訊安裝主機。

Tableau Server

- Amazon Linux 2
- 執行個體類型:m5a.8xlarge
- 安全性群組 ID:私有

- 儲存體：EBS、150 GiB、gp2 磁碟區類型。若部署將使用 Tableau 檔案存放區的外部儲存體，將需要計算適當的磁碟空間。請參閱*使用外部檔案存放區安裝 Tableau Server* (Linux)。
- 網絡：在每個私有子網路 (10.0.30.0/24 和 10.0.31.0/24) 中安裝兩個 EC2 主機。
- 將 Tableau Server 2021.2(或更高版本) rpm 套件的最新維護版本從 Tableau 下載頁面複製到每個 Tableau 主機。

Bastion 主機

- Amazon Linux 2
- 執行個體類型：t3.micro
- 安全性群組 ID：Bastion
- 儲存空間：EBS、50 GiB、gp2 磁碟區類型
- 網路：Bastion 子網路 10.0.0.0/24

Tableau Server 獨立閘道

- Amazon Linux 2
- 執行個體類型：t3.xlarge
- 安全性群組 ID：Public
- 儲存空間：EBS、100 GiB、gp2 磁碟區類型
- 網絡：在每個公用子網路(10.0.1.0/24 和 10.0.2.0/24) 中安裝一個 EC2 執行個體

PostgreSQL EC2 主機

- Amazon Linux 2
- 執行個體類型：r5.4xlarge
- 安全性群組 ID：Data
- 儲存體：磁碟空間要求取決於資料負載以及將會如何影響備份。請參閱主題*磁碟空間要求*中的*備份和還原程序*一節 (Linux)。
- 網路：資料子網路 10.0.50.0/24。(如果在 HA 叢集中複製 PostgreSQL，則在 10.0.51.0/24 子網路中安裝第二台主機)

驗證：VPC 連線

安裝主機電腦後，請驗證網路組態。使用 SSH 從 Bastion 安全性群組中的主機連線到每個子網路中的主機來驗證主機之間的連線。

範例：連線到 AWS 中的 Bastion 主機

1. 為 ssh-agent 設定管理員電腦。這讓您可以連線到 AWS 中的主機，而無需將私有金鑰檔案放置在任何 EC2 執行個體上。

   要在 Mac 上設定 ssh-agent，請執行以下命令：

   `ssh-add -K myPrivateKey.pem` 或用於最新 Mac OS，`ssh-add --apple-use-keychain myPrivateKey.pem`

   針對 Windows，請參閱主題安全連線到在私有 Amazon VPC 執行的 Linux 執行個體。

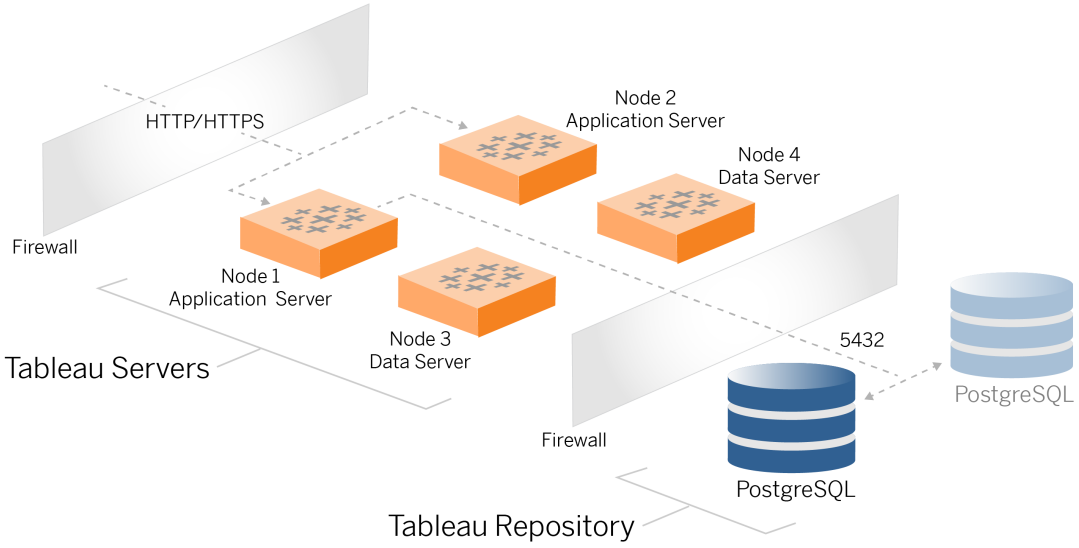
2. 執行以下命令連線到 Bastion 主機：

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

3. 然後，可以使用私有 IP 位址從 Bastion 主機連線到 VPC 中的其他主機，例如：

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

第 4 部分 - 安裝並設定 Tableau Server



本主題描述如何完成安裝和設定基準 Tableau Server 部署。此處的過程繼續 AWS 和 Linux 參考架構範例。

整個 Linux 範例的安裝過程展示了 RHEL-like 發行版的命令。具體來說，這裡的命令是使用 Amazon Linux 2 發行版開發的。若執行的是 Ubuntu 發行版，請編輯相對應的命令。

開始之前

第 3 部分 - 準備 Tableau Server Enterprise 部署所述，準備和驗證您的環境。

安裝、設定 PostgreSQL 和建立 tar 備份

此 PostgreSQL 執行個體託管 Tableau Server 部署的外部存放庫。在安裝 Tableau 之前，必須安裝和設定 PostgreSQL。

可在 Amazon RDS 或 EC2 執行個體上執行 PostgreSQL。有關在 RDS 與 EC2 執行個體上執行存放庫之間的差異詳情，請參閱 *Tableau Server 外部存放庫* (Linux)。

作為範例，以下過程顯示如何在 Amazon EC2 執行個體上安裝和設定 Postgres。此處顯示的範例是參考架構中 PostgreSQL 的通用安裝和設定。DBA 應根據您的資料大小和效能需求最佳化 PostgreSQL 部署。

要求：請注意，必須執行 PostgreSQL 1.6，並且必須安裝 uuid-ossp 模組。

PostgreSQL 版本

您必須為 Tableau Server 外部存放庫安裝相容的主要版本 PostgreSQL。此外，次要版本還必須滿足最低要求。

Tableau Server 版本	**PostgreSQL 最低相容版本**
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 2021.3.14 - 2021.3.15 2021.4.9 - 2021.4.10	12.10
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 - 2022.3.11 2023.1.5 - 2023.1.7	13.11

安裝 PostgreSQL

此範例安裝程序描述如何安裝 PostgreSQL 版本 13.6。

登入在之前「部分」中所建立的 EC2 主機。

1. 執行更新以將最新的修正套用到 Linux OS：

   ```
   sudo yum update
   ```

2. 在 `/etc/yum.repos.d/` 路徑中建立並編輯檔案 pgdg.repo.。將以下設定資訊填入檔案：

   ```
   [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. 安裝 Posgres 13.6：

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

4. 安裝 uuid-ossp 模組：

```
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
```

設定 Postgres

設定 Postgres 以完成基本安裝：

1. 請使用下列兩個項目來更新 pg_hba 組態檔，`/var/lib/pgsql/13/data/pg_hba.conf`。每個項目都必須包含將執行 Tableau Server 的子網路遮罩：

```
host all all 10.0.30.0/24 password

host all all 10.0.31.0/24 password
```

2. 要更新 Postgresql 檔，`/var/lib/pgsql/13/data/postgresql.conf`，請新增此行：

```
listen_addresses = '*'
```

3. 設定在重開機時啟動 Postgres：

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

4. 設定超級使用者密碼：

```
sudo su - postgres

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

> **附註**:設定強式密碼。切勿使用此處範例中所示的 'StrongPassword'。

```
exit
```

5. 重新啟動 Postgres:

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

使用 PostgreSQL 步驟 1 tar 備份

建立 PostgreSQL 設定的 tar 備份。如果在繼續部署時遇到故障,請建立當前設定的 tar 快照以節省時間。

我們將此稱為「步驟 1」備份。

在 PostgreSQL 主機上:

1. 停止 Postgres 資料庫執行個體:

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

2. 執行以下命令建立 tar 備份:

```
sudo su

cd /var/lib/pgsql

tar -cvf step1.13.bkp.tar 13

exit
```

3. 啟動 Postgres 資料庫:

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

還原步驟 1

如果 Tableau Server 初始節點在安裝過程中出現故障，則還原到步驟 1。

1. 在執行 Tableau 的電腦上，執行 obliterate 指令碼來完全移除主機中的 Tableau Server：

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

2. 還原 PostgreSQL Stage 1 tar。在執行 Postgres 的電腦上，執行以下命令：

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

繼續安裝 Tableau Server 初始節點的安裝流程。

安裝之前

如果根據本指南中描述的 AWS/Linux 範例實作部署 Tableau，則也許能夠執行自動安裝指令碼 TabDeploy4EDG。TabDeploy4EDG 指令碼會自動執行以下過程中描述的四節點 Tableau 部署的範例安裝。請參閱附錄 - AWS 部署工具箱。

安裝 Tableau Server 初始節點

此過程描述如何按照參考架構的定義，安裝 Tableau Server 的初始節點。除了安裝套件和初始化 TSM 之外，在此過程盡量使用 TSM 命令行。由於獨立於平台之外，使用 TSM CLI 能讓虛擬化和無頭環境的安裝更加順利。

執行安裝套件並初始化 TSM

登入到節點 1 主機伺服器。

1. 執行更新以將最新的修正套用到 Linux OS:

   ```
   sudo yum update
   ```

2. 從Tableau 下載頁面複製安裝組件到即將執行 Tableau Server 的主機電腦。

   例如，在執行 Linux RHEL-like 操作系統的電腦上，執行:

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

   其中，<version>是版本號。

3. 下載並安裝相依性:

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

4. 在根目錄中建立 /app/tableau_server 路徑:

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

5. 執行安裝程序並指定/app/tableau_server安裝路徑。例如，在類似於 Linux RHEL 的作業系統上，執行:

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

6. 更改為/app/tableau_server/packages/scripts.<version_code>/目錄並執行位於該處的 initialize-tsm指令碼:

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

7. 初始化完成後，離開 shell：

```
exit
```

啟用並註冊 Tableau Server。

1. 登入到節點 1 主機伺服器。

2. 在此步驟中提供 Tableau Server 產品金鑰。為您購買的每個授權金鑰執行以下命令：

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

3. 使用如下所示的格式建立一個 json 登錄檔：

```
{
"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. 儲存對檔案進行的變更之後，使用`--file`選項傳遞該檔案以註冊 Tableau Server：

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

設定身分識別存放區

> **附註**:如果部署使用 Tableau 檔案存放區的外部存放區,則需要在設定身分識別存放區之前啟用外部檔案存放區。請見*使用外部檔案存放區安裝 Tableau Server*。(Linux)。

參考架構預設使用本機身分識別存放區。要設定本機身分識別存放區的初始主機,請透過`tsm settings import`命令來傳遞`config.json`檔案。

根據作業系統匯入 `config.json` 檔:

`config.json`檔包含在 scripts.<version> 目錄路徑中(例如,`scripts.20204.21.0217.1203`),且其格式可用來設定識別身分存放區。

執行以下命令匯入`config.json`檔:

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

設定外部 Postgres

1. 使用下列組態設定建立外部資料庫 json 檔:

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

2. 儲存對檔案的變更後,使用以下命令傳遞檔案:

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

將會提示您輸入 Postgres 主要使用者名稱密碼。

選項 `--no-ssl` 將 Tableau 設定為僅在 Postgres 伺服器設定為使用 SSL/TLS 時使用 SSL/TLS。若 Postgres 沒有設定為使用 SSL/TLS，那麼連線不會被加密。第 6 部分 - 安裝後組態描述了在完成第一階段部署後如何為 Postgres 連線啟用 SSL/TLS。

3. 應用變更。

   執行此命令來套用變更並重新啟動 Tableau Server：

   ```
   tsm pending-changes apply
   ```

4. 刪除您在步驟 1 中使用的設定檔案。

完成節點 1 安裝

1. 安裝 Tableau Server 後，必須初始化伺服器。

   執行以下命令：

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

2. 初始化完成後，必須建立一個 Tableau Server 管理員帳戶。

   有別於用來安裝和管理 TSM 操作系統元件的電腦帳戶，Tableau Server 管理員帳戶是一種用來建立 Tableau Server 使用者、專案和網站的應用程式帳戶。Tableau Server 管理員還能將使用權限應用在 Tableau 資源。執行以下命令以建立初始管理員帳戶。在下面的範例中，使用者被稱為`tableau-admin`：

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

   Tabcmd 將提示您為此使用者設置密碼。

驗證:節點 1 組態

1. 執行以下命令以驗證 TSM 服務是否正在執行:

   ```
   tsm status -v
   ```

   Tableau 應傳回以下內容:

   ```
   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.
   ...
   ```

   將會列出所有服務。

2. 執行以下命令以驗證 Tableau 管理站點是否正在執行:

   ```
   curl localhost
   ```

   前幾行應該顯示 Vizportal html, 類似於:

   ```
   <!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 ...
```

進行步驟 2 tar 備份

驗證初始安裝後，請進行二個 tar 備份：

- PostgreSQL
- Tableau 初始節點（節點 1）

在大多數情況下，可透過還原這些 tar 檔案來恢復初始節點的安裝。還原 tar 檔案比將初始節點重新安裝和重新初始化要快得多。

建立步驟 2 tar 檔案

1. 在 Tableau 的初始節點上，停止 Tableau：

   ```
   tsm stop
   ```

   等待 Tableau 停止，然後再繼續下一步。

2. 在 PostgreSQL 主機上，停止 Postgres 資料庫執行個體：

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

3. 執行以下命令建立 tar 備份：

   ```
   sudo su
   cd /var/lib/pgsql
   tar -cvf step2.13.bkp.tar 13
   exit
   ```

4. 驗證 Postgres tar 檔案是使用 root 權限建立的：

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

5. 在 Tableau 主機上，停止 Tableau 管理服務：

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

6. 執行以下命令建立 tar 備份：

```
cd /data
```

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

7. 在 Postgres 主機上，啟動 Postgres 資料庫：

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

8. 啟動 Tableau 管理服務：

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

9. 在重新啟動之前執行 `tsm status` 命令以監測 TSM 狀態。

   在大多數情況下，該命令將先傳回 DEGRADED 或 ERROR 狀態。稍待片刻，然後再次執行該命令。如果傳回 ERROR 或 DEGRADED 狀態，則繼續等待。在傳回 STOPPED 狀態之前不要嘗試啟動 TSM。然後執行以下命令：

```
tsm start
```

還原步驟 2

此過程會將 Tableau 節點 1 和 Postgres 執行個體還原到步驟 2。還原到此步驟後，可以重新部署剩餘的 Tableau 節點。

1. 在初始 Tableau 主機（節點 1）上停止 tsm 服務：

```
tsm stop
```

2. 在 Tableau Server 部署的所有節點上停止 Tableau 管理服務。在每個節點上依次 (節點 1、節點 2、節點 3) 執行以下命令:

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

3. 在 Tableau 服務停止後, 還原 PostgreSQL 步驟 2 tar。在執行 Postgres 的電腦上, 執行以下命令:

   - ```
     sudo su
     ```

     ```
     systemctl stop postgresql-13
     ```

     ```
     cd /var/lib/pgsql
     ```

     ```
     tar  -xvf step2.13.bkp.tar
     ```

     ```
     systemctl start postgresql-13
     ```

     ```
     exit
     ```

4. 還原 Tableau 步驟 2 tar。在初始 Tableau 主機上, 執行以下命令:

   ```
   cd /data
   ```

   ```
   sudo rm -rf tableau_data
   ```

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

5. 在 Tableau 節點 1 電腦上, 移除下列檔案:

   - ```
     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. 啟動 Tableau 管理服務:

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

7. 重新載入 Tableau systmctl 檔，然後再次執行`start-administrative-services`：

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

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

8. 在節點 1 上，在重新啟動之前執行`tsm status`命令監測 TSM 狀態。

   在某些情況下，會收到錯誤訊息，`Cannot connect to server...`.出現此錯誤是因為 tabadmincontroller 服務尚未重新啟動。繼續定期執行`tsm status`。如果此錯誤在 10 分鐘後仍未消失，請再次執行`start-administrative-services`命令。

   片刻之後，`tsm status`命令將傳回 DEGRADED 狀態，然後傳回 ERROR。在傳回 STOPPED 狀態之前不要啟動 TSM。然後執行以下命令：

   ```
   tsm start
   ```

在其他節點繼續 Tableau Server 的安裝過程。

在其他節點安裝 Tableau Server

要繼續部署，請將 Tableau 安裝程式複製到每個節點。

Nodes 設定概觀

本節介紹設定節點 2-4的過程。以下各節提供了每個步驟的詳細設定和驗證過程。

Tableau Server 節點 2-4 的安裝要求您在節點安裝期間產生、複製和引用啟動程序檔案。

要產生啟動程序檔案，請在初始節點上執行 TSM 命令。然後，您會將啟動程序檔案複製到目標節點，並在其中將其當作節點初始化的一部分執行。

以下 json 內容顯示了啟動程序檔案範例。(為了使範例檔案易於閱讀，憑證與加密相關的數值已被截斷。)

