Tableau Server Scalability Explained

Author: Neelesh Kamkolkar

Tableau Software

July 2013

Executive Summary

In March 2013, we ran scalability tests to understand the scalability of Tableau 8.0. We wanted to better understand how Tableau Server 8.0 would scale across a variety of configurations and workloads.

There are a number of factors that can impact the scalability of a Tableau Server deployment including workbook complexity, data volumes, hardware, and browser and network settings.

We tried to simulate real-world usage based on what we typically see at customers. We defined a workload of “read-only” users and “interactor” users. Read-only users simply view the report while interactor users perform a selection, filter the view, change tabs and perform similar interactions with the report. Then—under increasing user loads and various workload mix ratios of read-only and interactor users—we studied the system behavior at saturation (maximum throughput).

The results demonstrate that Tableau Server 8.0 scales nearly linearly. Based on our testing and customer usage estimates, we are assuming that the number of concurrent users on the system is 10%. With this in mind, we demonstrated that Tableau Server scales from 980 total users on a single 8 core machine to 3390 total users on a 3 node cluster of 24 cores. This is for a typical workload mix where we see 40% of users interacting with the reports and the other 60% viewing it.

Workers	Max Clients	Total Users
1 Worker	98	980
2 Workers	194	1940
3 Workers	339	3390

Results for moderate workload and typical workload mix

Note: When running in a distributed environment, one physical machine is designated the primary server and the others are designated as worker servers.

We also tested a more active workload. In a scenario in which 100% of users are interacting with the report—again using the concurrency rate of 10%—Tableau Server can support from 890 total users on a single 8 core server up to a total of 2,220 total users on cluster of 3 nodes with 24 cores.

Workers	Concurrent Users	Total Users
1 Worker	89	890
2 Workers	159	1590
3 Workers	222	2220

Results for a moderate workbook with 100% interaction

This white paper explains the scalability tests, methodology and test results.

We will also provide some real world scale examples of Tableau Server, describe Tableau’s approach to performance and scalability, set some baselines to help you understand the various elements of scalability testing, review the results of the experiments, and finally provide guidance on how you could apply these outcomes to your environment.

Scaling from User to Enterprise

At Tableau, we know that data visualization significantly improves the ability to understand information.

We wanted a solution that would improve upon the standard “analyze data in text form and then create visualizations of the findings” process.

So we invented a technology that made visualization part of the analysis journey, rather than a final step. This invention, called VizQL, quickly caught users’ attention. As these users saw how easy it was to create their own data visualizations—and others saw how much value the visualizations provided to the business—enterprise organizations quickly gained interest.

In March 2013, we launched version 8.0 of our software. Many of the enhancements we made were in response to the growing demand for Tableau products capable of supporting large and enterprise-wide deployments.

As more users have discovered the power of visualization, self-service analysis and reporting, IT

finds itself being asked to configure and manage Tableau software and servers to support a larger number of users, groups and interactions.

It's quite natural, then, that CIOs, IT managers and IT architects are deeply interested in the scalability of Tableau Server. They want to be assured that Tableau can support an enterprise deployment. And they want to understand what to expect in terms of performance in order to help guide architecture decisions.

Eating Our Own Dog Food: the Tableau Public Story

As we improved the features of Tableau to support very large groups of users, we needed a way to test and refine these features. We wanted the testing to be as true-to-life as possible, replicating even the toughest business conditions.

As part of the product release and a core part of our engineering culture of using our own products, we rolled the latest Tableau Server pre-release software to Tableau Public. This allowed us to deploy our products at large scale in a production mission critical environment and also to understand, find and fix issues related to scalability before our customers encountered them.

Today, Tableau Server is running at high scale in our own data centers as part of the Tableau Public solution.

