**Tableau Server: Timeline and Overview of Deployment Options**

**Purpose of this document:**

The primary purpose of this document is to step you through several ways in which you can scale out Tableau Server software. It is meant to be a companion document to our [built-in administrative help](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#admin.htm), with a focus on discrete, actionable steps you can take as part of your Tableau Server rollout. Although we provide rough timelines and estimates below, your specific implementation and timeline may vary due to external factors such as hardware procurement, IT policies and so forth.

*At Tableau Software, we have an agile methodology known as “Tableau Drive” – this document is an example of that workflow process, and has a focus of fast and iterative design and product usage. The beginning of this document is broken out into “4 days” of effort, with tasks and review material for each day – this is “Tableau Drive” at work!*

A secondary goal of this document is to highlight the differences between the definitions of the phrases “Performance”, “Scalability”, “High Availability”, and “Disaster Recovery”. These four concepts are independently discussed throughout this document. Sometimes they overlap when applicable. However, it is important to note that they are four separate concepts which can be discussed in isolation of each other.

This document is procedural as well as workflow -oriented. You are being asked to take actionable steps such as installing the Tableau Server software, configuring components of the software, and testing the software for performance and scalability. Tableau Software is very hands on – the best way to learn these concepts is to step through the tasks shown below in conjunction with the built-in online administrative help.

Prerequisites: if you were to complete *every single step in this entire document*, you would need 4 machines for “production”, a second 4 machines for “dev”, a third 4 machines for “Disaster Recovery”, and three distinct external load balancers (not Tableau provided). You are welcome to skip as many of these configuration steps as needed based upon your available hardware (or virtual equivalent). You would end up with three independent installations of Tableau Server. All three installations would have High Availability configured. One of the installations would be for “Production”, the second for “test/qa/dev”, and the third for “Disaster Recovery”. This would be 12 machines in total plus 3 machines as load balancers (typically either open source such as Apache with the mod-balancer module, or commercial hardware load balancers products from F5 or similar)

* Provided by you: machines running a valid operating system as [specified here](http://www.tableausoftware.com/products/server/specs).
* Provided by Tableau: Tableau Server software, Tableau Server worker node software, and database drivers.

**Definitions:**

* *Process*: the various components that make up the Tableau Server software solution.
* *Node/Machine*: these words are interchangeable. They refer to a machine running Tableau Server primary software, or Tableau Server worker software. These can be physical machines or virtualized machines.
* *Instance*: the sum total of the Tableau Server topology or footprint. This can be one or N nodes/machines.
* *Hardware spec*: for this document, we assume all machines are dual quad (8 core), 32gb RAM, 500 GB drives… or similar, virtual equivalent.

Before you begin: review basic prerequisites prior to installing Tableau Server: <http://onlinehelp.tableausoftware.com/current/server/en-us/requ.htm>

**Outline of this document:**

* **4 day implementation steps**
* **Additional Scalability and Performance Considerations**
* **Auditing Tableau Server Usage**
* **Appendix A: Overview of processes/components in Tableau Server**
* **Appendix B: Configuring Apache as an external load balancer (ELB)**
* **Appendix C: Additional Information on SSL Certificates for Tableau Instances**
* **Appendix D: ELB with Netscaler**

**Day One**

* Install Tableau Server on “dev”, single-machine only, for backup, test and dev purposes. Set vizQL to 2, app to 1, and backgrounders to 1. You cannot disable data engine, data server, or repository for a single-server installation. Use our [built-in online server help](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#install.htm) for help with this installation.
* Install Tableau Server on “production”, second single-machine installation for production. Set vizQL to 2, app to 1, and backgrounders to 1. You cannot disable data engine, data server, or repository for a single-server installation. Use our [built-in online server help](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#install.htm) for help with this installation.

You now have a single node instance (two of them), as shown in the diagram below.



A list of the processes and their purposes is part of the built-in administrative help for Tableau Server:

<http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#processes.htm>

Some tests you can perform at this point:

* Publish a viz from Tableau Desktop to Tableau Server
* [Configure user accounts](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#users.htm)
* Enable various Tableau Server features such as enabling [performance recording](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#perf_record_create_server.htm) or [adding a new schedule](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#schedule_manage_create.htm).

**We strongly encourage you to spend the rest of day one using the Tableau Desktop software, with a focus on publishing content from Tableau Desktop to Tableau Server!**

**Day Two**

* Stop the Tableau Server.
* Install Tableau Server worker software on “dev”, adding a worker node to primary server. [Detailed instructions are located here](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#distrib_setup.htm).
* Please note that this online help page has several other URL links to additional information – please read all of these links!
* For this task, configure your worker node as follows (changing the IP address to reflect your worker node):



* You now have a two-machine cluster for “dev”. This is not yet High Availability (HA).
* Install Tableau Server worker on “production”, adding a worker node to the primary server.
* You now have a two-machine cluster for “production”. This is not yet High Availability HA.