```
{
"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"
       }
}
```

啟動程序檔案包括連線式的驗證，以對節點 1 進行身分驗證，並為啟動程序流程建立加密的管道。啟動程序工作階段是有時間限制的，設定和驗證節點非常耗時。設定節點時，計劃建立和複製新的啟動程序。

啟動程序檔案後，然後登入到初始 Tableau Server 節點並為新節點設定程序。完成節點設定後，必須套用變更並重新啟動初始節點。新節點已設定並啟動。新增節點時，部署的設定和重新啟動將繼續花費更長的時間才能完成。

整個 Linux 範例的安裝過程展示了 RHEL-like 發行版的命令。若執行的是 Ubuntu 發行版，請編輯相對應的命令。

1. 執行更新以將最新的修正套用到 Linux OS：

   ```
   sudo yum update
   ```

2. 下載並安裝相依性：

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

3. 在根目錄中建立 /app/tableau_server 路徑：

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

4. 執行安裝程式並指定/app/tableau_server安裝路徑。例如，在類似於 Linux RHEL 的作業系統上，執行：

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

產生、複製並使用啟動程序檔案以初始化 TSM

以下流程展示在另一個節點初始化 TSM 時，如何產生、複製並使用啟動程序檔案。在此範例中，啟動程序檔案稱為 boot.json。

在此範例中，主機在 AWS 中執行，其中 EC2 主機正在執行 Amazon Linux 2 。

1. 連線到初始節點 (節點 1) 並執行以下命令：

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

2. 複製啟動程序檔到節點 2 。

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

3. 連線到節點 2 並切換到 Tableau Server 指令碼目錄：

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

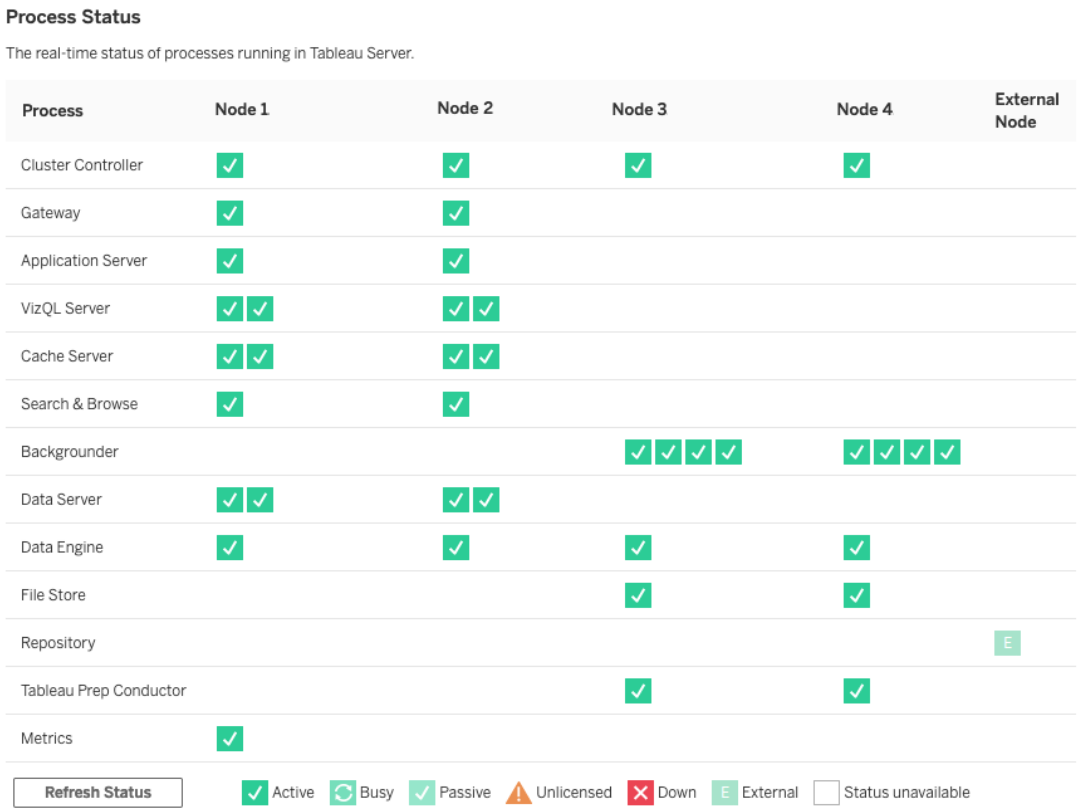
4. 執行 `initialize-tsm` 命令並參照啟動程序檔：

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

5. `initialize-tsm` 完成後，刪除 `boot.json`，然後結束或登出工作階段。

設定流程

您必須在執行 Tableau Server 管理控制項目（TSM 控制項目）的節點上設定 Tableau Server 叢集。TSM 控制項目在初始節點上執行。



設定節點 2

1. 在節點 2 上使用啟動程序檔初始化 TSM 後，請登入初始節點。

2. 要在節點 2 上設定程序，請在初始節點（`node1`）執行以下命令：

   ```
   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
   ```

   如果要安裝 2022.1 或更高版本，請同時新增索引和搜尋服務：

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

   如果要安裝版本 2023.3 或更高版本，請僅納入索引與搜尋服務：請勿新增搜尋與瀏覽 (searchserver) 服務

3. 在套用前檢視組態。執行以下命令：

   ```
   tsm pending-changes list
   ```

4. 確認您的變更在待處理清單中後（待處理清單中還有其他服務），請套用變更：

   ```
   tsm pending-changes apply
   ```

   所做的變更將需要重新啟動。設定與重新啟動會需要一些時間。

5. 驗證節點 2 組態。執行以下命令：

   ```
   tsm status -v
   ```

設定節點 3

在節點 3 上使用啟動程序初始化 TSM, 然後執行`tsm topology set-process`下面的命令。

每當您設定程序時, 都會顯示一個協調服務警告。您可以在設定程序時忽略此警告。

1. 在節點 3 上使用啟動程序檔案初始化 TSM 後, 請登入初始節點 (`node1`) 並執行以下命令來設定流程:

   ```
   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
   ```

   如果要安裝 2022.1 或更高版本, 請同時新增索引和搜尋服務:

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

2. 在套用前檢視組態。執行以下命令:

   ```
   tsm pending-changes list
   ```

3. 確認您的變更在待處理清單中後(該清單將包括其他自動設定的服務), 請套用變更:

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

   所做的變更將需要重新啟動。設定與重新啟動會需要一些時間。

4. 透過執行以下命令確認組態:

   ```
   tsm status -v
   ```

將協調服務組合部署到節點 1-3

對於標準參考架構四節點部署, 請執行以下流程:

1. 在節點 1 上執行以下命令:

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

   該流程包括重新啟動 TSM, 這需要一些時間。

2. 協調服務部署完成後, 啟動 TSM:

   ```
   tsm start
   ```

進行步驟 3 tar 備份

安裝驗證後, 請進行四個 tar 備份:

- PostgreSQL
- Tableau 初始節點(節點 1)
- Tableau 節點 2
- Tableau 節點 3

建立 Step 3 tar 檔案

1. 在 Tableau 的初始節點上, 停止 Tableau:

   ```
   tsm stop
   ```

2. TSM 停止後, 停止每個節點上的 Tableau 管理服務。在每個節點上依次(節點 1、節點 2 、節點 3) 執行以下命令:

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

3. 在 PostgreSQL 主機上, 停止 Postgres 資料庫執行個體:

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

4. 執行以下命令建立 tar 備份:

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

5. 驗證 Postgres tar 檔案是使用 root 權限建立的:

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

6. 在 Postgres 主機上, 啟動 Postgres 資料庫:

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

7. 在節點 1、節點 2 和節點 3 上建立 tar 備份。在每個節點上執行以下命令:

   - `cd /data`

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

   - 驗證是否使用 root 權限建立 Tableau tar 檔案:

     ```
     ls -al
     ```

8. 在每個節點上依次(節點 1、節點 2 、節點 3) 啟動 Tableau 管理服務 :

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

9. 在重新啟動之前執行 `tsm status` 命令以監測 TSM 狀態。

   在大多數情況下, 該命令將傳回 DEGRADED 或 ERROR 狀態。稍待片刻, 然後再次執行該命令。如果傳回 ERROR 或 DEGRADED 狀態, 則繼續等待。在傳回 STOPPED 狀態之前不要嘗試啟動 TSM。然後執行以下命令:

   ```
   tsm start
   ```

還原步驟 3

此過程將還原 Tableau 節點 1、節點 2 和節點 3。也會還原 Postgres 執行個體到步驟 3。還原到此步驟後，可以部署協調服務、節點 4，然後是最終節點組態。

1. 在初始 Tableau 主機（節點 1）上停止 tsm 服務：

   ```
   tsm stop
   ```

2. TSM 停止後，停止節點 1、節點 2 和節點 3 上的 Tableau 管理服務。在每個節點上執行以下命令：

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

3. 還原 PostgreSQL 步驟 3 tar。在執行 Postgres 的電腦上，執行以下命令：

   ```
   sudo su
   systemctl stop postgresql-12
   cd /var/lib/pgsql
   tar -xvf step3.12.bkp.tar
   systemctl start postgresql-12
   exit
   ```

4. 在節點 1、節點 2 和節點 3 上還原 Tableau 步驟 3 tar。在每個 Tableau 節點上執行以下命令：

   ```
   cd /data
   sudo rm -rf tableau_data
   sudo tar  -xvf step3.tableau_data.bkp.tar
   ```

5. 在 Tableau 節點 1 電腦上，移除下列檔案：

   - `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 傳回「找不到檔案」錯誤訊息，則可能需要變更路徑名稱以增加路徑此部分中的數字`<n>`：`.../appzookeeper/<n>/version-2/...`。

6. 重新啟動節點 1、節點 2 和節點 3 上的管理服務。在每個節點上執行以下命令：

```
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. 在節點 1 上，在重新啟動之前執行`tsm status`命令監測 TSM 狀態。

   在某些情況下，會收到錯誤訊息，`Cannot connect to server...`.出現此錯誤是因為 tabadmincontroller 服務尚未重新啟動。繼續定期執行`tsm status`。如果此錯誤在 10 分鐘後仍未消失，請再次執行`start-administrative-services`命令。

   片刻之後，`tsm status`命令將傳回 DEGRADED 狀態，然後傳回 ERROR。在傳回 STOPPED 狀態之前不要啟動 TSM。然後執行以下命令：

```
tsm start
```

在節點 1-3 上，繼續安裝流程以部署協調服務。

設定節點 4

節點 4 的設定過程與節點 3 相同。

將程序設定為與節點 3 設定的相同，執行與以上相同的命令集，但在命令中指定 node4 而不是 node3。

與節點 3 驗證一樣，透過執行tsm status -v來驗證節點 4 組態。

在繼續之前，請等待節點 4 上「檔案存放區」的過程完成同步。「檔案存放區」服務狀態將傳回is synchronizing直到它完成。當「檔案存放區」服務狀態傳回is running時，便可以繼續。

最終程序組態和驗證

程序設定的最後一步是從節點 1 刪除冗餘程序。

1. 連接到初始節點 (node1)。
2. 停用節點 1 上的檔案存放區。這將導致有關從相同位置的控制項目中刪除檔案存放區的警告。這個警告可以忽略。執行以下命令：

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

3. 在停止檔案存放區後，執行以下命令以從節點 1 中刪除後台流程：

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

4. 在套用前檢視組態。執行以下命令：

   ```
   tsm pending-changes list
   ```

5. 確認您的變更在待處理清單中之後，請套用變更：

   ```
   tsm pending-changes apply
   ```

   所做的變更將需要重新啟動。設定與重新啟動會需要一些時間。

6. 驗證組態：

   tsm status -v。

在繼續之前，請等待節點 4 上「檔案存放區」的過程完成同步。「檔案存放區」服務狀態將傳回`is synchronizing`直到它完成。當「檔案存放區」服務狀態傳回`is running`時，便可以繼續。

執行備份

完整復原 Tableau Server 所需的備份是由三個部分組合而成：

- 存放庫和檔案存放區資料的備份檔案。這個檔案是由`tsm maintenance backup`命令所產生的。
- 拓撲和組態匯出檔案。這個檔案是由`tsm settings export`命令所產生的。
- 身份驗證憑證、金鑰和 keytab 檔案。

有關備份和還原過程的完整說明，請參閱 Tableau Server 主題*執行 Tableau Server 的完整備份和還原*(Linux)。

在部署的這個階段，請執行`tsm maintenance backup`和`tsm settings export`命令，就能包含完整還原所需的所有相關檔案和資產。

1. 執行以下命令將組態和拓撲設定匯出到名為`ts_settings_backup.json`的檔案

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

2. 建立存放庫和檔案存放區資料的備份，請在名為`ts_backup-<yyyy-mm-dd>.tsbak`的檔案中執行以下命令忽略有關檔案存放區不在控制項目節點上的警告。

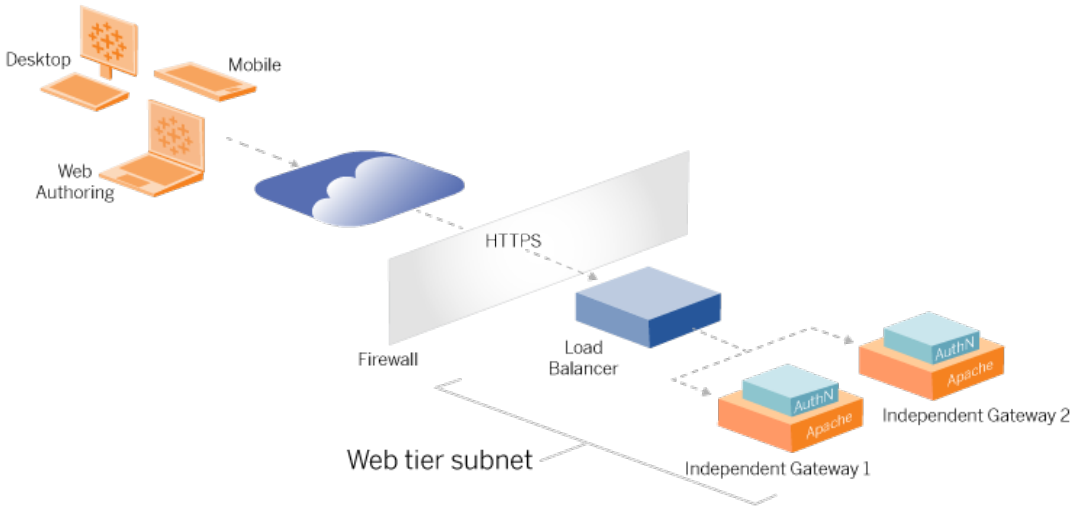
   ```
   tsm maintenance backup -f ts_backup -d --skip-compression
   ```

   備份檔的位置：

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

3. 複製這兩個檔案並保存在 Tableau Server 部署未共享的其他儲存資產上。

第 5 部分 - 配置 Web 層



參考架構的 Web 層應包括以下組件：

- 面向 Web 的應用程式負載平衡器，它接受來自 Tableau 用戶端的 HTTPS 請求並與反向 proxy 伺服器進行通訊。
- 反向 proxy：
  - 建議部署 Tableau Server 獨立閘道。
  - 建議至少使用兩個 proxy 伺服器為冗餘和處理用戶端加載。
  - 從負載平衡器接收 HTTPS 流量。
  - 支援 Tableau 主機的相黏工作階段。
  - 為執行閘道過程的每個 Tableau Server 設定循環負載平衡 proxy。
  - 處理來自外部 IdP 的身份驗證請求。
- 正向 Proxy：Tableau Server 需要存取網際網路獲得授權與對應功能。您必須為 Tableau 服務 URL 設定正向 proxy 允許清單。請參閱 *與網際網路通訊*（ Linux）。
- 所有與用戶端相關的流量都可以透過 HTTPS 加密：
  - 用戶端到應用程式負載平衡器
  - 應用程式負載平衡器到反向 proxy 伺服器
  - Proxy 伺服器到 Tableau Server
  - 在反向 proxy 上執行到 IdP 的身份驗證處理程序
  - Tableau Server 到 IdP

Tableau Server 獨立閘道

Tableau Server 版本 2022.1 中引入了 Tableau Server 獨立閘道。獨立閘道是 Tableau 閘道流程的獨立執行個體，用作 Tableau 感知反向 Proxy。

獨立閘道支援到後端 Tableau Server 的簡單循環配置資源負載平衡。但是，獨立閘道並非旨在充當企業應用程式負載平衡器。我們建議在企業級應用程式負載平衡器後面執行獨立閘道。

獨立閘道需要 Advanced Management 授權。

驗證和授權

預設參考架構會指定安裝 Tableau Server 並設定本機驗證。在此模型中，用戶端必須連線到 Tableau Server 才能透過本機 Tableau Server 本機驗證流程進行驗證。不建議在參考架構中使用此驗證方法，因為該情境要求未經驗證的用戶端與應用程式層通訊，這存在安全風險。

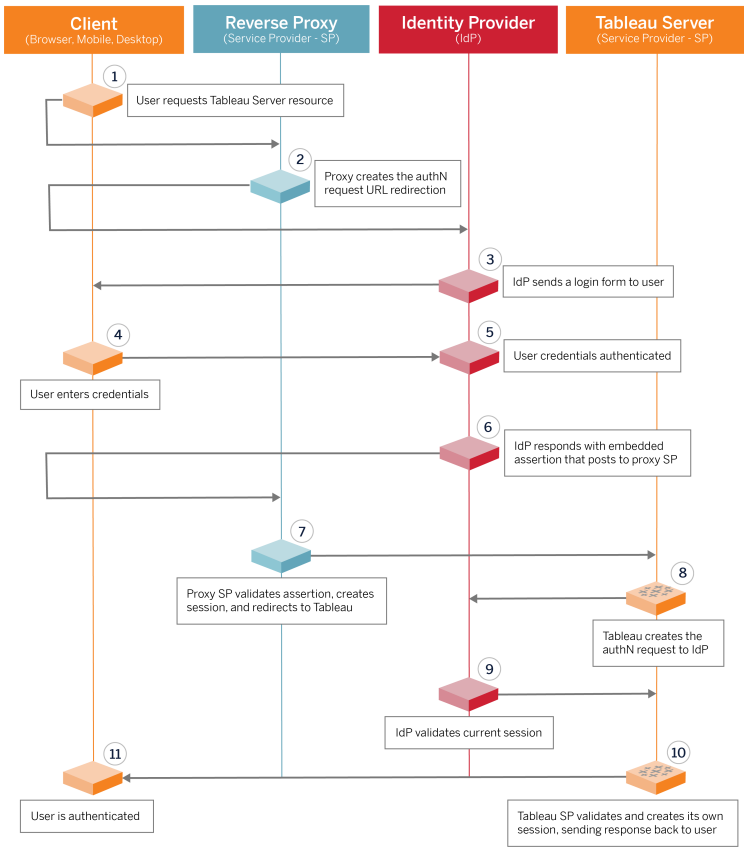
相反，我們建議設定企業級外部身分識別提供者和 AuthN 模組，以預先驗證所有到應用程式層的流量。使用外部 IdP 設定時，不使用本機 Tableau Server 本機驗證流程。在 IdP 對使用者進行驗證後，Tableau Server 會授權存取部署中的資源。

使用 AuthN 模組進行預先驗證

在本指南中記錄的範例中，已設定 SAML SSO，但可以使用大多數外部身分識別提供者和 AuthN 模組設定預先驗證流程。

在參考架構中，反向 Proxy 設定為在 Proxy 處理這些對 Tableau Server 的請求之前與 IdP 建立用戶端驗證工作階段。我們將此過程稱為*預驗證*階段。反向 Proxy 只會將經過驗證的用戶端工作階段重新導向到 Tableau Server。然後，Tableau Server 將建立一個工作階段，使用 IdP 驗證工作階段的身分驗證，然後傳回用戶端請求。

