TabJolt Installation Guide

TabJolt: A Tableau Server Point and Run Load Testing Tool

What is TabJolt?

TabJolt is a point and run load generator built on top of JMeter that is specifically designed for Tableau Server. It is available as a free download as-is from GitHub.

Unlike traditional load testing tools, TabJolt can drive load against your Tableau Server without you having to develop or maintain scripts. TabJolt is optimized for the Tableau presentation model and can load visualizations and interpret possible interactions during test execution.

TabJolt is a subset of our engineering load testing framework and we are making it available in hopes that it allows you to accelerate production go-live with Tableau Server, and also to help you with planning your specific on-site capacity needs.

Of course, running load tests does not replace understanding the Tableau Server architecture and following best practices for deployment during load testing. Treating Tableau Server as a black box for load testing is not recommended, and will probably yield unexpected results. For information about the Tableau Server architecture, see the Tableau Server Administration Guide.

When to use TabJolt

Here are the key questions TabJolt is designed to help you answer:

1. I want to deploy a brand new Tableau Server. How will Tableau Server scale on my hardware and with my workload?
2. I am moving from Tableau Server 8.x to version 9.0. Given my workbooks and hardware, how will Tableau Server 9.0 scale in my environment?
3. With guidance from Tableau, I want to tune my server configuration to suit my hardware, workbooks and environments. How do I measure and monitor the effects of configuration changes to select the best configuration?
4. I need to complete a load test as part of go-live testing for Tableau Server. How do I accelerate the time it takes to do a full load test?

Download package contents

When you download TabJolt, the download package includes this Installation Guide along with a READ.ME file that you should review. You will also find the following files:

File Name	What It Does
READ.ME	Contains information about the project.
postgresDBSchemaPart1.sql	Creates the performance results repository for use with TabJolt.
postgresDBSchemaPart2.sql	Creates the schema for the performance results for use with TabJolt and necessary data.
Tabjolt Installation Guide	This document.
TabJoltForTableauServer9.zip	Contains all of the necessary binary and configuration files for TabJolt for Tableau Server 9.0. Tabjolt does not work with earlier versions of Tableau Server.

Prerequisites

To install and use TabJolt, you need a Windows machine with a minimum of two cores with 8 GB or more RAM. As a best practice, you should monitor this machine for CPU and memory to ensure that your test runs don't create a bottleneck on the load injector.

You must also have downloaded the complete package as described above.

JAVA

If you don't have a recent, working version of JAVA, you must install the latest version. Get the latest JAVA installation and instructions at this website:
http://java.com/en/download/manual.jsp

Tableau Desktop

If you already have Tableau Desktop 9.0 or later, you can simply use that. If you don't, you can download a free trial version from this website:
http://www.tableau.com/products/desktop

Postgres

TabJolt uses a Postgres database as its performance result store. To ensure that the running your tests doesn't impact your Tableau Server, this should not your Tableau Server Postgres instance. Instead, you should install a standalone version of Postgres on the same machine where you are installing TabJolt.

NOTE: During Postgres installation, make sure that the database superuser password is set to `testresults`

Download and install the latest Postgres from the following website:

http://www.enterprisedb.com/products-services-training/pgdownload#windows

If you are running Postgres over a Remote Desktop Session, the install splash screen might not appear and the installer might not run. If this happens, navigate to the installation file from an elevated command prompt and then run the installer from the command prompt.

In some cases, we found that the easiest way to install Postgres is to use the default install path—changing the defaults might prevent your Postgres installation from completing in some cases. For help on Postgres, you should review community posts on the Postgres forums.

Configure Postgres

To configure Postgres, you must execute the Postgres SQL scripts that you downloaded (described in the preceding table) in the order that is specified in their file name.

To execute the scripts:

1. Open the administration tool PGADMIN III, which is installed as part of the Postgres installation.
2. Select one of the databases to launch the SQL query execution UI from PGADMIN III.

   IMPORTANT: The next step creates a new database.
3. After you open the SQL query execution UI, open and run the **postgresDBSchemaPart1.sql** script.