You now have the most basic clustered instance of Tableau Server possible, shown in the diagram below. In this example, we configured the second machine to serve Vizql server processes, application server processes, and background processes.



Some tests you can perform at this point:

* Verify all components are running correctly via the web admin console.
* Enable [custom administrative views (postgres auditing)](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#adminview_postgres.htm) and verify that both the primary and worker node are being “hit” by your end users. This is easily observed with a copy of Tableau Desktop by connecting to the \_http\_requests table and looking for the dimensions called “worker” and/or “action”.
* Scheduled extractions. You can publish a workbook which is connected to your original database (oracle, SQL server, etc), but also in extract mode, and test the refresh/automation of the extraction. More info from our built-in help is [located here](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#schedules.htm).

The remainder of Day Two should be spent publishing visualizations and data sources, and experimenting with your Tableau Server instance (remember: instance means the total sum of the parts of the installation, which currently is two nodes/machines).

**Day Three**

* Stop the Tableau Server Instance.
* Install a second worker node into the Tableau Server instance on “dev”, adding a second worker machine to the existing two-machine cluster. [Detailed instructions are located here.](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#distrib_setup.htm)
* Set this worker with “secondary repository” and “secondary data engine”



* Because you checked the box for “repository” as well as marked a “1” on Data Engine, you have implicitly enabled High Availability (HA) for these processes. No further HA configuration is required.
* Install a second worker node into the Tableau Server instance on “production”, adding second worker machine to the existing two-machine cluster. [Detailed instructions are located here.](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#distrib_setup.htm)
* Be patient. Even after Tableau Server is restarted, components may appear to be ‘down’ or not running. This may be due to the sync of content between nodes.
* You now have *partial* HA for two components: Tableau’s repository and the Tableau Data Engine. Your total configuration for the entire instance (again there are two of them, one for dev and one for production) should now look like this:



At this point, let’s stop and discuss a few items. First, why 4 VizQL processes on one machine versus 2 VizQL processes on a second machine? Mostly to show you that this type of asymmetric configuration is possible and that you have some options with the configuration. However, in the real world, you should start all nodes with 2 VizQL processes on any 64-bit instances. This is to increase the chance of memory cache hits occurring. There are edge cases where cache hits will rarely or never occur, in which case it may make more sense to have 4 or 8 vizql processes for concurrency reasons.

Second, why place backgrounder tasks on all three machines? Again, partially to show you that this can be done. In real world terms, you should try to restrict backgrounder processes to 2 machines. That way, if one of the nodes dies, background tasks can still run on the second node, but the background tasks don’t use up ram and CPU for any additional machines (3rd or Nth machines).

Why is this only “partial” HA? Because while you have configured an active secondary data engine and a standby repository, the primary nodes – which runs the Tableau gateway component as well as the license check - is still a point of failure. Later on, we will discuss the configuration of an external load balancer to solve for this situation.

Some tests you can perform at this point:

* Verify that the repository database is replicating correctly. One way to do this is to create [administrative access to postgres](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#adminview_postgres.htm) – you could then build a connection to the active repository as well as the standby, and compare the record counts found in various common tables.
* Verify that the data engine is replicating correctly. One way to do this is to compare folder sizes for C:\ProgramData\Tableau\Tableau Server\data\tabsvc on the two machines running the data engine processes.
* Observe the maintenance screen in the Tableau Server web UI. Here is an example (it is not the same configuration as shown above) of a two machine instance configured for HA:



**Day Four**

We are going to add a fourth machine running additional worker node components, as well as an eternal load balancer to the equation. This will enable us to achieve 100% high availability for all components.

* Stop the Tableau Server Instance for “dev”.
* Install a third worker node into the Tableau Server instance on “dev”, adding a third worker machine to the existing three-machine cluster. [Detailed instructions are located here.](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#distrib_setup.htm)
* Regardless of any previous configuration, set all components for all workers in this cluster as shown in the following picture. This achieves a very symmetrical installation across the nodes; HA for the repository and the data engine across two nodes; a backgrounder process on two nodes; and the gateway on all nodes.



* Restart “dev” and wait patiently. Sending the packages across the nodes can take some time.
* Now you can configure the system for an External Load Balancer.
* Detailed instructions are in our [online help](http://onlinehelp.tableausoftware.com/current/server/en-us/distrib_lb.htm). We have also included appendix information in this document specific to Apache as well as general comments about real world experiences with external load balancers.
* Configuration of an external load balancer is not an introductory task, and requires detailed knowledge about your load balancer (Apache, F5, etc). Please ensure that you configure your “dev” instance prior to working on your production instance.
* Once you have your “dev” instance configured, complete all of the steps shown above for your “production” instance.