下圖顯示了已設定 AuthN 模組的預先驗證和驗證流程的分步詳細資訊。反向 Proxy 可能為泛型協力廠商解決方案或 Tableau Server 獨立閘道：



設定概觀

以下為設定 Web 層的過程概觀。在每個步驟之後驗證連線：

1. 設定兩個反向 proxy 以提供對 Tableau Server 的 HTTP 存取。
2. 在 proxy 伺服器上使用相黏工作階段設定負載平衡邏輯，以連線到執行閘道處理序的每個 Tableau Server 執行個體。
3. 在 Internet 閘道使用相黏工作階段設定套用應用程式負載平衡，以將請求轉傳到反向 proxy 伺服器。
4. 使用外部 IdP 設定身份驗證。可以在反向 proxy 伺服器上安裝身份驗證處理程式來設定 SSO 或 SAML。AuthN 模組管理外部 IdP 和 Tableau 部署之間的驗證信號交換。Tableau 還將充當 IdP 服務提供者並使用 IdP 對使用者進行身份驗證。
5. 要在此部署中使用 Tableau Desktop 進行身份驗證，用戶端必須執行 Tableau Desktop 2021.2.1 或更高版本。

使用 Tableau Server 獨立閘道設定 Web 層的範例

本主題的其餘部分會介紹一個端到端的過程，該過程會描述如何使用 Tableau Server 獨立閘道在範例 AWS 參考架構中實作 Web 層。有關使用 Apache 作為反向 Proxy 的範例設定，請參閱附錄 - 具有 Apache 示例部署的 Web 層。

範例設定由以下元件組成：

- AWS 應用程式負載平衡器
- Tableau Server 獨立閘道
- Mellon 驗證模組
- Okta IdP
- SAML 驗證

> **附註**：本節中提供的範例 Web 層設定包括部署協力廠商軟體和服務的詳細程序。我們已盡最大努力驗證和記錄啟用 Web 層方案的過程。但是，協力廠商軟體可能會發生變化，或者您的方案可能與此處描述的參考架構不同。權威設定細節和支援請參考協力廠商文件。

整個區段中的 Linux 範例展示了 RHEL-like 發行版的命令。具體來說，這裡的命令是使用 Amazon Linux 2 發行版開發的。若執行的是 Ubuntu 發行版，請編輯相對應的命令。

在此範例中部署 Web 層遵循逐步設定和驗證程序。核心 Web 層設定包含以下步驟，用於在 Tableau 和 Internet 之間啟用 HTTP。獨立閘道執行並設定為在 AWS 應用程式負載均衡器後面進行反向 proxy/負載均衡：

1. 準備環境
2. 安裝獨立閘道
3. 設定獨立閘道伺服器
4. 配置 AWS 應用程式負載均衡器

設定 Web 層並驗證與 Tableau 的連線後，使用外部提供者設定身份驗證。

準備環境

在部署獨立閘道之前完成以下工作。

1. AWS 安全性群組變更。將公共安全群組設定為允許來自私人安全群組的傳入獨立閘道內務管理流量 (TCP 21319)。
2. 如第 4 部分 - 安裝並設定 Tableau Server中所述，在四節點 Tableau Server 叢集上安裝版本 22.1.1(或更高版本) 。
3. 如設定主機電腦中所述，在公共安全性群組中設定兩個 Proxy EC2 執行個體。

安裝獨立閘道

Tableau Server 獨立閘道需要 Advanced Management 授權。

部署 Tableau Server 獨立閘道包括安裝和執行 .rpm 套件，然後設定初始狀態。本指南中包含的程序為部署到參考架構中提供了說明性指導。

若您的部署與參考架構不同，請參閱核心 Tableau Server 文件*使用獨立閘道安裝 Tableau Server* (Linux)。

> **重要提示**：設定獨立閘道是可能出錯的程序。跨獨立閘道伺服器的兩個執行個體對設定問題進行疑難排解非常困難。因此建議一次設定一個獨立閘道伺服器。設定第一台伺服器並驗證功能後，應設定第二台獨立閘道伺服器。

儘管您將分別設定每個獨立閘道伺服器，請在安裝到公共安全群組中的兩個 EC2 執行個體中執行此安裝過程：

1. 執行更新以將最新的修正套用到 Linux OS：

   ```
   sudo yum update
   ```

2. 如果安裝到 Apache，請將其移除：

```
sudo yum remove httpd
```

3. 將版本 2022.1.1(或更高版本)獨立閘道安裝套件從 Tableau 下載頁面複製到將執行 Tableau Server 的主機電腦。

   例如,在執行 Linux RHEL-like 操作系統的電腦上,執行:

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

4. 執行安裝程式:例如,在類似於 Linux RHEL 的作業系統上,執行:

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

5. 變更為 `/opt/tableau/tableau_tsig/packages/scripts.<version_code>/` 目錄,並執行位於該處的 `initialize-tsig` 指令碼。除了 `--accepteula` 標幟外,還必須包括執行 Tableau Server 部署的子網路之 IP 範圍。使用 `-c` 選項指定 IP 範圍。下面的範例顯示了包含指定範例 AWS 子網路的命令:

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

6. 初始化完成後,開啟 `tsighk-auth.conf` 檔案,並複製檔案中的驗證密碼。作為後端 Tableau Server 設定的一部分,需要為每個獨立閘道執行個體提交此程式碼:

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

7. 在獨立閘道的兩個執行個體上執行上述步驟後,準備 `tsig.json` 設定檔。設定檔由「independentGateways」陣列組成。該陣列包含設定物件,每個物件定義獨立閘道執行個體的連線詳細資訊。

   複製以下 JSON,並根據部署環境對其進行自訂。此處的範例顯示了範例 AWS 參考架構的檔案。

下面的示例 JSON 檔案僅包含一個獨立閘道連線資訊。在此程序的後續部分，您將包括第二個獨立閘道伺服器連線資訊。

將檔案儲存為 tsig.json，以用於以下程序。

```
{
"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"- 執行獨立閘道的 AWS EC2 執行個體的私人 DNS 名稱。
- "host" - 與 "id" 相同。
- "port"- 內務處理連接埠，預設為 "21319"。
- "protocol"- 用戶端流量的通訊協定。將此保留為 http，用戶端流量的通訊協定。
- "authsecret"- 在上一步中複製的密碼。

獨立網關：直接與轉送連線

在繼續之前，必須決定在部署中設定哪種連線配置：直接連線或轉送連線。此處簡要描述了每個選項以及相關的決策資料點。

轉送連線：可以將獨立閘道設定為透過單個連接埠將用戶端通訊轉送到 Tableau Server 上的閘道流程。我們將此稱為*轉送*連線：

- 轉送流程會導致從獨立閘道到後端 Tableau Server 閘道流程的過程中產生額外的躍點。與直接連線設定相比，額外的躍點會降低效能。
- 轉送模式支援 TLS。轉送模式下的所有通訊都僅限於單一通訊協定（HTTP 或 HTTPS），因此可以使用 TLS 進行加密和驗證。

直接連線：獨立閘道可以透過多個連接埠直接與後端 Tableau Server 流程通訊。我們將這種通訊稱為*直接*連線：

- 由於直接連線到後端 Tableau Server，因此與轉送連線選項相比，用戶端效能顯著提高。
- 需要開啟 16 個以上從公共子網路到私人子網路的連接埠，以便實現從獨立閘道到 Tableau Server 電腦的直接流程通訊。
- 從獨立閘道到 Tableau Server 的流程尚不支援 TLS。

設定轉送連線

要在 Tableau Server 和 Independent Gateway 之間執行 TLS，您必須設定中繼連線。EDG 中的範例場景設定中繼連線。

1. 將 `tsig.json` 複製到 Tableau Server 部署的節點 1。

2. 在節點 1 上執行以下命令，以啟用獨立閘道。

```
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
```

設定直接連線

由於直接連線不支援 TLS，因此我們建議僅在能夠以其他方式保護所有網路流量的情況下才設定直接連線。要在 Tableau Server 和 Independent Gateway 之間執行 TLS，您必須設定中繼連線。EDG 中的範例場景設定中繼連線。

若將獨立閘道設定為直接連線到 Tableau Server，則必須啟用設定以觸發通訊。Tableau Server 與獨立閘道通訊後，將建立通訊協定目標。然後，必須從獨立閘道電腦中擷取 `proxy_targets.csv`，並開啟從 AWS 中公共安全性群組到私人安全性群組的相應連接

埠。

1. 將 `tsig.json` 複製到 Tableau Server 部署的節點 1。

2. 在節點 1 上執行以下命令，以啟用獨立閘道。

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

3. 在獨立閘道電腦上執行以下命令，以檢視 Tableau Server 叢集正在使用的連接埠：

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

4. 設定 AWS 安全性群組。新增 `proxy_targets.csv` 中列出的 TCP 連接埠，以允許從公共安全性群組到私人安全性群組的通訊。

   建議自動化連接埠輸入設定，因為若 Tableau Server 部署發生變更，連接埠可能會變更。在 Tableau Server 部署上新增節點或重新設定流程將觸發對獨立閘道所需的連接埠存取權的變更。

驗證：基本拓撲組態

應該能夠藉由瀏覽至 `http://<gateway-public-IP-address>`.來存取 Tableau Server 管理員頁面。

如果 Tableau Server 登入頁面未載入，或者 Tableau Server 未啟動，請執行以下疑難排解步驟：

網路

- 在 Tableau Server 節點 1 執行以下 `wget` 命令來驗證 Tableau 部署和 Independent Gateway 執行個體之間的連線：wget http://<internal IP address of Independent Gateway> :21319，例如：

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

如果連線遭拒或失敗，則驗證公用安全性群組是否設定為允許來自私有安全性群組的 Independent Gateway 內務處理流量 (TCP 21319)。

如果安全性群組設定正確，則驗證是否在 Independent Gateway 初始化期間指定正確的 IP 位址或 IP 範圍。您可以檢視並變更位於 `/etc/opt/tableau/tableau_tsig/environment.bash` 的 `environment.bash` 檔案設定。如果變更此檔案，則依照如下所述重新啟動 tsig-http 服務。

在 Proxy 1 主機上：

1. 用 Independent Gateway 存虛設常式覆寫 httpd.conf 檔案：

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

2. 重新啟動 tsig-httpd 作為疑難排解的第一步：

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

在 Tableau 節點 1

- 仔細檢查 `tsig.json` 檔案。若發現錯誤，請進行修復，然後執行 `tsm topology external-services gateway update -c tsig.json`。
- 若執行直接連線，請驗證 `proxy_targets.csv` 中列出的 TCP 連接埠是否設定為從公共到私人安全性群組的輸入連接埠。

配置 AWS 應用程式負載均衡器

將負載平衡器設定為 HTTP 偵聽器。此處的過程描述瞭如何在 AWS 中新增負載平衡器。

步驟 1：建立目標群組

目標群組是定義執行 Proxy 伺服器的 EC2 實例的 AWS 配置。這些是來自 LBS 的流量的目標。

1. EC2 > **目標群組** > **建立目標群組**

2. 在建立頁面上:

   - 輸入目標群組名稱,例如 `TG-internal-HTTP`
   - 目標類型:執行個體
   - 協定:HTTP
   - 連接埠:80
   - VPC:選取您的 VPC
   - 在**健康情況檢查>進階健康情況檢查設定>成功代碼**下,附加代碼清單以讀入:`200,303`。
   - 按一下**「建立」**

3. 選取剛剛建立的目標群組,然後按一下**「目標」**索引標籤:

   - 按一下 **[編輯]**。
   - 選取要執行 Proxy 應用程式的 EC2 執行個體(或單個執行個體,如果一次設定一個),然後按一下**新增到已註冊**。
   - 按一下**「儲存」**。

步驟 2:啟動負載均衡器精靈

1. EC2 > **負載平衡器** > **建立負載平衡器**

2. 在「選取負載平衡器類型」頁面上,建立一個應用程式負載均衡器。

> **附註**:為設定負載平衡器而顯示的 UI 在 AWS 資料中心之間不一致。下面的「精靈設定」過程與從**步驟 1 設定負載平衡器**開始的 AWS 設定精靈一致。
>
> 如果資料中心在底部包含**建立負載平衡器**按鈕的單個頁面中顯示所有設定,請按照下方「單一頁面設定」過程進行操作。

精靈設定

1. **設定負載平衡器**頁面:

    - 指定名稱
    - 結構:以網際網路為對象(預設)
    - IP 位址類型:ipv4(預設)
    - 接聽程式(接聽程式和路由):
        a. 保留預設的 HTTP 接聽程式
        b. 按一下**新增接聽程式**並新增 `HTTPS:443`
    - VPC:選取已安裝所有內容的 VPC
    - 可用區域:
        - 為資料中心區域選取 **a** 和 **b**
        - 在每個相應的下拉式選取器中,選取公用子網路(proxy 伺服器所在位置)。
    - 按一下:**設定安全設定**

2. **設定安全性設定**頁面

    - 上傳公用 SSL 憑證。
    - 按一下**「下一步:設定安全群組」**。

3. **設定安全性群組**頁面:

    - 選取公用安全性群組。如果選取預設安全性群組,則清除該選取。
    - 按一下**「下一步:設定路由」**。

4. **設定路由**頁面

    - 目標群體:現有的目標群體。
    - 名稱:選取之前建立的目標群組
    - 按一下**「下一步:註冊目標」**。

5. **註冊目標**頁面

    - 應顯示之前設定的兩個 proxy 伺服器執行個體。
    - 按一下**「下一步:檢閱」**:。

6. **評論**頁面

    按一下**「建立」**。

單一頁面設定

基本設定

- 指定名稱
- 結構:以網際網路為對象(預設)
- IP 位址類型:ipv4(預設)

網路對應

- VPC:選取已安裝所有內容的 VPC
- 對應:
  - 為您的數據中心區域選擇**a**和**b**(或類似的)「可用區域」
  - 在每個相應的下拉式選取器中,選取公用子網路(proxy 伺服器所在位置)。

安全性群組

選取公用安全性群組。如果選取預設安全性群組,則清除該選取。

接聽程式和路由

- 保留預設的 HTTP 接聽程式。對於**預設動作**,請指定之前設定的目標群組。
- 按一下**新增接聽程式**並新增 `HTTPS:443`。對於**預設動作**,請指定之前設定的目標群組。

安全接聽程式設定

- 上傳公用 SSL 憑證。

按一下**建立負載平衡器**。

步驟 3:啟用綁定

1. 建立負載均衡器後,必須在目標群組上啟用綁定。
   - 開啟 AWS 目標群組頁面(**EC2> 負載平衡> 目標群組**),選取剛剛設定的目標群組執行個體。在**「動作」**功能表上,選取**「編輯屬性」**。
   - 在**「編輯屬性」**頁面上,選取**「綁定」**,指定持續時間為 `1 day`,然後**儲存變更**。

2. 在負載均衡器上，在 HTTP 偵聽器上啟用綁定。選取剛剛設定的負載均衡器，然後按一下**「偵聽器」**索引標籤：
    - 對於 **HTTP:80**，按一下**「檢視/編輯規則」**。在產生的**「規則」**頁面上，按一下編輯圖示（原本位於頁面頂端，然後接著位於規則旁邊）以編輯規則。刪除現有 THEN 規則並按一下**新增動作>轉傳至...**以取代它。在產生的 THEN 設定中，指定建立的相同目標群組。在群組層級綁定下，啟用綁定並將持續時間設定為 1 天。儲存設定然後按一下**「更新」**。

步驟 4：在負載平衡器上設定空閒逾時

在負載平衡器上，將空閒逾時更新為 400 秒。

選取為此部署設定的負載平衡器，然後按一下**動作 > 編輯屬性**。將**空閒逾時**設定為 400 秒，然後按一下**儲存**。

步驟 5：驗證 LBS 連線

打開 AWS 負載平衡器頁面（**EC2>負載平衡器**），選取剛剛設定的負載平衡器實例。

在**「描述」**下，複製 DNS 名稱並將其貼上到瀏覽器中以存取 Tableau Server 登入頁面。

如果收到 500 級錯誤，那可能需要重新啟動 Proxy 伺服器。

使用公用 Tableau URL 更新 DNS

使用 AWS 負載均衡器描述中的網域 DNS 區域名稱在 DNS 中建立 CNAME 值。流向您的 URL (tableau.example.com) 的流量應傳送到 AWS 公用 DNS 名稱。

驗證連線

DNS 更新完成後，應該能夠輸入公用 URL 巡覽到 Tableau Server 登入頁面，例如，`https://tableau.example.com`。

身份驗證組態範例:帶有外部 IdP 的 SAML

以下範例介紹如何為 AWS 參考架構中執行的 Tableau 部署,設定和配置帶有 Okta IdP 跟 Mellon 驗證模組的 SAML。

此範例取自上一區段,並假設您一次設定一個獨立閘道。

該範例會介紹如何透過 HTTP 設定 Tableau Server 和獨立閘道。Okta 將以 HTTPS 向 AWS 負載均衡器傳送請求,但所有內部流量都將以 HTTP 傳輸。在為此案例進行設定時,請在設定 URL 字串時注意 HTTP 與 HTTPS 協定。

此範例使用 Mellon 作為 Independent Gateway 伺服器上的預先身份驗證服務提供者模組。此設定可確保只有經過身份驗證的流量連線到 Tableau Server, Tableau Server 還充當 Okta IdP 的服務提供者。因此,必須設定兩個 IdP 應用程式:一個用於 Mellon 服務提供者,另一個用於 Tableau 服務提供者。

建立 Tableau 管理員帳戶

配置 SAML 時的一個常見錯誤是在啟用 SSO 之前忘記在 Tableau Server 上建立管理員帳戶。

第一步是在 Tableau Server 上建立一個具有伺服器管理員角色的帳戶。在 Okta 情境中,使用者名稱必須採用有效電子郵件位址的形式,例如 user@example.com。必須為此使用者設定密碼,但在設定 SAML 後將不再使用該密碼。

配置 Okta 預身分驗證應用程式

本區段描述的端到端方案需要兩個 Okta 應用程式:

- Okta 預身分驗證應用程式
- Okta Tableau Server 應用程式

這些應用程式都與您需要分別在反向 proxy 和 Tableau Server 上設定的不同中繼資料關聯。

此過程描述如何建立和設定 Okta 預身份驗證應用程式。在本主題的後面，您將建立 Okta Tableau Server 應用程式。有關使用者受限的免費測試 Okta 帳戶，請參閱Okta 開發者網頁。

為 Mellon 預先身份驗證服務提供者建立 SAML 應用程式整合。

1. 打開 Okta 管理儀表板 >**應用程式**>**建立 App 集合**。

2. 在**新建 app 集合**頁面上，選擇**SAML 2.0**，然後按一下**「下一步」**。

3. 在**「一般設定」**索引標籤上，輸入應用程式名稱，例`Tableau Pre-Auth`，然後按一下**「下一步」**。

4. 在**設定 SAML**索引標籤上：

   - 單一登入 (SSO) URL。單一登入 URL 路徑的最後一個元素是指接在此程序之後，在`mellon.conf`組態檔中的`MellonEndpointPath`。可以指定想要的任何端點。在這個範例中，`sso`為端點。最後一個元素，`postResponse` 是必需的：`https://tableau.example.com/sso/postResponse`。
   - 清除核取方塊：**將此用於收件者 URL 和終點 URL**。
   - 收件者 URL：與 SSO URL 相同，但使用 HTTP。例如，`http://tableau.example.com/sso/postResponse`。
   - 目的地 URL：與 SSO URL 相同，但使用 HTTP。例如，`http://tableau.example.com/sso/postResponse`。
   - 受眾 URI(SP 實體 ID)。例如，`https://tableau.example.com`。
   - 名稱 ID 格式：`EmailAddress`
   - 應用程式使用者名稱：`Email`
   - 屬性聲明：名稱 = `mail`；名稱格式 = `Unspecified`；值 = `user.email`。

   按一下**「下一步」**。

5. 在**「回饋」**索引標籤上，選取：

   - **我是新增內部應用程式的 Okta 客戶**
   - **這是我們建立的內部應用程式**
   - 按一下**「完成」**。

6. 建立預授權 IdP 中繼資料檔案：

- 在 Okta 中：**應用程式>應用程式>**您的新應用程式（例如 `Tableau Pre-Auth`）**>登入**
- 在 **SAML 簽署憑證**旁邊，按一下**檢視 SAML 設定說明**。
- 在**如何為<預授權>應用程式設定 SAML 2.0** 頁面上，向下捲動到**可選**部分，**向您的 SP 提供者提供以下 IDP 中繼資料**。
- 複製 XML 欄位的內容，並將它們儲存在名為 `pre-auth_idp_metadata.xml` 的檔案中。

7. （可選）設定多重要素驗證：

   - 在 Okta 中：**應用程式>應用程式>**您的新應用程式（例如 `Tableau Pre-Auth`）**>登入**
   - 在**「登入原則」**下，按一下**「新增規則」**。
   - 在**「應用程式登入規則」**上，指定名稱和不同 MFA 選項。要測試功能，可以將所有選項保留為預設值。但是，在**「動作」**下，必須選取**「提示要素」**，然後指定使用者必須登入的頻率。按一下**「儲存」**。

建立和指派 Okta 使用者

1. 在 Okta 中，使用在 Tableau 中建立的相同使用者名稱建立一個使用者 (user@example.com)：**「目錄」>「人員」>「新增人員」**。
2. 建立使用者後，將新的 Okta 應用程式指派給該人員：按一下使用者名稱，然後在**「指派應用程式」中指派應用程式**。

安裝 Mellon 進行預身份驗證

此範例使用流行的開源模組 mod_auth_mellon。一些 Linux 發行版從較舊的存放庫中封裝了過時的 mod_auth_mellon 版本。這些過時的版本可能包含未知的安全性漏洞或功能問題。若選擇使用 mod_auth_mellon，請檢查您使用的是目前版本。

mod_auth_mellon 模組是協力廠商軟體。我們已盡最大努力驗證和記錄啟用此方案的程序。但是，協力廠商軟體可能會發生變更，或者您的方案可能與此處描述的參考架構不同。權威設定細節和支援請參考協力廠商文件。

1. 在執行 Independent Gateway 的使用中 EC2 執行個體上，安裝目前版本的 Mellon 驗證模組。

2. 建立/etc/mellon目錄:

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

將 Mellon 設定為預身份驗證模組

在 Independent Gateway 的第一個執行個體上執行此過程。

您必須擁有從 Okta 設定建立的 pre-auth_idp_metadata.xml 檔案複本。

1. 變更目錄:

   ```
   cd /etc/mellon
   ```

2. 建立服務提供者後設資料。執行 mellon_create_metadata.sh 指令碼。命令中必須包含您組織的實體 ID 和返回 URL。

   在 Okta 中，返回 URL 被稱為*單一登入 URL*。返回 URL 路徑的最後一個元素是指接在此程序之後，在mellon.conf組態檔中的MellonEndpointPath。在這個範例中，我們指定sso作為路徑終點。

   例如:

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

   該指令碼回傳服務提供者憑證、金鑰和後設資料檔。

3. 為了容易閱讀，請重新命名mellon目錄中的服務提供者檔案。我們將使用以下名稱來引用文件中的檔案:

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

4. 將 pre-auth_idp_metadata.xml 檔案複製到同一目錄中。

5. 變更 /etc/mellon 目錄中所有檔案的擁有權和權限:

```
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. 建立/etc/mellon/conf.d目錄:

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

7. 在 /etc/mellon/conf.d 目錄中建立 global.conf 檔案。

   如下所示複製檔案內容,但使用您的根網域名稱更新 MellonCookieDomain。例如,若 Tableau 的網域名稱是 tableau.example.com, 請輸入 example.com 作為根網域。

```
<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 目錄中建立 mellonmod.conf 檔案。

該檔案包含指定 `mod_auth_mellon.so` 檔案位置的單一指示詞。此處範例中的位置為檔案的預設位置。請驗證檔案是否在此位置，或變更此指示詞中的路徑以符合 `mod_auth_mellon.so` 的實際位置：

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

在 Okta 中建立 Tableau Server 應用程式

1. 在 Okta 儀表板中：**「應用程式」>「應用程式」>「瀏覽應用程式目錄」**
2. 在**瀏覽應用程式整合目錄**中，搜尋 `Tableau`，選取 Tableau 伺服器 動態磚，然後按一下**「新增」**。
3. 在**「新增 Tableau Server」>「一般設定」**上，輸入標籤，然後按一下**「下一步」**。
4. 在登入選項中，選取 **SAML 2.0** 並向下滾動到進階登入設定：
   - **SAML 實體 ID**：輸入公用 URL，例如 https://tableau.example.com。
   - **應用程式使用者名稱格式**：Email
5. 按一下**「身份提供者中繼資料」**連結以啟動瀏覽器。複製瀏覽器連結。這是在以下過程中設定 Tableau 時將使用的連結。
6. 按一下 **[完成]**。
7. 將新的 Tableau Server Okta 應用程式指派給您的使用者 (user@example.com)：按一下使用者名稱，然後在**「指派應用程式」**中指派應用程式。

在 Tableau Server 上設定驗證模組設定

在 Tableau Server 節點 1 上執行以下命令。這些命令指定遠端 Independent Gateway 電腦上 Mellon 設定檔的檔案位置。仔細檢查這些命令中指定的檔案路徑是否對應到遠端 Independent Gateway 電腦上的路徑和檔案位置。

```
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
```

為減少停機時間，請在啟用 SAML 之前不要套用變更，如下一節所述。

在 Tableau Server 上為 IdP 啟用 SAML

在 Tableau Server 節點 1 上執行此過程。

1. 從 Okta 下載 Tableau Server 應用程式中繼資料。使用在上一過程中儲存的連結：

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

2. 將 TLS 憑證和相關金鑰檔案複製到 Tableau Server。金鑰檔案必須是 RSA 金鑰。有關 SAML 憑證和 IdP 要求詳細資訊，請參閱 *SAML 要求* (Linux)。

   為簡化憑證管理和部署，並作為安全性最佳做法，我們建議使用由受信任的主要協力廠商憑證頒發授權單位 (CA) 產生的憑證。或者，您可以產生自我簽署憑證或使用針對 TLS 的 PKI 憑證。

   如果您沒有 TLS 憑證，則可以使用下面的內嵌過程產生自簽章憑證。

產生自簽章憑證

在 Tableau Server 節點 1 上執行此過程。

a. 產生簽署根憑證頒發授權單位 (CA) 金鑰：

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

b. 建立根 CA 憑證：

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

   系統將提示輸入憑證欄位的值。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. 建立憑證和相關金鑰(下方範例中的 `server-saml.csr` 和 `server-saml.key`)。憑證的主題名稱必須與 Tableau 主機的公用主機名稱相符。主題名稱設定為 `-subj` 選項,帶有格式 `"/CN=<host-name>"`,例如:

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

d. 使用您在上述步驟中建立的 CA 憑證簽署新憑證。以下命令也將以 `crt` 格式輸出憑證:

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

e. 將金鑰檔案轉換為 RSA。Tableau 需要用於 SAML 的 RSA 金鑰檔案。若要轉換金鑰,請執行以下命令:

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

3. 配置 SAML。執行以下命令,指定您的實體 ID 和傳回 URL,以及中繼資料檔、憑證檔案和金鑰檔案的路徑:

```
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. 如果您的組織執行 Tableau Desktop 2021.4 或更高版本，則必須執行以下命令以用反向 proxy 伺服器啟用身份驗證。

   Tableau Desktop 2021.2.1 到 2021.3 的版本無須執行此命令即可執行，前提是您的預先身份驗證模組（例如 Mellon）設定為允許保留頂層網域 cookie。

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

5. 套用組態變更：

   ```
   tsm pending-changes apply
   ```

重新啟動 tsig-httpd 服務

Tableau Server 部署套用變更時，重新登入到 Tableau Server Independent Gateway 電腦，並執行以下命令，以重新啟動 tsig-httpd 服務：

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

驗證 SAML 功能

要驗證端到端 SAML 功能，請使用您在此過程開始時建立的 Tableau 管理員帳戶，透過公用 URL（例如 https://tableau.example.com）登入到 Tableau Server。

如果 TSM 未啟動（「閘道錯誤」），或者在嘗試連線時遭遇瀏覽器錯誤，請參見 解除安裝 Tableau Server 獨立閘道。

設定 Independent Gateway 的第二個執行個體

成功設定獨立閘道的第一個執行個體後，部署第二個執行個體。此處的範例是安裝本主題中描述的 AWS/Mellon/Okta 方案的最終過程。該過程假定您已在第二個執行個體中安裝了獨立閘道，如本主題前面所述（安裝獨立閘道）。

部署第二個獨立閘道的過程需要以下步驟:

1. 在獨立閘道的第二個執行個體中:安裝 Mellon 驗證模組。

   不要按照本主題前面的描述設定 Mellon 驗證模組。相反地,必須按照後續步驟中的說明複製設定。

2. 在已設定的(第一個)獨立閘道執行個體中:

   獲取現有 Mellon 設定的 tar 副本。tar 備份將保留所有目錄層次結構和權限。執行以下命令:

   ```
   cd /etc
   sudo tar -cvf mellon.tar mellon
   ```

   將 `mellon.tar` 複製到獨立閘道的第二個執行個體。

3. 在獨立閘道的第二個執行個體:

   擷取(「解壓縮」) tar 檔案到 `/etc`目錄中的第二個執行個體。執行以下命令:

   ```
   cd /etc
   sudo tar -xvf mellon.tar
   ```

4. 在 Tableau Server 部署的節點 1 上:使用第二個獨立閘道的連線資訊更新連線檔案 (`tsig.json`)。需要按照本主題前面(安裝獨立閘道)中的描述檢索身份驗證金鑰。

   此處顯示範例連線檔案 (`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 部署的節點 1 中:執行以下命令以更新設定:

```
tsm stop

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

tsm start
```

6. 在 Independent Gateway 的兩個執行個體中:Tableau Server 啟動後,重新啟動 `tsig-httpd` 過程:

```
sudo su - tableau-tsig

systemctl --user restart tsig-httpd

exit
```

7. 在 AWS **EC2>目標群組**中:更新目標群組以包含執行第二個獨立閘道執行個體的 EC2 執行個體。

   選取剛剛建立的目標群組,然後按一下「目標」索引標籤:

   - 按一下 **[編輯]**。
   - 選取第二台獨立閘道電腦的 EC2 執行個體,然後按一下**「新增至已註冊」**。按一下**「儲存」**。

第 6 部分 - 安裝後組態

設定從負載平衡器到 Tableau Server 的 SSL/TLS

一些組織需要從用戶端到後端服務的端到端加密通道。到目前為止所述的預設參考架構可指定從用戶端到在您組織的 Web 層中執行的負載平衡器的 SSL。

本部分會介紹如何在範例 AWS 參考架構中為 Tableau Server 和獨立閘道設定 SSL/TLS。有關描述如何在 AWS 參考架構中在 Apache 上設定 SSL/TLS 的設定範例，請參閱範例：在 AWS 參考架構中設定 SSL/TLS。

目前，在 8000-9000 範圍內執行的後端 Tableau Server 流程不支援 TLS。若要啟用 TLS，必須為獨立閘道設定與 Tableau Server 的轉送連線。

本程序會介紹如何在連線到 Tableau Server 的獨立閘道上以及在連線到獨立閘道的 Tableau Server 上啟用和設定 TLS。本程序會加密通過 HTTPS/443 的轉送流量和通過 HTTPS/21319 的內務處理流量。

此範例的整個 Linux 程序展示了 RHEL-like 發行版的命令。具體來說，這裡的命令是使用 Amazon Linux 2 發行版開發的。若執行的是 Ubuntu 發行版，請編輯相對應的命令。

此處的指導是對本指南中介紹的特定 AWS 範例參考架構的規定性說明。因此，不包括可選設定。有關完整的參考文件，請參閱*在獨立閘道上設定 TLS*(Linux)。

設定 TLS 之前

請在工作時間之外執行 TLS 設定。設定需要至少重新啟動 Tableau Server 一次。若正在執行完整的四節點參考架構部署，則重新啟動可能需要一段時間。

- 驗證用戶端是否可以透過 HTTP 連線到 Tableau Server。使用獨立閘道設定 TLS 是一個多步驟過程，可能需要進行疑難排解。因此，我們建議在設定 TLS 之前從完全可操作的 Tableau Server 部署開始。

- 收集 TLS/SSL 憑證、金鑰和相關資產。您將需要用於獨立閘道和 Tableau Server 的 SSL 憑證。為簡化憑證管理和部署，並作為安全性最佳做法，我們建議使用由受信任的主要協力廠商憑證頒發授權單位 (CA) 產生的憑證。或者，您可以產生自我簽署憑證或使用針對 TLS 的 PKI 憑證。

  本主題中的範例設定使用以下資產名稱進行說明：
  - `tsig-ssl.crt`：Independent Gateway 的 TLS/SSL 認證。
  - `tsig-ssl.key`：Independent Gateway 的私密金鑰 `tsig-ssl.crt`。
  - `ts-ssl.crt`：Tableau Server 的 TLS/SSL 認證。
  - `ts-ssl.key`：Tableau Server 的私密金鑰 `tsig-ssl.crt`。
  - `tableau-server-CA.pem`：為 Tableau Server 電腦產生認證的 CA 根認證。如果使用來自主要信任第三方的認證，則通常不需要此認證。
  - `rootTSIG-CACert.pem`：為 Independent Gateway 電腦產生認證的 CA 根認證。如果使用來自主要信任第三方的認證，則通常不需要此認證。
  - SAML 還需要其他認證和金鑰檔案資產，詳見本指南的第 5 部分。
  - 若實作需要使用憑證鏈結檔案，請參閱知識庫文章在使用具有憑證鏈結的憑證時在獨立閘道上設定 TLS。

- 驗證是否有權存取 IdP。若使用 IdP 進行驗證，可能需要在設定 SSL/TLS 後變更 IdP 的接收者和目標 URL。

為 TLS 設定獨立閘道電腦

設定 TLS 可能是一個可能出錯的程序。由於跨獨立閘道的兩個執行個體進行疑難排解可能非常耗時，建議僅使用一個獨立閘道在 EDG 部署上啟用和設定 TLS。經驗證 TLS 在整個部署中正常工作後，設定第二台獨立閘道電腦。

第 1 步：將憑證和金鑰分發到獨立閘道電腦

只要 tsig-httpd 使用者具有對檔案的讀取權限，您就可以將資產分發到任意目錄。這些檔案的路徑會在其他程序中被引用。我們將在整個主題中使用 `/etc/ssl` 下的範例路徑，如下所示。

1. 為私密金鑰建立目錄:

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

2. 將憑證和金鑰檔案複製到 /etc/ssl 路徑。例如,

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

3. (可選)若在 Tableau Server 上為 SSL/TLS 使用自我簽署或 PKI 憑證,則還必須將 CA 根憑證檔案複製到獨立閘道電腦。例如,

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

第 2 步:為 TLS 更新環境變數

必須更新獨立閘道設定的連接埠和通訊協定環境變數。

透過更新檔案 /etc/opt/tableau/tableau_tsig/environment.bash 來變更這些值,如下所示:

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

第 3 步:為 HK 通訊協定更新虛設常式設定檔

手動編輯虛設常式設定檔 (/var/opt/tableau/tableau_tsig/config/httpd.conf.stub), 以為內務處理 (HK) 通訊協定設定與 TLS 相關的 Apache httpd 指示詞。

虛設常式設定檔包含與 TLS 相關的指示詞區塊,這些指示詞透過 #TLS# 標記進行註解。從指示詞中移除標記,如下例所示。請注意,該範例顯示了將根 CA 憑證用於具有 SSLCACertificateFile 選項的 Tableau Server 上使用的 SSL 憑證。

```
#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>
```

若重新安裝獨立閘道，這些變更將丟失。我們建議製作備份複本。

第 4 步：複製虛設常式檔案，並重新啟動服務

1. 複製在上一步中更新的檔案，以使用變更更新 httpd.conf：

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

2. 重新啟動獨立閘道服務：

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

重新啟動後，獨立閘道將無法執行，直到您在 Tableau Server 上執行下一組步驟。在 Tableau Server 上完成這些步驟後，獨立閘道將揀選變更並上線。

為 TLS 設定 Tableau Server 節點 1

在 Tableau Server 部署的節點 1 上執行這些步驟。

步驟 1：複製憑證和金鑰，並停止 TSM

1. 確認已將 Tableau Server「外部 SSL」憑證和金鑰複製到節點 1。

2. 為最大限度地減少停機時間，我們建議停止 TSM，執行以下步驟，然後在套用變更後啟動 TSM：

   ```
   tsm stop
   ```

步驟 2:設定憑證資產, 並啟用獨立閘道設定

1. 為獨立閘道指定憑證和金鑰檔案的位置。這些路徑會引用獨立閘道電腦上的位置。請注意, 此範例假設使用相同的憑證和金鑰對來保護 HTTPS 和內務處理流量:

   ```
   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. 為獨立閘道的 HTTPS 和 HK 通訊協定啟用 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. (可選) 若在獨立閘道上為 SSL/TLS 使用自我簽署或 PKI 憑證, 則必須上傳 CA 根憑證檔案。CA 根認證檔案是用於為 Independent Gateway 產生認證的根認證。例如,

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

4. (可選) 若在 Tableau Server 上為 SSL/TLS 使用自我簽署或 PKI 憑證, 則還必須將 CA 根憑證檔案複製到 Independent Gateway 的 `/etc/ssl/certs` 目錄。CA 根認證檔案是用於為 Tableau Server 電腦產生認證的根認證。將認證複製到 Independent Gateway 後, 必須使用以下 tsm 命令指定節點 1 上認證的位置。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. (可選:僅用於測試目的) 若在電腦之間使用共用自我簽署或 PKI 憑證, 因此憑證上的主題名稱與電腦名稱不相符, 那麼必須停用憑證驗證。

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

步驟 3: 為 Tableau Server 啟用「外部 SSL」, 並套用變更

1. 在 Tableau Server 上啟用和設定「外部 SSL」:

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

2. 套用變更。

   ```
   tsm pending-changes apply
   ```

第四步:更新閘道設定 JSON 檔案, 並啟動 tsm

1. 在 Tableau Server 端上更新獨立閘道設定檔(例如, `tsig.json`), 為獨立閘道物件指定 `https` 通訊協定:

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

2. 移除(或取消註解)獨立閘道第二個執行個體的連線資訊。請務必在儲存前在外部編輯器中驗證 JSON。

   為獨立閘道的單個執行個體設定和驗證 TLS 後, 可使用獨立閘道的第二個執行個體連線資訊更新此 JSON 檔案。

3. 執行以下命令以更新獨立閘道設定:

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

4. 啟動 TSM

   ```
   tsm start
   ```

5. 啟動 TSM 時, 登入獨立閘道執行個體並重新啟動 `tsig-httpd` 服務:

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

```
systemctl --user restart tsig-httpd

exit
```

將 IdP 驗證模組 URL 更新為 HTTPS

如果已為 Tableau 設定了外部身分識別提供者，則可能需要更新 IdP 管理儀表板中的返回 URL。

例如，如果使用 Okta 預身分驗證應用程式，則需要更新應用程式以對接收 URL 和目標 URL 使用 HTTPS 協定。

為 HTTPS 設定 AWS 負載平衡器

若按照本指南中的說明使用 AWS 負載平衡器進行部署，則可重新設定 AWS 負載平衡器，以將 HTTPS 流量傳送到執行獨立閘道的電腦：

1. 刪除現有的 HTTP 目標群組：

   在**「目標群組」**中，選取已為負載平衡器設定的 HTTP 目標群組，按一下**「動作」**，然後按一下**「刪除」**。

2. 建立 HTTPS 目標群組：

   目標群組 > 建立目標群組

   - 選取「執行個體」
   - 輸入目標群組名稱，例如 `TG-internal-HTTPS`
   - 選取 VPC
   - 協定：HTTPS 443
   - 在**健康情況檢查>進階健康情況檢查設定>成功代碼**下，附加代碼清單以讀入：`200,303`。
   - 按一下**建立**。

3. 選擇剛剛建立的目標群組，然後按一下**目標**索引標籤：

- 按一下 **編輯**
- 選取正在執行 Tableau Server 獨立閘道的 EC2 執行個體，然後按一下**「新增至已註冊」**。
- 按一下**儲存**。

4. 建立目標群組後，必須啟用相黏：

   - 開啟 AWS 目標群組頁面（**EC2> 負載平衡> 目標群組**），選取剛剛設定的目標群組執行個體。在**動作**功能表上，選取**編輯屬性**。
   - 在**「編輯屬性」**頁面上，選取**「綁定」**，指定持續時間為 `1 day`，然後**儲存變更**。

5. 在負載平衡器上，更新收聽器規則。選擇為此部署設定的負載平衡器，然後按一下**收聽器**索引標籤。

   - 對於**HTTP:80**，按一下**查看/編輯規則**。在生成的**規則**頁面上，按一下編輯圖示（一次位於頁面上方，然後再次位於規則旁邊）以編輯規則。刪除現有 THEN 規則並按一下**新增動作 > 重新導向至...**來替換它。在生成的 THEN 組態中，指定`HTTPS`和連接埠`443`並將其他選項保留為預設值。儲存設定然後按一下**更新**。
   - 對於 **HTTPS:443**，按一下**「檢視/編輯規則」**。在生成的**規則**頁面上，按一下編輯圖示（一次位於頁面上方，然後再次位於規則旁邊）以編輯規則。刪除現有 THEN 規則並按一下**新增動作>轉傳至...**以取代它。將目標群組指定為剛剛建立的 HTTPS 群組。在**群組層級綁定**下，啟用綁定並將持續時間設定為 1 天。儲存設定然後按一下**更新**。

6. 在負載平衡器上，將空閒逾時更新為 400 秒。選取為此部署設定的負載平衡器，然後按一下**動作 > 編輯屬性**。將**空閒逾時**設定為 `400` 秒，然後按一下**儲存**。

驗證 TLS

要驗證 TLS 功能，請使用您在此過程開始時建立的 Tableau 管理員帳戶，透過公用 URL（例如 https://tableau.example.com）登入 Tableau Server。

如果 TSM 未啟動或收到其他錯誤，請參見解除安裝 Tableau Server 獨立閘道。

為 SSL 設定獨立閘道的第二個執行個體

成功設定獨立閘道的第一個執行個體後，部署第二個執行個體。

部署第二個獨立閘道的過程需要以下步驟：

1. 在 Independent Gateway 的已設定（第一個）執行個體上：將以下檔案複製到 Independent Gateway 的第二個執行個體上的相應位置：
   - `/etc/ssl/certs/tsig-ssl.crt`
   - `/etc/ssl/private/tsig-ssl.key`（需要在第二個執行個體上建立 `private` 目錄）。
   - `/var/opt/tableau/tableau_tsig/config/httpd.conf.stub`
   - `/etc/opt/tableau/tableau_tsig/environment.bash`
2. 在 Tableau Server 部署的節點 1 上：使用第二個獨立閘道的連線資訊更新連線檔案 (`tsig.json`)。

   此處顯示範例連線檔案 (`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 部署的節點 1 中：執行以下命令以更新設定：

```
tsm stop

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

tsm start
```

4. 在獨立獨立閘道的兩個執行個體中：Tableau Server 啟動後，獨立閘道的兩個執行個體重新啟動 `tsig-httpd`：

```
sudo su - tableau-tsig

systemctl --user restart tsig-httpd

exit
```

5. 在 AWS **EC2>目標群組**中：更新目標群組以包含執行第二個獨立閘道執行個體的 EC2 執行個體。

   選取剛剛建立的目標群組，然後按一下「目標」索引標籤：

   - 按一下**「編輯」**。
   - 選取第二台獨立閘道電腦的 EC2 執行個體，然後按一下**「新增至已註冊」**。按一下**「儲存」**。

為 Postgres 設定 SSL

可以選擇性為 Tableau Server 上的外部存放庫連線的 Postgres 連線設定 SSL (TLS)。

為簡化憑證管理和部署，並作為安全性最佳做法，我們建議使用由受信任的主要協力廠商憑證頒發授權單位 (CA) 產生的憑證。或者，您可以產生自我簽署憑證或使用針對 TLS 的 PKI 憑證。

此流程描述如何使用 OpenSSL 在範例 AWS 參考架構中類似於 RHEL 的 Linux 發行版上的 Postgres 主機上產生自我簽署憑證。

產生並簽署 SSL 憑證後，必須將 CA 憑證複製到 Tableau 主機。

在執行 Postgress 的主機上：

1. 產生簽署根憑證頒發授權單位 (CA) 金鑰：

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

2. 建立根 CA 憑證：

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

   系統將提示輸入憑證欄位的值。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. 為 Postgres 電腦建立憑證和相關金鑰(下方範例中的 `server.csr` 和 `server.key`)。憑證的主題名稱必須與 Postgres 主機的 EC2 私人 DNS 名稱相符。主題名稱設定為 `-subj` 選項，帶有格式 `"/CN=<private DNS name>"`，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. 使用您在步驟 2 中創建建立的 CA 憑證籤署新憑證。以下命令也將以 `crt` 格式輸出憑證：

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

5. 將 crt 和 key 檔案複製到 Postgres `/var/lib/pgsql/13/data/` 路徑：

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

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

6. 切換到根使用者：

```
sudo su
```

7. 設定 cer 和 key 檔案的權限。執行以下命令：

```
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 設定檔，`/var/lib/pgsql/13/data/pg_hba.conf` 以指定 md5 信任：

   變更現有的連線語句

   `host all all 10.0.30.0/24 password` 和

   `host all all 10.0.31.0/24 password`

   為

   `host all all 10.0.30.0/24 md5` 和

   `host all all 10.0.31.0/24 md5`。

9. 新增這行以更新 postgresql 檔案，`/var/lib/pgsql/13/data/postgresql.conf`：

```
ssl = on
```

10. 退出根使用者模式：

```
exit
```

11. 重新啟動 Postgres：

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

可選:在 Tableau Server 上為 Postgres SSL 啟用認證信任驗證

如果按照第 4 部分 - 安裝並設定 Tableau Server 中的安裝過程進行動作, 則 Tableau Server 會設定用於 Postgres 連線的可選 SSL。這代表在 Postgres 上設定 SSL(如上所述) 將形成加密連線。

如果想要對連線進行認證信任驗證, 則必須在 Tableau Server 上執行以下命令來重新設定 Postgres 主機連線:

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

`<filename>.json` 所在處是設定外部 Postgres 中描述的連線檔案。而 `CACert.pem` 是 Postgres 使用的 SSL/TLS 認證的 CA 認證檔案。

可選:驗證 SSL 連線

要驗證 SSL 連線, 必須:

- 在 Tableau Server 節點 1 上安裝 Postgres 用戶端。
- 將在上一個過程中建立的根認證複製到 Tableau 主機。
- 從節點 1 連線到 Postgres 伺服器

在節點 1 上安裝 Postgres 用戶端

此範例示範如何安裝 Postgres 版本 13.4。安裝您為外部存放庫執行的相同版本。

1. 在節點 1 上, 在 `/etc/yum.repos.d` 路徑中建立並編輯檔案 pgdg.repo.。將以下設定資訊填入檔案。

   ```
   [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. 安裝 Postgres 用戶端：

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

將根認證複製到節點 1

將 CA 憑證 (`pgsql-rootCACert.pem`) 複製到 Tableau 主機：

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

從節點 1 以 SSL 連線到 Postgres 主機：

從 Node1 執行以下命令，指定 Postgres 伺服器主機 IP 位址和根 CA 認證：

```
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 將提示您輸入密碼。登入成功後，shell 將傳回：

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

設定 SMTP 和事件通知

Tableau Server 向管理員和使用者傳送電子郵件通知。要啟用此功能，必須設定 Tableau Server 才能將電子郵件傳送到電子郵件伺服器。還必須指定要發送的事件類型、閾值和訂閱資訊。

對於 SMTP 和通知的初始組態，建議使用下面的組態檔範本建立一個 json 檔。還可以使用 *tsm configuration set* 中描述的語法來設定下面列出的任何單一組態金鑰(Linux)。

在 Tableau Server 部署中的節點 1 上執行此過程：

1. 將以下 json 範本複製到檔案。使用 SMTP 組態選項以及組織的訂閱和通知來自訂檔案。
   - 要查看所有 SMTP 選項的清單和說明，請參閱*SMTP CLI 組態參照*(Linux)。
   - 要查看所有通知事件選項的清單和說明，請參閱*設定伺服器事件通知*(Linux)的 CLI 一節。

```
{
"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. 執行`tsm settings import -f file.json`將 json 檔傳遞給 Tableau 服務管理員。
3. 執行 `tsm pending-changes apply` 命令以套用變更。
4. 執行 `tsm email test-smtp-connection` 以檢視並驗證連線組態。

安裝 PostgreSQL 驅動程式

要在 Tableau Server 上查看管理員檢視，必須在 Tableau Server 部署的節點 1 上安裝 PostgreSQL 驅動程式。

1. 前往Tableau 驅動程序下載頁面並複制 PostgreSQL jar 檔的 URL。
2. 在 Tableau 部署的每個節點上執行以下過程：
   - 建立以下檔案路徑：

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

   - 從新路徑下載最新版本的 PostgreSQL jar 檔：例如：

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

3. 在初始節點上，重新啟動 Tableau Server：

```
tsm restart
```

設定強式密碼原則

若您不使用 IdP 驗證解決方案部署 Tableau Server 驗證解決方案，我們建議您加強預設 Tableau 密碼原則的安全性。

若您使用 IdP 部署 Tableau Server，則您必須使用 IdP 管理密碼原則。

以下流程包括用於在 Tableau Server 上設定密碼原則的 json 設定。有關以下選項的更多資訊，請參閱*本機驗證* (Linux)。

1. 將以下 json 範本複製到檔案。使用您的密碼原則設定填入機碼值。

```
{
 "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,
        "features.PasswordReset": true
  }
}
```

2. 執行 `tsm settings import -f file.json`, 將 json 檔傳遞給 Tableau 服務管理員, 以設定 Tableau Server。

3. 執行 `tsm pending-changes apply` 命令以套用變更。

第 7 部分 - 驗證、工具和疑難排解

本部分包括安裝後驗證步驟和疑難排解指南。

容錯移轉系統驗證

設定部署後，我們建議執行簡單的容錯移轉測試來驗證系統冗餘。

我們建議執行以下步驟來驗證容錯移轉功能：

1. 關閉 Independent Gateway (TSIG1) 的第一個執行個體。所有輸入流量都應由 Independent Gateway (TSIG2) 的第二個執行個體進行路由。
2. 重新啟動 TSIG1，然後關閉 TSIG2。所有輸入流量都應由 TSIG1 路由。
3. 重新啟動 TSIG2。
4. 關閉 Tableau Server 節點 1。所有 Vizportal/Application 服務流量都將容錯移轉到節點 2。

> **注意** 自 2022 年 9 月起，節點 1 的高可用性在特定 Tableau Server 2021.4 及更高版本上受到影響。如果節點 1 關閉，用戶端連線將失敗。此問題已在這些維護版本中得到修復：
>
> - 2021 年 4 月 15 日及更高版本
> - 2022 年 1 月 11 日及更高版本
> - 2023 年 1 月 3 日及更高版本
>
> 為了確保使用 ATR 啟動的 Tableau Server 安裝在初始節點故障後有 72 小時的寬限期，請安裝或升級到這些版本之一。有關詳情，請參閱 Tableau 知識庫中的 使用 ATR 的 Tableau Server HA 在初始節點故障後沒有寬限期。

5. 重新啟動節點 1 並關閉節點 2。所有 Vizportal/Application 服務流量都將容錯移轉到節點 1。
6. 重新啟動節點2。

在這種情況下,「關閉」或「重新啟動」是以關閉作業系統或虛擬機來完成,無需事先嘗試正常關閉應用程式。目標是模擬硬體或虛擬機故障。

每個容錯移轉測試的最小驗證步驟是向使用者進行身份驗證並執行基本的檢視動作。

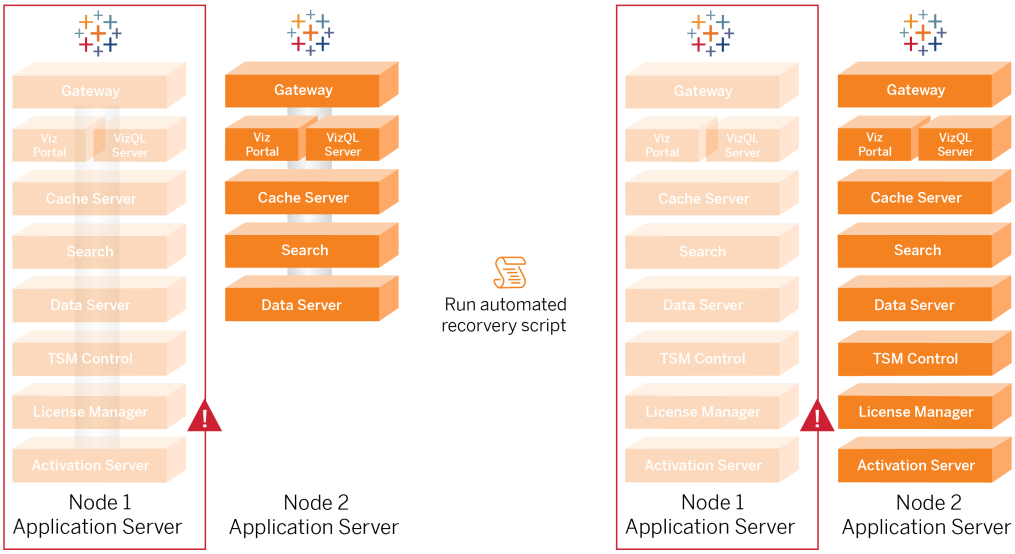
在模擬失敗後嘗試登入時,您可能會收到「錯誤請求」瀏覽器錯誤。即使清除瀏覽器中的快取,也可能會看到此錯誤。當瀏覽器快取來自先前 IdP 工作階段的資料時,通常會發生此問題。如果清除本機瀏覽器快取後此錯誤仍然存在,請與其他瀏覽器連線來驗證 Tableau 方案。

初始節點自動復原

Tableau Server 版本 2021.2.4 及更高版本在指令碼目錄 (`/app/tableau_server/packages/scripts.<version>`) 中包含自動初始節點復原指令碼 `auto-node-recovery`。

若初始節點出現問題,並且節點 2 上有冗餘流程,則無法保證 Tableau Server 將繼續執行。Tableau Server 可以在初始節點出現故障後繼續執行長達 72 小時,然後才會因為缺少授權服務而影響其他流程。若是這樣,您的使用者可能能夠在初始節點失敗後繼續登入並查看和使用他們的內容,但您將無法重新設定 Tableau Server,因為您將無權存取管理控制器。

即使已設定冗餘流程,在初始節點出現故障後,Tableau Server 也可能無法繼續執行。



要復原初始節點(節點 1)故障:

1. 登入到 Tableau Server 節點 2。
2. 變更為指令碼目錄:

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

3. 執行以下命令以啟動指令碼:

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

   其中 `<license keys>` 是以逗號分隔(無空格)的部署授權金鑰清單。若無權存取授權金鑰,請造訪 Tableau 客戶入口網站以進行檢索。例如:

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

auto-node-recovery 指令碼將執行大約 20 個步驟以將服務復原到節點 2。每個步驟都會隨著指令碼的進度顯示在終端中。更詳細的狀態會記錄到 `/data/tableau_data/logs/app-controller-move.log` 中。在大多數環境中,指令碼需要 35 到 45 分鐘才能完成復原。

對初始節點復原進行疑難排解

若節點復原失敗，您可能會發現以互動方式執行指令碼以允許或不允許流程中的離散步驟很有用。例如，若指令碼在流程中途失敗，可以查看記錄檔、變更設定，然後再次執行指令碼。透過以互動模式執行，可以跳過所有步驟，直到到達失敗的步驟。

要以互動模式執行，請將 `-i` 開關新增到指令碼引數中。

重建故障節點

執行指令碼後，節點 2 將執行以前在出現故障的節點 1 主機上的所有服務。要新增 4 節點，需要使用啟動程序檔案部署新的 Tableau Server 主機，並按照第 4 部分中指定的原始節點 2 的設定方式對其進行設定。請參閱設定節點 2。

switchto

Switchto 是 Tim 的一個指令碼，可讓視窗之間的切換變得輕而易舉。

1. 將以下程式碼複製到堡壘主機上主目錄中名為 `switchto` 的檔案中。