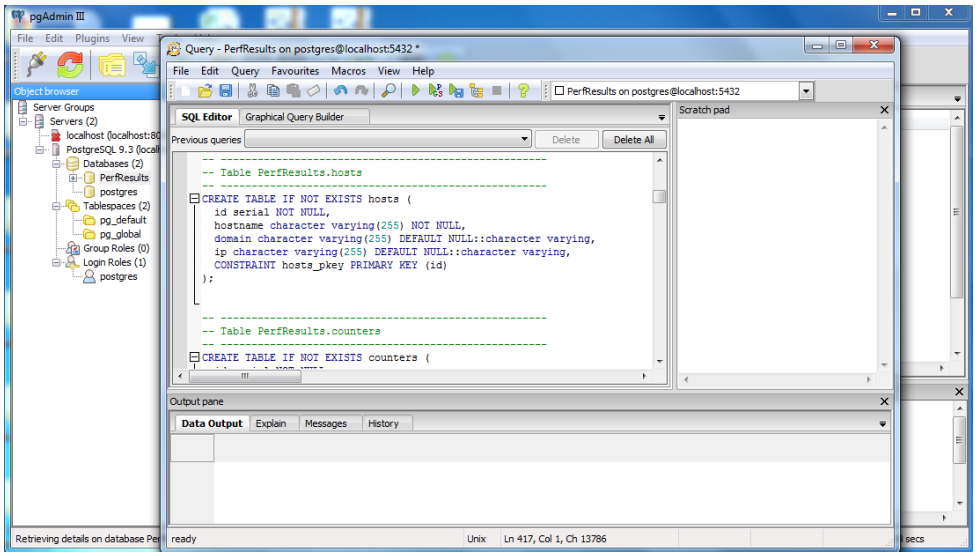

Figure 1: The PGADMIN III console.

Next, connect to the newly created database (PerfResults) and then run the second script, **postgresDBSchemaPart2.sql**, which creates the schema and necessary users to allow TabJolt to run.

Install and configure TabJolt

After the prerequisites are installed and working, unzip TabJoltForTableauServer9.zip to c:\tabjolt. (If your install directory is in a different location, you should factor that in for all the following instructions).

To configure TabJolt, first, edit **c:\tabjolt\config\ServerTestConfig.yaml** and update the following field to point to your Tableau Server URL.

```
hostName: http://yourhost
```

Configure for metrics collection

TabJolt can collect both Windows performance metrics (perfmon) data and JMX data from Tableau Server very easily.

Windows performance metrics

TabJolt includes a set of perfmon metrics out of the box. You can add your own by updating the **c:\tabjolt\config\dataretriever.config** file. In the <hosts> </hosts> section, you can specify the hosts (with the applicable counters) that you want to collect performance counters from.

JMX metrics

JMX metrics give you better visibility into the performance of key server processes under load. To enable JMX data collection, ensure that JMX is enabled on Tableau Server (by default it is disabled) and update your **c:\tabjolt\config\dataretriever.config** file.

To enable collection of JMX metrics on Tableau Server, run the following tabadmin commands:

```
tabadmin set service.jmx_enabled true

tabadmin stop

tabadmin configure

tabadmin start
```

In addition, in **the c:\tabjolt\config\dataretriver.config** file, you must identify and un-comment the following lines in the <hosts> … </hosts> section of the file.

```
<!--enable the following section only after you jmx counter for
tableau-->

<!--

      <applicableCounterGroup>vizqlserver</applicableCounterGroup>
            <applicableCounterGroup>dataserver</applicableCounterGroup>
      <applicableCounterGroup>vizportal</applicableCounterGroup>

-->
```

Choose your workload

Workload on Tableau Server that is user facing is related to loading workbooks and interacting with visualizations within the workbook. In addition, there are background workloads, like extract refreshes and subscriptions. Often these background processes are set up to run in the middle of the night or off peak hours so they don't interfere with user loads.