At this point, you have two environments:

* A production environment with four nodes plus an ELB
* A dev environment with four nodes plus an ELB

Some tests you can perform at this point include:

* Migrating content from prod to dev (or dev to prod) using “tabadmin backup”. See our [built-in help](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#db_backup.htm) or our [KB article](http://kb.tableausoftware.com/articles/knowledgebase/server-maintenance).
* Perform addition auditing across all of the nodes in the instance. Please review the section in this document entitled “Auditing Tableau Server Usage”.
* Verify that traffic from the web browser to the Tableau instance nodes is NOT allowed (via firewall). All traffic should be required to go through the external load balancer. Please review the section in this document entitled “Additional Information on SSL Certificates for Tableau Instances”.
* Verify system behavior by “pulling the plug” on one of the workers. You should be able to review the maintenance page and see in real time the system attempting to correct itself when one of the nodes goes down.
* Continue reading this document for information on “hardening” your servers as well as auditing and performance topics.

**Additional Scalability and Performance Considerations**

**Caching**

Caching (the storage of ‘results’ for later use) exists in multiple components of Tableau Server, including:

* The Tableau Data Engine, which will cache answers into RAM that it has already processed.
* The Tableau VizQL processes, which will cache answers into RAM that it has already processed.
* The Tableau Web Application, which will cache the retrieval of content (workbooks) as well as the general user interface

**Specific Hardware Concepts**

Hard Drives:

* Whenever possible, use solid state hard drives. Tableau Server will frequently read and write to the disk. There is simply no good reason to *not* use solid state drives in 2014 and beyond!
* Temp space. Tableau Server uses both the windows TEMP variable as well as our own temp area in C:\ProgramData\Tableau\Tableau Server\temp. A general rule is to have *at least 5X free space* as a ratio of all of the Tableau data files on the system. Example: if your data directory (“C:\ProgramData\Tableau\Tableau Server\data\tabsvc\dataengine”) is consistently 2 GB in size, you should always have 10GB free on that hard drive. In fact, a good system administrator would tell you that a 10X ratio is preferred, meaning you would have 20GB free.
* If you installed Tableau Server to a non-C drive, Tableau will use temp space on that drive instead.

CPU’s:

* TBD

RAM:

* TBD
* Tableau Server can operate on 64-bit machines and is designed to leverage 64-bit memory addressing. This means you can have fewer VizQL processes running than on an equivalent 32-bit system. Each process can consume large quantities of RAM, which will help increase the chance that caching is used.

Ethernet/Network:

* TBD

**Checklist for “Hardening” your Tableau Server:**

* Disable Windows OS “auto updates” - you don’t want windows restarting during a tabadmin backup, or while any other Tableau Server process is running!
* Same thing with virus software updates!
* Exclude c:\ProgramData\Tableau\ from virus checks
* Turn off any windows services which are not applicable.
* Disable windows firewall. Tableau Server only allows two ways into the system: the “front door” which is port 80 or 443, and the “back door” which is Windows Remote Desktop. Thus, you do not need a firewall rule on the windows machine.
* Enable Remote Desktop encryption. [Read more about that here](http://technet.microsoft.com/en-us/magazine/ff458357.aspx).

**Auditing Tableau Server Usage**

Tableau Server includes built-in auditing which is logged in the Tableau Server database itself. To access this data, you need to [enable the feature](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#adminview_postgres.htm); it is not enabled by default. You can then use Tableau Desktop to connect to this audit model.

**Basic steps to configure auditing:**

* Configure the feature using the link shown above.
* Create a new Tableau workbook connecting to “postgreSQL”.
* Port is “8060”; database is “workgroup”; username is “tableau”; password is whatever you assigned in the instructions shown above.

**There are two types of auditing information.**

* There is a snapshot of the “current system” – these are live database views which all begin with an underscore (“\_”)
* There is also detailed historical auditing. These are tables which are continually updated by the system in real time.

**For the snapshot of the “current system”,** you can connect to any of the following views with Tableau Desktop:

\_background\_tasks

\_comments

\_customized\_views

\_datasources

\_datasources\_stats

\_groups

\_http\_requests

\_projects

\_schedules

\_sessions

\_sites

\_subscriptions

\_system\_users

\_tags

\_users

\_views

\_views\_stats

\_workbooks

users\_view

Many of these views have obvious joins between them. For example: \_views.workbook\_id = \_workbooks.id

Others are not necessarily meant to be joined, e.g. “\_background\_tasks”

**For historical auditing,** the two primary tables are shown here:



The important fields to use are:

* **historical\_event\_types.name** – the detailed name of the audit event. There are dozens and dozens of actions which are audited by the system.
* **historical\_event\_types.action\_type** – a higher level classification, some choices are “publish”, “delete”, “access” et cetera
* **historical\_events.created\_at** – a timestamp for the audit event, recorded in GMT time.