```
#!/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. 更新指令碼中的 IP 位址，以對應到 EC2 執行個體，然後儲存檔案。
3. 對指令碼檔案套用權限：

   ```
   sudo chmod +x switchto
   ```

用法：

若要切換到主機，請執行以下命令：

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

例如，若要切換到節點 1，請執行以下命令：

```
./switchto node1
```

解除安裝 Tableau Server 獨立閘道

在 Tableau Server 上設定獨立閘道、Okta、Mellon 和 SAML 可能是容易出錯的過程。最常見的失敗根本原因是字串錯誤。例如，設定過程中指定 Okta URL 的結尾斜線 (/) 可能會導致與 SAML 宣告相關的不相符錯誤。這只是一個範例。在設定過程中有很多機會在任何應用程式中輸入不正確的字串。

重啟 tableau-tsig 服務

始終透過重新啟動獨立閘道電腦中的 tableau-tsig 服務開始(和完成)疑難排解。重新啟動此服務很快，並且通常會觸發從 Tableau Server 加載更新的設定。

在獨立閘道電腦中執行以下命令：

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

找到不正確的字串

如果您犯了字串錯誤(複製/貼上錯誤、字串被截斷等)，請花時間瀏覽您配置的每個設定：

- Okta 預身份驗證設定。仔細查看您設定的 URL。尋找結尾斜線。驗證 HTTP 與 HTTPS。
- 節點 1 中 SAML 設定的 Shell 歷程記錄。檢閱您執行的 `tsm authentication saml configure` 命令。驗證所有 URL 是否與您在 Okta 中設定的 URL 相符。在查看節點 1 的 shell 歷史記錄時，請確認 `tsm configuration set` 指定 Mellon 設定檔路徑的命令完全對應在Independent Gateway上複製檔案的檔案路徑。
- 獨立閘道中的 Mellon 設定。檢閱 shell 歷程記錄，驗證所建立的中繼資料是否使用您在 Okta 和 Tableau SAML 中設定的相同 URL 字串。驗證 `/etc/mellon/conf.d/global.conf` 中指定的所有路徑是正確的，並且將 `MellonCookieDomain` 設定為您的根網域，而不是您的 Tableau 子網域。

搜尋相關記錄

如果所有字串均正確設定，則應該檢查記錄是否有錯誤。

Tableau Server 將錯誤和事件記錄到數十個不同的記錄檔中。獨立閘道也記錄到一組本機檔案中。建議按以下順序檢查這些記錄。

獨立閘道記錄檔

獨立閘道記錄檔的預設位置為 `/var/opt/tableau/tableau_tsig/logs`。

- `access.log`：此記錄非常有用，因為它具有顯示來自 Tableau Server 節點連線的項目。如果嘗試啟動 TSM 時遇到閘道錯誤（不會啟動），並且 `access.log` 記錄檔中沒有項目，則存在核心連線能力問題。始終首先驗證 AWS 安全性群組設定。另一個常見問題是 `tsig.json` 中的拼字錯誤。如果更新 tsig.json，請執行 `tsm stop`，然後執行 `tsm topology external-services gateway update -c tsig.json`。更新 tsig.json 後，請執行 `tsm start`。
- `error.log`：除其他項目外，此記錄還包括 SAML 和 Mellon 錯誤。

Tableau Server tabadminagent 記錄檔

`tabadminagent`（非 `tabadmincontroller`）檔案集是解決 Independent Gateway 相關錯誤的唯一相關記錄檔。

必須找到在 `tabdminagent` 中記錄的 Independent Gateway 錯誤。這些錯誤可以在任何節點中，但只在一個節點上。在 Tableau Server 叢集中的每個節點中執行以下步驟，直到找到「independent」字串：

1. 在 EDG 設定中找到 Tableau Server 節點 1-4 上的 tabadminagent 記錄檔位置：

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

2. 開啟最新記錄讀取：

   ```
   less tabadminagent_nodeN.log
   ```

   （使用節點號取代 N）

3. 搜尋所有「Independent」和「independent」執行個體 - 使用以下搜尋字串：

   ```
   /ndependent
   ```

   如果沒有相符項目，則轉到下一個節點並重複步驟 1-3。

4. 找到相符項目後：`Shift + G` 移至底部以獲取最新錯誤資訊。

重新載入 httpd 虛設常式檔案

Independent Gateway 管理 Apache 的 httpd 設定。通常會修復暫時性問題的一般操作是重新載入作為基礎 Apache 設定植入的 httpd 虛設常式檔案。在 Independent Gateway 的兩個執行個體上執行以下命令。

1. 將虛設常式檔案複製到 httpd.conf：

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

2. 重新啟動獨立閘道服務：

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

刪除或移動記錄檔

Independent Gateway 記錄所有存取事件。將需要管理記錄檔儲存以避免填滿磁碟空間。如果磁碟已滿，Independent Gateway 將無法寫入存取事件並且服務將失敗。以下訊息將記錄到 Independent Gateway 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
```

此故障將導致在 Tableau 節點 1 執行 `tsm status -v` 時 `external` 節點狀態為 `DEGRADED`。狀態輸出中的 `external` 節點是指 Independent Gateway。

要解決此問題，請刪除或移動磁碟上的 access.log 檔案。Access.log 的存放位置為 `/var/opt/tableau/tableau_tsig/logs`。清理磁碟後，重新啟動 tableau-tsig 服務。

瀏覽器錯誤

錯誤的要求：這種情況下的一個常見錯誤是來自 Okta 的「錯誤的要求」錯誤。當瀏覽器快取來自先前 Okta 工作階段的資料時，通常會發生此問題。例如，如果您以 Okta 管理員身份管理 Okta 應用程式，然後嘗試使用不同的啟用 Okta 帳戶存取 Tableau，則來自管理員資料的工作階段資料可能會導致「錯誤的要求」錯誤。如果清除本機瀏覽器快取後此錯誤仍然存在，請嘗試與其他瀏覽器連線來驗證 Tableau 方案。

「錯誤請求」錯誤的另一個原因是在 Okta、Mellon 和 SAML 設定過程中輸入的眾多 URL 之一中發生拼寫錯誤。檢查您輸入的所有這些內容都沒有錯誤。

通常情況下，獨立閘道伺服器中的 `error.log` 檔案會指出造成此錯誤的 URL。

找不到 - 在此伺服器上找不到請求的 URL：此錯誤代表許多設定錯誤的其中一種。

如果使用者使用 Okta 進行身份驗證，然後收到此錯誤，則很可能是在設定 SAML 時將 Okta 預身份驗證應用程式上傳到 Tableau Server。驗證出您在 Tableau Server 上設定了 Okta Tableau Server 應用程式中繼資料，而不是 Okta 預身份驗證應用程式中繼資料

其他疑難排解步驟:

- 檢查 Okta 預身份驗證應用程式設定。確定 HTTP 與 HTTPS 協定按照此主題指定步驟設定。
- 在兩個獨立閘道伺服器上重新啟動 tsig-httpd。
- 請驗證 `sudo apachectl configtest` 是否在兩個獨立閘道上都傳回「語法正常」。
- 驗證測試使用者是否已指派給 Okta 中的兩個應用程式。
- 驗證是否在負載平衡器和關聯的目標群組上設定綁定。

用於從 Tableau Server 到獨立閘道的 TLS 驗證

使用 `wget` 命令驗證從 Tableau Server 到獨立閘道的連線和存取。此命令的變體有助於您了解憑證問題是否導致連線問題。

例如執行這個 `wget` 驗證 Tableau Server 內務吹 (HK) 通訊協定的命令:

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

使用 tsig.json 檔案的 host 選項包含的相同主機地址建構 URL。指定 `https` 通訊協議並在 URL 後面加上 HK 連接埠 `21319`。

要檢查連線能力並忽略憑證驗證:

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

驗證 TSIG 的根 CA 證書是否有效:

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

如果 Tableau 能夠通信,則仍會收到與內容相關的錯誤,但不會收到與連線相關的錯誤。如果 Tableau 根本無法連線,則首先驗證防火牆/安全組中的通訊協議設定。例如,用於獨立閘道所在的安全群組的輸入規則必須允許 TCP 21319。

附錄 - AWS 部署工具箱

本主題會介紹在 AWS 中進行部署時參考架構的工具和備用部署選項。具體而言，本主題會介紹如何自動化整個 EDG 中所述的範例 AWS 部署。

TabDeploy4EDG 自動安裝指令碼

TabDeploy4EDG 指令碼會自動實作第 4 部分 - 安裝並設定 Tableau Server 中所述的四節點 Tableau 部署。如果遵循本指南中所述的範例 AWS 實作，就能夠執行 TabDeploy4EDG。

要求。要執行該指令碼，必須根據 第 3 部分 - 準備 Tableau Server Enterprise 部署中的範例實作來準備和設定 AWS 環境：

- VPC、子網路和安全性群組已按所述進行設定。IP 位址不必與範例實作中顯示的位址相符。
- 四個 EC2 執行個體，正在執行最新的已更新 AWS Linux 2 版本
- PostgreSQL 已安裝，並已如安裝、設定 PostgreSQL 和建立 tar 備份 中所述進行設定。
- 步驟 1 的 tar 備份文件位於安裝 PostgreSQL 的 EC2 執行個體中，如使用 PostgreSQL 步驟 1 tar 備份中所述。
- 執行 Tableau Server 部署節點 1 的 EC2 執行個體已設定為與 PostgreSQL 通訊，如第 4 部分 - 安裝並設定 Tableau Server中所述。
- 您已使用來自堡壘主機的 SSH 工作階段登入到每個 EC2 執行個體。

該指令碼大約需要 1.5-2 小時來安裝和設定四個 Tableau 伺服器。該指令碼會根據參考架構的指定設定來設定 Tableau。該指令碼會執行以下動作：

- 若您指定 PostgreSQL 主機 tar 檔案的路徑，則會還原 PostgreSQL 主機的第 1 階段備份。
- 刪除所有節點上的現有 Tableau 安裝。
- 在所有節點上執行 `sudo yum update`。
- 將 Tableau rpm 套件下載並複製到每個節點。
- 將相依項下載並安裝到每個節點。
- 建立 `/app/tableau_server`，並在所有節點上安裝套件。

- 使用本機身分存放區安裝節點 1，並使用 PostgreSQL 設定外部存放庫。
- 執行節點 2 - 節點 4 的啟動程序安裝和初始設定。
- 刪除啟動程序檔案和 TabDeploy4EDG 的設定檔案。
- 根據參考架構規格跨 Tableau 叢集設定服務。
- 驗證安裝並傳回每個節點的狀態。

將指令碼下載並複制到堡壘主機

1. 從 TabDeploy4EDG 範例頁面複製指令碼，並將程式碼貼上到名為 `TabDeploy4EDG` 的檔案中。
2. 將檔案儲存到作為堡壘主機的 EC2 主機上的主目錄。
3. 執行以下命令，變更檔案的模式以使其可執行：

```
sudo chmod +x TabDeploy4EDG
```

執行 TabDeploy4EDG

TabDeploy4EDG 必須從堡壘主機執行。編寫該指令碼時已假設該指令碼將在 ssh 轉送代理程式內容中執行，如範例：連線到 AWS 中的 Bastion 主機中所述。若未在 ssh 轉送代理程式內容中執行，那麼在整個安裝過程中將提示您輸入密碼。

1. 建立、編輯和儲存登錄檔（`registration.json`）。該檔案必須為格式正確的 json 檔案。複製並自訂以下範本：

```
{
        "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. 執行以下命令，產生範本設定檔案：

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

3. 開啟設定檔案進行編輯：

   ```
   sudo nano edg.config
   ```

   至少必須新增每個 EC2 主機的 IP 位址、登錄檔的檔案路徑以及有效的授權金鑰。

4. 完成設定檔案的編輯後，儲存然後關閉。

5. 要執行 TabDeploy4EDG，請執行以下命令：

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

範例：使用 Terraform 自動化 AWS 基礎架構部署

本區段介紹如何設定和執行 Terraform 以在 AWS 中部署 EDG 參考架構。此處提供的範例 Terraform 設定部署了一個 AWS VPC，其中包含第 3 部分 - 準備 Tableau Server Enterprise 部署中描述的子網域、安全性群組和 EC2 執行個體。

Tableau 範例網站上提供了範例 Terraform 範本，網址為 https://help.tableau.com/samples/en-us/edg/edg-terraform.zip 。必須為您的組織設定和自訂這些範本。本區段中提供的設定內容描述了必須自訂部署所需的最少範本變更。

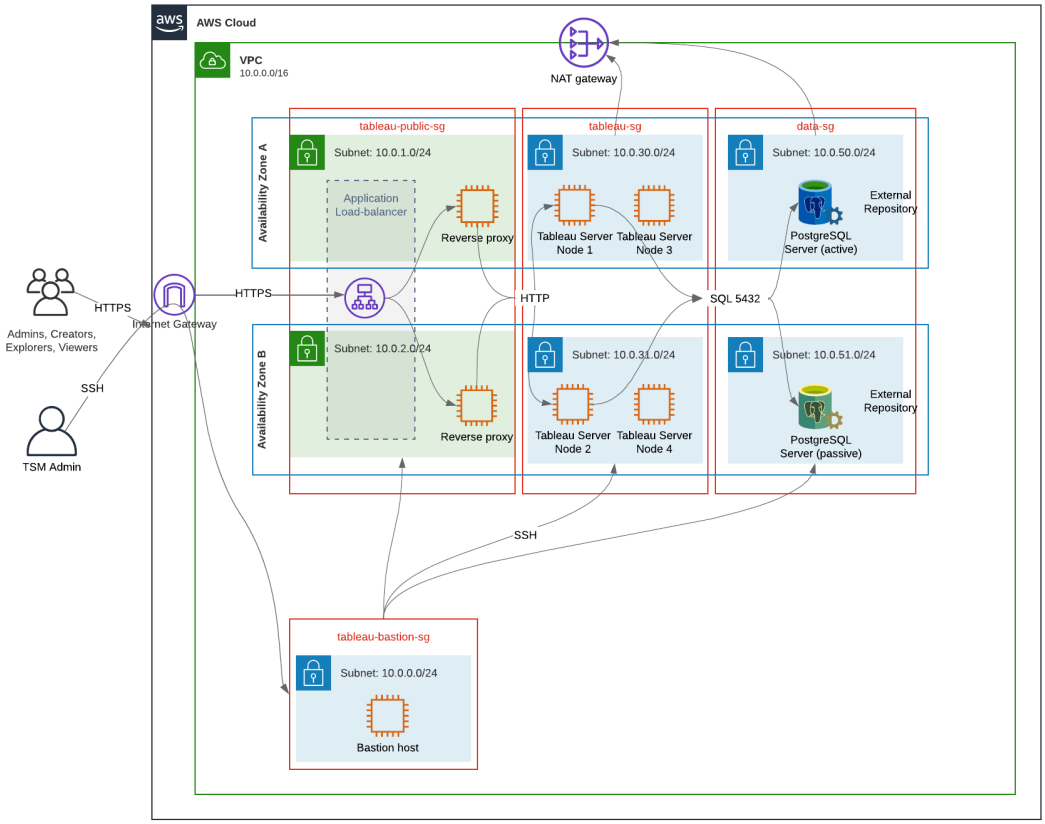
目標

此處提供的 Terraform 範本和內容旨在提供工作範例，使您能夠在開發測試環境中快速部署 EDG。

我們已盡最大努力測試和記錄範例 Terraform 部署。但是，使用 Terraform 在生產環境中部署和維護 EDG 需要 Terraform 專業知識，這超出了本範例的範圍。Tableau 不為此處記錄的範例 Terraform 解決方案提供支援。

結束狀態

按照本區段中的過程在 AWS 中設定 VPC，該 VPC 的功能等同於第 3 部分 - 準備 Tableau Server Enterprise 部署中指定的 VPC。



本區段中的範例 Terraform 範本和支援內容：

- 如上所示建立彈性 IP 位址、兩個可用區域和子網域組織的 VPC(IP 位址不同)
- 建立 Bastion、Public、Private 和 Data 安全性群組。
- 在安全性群組上設定大多數入口和出口規則。在 Terraform 執行後，將需要編輯安全性群組。

- 建立以下 EC2 主機(每個都執行 AWS Linux2):bastion、proxy 1 proxy 2、Tableau 節點 1、Tableau 節點 2、Tableau 節點 3、Tableau 節點 4。
- 未建立 PostgreSQL 的 EC2 主機。必須在資料安全性群組中手動建立 EC2,然後按照安裝、設定 PostgreSQL 和建立 tar 備份。

需求

- AWS 帳戶 - 必須有權存取允許建立 VPC 的 AWS 帳戶。
- 如果從 Windows 電腦執行 Terraform,則需要安裝 AWS CLI。
- 您的 AWS 帳戶中可用的彈性 IP 位址。
- 在 AWS Route 53 中註冊的網域。Terraform 將在 Route 53 中建立 DNS 區域和相關的 SSL 憑證。因此,執行 Terraform 的設定檔也必須在 Route 53 中具有適當的權限。

開始之前

- 此過程中的命令列範例適用於帶有 Apple OS 的終端。要在 Windows 上執行 Terraform,可能需要根據需要調整帶有檔案路徑的命令。
- Terraform 專案由許多文字設定檔案(.tf 檔案副檔名)組成。可以自訂這些檔案來設定 Terraform。如果沒有強大的文字編輯器,請安裝 Atom 或 Text++。
- 如果要與他人共用 Terraform 專案,我們建議將專案存放在 Git 中以進行變更管理。

步驟 1 - 準備環境

下載並安裝 Terraform

https://www.terraform.io/downloads

B. 產生公私鑰配對

這是您將用於存取 AWS 和產生的 VPC 環境的金鑰。執行 Terraform 時,將包括公鑰。

開啟命令提示字元並執行以下命令:

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

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

2. 建立公鑰。此金鑰格式不用於 Terraform。稍後將在此過程中將其轉換為 ssh 金鑰：

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

3. 設定私鑰權限：

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

   在 Windows 上設定權限：

   - 在 Windows 資源管理器中找到該檔案，右鍵點一下它，然後選取**「屬性」**。巡覽到**「安全性」**索引標籤，然後點一下**「進階」**。
   - 將擁有者變更為您，停用繼承並刪除所有權限。授予自己**「完全控制權」**，然後點一下**「儲存」**。將檔案標記為唯讀。