First, pick the set of workbooks you want to test. To point TabJolt to your workbooks and visualizations, ensure the target workbook is published to Tableau Server.

Populate vizpool.csv

Go to **c:\tabjolt\config\vizpool.csv** and add the link to the target visualization in the vizpool.csv file. The best way to get this link is to manually navigate to it from the browser and copy the URL to the clipboard, and then paste it in the vizpool.csv file.

IMPORTANT: You must remove the "#" in the URL path and add a 'magic number' at the end of the URL. It doesn't matter what the number is, but should be unique across all the numbers in your vizpool.csv.

For example, if your view URL in the browser looks like this:

http://localhost/#/views/WorldIndicators/GDPpercapita?:iid=1

Your vizpool.csv should look like this:

http://localhost/views/WorldIndicators/GDPpercapita/25

As you can see, we removed the "#" and added "25" as our magic number.

IMPORTANT: Before you further scale your load tests, you must ensure that a single user test on this workbook performs within your expectations. If not, you should optimize the workbook by following best practices for workbook authoring.

Choose your load mix

The workload mix that really matters for Tableau Server scale testing is not how many people are logged into the sever, but how many users are concurrently loading and interacting with visualizations.

To that end, we have provided the following two load mixes out of the box. Choose the one that best meets your needs.

- InteractVizLoadTest.jmx
- ViewVizLoadTest.jmx

These load mixes are located in the **testplans** directory under the main install location. They are used by the underlying JMeter execution engine, so you should not edit or change them.

For the interaction mix (InteractVizLoadTest.jmx), TabJolt selects the URLs for the views, which you have provided in vizpool.csv file, based on a uniform distribution. Then it tries to load the view. After the viz is loaded, TabJolt checks whether the viz has any elements that allow interaction with the view (such as a slider bar, drop-down menu, and so on). If the

view has interaction elements, TabJolt performs those interactions without requiring script development. If the view doesn't have any interactions, TabJolt selects marks on view.

The ViewVizLoadTest.jmx, on the other hand, simply loads the visualization without doing any interactions.

You can specify the load mix you want on the command line using the `--t` command line parameter.

The command line parameter `--d` is the test run duration in seconds, and `--c` is the number of virtual users (clients) that you want TabJolt to use.

Manage user login

User login and logout requests are handled by the Application Server component of Tableau Server and do not follow the execution path to VizQLServer, which services end user requests. TabJolt allows you to either leverage the built-in guest account, or specify a synthetic test user during test execution.

To use the built-in guest account, you must have a CORE server license, and you should have configured guest access on your server by checking the **Enable Guest account** option in your administration area on the Tableau Server General Settings page (shown below). You can then use TabJolt to stress or load-test Tableau Server using workbooks or visualizations that you have designed and developed for your users by using the built-in guest account.

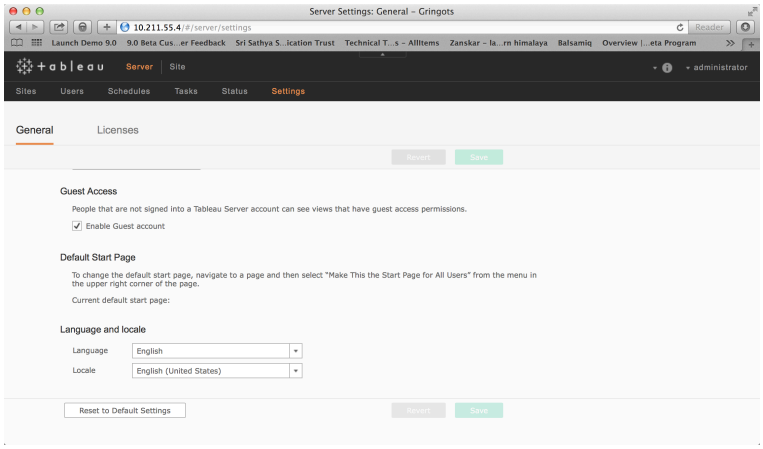

Figure 2: Tableau Server General Settings page, showing guest access.