There are several additional tables with a prefix of “hist”. These are to be left-joined to the primary fact table shown above. The reason for the left join is that not all audit events require all types of dimensions. If you perform an inner join on all these “hist” tables, you may get incomplete results. An example of the joins is shown below.

 LEFT JOIN "public"."historical\_event\_types" "historical\_event\_types" ON ("historical\_events"."historical\_event\_type\_id" = "historical\_event\_types"."type\_id")

 LEFT JOIN "public"."hist\_users" "hist\_users\_actor" ON ("historical\_events"."hist\_actor\_user\_id" = "hist\_users\_actor"."id")

 LEFT JOIN "public"."hist\_views" "hist\_views" ON ("historical\_events"."hist\_view\_id" = "hist\_views"."id")

 LEFT JOIN "public"."hist\_sites" "hist\_sites" ON ("historical\_events"."hist\_target\_site\_id" = "hist\_sites"."id")

 LEFT JOIN "public"."hist\_tasks" "hist\_tasks" ON ("historical\_events"."hist\_task\_id" = "hist\_tasks"."id")

 LEFT JOIN "public"."hist\_projects" "hist\_projects" ON ("historical\_events"."hist\_project\_id" = "hist\_projects"."id")

 LEFT JOIN "public"."hist\_groups" "hist\_groups" ON ("historical\_events"."hist\_group\_id" = "hist\_groups"."id")

 LEFT JOIN "public"."hist\_workbooks" "hist\_workbooks" ON ("historical\_events"."hist\_workbook\_id" = "hist\_workbooks"."id")

 LEFT JOIN "public"."hist\_comments" "hist\_comments" ON ("historical\_events"."hist\_comment\_id" = "hist\_comments"."id")

 LEFT JOIN "public"."hist\_datasources" "hist\_datasources" ON ("historical\_events"."hist\_datasource\_id" = "hist\_datasources"."id")

 LEFT JOIN "public"."hist\_schedules" "hist\_schedules" ON ("historical\_events"."hist\_schedule\_id" = "hist\_schedules"."id")

 LEFT JOIN "public"."hist\_capabilities" "hist\_capabilities" ON ("historical\_events"."hist\_capability\_id" = "hist\_capabilities"."id")

 LEFT JOIN "public"."\_users" "\_users" ON ("hist\_users\_actor"."user\_id" = "\_users"."id")

 LEFT JOIN "public"."hist\_users" "hist\_users\_target" ON ("historical\_events"."hist\_target\_user\_id" = "hist\_users\_target"."id")

A comment about the “user” dimension tables: there are two joins possible here, one for the “actor” and a second one for the “recipient”. Take note that in table “historical\_events” there is an ID “hist\_target\_user\_id” as well as an ID “hist\_target\_user\_id ”

Please see the companion workbook which attempts to connect to a “localhost” (lookback, i.e. 127.0.0.1) machine running Tableau Server. You need to edit the database connection in this Tableau workbook to point to your installation.

Example audit dashboard using the companion workbook:



**Tableau Auditing combined with Windows Performance monitor**

Analyzing Tableau Server performance can be challenging. This is due to the fact that no one installation is like any other installation. Also, by design, the same technology that allows Tableau Server to scale out to massive levels, makes it harder to analyze the entire system from a holistic perspective. Tableau has some great built-in tools including the [performance recorder](http://onlinehelp.tableausoftware.com/current/server/en-us/perf_record_create_server.htm) as well as a [white paper](http://www.tableausoftware.com/learn/whitepapers/tableau-server-scalability-explained) wherein they benchmark a given installation and response times.

In addition, there are a few different disparate data sources that we can blend together into a unified view in order to analyze Tableau Server performance. They include:

* Windows Performance Monitor
* Tableau Server HTTP requests
* Tableau Server audit events and
* Tableau Server background tasks.

Here is a short list of steps to get this type of unified view; at which point, you could download the workbook shown below and swap out the data sources using the replace data source feature.