4. 建立 ssh 公鑰。這是稍後將複製到 Terraform 中的金鑰。

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

C. 下載專案並新增狀態目錄

1. 下載並解壓縮 EDG Terraform 專案並將它們儲存到本機電腦。解壓縮下載後，您將擁有一個頂層目錄、edg-terraform 和一系列子目錄。

2. 建立一個名為 `state`，作為頂層 `edg-terraform` 目錄的同儕目錄。

步驟 2：自訂 Terraform 範本

必須自訂 Terraform 範本以符合 AWS 和 EDG 環境。此處的範例提供大多數組織需要進行的最少範本自訂。特定環境可能需要其他自訂。

本區段按範本名稱組織。

在繼續執行*步驟 3 - 執行 Terraform* 之前，請務必儲存所有變更。

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. 開啟步驟 1B 中產生的公鑰並複製該金鑰：

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

   Windows：複製公鑰的內容。

2. 將公鑰字串複製到 public_key 參數中，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

將 `user.owner` 更新為您的名稱或別名。您在此處輸入的值將用於 Terraform 建立的資源上 AWS 中的「名稱」標記。

providers.tf

1. 根據組織的要求新增標籤。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

更新根網域名稱。此名稱必須與在 Route 53 中註冊的網域相符。

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

預設情況下，子網域，`tableau`，指定為 VPC DNS 網域名稱。要變更此設定，請更新 `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，請新增標籤 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"
   }
  }
  ```

- 根據需要，可選取更新存放區以處理資料要求：

根卷:

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

應用卷:

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

步驟 3 - 執行 Terraform

A. 初始化 Terraform

在終端中,切換到 `edg-terraform` 目錄並執行以下命令:

```
terraform init
```

如果初始化成功,請繼續下一步。如果初始化失敗,請按照 Terraform 輸出中的說明進行動作。

B. 規劃 Terraform

在同一目錄中,執行計劃命令:

```
terraform plan
```

該命令可以多次執行。根據需要執行多次以修復錯誤。當此命令執行無誤時,繼續下一步。

C. 套用 Terraform

從同一目錄執行應用命令:

```
terraform apply
```

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

可選:銷毀 Terraform

可以執行 destroy 命令銷毀整個 VPC:

```
terraform destroy
```

destroy 命令只會銷毀它建立的內容。如果對 AWS 中的某些物件(即安全性群組、子網域等)進行手動變更,則 `destroy` 將失敗。要退出失敗/當機的銷毀動作,請輸入 `Control + C`。然後,必須手動將 VPC 清理到 Terraform 最初建立它時的狀態。然後可以執行 `destroy` 命令。

步驟 4 - 連線到 bastion

所有命令列連線都通過 TCP 22(SSH 協定)上的 bastion 主機。

1. 在 AWS 中,在 bastion 安全性群組中建立輸入規則(**「AWS」>「安全性群組」>「Bastion SG」>「編輯輸入規則」**)並建立規則以允許來自將執行終端命令的 IP 位址或子網域遮罩的 SSH (TCP 22) 連線.

   可選:您可能會發現在部署期間允許在「私人」和「公開」群組中的 EC2 執行個體之間複製檔案很有幫助。建立輸入 SSH 規則:

   - 私有:建立輸入原則以允許來自「公開」的 SSH
   - Public:建立輸入原則以允許來自「私人」和來自「公開」的 SSH

2. 使用在步驟 1.B 中建立的 pem 金鑰連線到 bastion 主機:

   在 Mac 終端上:

   從存放 pem 金鑰的目錄執行以下命令:

   ```
   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`

例如: `ssh -A ec2-user@3.15.12.112`。

在使用 PuTTY 和 Pageant 的 Windows 上:

a. 從 pem 金鑰建立 ppk: 使用 PuTTY 金鑰產生器。載入在步驟 1.B 中建立的 pem 金鑰。金鑰匯入後, 點一下**「儲存私鑰」**。這將建立一個 ppk 檔案。

b. 在 PuTTY 中 - 開啟設定並進行以下變更:

   - Sessions>Host Name: 新增 bastion 主機的IP地址。
   - Sessions>Port: 22
   - Connection>Data>Auto-login username: ec2-user
   - Connection>SSH>Auth>Allow agent forwarding
   - Connection>SSH>Auth> 對於私鑰, 點一下 Browse 並選取剛剛建立的 .ppk 檔案。

c. 安裝 Pageant 並將 ppk 載入到應用程式中。

步驟 5: 安裝 PostgreSQL

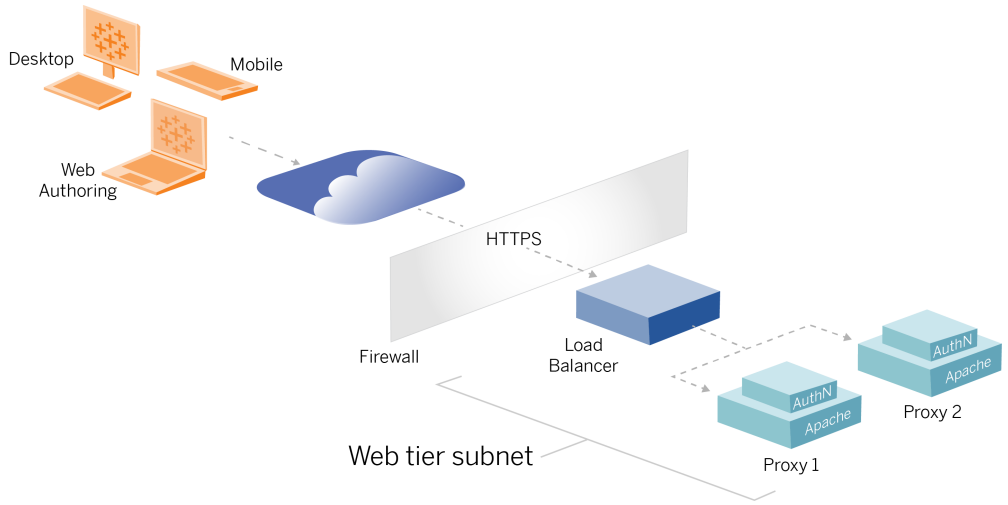
Terraform 範本不安裝 PostgreSQL 用作外部存放庫。但是, 會建立關聯的安全性群組和子網域。如果您在執行 PostgreSQL 的 EC2 執行個體上安裝外部存放庫, 則必須按照第 3 部分 - 準備 Tableau Server Enterprise 部署中所述部署 EC2 執行個體。

然後安裝、設定和 tar 備份 PostgreSQL, 如第 4 部分 - 安裝並設定 Tableau Server 中所述。

步驟 6 -(可選) 執行 DeployTab4EDG

TabDeploy4EDG 指令碼自動執行步驟 4 中描述的四節點 Tableau 部署。請參見 TabDeploy4EDG 自動安裝指令碼。

附錄 - 具有 Apache 示例部署的 Web 層



本主題的其餘部分會介紹一個端到端的過程，該過程描述如何在範例 AWS 參考架構中實作 Web 層。範例設定由以下元件組成：

- AWS 應用程式負載平衡器
- Apache Proxy 伺服器
- Mellon 驗證模組
- Okta IdP
- SAML 驗證

附註：本節中提供的範例 Web 層設定包括部署協力廠商軟體和服務的詳細程序。我們已盡最大努力驗證和記錄啟用 Web 層方案的過程。但是，協力廠商軟體可能會發生變化，或者您的方案可能與此處描述的參考架構不同。權威設定細節和支援請參考協力廠商文件。

整個區段中的 Linux 範例展示了 RHEL-like 發行版的命令。具體來說，這裡的命令是使用 Amazon Linux 2 發行版開發的。若執行的是 Ubuntu 發行版，請編輯相對應的命令。

在此範例中部署 Web 層遵循逐步設定和驗證程序。核心 Web 層設定包含以下步驟，用於在 Tableau 和 Internet 之間啟用 HTTP。Apache 執行並設定為在 AWS 應用程式負載均衡器後面進行反向 proxy/負載均衡：

1. 安裝 Apache
2. 設定反向 Proxy 測試與 Tableau Server 的連線
3. 在 proxy 上設定負載平衡
4. 配置 AWS 應用程式負載均衡器

設定 Web 層並驗證與 Tableau 的連線後，使用外部提供者設定身份驗證。

安裝 Apache

在兩個 EC2 主機（proxy 1 和 proxy 2）上執行以下過程。如果根據參考架構範例在 AWS 中進行部署，那麼應該擁有兩個可用區域且在每個區域中執行單個 Proxy 伺服器。

1. 安裝 Apache：

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

2. 設定在重新啟動時啟動 Apache：

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

3. 驗證安裝的 httpd 版本包括 `proxy_hcheck_module`：

   ```
   sudo httpd -M
   ```

   `proxy_hcheck_module` 是必需的。如果 httpd 版本不包含此模組，則更新到包含它的 httpd 版本。

配置 Proxy 測試與 Tableau Server 的連線

在其中一台 proxy 主機（proxy 1）上執行此過程。此步驟的目的是驗證 Internet 到Proxy 伺服器與私人安全群組中的 Tableau Server 之間的連線。

1. 建立名為 `tableau.conf` 的檔案，並將其新增到 `/etc/httpd/conf.d` 目錄。

   複製以下指令碼並指定 `ProxyPass` 和 `ProxyPassReverse` 具有 Tableau Server 節點 1 的私有 IP 位址的金鑰。

   > **重要提示**：下面顯示的設定不安全，不應在生產中使用。此設定僅應在安裝過程中用於驗證端到端連線。

   例如，如果節點 1 的 私有 IP 位址是 `10.0.30.32`，則`tableau.conf`檔案的內容會是：

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

2. 重新啟動 httpd：

   ```
   sudo systemctl restart httpd
   ```

驗證：基本拓撲組態

應該能夠藉由瀏覽至 `http://<proxy-public-IP-address>`.來存取 Tableau Server 管理員頁面。

如果您的瀏覽器中未載入 Tableau Server 登入頁面，請在 Proxy 1 主機上執行以下疑難排解步驟：

- 停止，然後啟動 httpd 作為疑難排解的第一步。
- 仔細檢查 `tableau.conf` 檔案。驗證節點 1 私有 IP是否正確。驗證雙引號並仔細檢查語法。
- 在帶有節點 1 私有 IP 位址的反向 Proxy 伺服器上執行 `curl` 命令，例如 `curl 10.0.1.90`。如果 shell 不返回 html，或者如果 Apache 測試網頁返回 html，則請驗證「公共」和「私有」安全性群組之間的通訊協定/連接埠設定。

- 執行帶有 Proxy 1 私有 IP 位址的 `curl` 命令，例如 `curl 10.0.0.163`。如果 shell 返回 Apache 測試網頁的 html 代碼，則說明 proxy 設定不正確。
- 在對 proxy 檔案或安全性群組進行組態變更之後，請務必重新啟動 httpd (`sudo systemctl restart httpd`)。
- 確保 TSM 在節點 1 上執行。

在 proxy 上設定負載平衡

1. 在建立的同一 proxy 主機 (proxy 1) 上 `tableau.conf` 檔案，移除現有的虛擬主機設定並編輯檔案以包含負載平衡邏輯。

   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，然後啟動 httpd：

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

3. 瀏覽到 Proxy 1 的公用 IP 位址驗證設定。

將設定複製到第二個 proxy 伺服器

1. 從 proxy 1 複製 `tableau.conf`檔案並將其儲存到 proxy 2 主機上的 `/etc/httpd/conf.d`目錄。

2. 停止，然後啟動 httpd：

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

3. 瀏覽到 Proxy 2 的公用 IP 位址來驗證設定。

配置 AWS 應用程式負載均衡器

將負載平衡器設定為 HTTP 偵聽器。此處的過程描述瞭如何在 AWS 中新增負載平衡器。

步驟 1：建立目標群組

目標群組是定義執行 Proxy 伺服器的 EC2 實例的 AWS 配置。這些是來自 LBS 的流量的目標。

1. EC2 > **目標群組** > **建立目標群組**

2. 在建立頁面上：

   - 輸入目標群組名稱，例如 `TG-internal-HTTP`
   - 目標類型：執行個體
   - 協定：HTTP
   - 連接埠：80
   - VPC：選取您的 VPC

- 在**健康情況檢查>進階健康情況檢查設定>成功代碼**下，附加代碼清單以讀入：`200,303`。
- 按一下**「建立」**

3. 選取剛剛建立的目標群組，然後按一下**「目標」**索引標籤：

    - 按一下 **[編輯]**。
    - 選取要執行 Proxy 應用程式的 EC2 執行個體（或單個執行個體，如果一次設定一個），然後按一下**新增到已註冊**。
    - 按一下**「儲存」**。

步驟 2：啟動負載均衡器精靈

1. EC2 > **負載平衡器** > **建立負載平衡器**

2. 在「選取負載平衡器類型」頁面上，建立一個應用程式負載均衡器。

> **附註**：為設定負載平衡器而顯示的 UI 在 AWS 資料中心之間不一致。下面的「精靈設定」過程與從**步驟 1 設定負載平衡器**開始的 AWS 設定精靈一致。
>
> 如果資料中心在底部包含**建立負載平衡器**按鈕的單個頁面中顯示所有設定，請按照下方「單一頁面設定」過程進行操作。

精靈設定

1. **設定負載平衡器**頁面：

    - 指定名稱
    - 結構：以網際網路為對象（預設）
    - IP 位址類型：ipv4（預設）
    - 接聽程式（接聽程式和路由）：
        a. 保留預設的 HTTP 接聽程式
        b. 按一下**新增接聽程式**並新增 `HTTPS:443`

- VPC:選取已安裝所有內容的 VPC
- 可用區域:
  - 為資料中心區域選取 **a** 和 **b**
  - 在每個相應的下拉式選取器中,選取公用子網路(proxy 伺服器所在位置)。
- 按一下:**設定安全設定**

2. **設定安全性設定**頁面
   - 上傳公用 SSL 憑證。
   - 按一下**「下一步:設定安全群組」**。
3. **設定安全性群組**頁面:
   - 選取公用安全性群組。如果選取預設安全性群組,則清除該選取。
   - 按一下**「下一步:設定路由」**。
4. **設定路由**頁面
   - 目標群體:現有的目標群體。
   - 名稱:選取之前建立的目標群組
   - 按一下**「下一步:註冊目標」**。
5. **註冊目標**頁面
   - 應顯示之前設定的兩個 proxy 伺服器執行個體。
   - 按一下**「下一步:檢閱」**:。
6. **評論**頁面

   按一下**「建立」**。

單一頁面設定

基本設定

- 指定名稱
- 結構:以網際網路為對象(預設)
- IP 位址類型:ipv4(預設)

網路對應

- VPC:選取已安裝所有內容的 VPC
- 對應:
  - 為您的數據中心區域選擇**a**和**b**(或類似的)「可用區域」
  - 在每個相應的下拉式選取器中,選取公用子網路(proxy 伺服器所在位置)。

安全性群組

選取公用安全性群組。如果選取預設安全性群組,則清除該選取。

接聽程式和路由

- 保留預設的 HTTP 接聽程式。對於**預設動作**,請指定之前設定的目標群組。
- 按一下**新增接聽程式**並新增 `HTTPS:443`。對於**預設動作**,請指定之前設定的目標群組。

安全接聽程式設定

- 上傳公用 SSL 憑證。

按一下**建立負載平衡器**。

步驟 3:啟用綁定

1. 建立負載均衡器後,必須在目標群組上啟用綁定。
   - 開啟 AWS 目標群組頁面(**EC2> 負載平衡> 目標群組**),選取剛剛設定的目標群組執行個體。在**「動作」**功能表上,選取**「編輯屬性」**。
   - 在**「編輯屬性」**頁面上,選取**「綁定」**,指定持續時間為 `1 day`,然後**儲存變更**。
2. 在負載均衡器上,在 HTTP 偵聽器上啟用綁定。選取剛剛設定的負載均衡器,然後按一下**「偵聽器」**索引標籤:
   - 對於 **HTTP:80**,按一下**「檢視/編輯規則」**。在產生的**「規則」**頁面上,按一下編輯圖示(原本位於頁面頂端,然後接著位於規則旁邊)以編輯規則。刪除現有 THEN 規則並按一下**新增動作>轉傳至...**以取代它。在產生的 THEN 設定中,指定建立的相同目標群組。在群組層級綁定下,啟用綁定並將持續時間設定為 1 天。儲存設定然後按一下**「更新」**。

步驟 4:在負載平衡器上設定空閒逾時

在負載平衡器上,將空閒逾時更新為 400 秒。

選取為此部署設定的負載平衡器,然後按一下**動作 > 編輯屬性**。將**空閒逾時**設定為 `400` 秒,然後按一下**儲存**。

步驟 5:驗證 LBS 連線

打開 AWS 負載平衡器頁面(**EC2>負載平衡器**),選取剛剛設定的負載平衡器實例。

在**「描述」**下,複製 DNS 名稱並將其貼上到瀏覽器中以存取 Tableau Server 登入頁面。

如果收到 500 級錯誤,那可能需要重新啟動 Proxy 伺服器。

使用公用 Tableau URL 更新 DNS

使用 AWS 負載均衡器描述中的網域 DNS 區域名稱在 DNS 中建立 CNAME 值。流向您的 URL (tableau.example.com) 的流量應傳送到 AWS 公用 DNS 名稱。

驗證連線

DNS 更新完成後,應該能夠輸入公用 URL 巡覽到 Tableau Server 登入頁面,例如,`https://tableau.example.com`。

身份驗證組態範例:帶有外部 IdP 的 SAML

以下範例介紹如何為 AWS 參考架構中執行的 Tableau 部署,設定和配置帶有 Okta IdP 跟 Mellon 驗證模組的 SAML。該範例介紹如何設定 Tableau Server 和 Apache proxy 伺服器以使用 HTTP。Okta 將以 HTTPS 向 AWS 負載均衡器傳送請求,但所有內部流量都將以 HTTP 傳輸。在為此案例進行設定時,請在設定 URL 字串時注意 HTTP 與 HTTPS 協定。

此範例使用 Mellon 作為反向 proxy 伺服器上的預先身份驗證服務提供者模組。此設定可確保只有經過身份驗證的流量連線到 Tableau Server，Tableau Server 還充當 Okta IdP 的服務提供者。因此，必須設定兩個 IdP 應用程式：一個用於 Mellon 服務提供者，另一個用於 Tableau 服務提供者。

建立 Tableau 管理員帳戶

配置 SAML 時的一個常見錯誤是在啟用 SSO 之前忘記在 Tableau Server 上建立管理員帳戶。

第一步是在 Tableau Server 上建立一個具有伺服器管理員角色的帳戶。在 Okta 情境中，使用者名稱必須採用有效電子郵件位址的形式，例如 user@example.com。必須為此使用者設定密碼，但在設定 SAML 後將不再使用該密碼。