However, if you want to use a test user to log in to server and then access the visualization, you must set up TabJolt to use a specified test user account. To provide a test user name for login, you need to provide user name and password in **c:\tabjolt\config\ServerTestConfig.yaml** for Tabjolt to use.

```
com.tableausoftware.test.server.configuration.User

name:     < test user name>

password:  <password for test user name>
```

This user can be an active directory user or a local Tableau Server user. The user must be valid and active. If you don't add a user name, TabJolt tries to use the built-in guest account and fails if you don't have your guest access set up correctly.

Run a TabJolt test

Now, you can start your scale test by using a Windows command prompt, navigating to **c:\tabjolt** and running the example command below for a short test. The command tells TabJolt to run the test for 60 seconds (using the `-d` parameter) and to run a single user client (`--c`). You can, of course, change these parameters.

```
go --t=testplans\InteractVizLoadTest.jmx --d=60 --c=1
```

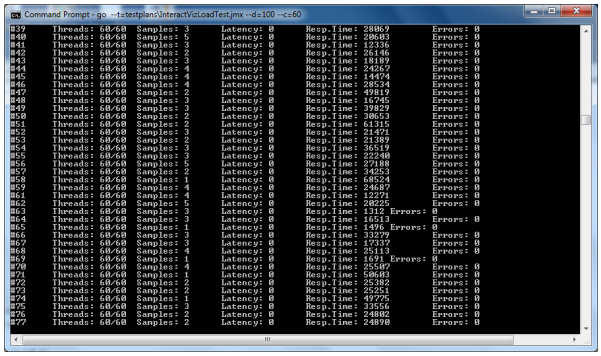

Figure 3: TabJolt running a test.

At the end of the run, on the command line, you will get the run ID, which you need to use as a filter when you analyze your data in Tableau Desktop.

You should explore all the options with the `go` command. If you expect to do a lot of runs, you should give your runs a useful description, by using the command `--r` followed by the description for the run.

Analyze results

After the run is finished, open the analysis workbook located at **c:\tabjolt\PerformanceViz.twb** using Tableau Desktop from the same computer. You

must provide the user name and password for your TabJolt Postgres repository that you installed above (remember, this is not your Tableau Server repository).

You can view the test results and review each of the worksheets.