* Enable the Custom Administrative Views feature of Tableau Server.
* Learn about and enable Windows Performance Monitor. Tableau has a great [KB article](http://kb.tableausoftware.com/articles/knowledgebase/monitoring-tableau-server-performance) to read. The companion workbook “TableauPerfmonBook.twbx” uses 5-second intervals.
* Take note that Tableau Server uses GMT Time. The companion workbook uses a Tableau calculation called "Tableau Timestamp" to offset all relevant dates by 8 hours to pacific time zone in order to match the Windows performance monitor timestamps. You will need to change this to your particular time zone.
* The CSV output of your Windows Performance Monitor data collector will have really ugly field header names like "\\machineName\processName\PrivateBytes" - we changed these inside the workbook shown below - you will need to ensure that you can correctly swap the field names out with your own data.
* Also note that the companion workbook comes from a small, lightly used system. We extracted all the data sources and inside the extract We filtered to "today only" which was Feb 6th, 2014. Thus, you will want to disable extracts, or re-extract accordingly.

You will be hard-pressed to come up with repeatable statistics e.g.

* "Total private Vizql RAM divided by number of unique users logged in" - this fails because you cannot predict how long a particular user sits on a viz, or, what the viz looks like, or, what the data source looks like, or
* "Total RAM divided by HTTP requests" - this fails because there are lots of http requests for a given viz, or
* "Total RAM divided by distinct users" - this fails because not all users are doing the same thing at the same time, or

Instead of trying to do any of that, we recommend simple "immersion analysis". Immersion analysis is covered from several different angles in Dick Hauer's seminal work [Psychology of Intelligence Analysis](https://www.cia.gov/library/center-for-the-study-of-intelligence/csi-publications/books-and-monographs/psychology-of-intelligence-analysis/). The basic premise of immersion analysis - specific to the Tableau Server platform and performance analysis - is that you should not be looking for a discrete and conclusive answer to the question of performance, e.g. "this thing times that thing must equal this other thing". Instead you should be using the available data to gather hypotheses with which to perform further research. Specifically, you are looking for:

* Time Patterns: is there a spike or peak at a recurring or periodic time? (this is achieved with any of the data sources shown below)
* Content Patterns: is there a particularly painful workbook or dashboard? (this is achieved by cross referencing RAM spikes against the audit tables. You will want to see which workbooks or dashboards were being looked at when the RAM spiked)
* Task Patterns: Same as above, but this time you are cross-referencing background task details against RAM spikes (e.g. is a particular "refresh extract" pinning down Tableau Server?)

It might be considered a fruitless task to come up with known-good and reproducable performance benchmarks with the Tableau platform. Instead, you should gather information, come up with a hypothesis (e.g. a likely root cause for a performance spike), and then continue researching that hypothesis.

**Appendix A: Overview of processes/components in Tableau Server**

Apache (httpd.exe)

* Apache serves as a load balancer for Tableau Server, directing requests to the right processes.
* With 8.1 external load balancers are supported, but the apache component is not removable or replaceable in Tableau Server.
* Routes traffic to the appropriate worker machines or services.
* Referred to as
	+ "Web Server"
	+ "Load Balancer"

Web Application Server (wgserver.exe)

* Application Server creates the main interface for Tableau Server.
* Handles Several tasks in Tableau Server:
* Tableau Server Log-ins
* Content Searches
* User/Group/Permissions Management
* Other tasks not related to the visualization of data
* Also known as:
	+ "Workgroup Server"
	+ "Application Server"
	+ "WG Server"

Vizql Server (vizqlserver.exe)

* Vizql Server Renders Visualizations for Tableau Server.
* Same functionality as Tableau Desktop, the actual vizql.dll files.
* When a view is requested, the VizQL Server:
* Loads the workbook from the Repository
* Connects to the appropriate data source
* Pulls only the data needed to generate the visualization
* Sends the completed visualization to the client.
* Includes built in caching
* If another user requests information from the same data source, the information is retrieved from the cache first.
* Also known as:
	+ VizQL Engine

Repository/PostgreSQL (pgsql.exe)

* The Repository is a PostgreSQL database used to store the Tableau Server metadata such as:
	+ Users
	+ Groups
	+ Permissions
	+ Workbooks
	+ Connections
	+ Background Jobs
* Stores the files used to display file-based views (.csv, Excel, Access)
* Does not store extracts
* Extracts are stores in the file system or left in the original database.
* Meta-data in the repository can be accessed (read-only) for reporting purposes using the ‘tableau’ user.

Backgrounder (backgrounder.exe)

* Backgrounder refreshes data, sends subscriptions and does lots of background tasks, including cleaning up things on the server.
* Example tasks performed by Backgrounder include:
* Extract Refreshes
* Sending subscriptions
* Removing temporary extract files
* When using the Fast Data Engine, backgrounder handles data refreshes.

Tableau Data Engine (tdeserver64.exe)

* Tableau Server’s Data Engine Process, is used to read TDEs which are extracts (snapshot) of a database.
* TDEs are specially designed data sources intended for reading. They are arranged into columns not rows.
* Extracts are stored under the dataengine folder.
* Also Called:
	+ "Fast Data Engine"
	+ "extract host."