配置 Okta 預身分驗證應用程式

本區段描述的端到端方案需要兩個 Okta 應用程式：

- Okta 預身分驗證應用程式
- Okta Tableau Server 應用程式

這些應用程式都與您需要分別在反向 proxy 和 Tableau Server 上設定的不同中繼資料關聯。

此過程描述如何建立和設定 Okta 預身份驗證應用程式。在本主題的後面，您將建立 Okta Tableau Server 應用程式。有關使用者受限的免費測試 Okta 帳戶，請參閱Okta 開發者網頁。

為 Mellon 預先身份驗證服務提供者建立 SAML 應用程式整合。

1. 打開 Okta 管理儀表板 **>應用程式>建立 App 集合**。
2. 在**新建 app 集合**頁面上，選擇**SAML 2.0**，然後按一下**「下一步」**。

3. 在**「一般設定」**索引標籤上，輸入應用程式名稱，例`Tableau Pre-Auth`，然後按一下**「下一步」**。

4. 在**設定 SAML**索引標籤上：

   - 單一登入 (SSO) URL。單一登入 URL 路徑的最後一個元素是指接在此程序之後，在`mellon.conf`組態檔中的`MellonEndpointPath`。可以指定想要的任何端點。在這個範例中，`sso`為端點。最後一個元素，`postResponse`是必需的：`https://tableau.example.com/sso/postResponse`。
   - 清除核取方塊：**將此用於收件者 URL 和終點 URL** 。
   - 收件者 URL：與 SSO URL 相同，但使用 HTTP。例如，`http://tableau.example.com/sso/postResponse`。
   - 目的地 URL：與 SSO URL 相同，但使用 HTTP。例如，`http://tableau.example.com/sso/postResponse`。
   - 受眾 URI(SP 實體 ID) 。例如，`https://tableau.example.com`。
   - 名稱 ID 格式：`EmailAddress`
   - 應用程式使用者名稱：`Email`
   - 屬性聲明：名稱 = `mail`；名稱格式 = `Unspecified`；值 = `user.email`。

   按一下**「下一步」**。

5. 在**「回饋」**索引標籤上，選取：

   - **我是新增內部應用程式的 Okta 客戶**
   - **這是我們建立的內部應用程式**
   - 按一下**「完成」**。

6. 建立預授權 IdP 中繼資料檔案：

   - 在 Okta 中：**應用程式>應用程式>**您的新應用程式(例如 `Tableau Pre-Auth`) **>登入**
   - 在 **SAML 簽署憑證**旁邊，按一下**檢視 SAML 設定說明**。
   - 在**如何為<預授權>應用程式設定 SAML 2.0** 頁面上，向下捲動到**可選**部分，**向您的 SP 提供者提供以下 IDP 中繼資料**。
   - 複製 XML 欄位的內容，並將它們儲存在名為 `pre-auth_idp_metadata.xml` 的檔案中。

7. (可選) 設定多重要素驗證：

- 在 Okta 中：**應用程式>應用程式>**您的新應用程式（例如 `Tableau Pre-Auth`）**>登入**
- 在**「登入原則」**下，按一下**「新增規則」**。
- 在**「應用程式登入規則」**上，指定名稱和不同 MFA 選項。要測試功能，可以將所有選項保留為預設值。但是，在**「動作」**下，必須選取**「提示要素」**，然後指定使用者必須登入的頻率。按一下**「儲存」**。

建立和指派 Okta 使用者

1. 在 Okta 中，使用在 Tableau 中建立的相同使用者名稱建立一個使用者 (user@example.com)：**「目錄」>「人員」>「新增人員」**。
2. 建立使用者後，將新的 Okta 應用程式指派給該人員：按一下使用者名稱，然後在**「指派應用程式」中指派應用程式**。

安裝 Mellon 進行預身份驗證

1. 在執行 Apache proxy 伺服器的 EC2 執行個體上以下命令以安裝 PHP 和 Mellon 模組：

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

2. 建立 `/etc/httpd/mellon` 目錄

將 Mellon 設定為預身份驗證模組

在兩個 proxy 伺服器上執行此過程。

您必須擁有從 Okta 設定建立的 `pre-auth_idp_metadata.xml` 檔案複本。

1. 變更目錄：

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

2. 建立服務提供者後設資料。執行 `mellon_create_metadata.sh` 指令碼。命令中必須包含您組織的實體 ID 和返回 URL。

在 Okta 中，返回 URL 被稱為*單一登入 URL*。返回 URL 路徑的最後一個元素是指接在此程序之後，在mellon.conf組態檔中的MellonEndpointPath。在這個範例中，我們指定sso作為路徑終點。

例如：

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

該指令碼回傳服務提供者憑證、金鑰和後設資料檔。

3. 為了容易閱讀，請重新命名mellon目錄中的服務提供者檔案。我們將使用以下名稱來引用文件中的檔案：

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

4. 將 pre-auth_idp_metadata.xml 檔案複製到同一目錄中。

5. 在 /etc/httpd/conf.d 目錄中建立mellon.conf檔案：

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

6. 將以下內容複製到mellon.conf.

```
<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. 將以下內容新增到現有的 tableau.conf 檔案：

在 `<VirtualHost *:80>` 區塊中，新增以下內容。使用實體 ID 中的公共主機名稱更新 `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>` 區塊外新增「位置」區塊。使用頂層網域更新 `MellonCookieDomain` 來保留 cookie 資訊，如下所示：

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

完整的 `tableau.conf` 檔案看起來可能類似於以下範例：

```
<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. 驗證組態。執行以下命令:

   ```
   sudo apachectl configtest
   ```

   如果組態測試回傳錯誤,請修復所有錯誤並再次執行組態測試。成功的組態會回傳,`Syntax OK`。

9. 重新啟動 httpd:

   ```
   sudo systemctl restart httpd
   ```

在 Okta 中建立 Tableau Server 應用程式

1. 在 Okta 儀表板中:**「應用程式」**>**「應用程式」**>**「瀏覽應用程式目錄」**
2. 在**瀏覽應用程式整合目錄**中,搜尋 `Tableau`,選取 Tableau 伺服器 動態磚,然後按一下**「新增」**。
3. 在**「新增 Tableau Server」**>**「一般設定」**上,輸入標籤,然後按一下**「下一步」**。
4. 在登入選項中,選取 **SAML 2.0** 並向下滾動到進階登入設定:
   - **SAML 實體 ID**:輸入公用 URL,例如 https://tableau.example.com。
   - **應用程式使用者名稱格式**:Email
5. 按一下**「身份提供者中繼資料」**連結以啟動瀏覽器。複製瀏覽器連結。這是在以下過程中設定 Tableau 時將使用的連結。
6. 按一下 **[完成]**。

7. 將新的 Tableau Server Okta 應用程式指派給您的使用者 (user@example.com)：按一下使用者名稱，然後在**「指派應用程式」**中指派應用程式。

在 Tableau Server 上為 IdP 啟用 SAML

在 Tableau Server 節點 1 上執行此過程。

1. 從 Okta 下載 Tableau Server 應用程式中繼資料。使用在上一過程中儲存的連結：

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

2. 將 TLS 憑證和相關金鑰檔案複製到 Tableau Server。金鑰檔案必須是 RSA 金鑰。有關 SAML 憑證和 IdP 要求詳細資訊，請參閱 *SAML 要求* (Linux)。

   為簡化憑證管理和部署，並作為安全性最佳做法，我們建議使用由受信任的主要協力廠商憑證頒發授權單位 (CA) 產生的憑證。或者，您可以產生自我簽署憑證或使用針對 TLS 的 PKI 憑證。

   如果您沒有 TLS 憑證，則可以使用下面的內嵌過程產生自簽章憑證。

產生自簽章憑證

在 Tableau Server 節點 1 上執行此過程。

a. 產生簽署根憑證頒發授權單位 (CA) 金鑰：

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

b. 建立根 CA 憑證：

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

系統將提示輸入憑證欄位的值。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. 建立憑證和相關金鑰(下方範例中的 `server-saml.csr` 和 `server-saml.key`)。憑證的主題名稱必須與 Tableau 主機的公用主機名稱相符。主題名稱設定為 `-subj` 選項,帶有格式 `"/CN=<host-name>"`,例如:

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

d. 使用您在上述步驟中建立的 CA 憑證簽署新憑證。以下命令也將以 `crt` 格式輸出憑證:

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

e. 將金鑰檔案轉換為 RSA。Tableau 需要用於 SAML 的 RSA 金鑰檔案。若要轉換金鑰,請執行以下命令:

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

3. 配置 SAML。執行以下命令,指定您的實體 ID 和傳回 URL,以及中繼資料檔、憑證檔案和金鑰檔案的路徑:

```
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. 如果您的組織執行 Tableau Desktop 2021.4 或更高版本，則必須執行以下命令以用反向 proxy 伺服器啟用身份驗證。

   Tableau Desktop 2021.2.1 到 2021.3 的版本無須執行此命令即可執行，前提是您的預先身份驗證模組（例如 Mellon）設定為允許保留頂層網域 cookie。

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

5. 套用組態變更：

   ```
   tsm pending-changes apply
   ```

驗證 SAML 功能

要驗證端到端 SAML 功能，請使用您在此過程開始時建立的 Tableau 管理員帳戶，透過公用 URL（例如 https://tableau.example.com）登入到 Tableau Server。

驗證疑難排解

錯誤的要求：這種情況下的一個常見錯誤是來自 Okta 的「錯誤的要求」錯誤。當瀏覽器快取來自先前 Okta 工作階段的資料時，通常會發生此問題。例如，如果您以 Okta 管理員身份管理 Okta 應用程式，然後嘗試使用不同的啟用 Okta 帳戶存取 Tableau，則來自管理員資料的工作階段資料可能會導致「錯誤的要求」錯誤。如果清除本機瀏覽器快取後此錯誤仍然存在，請嘗試與其他瀏覽器連線來驗證 Tableau 方案。

「錯誤請求」錯誤的另一個原因是在 Okta、Mellon 和 SAML 設定過程中輸入的眾多 URL 之一中發生拼寫錯誤。請仔細檢查所有這些輸入。

通常 Apache 伺服器上的 httpd `error.log` 檔案會指定導致錯誤的 URL。

找不到 - 在此伺服器上找不到請求的 URL：此錯誤代表許多設定錯誤的其中一種。

如果使用者使用 Okta 進行身份驗證，然後收到此錯誤，則很可能是在設定 SAML 時將 Okta 預身份驗證應用程式上傳到 Tableau Server。驗證出您在 Tableau Server 上設定了 Okta Tableau Server 應用程式中繼資料，而不是 Okta 預身份驗證應用程式中繼資料

其他疑難排解步驟：

- 仔細檢查 `tableau.conf` 是否有錯字或設定錯誤
- 檢查 Okta 預身份驗證應用程式設定。確定 HTTP 與 HTTPS 協定按照此主題指定步驟設定。
- 在兩個 proxy 伺服器上皆重新啟動 httpd。
- 驗證 `sudo apachectl configtest` 在兩個 proxy 伺服器上都傳回「語法正常」。
- 驗證測試使用者是否已指派給 Okta 中的兩個應用程式。
- 驗證是否在負載平衡器和關聯的目標群組上設定綁定

設定從負載平衡器到 Tableau Server 的 SSL/TLS

一些組織需要從用戶端到後端服務的端到端加密通道。到目前為止所述的預設參考架構可指定從用戶端到在您組織的 Web 層中執行的負載平衡器的 SSL。

要設定從負載平衡器到 Tableau Server 的 SSL，必須：

- 在 Tableau 和 Proxy 伺服器上安裝有效的 SSL 憑證。
- 設定從負載平衡器到反向 Proxy 伺服器的 SSL。
- 設定從 Proxy 伺服器到 Tableau Server 的 SSL。
- 還可以設定從 Tableau Server 到 PostgreSQL 執行個體的 SSL。

本主題的其餘部分在範例 AWS 範例參考架構的內容中描述了此實作。

範例:在 AWS 參考架構中設定 SSL/TLS

本節介紹如何在 Tableau 上設定 SSL 以及如何在 Apache proxy 伺服器上設定 SSL, 這些都在範例 AWS 參考架構中執行。

此範例的整個 Linux 程序展示了 RHEL-like 發行版的命令。具體來說, 這裡的命令是使用 Amazon Linux 2 發行版開發的。若執行的是 Ubuntu 發行版, 請編輯相對應的命令。

第 1 步:收集憑證和相關金輪

為簡化憑證管理和部署, 並作為安全性最佳做法, 我們建議使用由受信任的主要協力廠商憑證頒發授權單位 (CA) 產生的憑證。

或者, 您可以產生自我簽署憑證或使用針對 TLS 的 PKI 憑證。

以下過程如何產生自我簽署憑證。如果按照我們的建議使用協力廠商憑證, 則可以跳過此過程。

在其中一台 proxy 主機上執行此過程。生成憑證和相關金鑰後, 會將它們分享到其他 proxy 主機和 Tableau Server 節點 1 。

1. 產生簽署根憑證頒發授權單位 (CA) 金鑰:

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

2. 建立根 CA 憑證:

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

   系統將提示輸入憑證欄位的值。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. 建立憑證和相關金鑰(下方範例中的 `serverssl.csr` 和 `serverssl.key`)。憑證的主題名稱必須與 Tableau 主機的公用主機名稱相符。主題名稱設定為 `-subj` 選項,帶有格式 `"/CN=<host-name>"`,例如:

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

4. 使用您在步驟 2 中創建建立的 CA 憑證簽署新憑證。以下命令也將以 `crt` 格式輸出憑證:

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

步驟 2:為 SSL 設定 proxy 伺服器

在兩個 proxy 伺服器上執行此過程。

1. 安裝 Apache ssl 模組:

```
sudo yum install mod_ssl
```

2. 建立`/etc/ssl/private`目錄:

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

3. 將 crt 和金鑰檔複製到以下`/etc/ssl/`路徑:

```
sudo cp serverssl.crt /etc/ssl/certs/

sudo cp serverssl.key /etc/ssl/private/
```

4. 使用以下最新內容來更新現有的tableau.conf:

   - 加入 SSL 重寫區塊:

     ```
     RewriteEngine on
     RewriteCond %{SERVER_NAME} =tableau.example.com
     RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI}
     [END,NE,R=permanent]
     ```

   - 在 SSL 重寫區塊中,更新RewriteCond伺服器名稱:加入您的公用主機名稱,例如,tableau.example.com
   - 將<VirtualHost *:80>變更為 <VirtualHost *:443>。
   - 將 <VirtualHost *:443>和具有 <IfModule mod_ssl.c>的 <Location />區塊回繞。</IfModule>.
   - BalancerMember:將協定http變更到https。
   - 將SSL*元素加入到 <VirtualHost *:443>區塊:

     ```
     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:加入ssl:warn。
   - 可選:如果已安裝並設定了身份驗證模組,則 tableau.conf 檔中可能包含其他元素。例如, <Location /> </Location>區塊將包含元素。

   此處顯示的範例是為 SSL 設定的 tableau.conf 檔:

   ```
   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. 新增 索引.html 檔以抑制 403 錯誤：

```
sudo touch /var/www/html/index.html
```

6. 重新啟動 httpd:

```
sudo systemctl restart httpd
```

步驟 3:為外部 SSL 設定 Tableau Server

將 serverssl.crt 和 serverssl.key 檔從 Proxy 1 主機複製到初始 Tableau Server(節點 1) 。

在節點 1 上執行以下命令:

```
tsm security external-ssl enable --cert-file serverssl.crt --key-
file serverssl.key
tsm pending-changes apply
```

步驟 4:身分驗證組態(選用)

如果已為 Tableau 設定了外部身分識別提供者, 則可能需要更新 IdP 管理儀表板中的返回 URL。

例如, 如果使用 Okta 預身分驗證應用程式, 則需要更新應用程式以對接收 URL 和目標 URL 使用 HTTPS 協定。

步驟 5:為 HTTPS 設定 AWS 負載平衡器

如果按照本指南中的說明使用 AWS 負載平衡器進行部署, 則可以重新設定 AWS 負載平衡器以將 HTTPS 流量發送到 proxy 伺服器:

1. 註銷現有的 HTTP 目標群組:

   在**目標群組**中, 選擇已為負載平衡器設定的 HTTP 目標群組, 按一下**動作**, 然後按一下**註冊和註銷執行個體**。

   在**註冊和註銷目標**頁面, 選擇目前設定的執行個體, 按一下**註銷**, 然後按一下**儲存**。

2. 建立 HTTPS 目標群組:

目標群組 > 建立目標群組

- 選取「執行個體」
- 輸入目標群組名稱，例如 `TG-internal-HTTPS`
- 選取 VPC
- 協定：HTTPS 443
- 在**健康情況檢查>進階健康情況檢查設定>成功代碼**下，附加代碼清單以讀入：`200,303`。
- 按一下**「建立」**。

3. 選取剛剛建立的目標群組，然後按一下**「目標」**索引標籤：

   - 按一下 **編輯**
   - 選取正在執行 Proxy 應用程式的 EC2 執行個體，然後按一下**新增至已註冊**。
   - 按一下**「儲存」**。

4. 建立目標群組後，必須啟用相黏：

   - 開啟 AWS 目標群組頁面（**EC2> 負載平衡> 目標群組**），選取剛剛設定的目標群組執行個體。在**「動作」**功能表上，選取**「編輯屬性」**。
   - 在**「編輯屬性」**頁面上，選取**「綁定」**，指定持續時間為 `1 day`，然後**儲存變更**。

5. 在負載平衡器上，更新收聽器規則。選擇為此部署設定的負載平衡器，然後按一下**收聽器**索引標籤。

   - 對於 **HTTP:80**，按一下**「檢視/編輯規則」**。在產生的**「規則」**頁面上，按一下編輯圖示（原本位於頁面頂端，然後接著位於規則旁邊）以編輯規則。刪除現有 THEN 規則並按一下**新增動作 > 重新導向至...**來替換它。在生成的 THEN 組態中，指定`HTTPS`和連接埠`443`並將其他選項保留為預設值。儲存設定然後按一下**「更新」**。
   - 如果是 **HTTP:443**，請按一下**「檢視/編輯規則」**。在產生的**「規則」**頁面上，按一下編輯圖示（原本位於頁面頂端，然後接著位於規則旁邊）以編輯規則。在 **THEN** 設定中的**傳送至...**下，將目標群組變更為剛剛建立的 HTTPS 群組。在**群組層級綁定**下，啟用綁定並將持續時間設定為 1 天。儲存設定然後按一下**「更新」**。

步驟 6: 驗證 SSL

透過瀏覽到 https://tableau.example.com 來驗證組態。