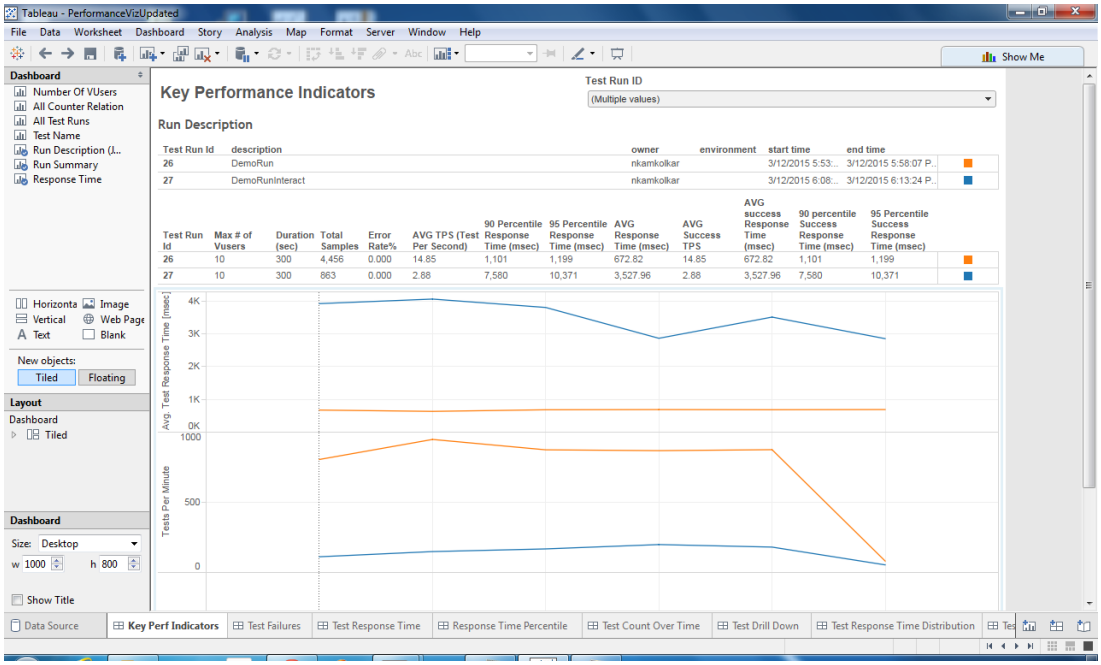

Figure 4: Tableau Desktop showing the results captured from a test run.

Analyzing your load test data is as exploratory as working with any data with Tableau. TabJolt has some key worksheets as starting point. Standard KPI metrics like response times, test cases per second (TPS), host metrics, and JMX metrics (if configured) appear in the workbooks automatically.

A test case is defined as either a “view” or an “interact” test case as described above. These parent test cases might have many child test steps to run. For example, as part of loading a new view, we might create a bootstrap session for the view, get a customized view, or perform operations after the load. These are a subtests of the parent test case:

Test Case (Load View)

- Boot Strap Request
- Get Customized View
- Perform Post Load Operations

When you look at these results, you can quickly find patterns and check how your workload is behaving under load.

Before proceeding with a larger scale test, best practice is to ensure that your workbook is optimized for a single user. If your workbook takes a very long time just for one user, you should either follow best practices on how to author workbooks for performance or request Tableau to help.

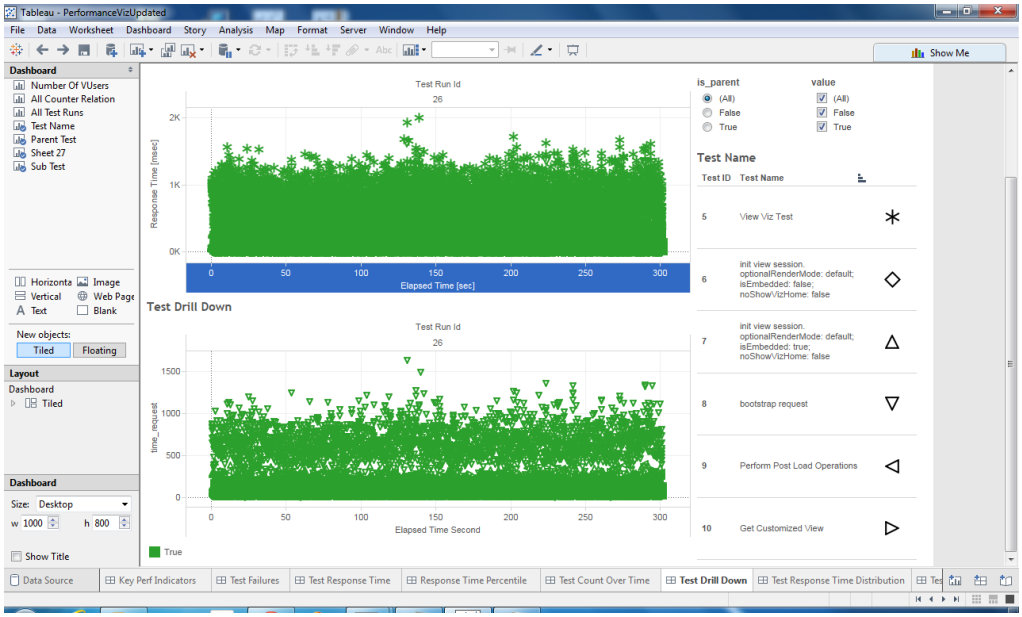

Figure 5: Tableau Desktop showing test drill down from a test run.

Give feedback

If you have questions or comments, please email me at nkamkolkar@tableau.com or contact me on Twitter: @neeleshkamkol.

You can also post questions in the forum for the Server Administration Community on Tableau.com.