Data Server (dataserver.exe)

* Data Server acts as a proxy between Tableau Server and data sources such that users can centrally manage and store Tableau Server data sources.
* Data Server is only used for Published Data Sources.
* Data Server does not store any data, and should not be confused with Tableau Data Engine.
* Data Server Allows centralized metadata management of the following:
	+ Field names, calculations, hierarchies, sets, groups, table calcs, et cetera
	+ Database drivers.
* Allows multiple workbooks to use the same data base.
* When connecting to a Published Data Source from Tableau Desktop, no drivers are needed.
* [Additional information for Data Server](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#datasource_about.htm)

Other components will appear in Windows Task Manager and are required for some or all operations. These components will help with determining more complex cases.

SOLR (tabrepo.exe) (Search Index)

* Used as a text search index for Tableau Server.
* Examples of searchable content:
* Users
* Groups
* Projects
* Data Sources
* Workbooks
* Views
* Tags
* Comments
* Rebuilding the Search Index can help with odd permissions issues and behavior which relies on searching.

FLEXNet (fnplicensingservice.exe)

* Licensing component for Tableau Server.
* Lmgrd.exe is the Flexnet Licensing manager and loads the licensing information from Trusted Storage.
* Tableau.exe and lmgrd.exe work together to confirm licensing.
* tabsrvlic.log logs information from the Manage Product Keys dialog.
* tablicsrv.log tracks licensing when Tableau Server starts and checks licenses.
* tabsvc (tabsvc.exe)
* Overarching component for Tableau Server
* Starts all other processes.
* Location: C:\Program Files (x86)\Tableau\Tableau Server\8.0\bin

tabadmin (tabadmin.exe)

* Command line utility for administrating Tableau Server.
* Used to start, stop, and set variables for Tableau Server.
* Used to execute commands that cannot be performed via the web interface.
* Location: C:\Program Files (x86)\Tableau\Tableau Server\8.0\bin
* Also see the [online help for tabadmin](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#reconfig_tabadmin.htm)

tabcmd (tabcmd.exe)

* Utility for users who prefer command line over GUI
* Available under the bin directory
* Can be installed on any machine.
* Used to script tasks:
	+ Extract Refreshes
	+ AD group syncing
	+ Export .PDF or .PNG
	+ Publishing
* Tasks performed using tabcmd can also be performed using the Web interface.
* Installer location: C:\Program Files (x86)\Tableau\Tableau Server\8.1\extras
* Location: C:\Program Files (x86)\Tableau\Tableau Server\8.0\bin
* Also see [the online help for tabcmd](http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#tabcmd.htm)

tabconfig (tabconfig.exe)

* "Configure Tableau Server" GUI that can be found within Program Files.

tabspawn.exe

* Wraps most services, starts and stops services as needed.

tabspawnde.exe

* Special start up code for High Availability Data Engine services

tabspawnpg.exe

* Special start up code for High Availability PostgreSQL services

**Appendix B: Configuring Apache as an external load balancer (ELB)**

1. Load balancing using Apache 2.4.4 (this is round robin, giving equal balance between all machines)
2. Get a recent version of Apache and install or place it in C:\ of the computer you want to use as the load balancer. This can be a VM or a physical machine. We recommend the latest version of Apache, as of this writing that is 2.4.9.
3. For these instructions, we assume an install path of c:\Apache, with directories such as c:\apache\conf, c:\apache\logs, et cetera.
4. With a text editor, find “ServerRoot” in httpd.conf and make sure it is set to "C:\apache"
5. cd into c:\apache\bin
6. Go to C:\Apache\conf and open up “httpd.conf” with your favorite text editor
7. Find “DocumentRoot” and make sure it is set to: DocumentRoot "c:\apache\" <Directory "C:/apache/">
8. Uncomment out (delete the # from the line) the following modules

LoadModule proxy\_module modules/mod\_proxy.so

LoadModule proxy\_balancer\_module modules/mod\_proxy\_balancer.so

LoadModule lbmethod\_byrequests\_module modules/mod\_lbmethod\_byrequests.so

LoadModule slotmem\_shm\_module modules/mod\_slotmem\_shm.so

LoadModule proxy\_http\_module modules/mod\_proxy\_http.so

LoadModule headers\_module modules/mod\_headers.so

LoadModule proxy\_connect\_module modules/mod\_proxy\_connect.so

1. Add this code somewhere below all the load modules (change the “balancermember ipaddress” to the addresses of your machines). You can have as many balancerMembers as you want.

# Ensure that encoded slashes are excepted, as Tableau Server uses them.

AllowEncodedSlashes On

# Timeout setting to prevent long-running sessions from blocking new connections

TimeOut 1800

#Set a cookie to ensure session stickiness for the authentication work.

Header add Set-Cookie "ROUTEID=.%{BALANCER\_WORKER\_ROUTE}e; path=/" env=BALANCER\_ROUTE\_CHANGED

#Configure the load balancer proxy, 'loadfactor' determines the weight in the round robin, 'route' is an id for session stickiness used in auth

<Proxy balancer://tableau-server>

BalancerMember http://10.17.136.103:80 loadfactor=1 route=1

BalancerMember http://10.17.136.104:80 loadfactor=1 route=2

BalancerMember http://10.17.136.105:80 loadfactor=1 route=3

ProxySet lbmethod=byrequests

</Proxy>

#Proxy passthrough for the root server context. Forwards addresses to the balancer cluster defined above

ProxyPass / balancer://tableau-server/

ProxyPassReverse / balancer://tableau-server/

# A secondary passthrough to account for wgserver returning the LB’s DNS name in the authentication 302

ProxyPass /ntlm/auth balancer://tableau-server/ntlm/auth stickysession=ROUTEID

ProxyPassReverse /ntlm/auth balancer://tableau-server/ntlm/auth

ProxyPass /balancer-manager !

1. Save the changes you made
2. Open up a cmd prompt as an administrator
3. Cd into c:\apache\bin
4. Optionally install apache as a service, by typing “httpd.exe -k install” then hit enter. Close cmd prompt. You can also manually run apache each time with just “httpd.exe”
5. Optionally go to C:\Apache\bin and double click “ApacheMonitor”. This gives you a convenient way to start and stop apache in the system tray, and also to see if it is running.
6. Start apache or restart it if it was already running
7. Follow the instructions on <http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#distrib_lb.htm> (Configure Tableau Server to Work with a Load Balancer).
8. The basic steps in the online help are:

tabadmin set gateway.public.host "name"

tabadmin set gateway.public.port "port\_number"

tabadmin set gateway.trusted "IPv4\_address"

tabadmin set gateway.trusted\_hosts "name1, name2, name3"

tabadmin config

tabadmin restart

1. you are now ready to use the load balancer. Put the URL of the load balancer into a browser and you will be taken to Tableau Server.

**Configure for SSL termination at load balancer**

This is additional configuration to cover the case when traffic from the client (browser, Tableau Desktop, etc..) to Tableau Server goes over HTTPS, and then over plain HTTP from load balancer to Tableau Server.

1. Create SSL certificates (see [Creating SSL Certs for Server](http://kb.tableausoftware.com/articles/knowledgebase/creating-ssl-certificate-and-key-tableau-server) for details)
2. Using a text editor, uncomment out the "mod\_ssl" module in httpd.conf

LoadModule ssl\_module modules/mod\_ssl.so

1. Find "Listen 80", and add this after:

Listen 443

1. Add the following code after "</Proxy>" but before the first "ProxyPass" entries:

httpd.conf

SSLEngine on

SSLProxyEngine On

SSLCertificateFile "<drive:>/path/to/my.crt"

SSLCertificateKeyFile "<drive:>/path/to/my.key"

RequestHeader set X\_FORWARDED\_PROTO "https"

1. Start Apache and connect with "https://" in front of the machine name.
2. If you don't want to type "https://" in the browser each time, you can setup automatic redirect from default port (80) to HTTPS port. Here's what you need to do:
3. Uncomment out "mod\_rewrite" module in httpd.conf

LoadModule rewrite\_module modules/mod\_rewrite.so

1. In httpd.conf, add "VirtualHost" for your server on the default port with the following settings:

<VirtualHost \*:80>

RewriteEngine On

RewriteCond %{HTTPS} off

RewriteRule (.\*) https://%{HTTP\_HOST}%{REQUEST\_URI}

</VirtualHost>

1. In httpd.conf, wrap the SSL configuration into its own "VirtualHost":

<VirtualHost \*:443>

SSLEngine on

...

</VirtualHost>

Now all your HTTP requests will be redirected to HTTPS version of the server. Be careful when testing with this redirect on, as it might "fix" incorrectly generated URLs which miss the protocol part.

**Appendix C: Additional Information on SSL Certificates for Tableau Instances**

**Should SSL Certificate be on External Load Balancer or Tableau Server?**

At the end of the day there isn't much difference in functionality whether the SSL Certificate is on the External Load Balancer or on the Tableau Machine. Generally, it is prefered that the SSL Certificate be on the External Load Balancer for a few reasons.

There is a small amount of added overhead with SSL, extra CPU cycles are needed to encrypt/decrypt traffic. It is best to offload that processing to the External Load Balancer or Proxy Server or somewhere outside of the Tableau Server so as to free up the request processing at the Tableau Level, removing the encrypt/decrypt step. It is a pretty common industry practice for the SSL Certification for an application to be offloaded to the External Load Balancer level.

If in the future an organization wants to allow external non VPN access to reports through a Proxy Server then it would be preferable to have the SSL Certificate on the External Load Balancer. That way traffic over the internet from the user browser to the inside of the corporate network will all be encrypted. If the SSL Certificate is on Tableau then only the traffic from the External Load Balancer to Tableau Server is encrypted

If the SSL Certificate is on the Tableau Server then the URL would change be http. Clients would hit the non secure URL, External Load Balancer would then connect to the Secure Tableau server and utilize https between the External Load Balancer and Tableau Server.

If the SSL Certificate is at the Tableau Server level then it is also required that each Gateway worker would have the SSL Certificate installed. The “Configure SSL for a Cluster” section of Online help details this further. <http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#ssl_config.htm>

**Prevent bypassing of an external load balancer on Tableau Server**

As Tableau Server does not prevent direct access to gateways via IP address after an External Load Balancer has been added in a distributed installation, how do we prevent client browsers from bypassing the external load balancer?

The answer is to implement a firewall between Tableau Server and the external load balancer as follows:

* Put Tableau Server behind the firewall.
* Put the external load balancer in front of the firewall.
* Allow access to the Tableau Server machines only from the external load balancer.

**Appendix D: ELB with Netscaler**

We recently implemented a NetScaler load balancer in two different 16 core environments. We did SSL Termination and had to work through the correct value of the X\_FORWARD\_PROTO proprtey.

This is from the NetScaler Engineering team:

Create Rewrite Action:

add rewrite action Insert\_X\_FORWARDED\_PROTO insert\_http\_header X\_FORWARDED\_PROTO "\"https\""

Create Rewrite Policy:

add rewrite policy CheckForX\_FORWARDED\_PROTO "HTTP.REQ.HEADER(\"X\_FORWARDED\_PROTO\").CONTAINS(\"https\").NOT" Insert\_X\_FORWARDED\_PROTO

Bind Rewrite Policy to Vserver:

bind lb vserver VSRV-tableau -policyName CheckForX\_FORWARDED\_PROTO -priority 100 -gotoPriorityExpression END -type REQ

General help for Load Balancer: http://onlinehelp.tableausoftware.com/current/server/en-us/help.htm#distrib\_lb.htm

Issues with Netscaler & Tableau Implementation (SSL Offload Specifically)

http://kb.tableausoftware.com/articles/issue/https-redirected-to-http-reverse-proxy

http://kb.tableausoftware.com/articles/issue/cannot-connect-via-ssl-proxy

X\_FORWARDED\_PROTO

Had to add ELB ip address to gateway.trusted

Redirecting from HTTPS to HTTP Causing 304

Need to set X\_FORWARDED \_PROTO to include 'https' to keep the protocol

Policies > ReWrite Rules to X\_FORWARDED \_PROTO to include 'https'

Doco Coming from Fareed:

As promised here are the configuration lines which allows Netscalers to be setup with “X\_Forwarded\_Proto == https”

Create Rewrite Action:

add rewrite action Insert\_X\_FORWARDED\_PROTO insert\_http\_header X\_FORWARDED\_PROTO "\"https\""

Create Rewrite Policy:

add rewrite policy CheckForX\_FORWARDED\_PROTO "HTTP.REQ.HEADER(\"X\_FORWARDED\_PROTO\").CONTAINS(\"https\").NOT" Insert\_X\_FORWARDED\_PROTO

Bind Rewrite Policy to Vserver:

bind lb vserver VSRV-tableau -policyName CheckForX\_FORWARDED\_PROTO -priority 100 -gotoPriorityExpression END -type REQ

http://blog.b3rg.nl/netscaler/insert-client-ip-x-forwarded-for-header-by-using-rewrite-feature/

NetScaler Insert Client IP X-Forwarded-For header with Rewrite featureMatthijs' Blog

Insert the Client IP via a X-Forwarded-For HTTP header into traffic send to back-end web servers by using the NetScaler Rewrite feature.

Read more...

http://discussions.citrix.com/topic/315307-http-x-forwarded-proto/

HTTP\_X\_FORWARDED\_PROTO - NetScaler VPX - Discussions

HTTP\_X\_FORWARDED\_PROTO - posted in NetScaler VPX: Hi All,Is it possible to add an HTTP\_X\_FORWARDED\_PROTO Header on Netscaler?So I can Implement the code below?My hosting provider refuses to change the headers saying it could cause a security issueUsing a Reverse ProxyIf WordPress is hosted behind a reverse proxy that provides SSL, but is hosted itself without SSL, these options will initially send any requests into an infinite redirect loop.