For those unfamiliar with the product, Tableau Public is

Figure 1:
Tableau Public gallery

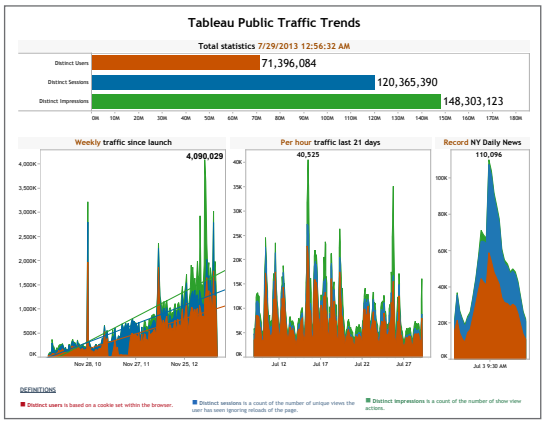

Figure 2:
Tableau Public Traffic Trends

a free service that lets anyone publish interactive data to the web. Once the data is uploaded, anyone can interact with the data, download it, or create their own visualizations with it—no programming skills needed.

Today, Tableau Public supports over 70 million distinct users and over 140 million distinct impressions—and continues to grow at a fast pace. We have seen more than 4 million impressions in a week with a peak of 94,000 views in one hour.

This traffic is powered by Tableau Server leveraging its scale-up and scale-out architecture.

The Tableau Public configuration is similar to a corporate deployment of Tableau Server with a few exceptions:

While the core components of Tableau Public are the same as Tableau Server, Tableau Public users are limited to extracts of 50 MB or less. Tableau Public users also do not face data security issues, as all data is public.

But Tableau Public runs tens of thousands of queries every single day. And while data sizes are relatively small, they have a high degree of variability.

In addition to Tableau Public, Tableau internally deploys and uses Tableau Server across the enterprise to support sales, engineering, support, operations and other key business functions. Using our own products extensively is a core part of Tableau culture.

How We Tested Version 8.0

At Tableau, we see performance and scalability as journey that continues throughout the product lifecycle.

As data sets get larger and analysis gets more complex, performance and scalability become an ongoing, foundational investment.

Traditional Performance Testing

Traditionally, software engineering processes have scheduled testing efforts toward the end of the release cycle. Performance testing is often done toward the end of the release which is too often too little. The pressure to release trumps the priority of fixing performance issues.

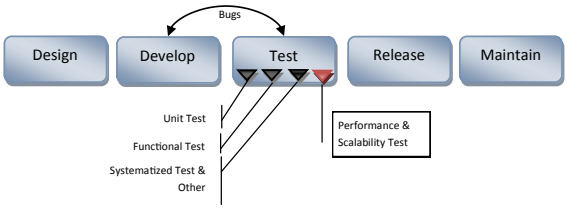

Figure 3:
A traditional approach to performance testing

Tableau Performance Lifecycle

At Tableau, the performance and scalability lifecycle of release starts at design and continues throughout the product lifecycle and across releases.

Tableau is continually enhancing and building performance and scalability into the product to support key scenarios. We measure, monitor and report our performance measurements on a daily basis—and not just for the current release. We also compare test results across current and previous releases.

Tableau's performance engineering practices are backed by a dedicated performance engineering team that is responsible for continually monitoring, managing and reporting.

Typically, we run performance tests nightly across internal workbooks as well as customer's workbooks. The performance engineering team reviews these results each morning and works closely with the engineering team to resolve performance bugs.

At Tableau, performance is a central foundational theme across the entire product lifecycle. We continually invest in improvements to address performance scenarios.

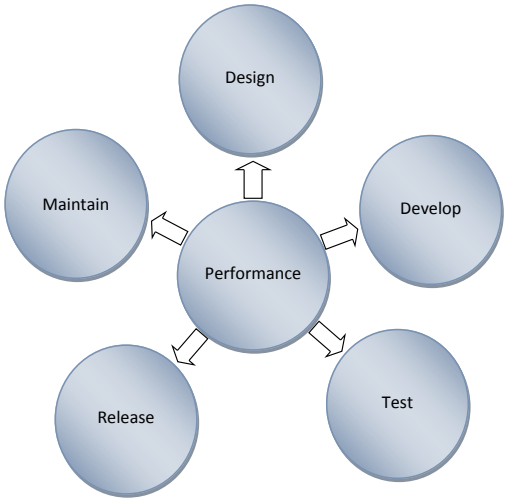

Figure 4:
Tableau Performance Lifecycle

How We Tested Tableau Server Scalability

The goal of these tests was to determine the point at which the system reaches maximum capacity, including the total number of users at that point. This is the point at which no more users can be served until another user's session ends.

Little's Law helps us illustrate this point very well:

Little's Law *Imagine a small coffee shop that has one worker who can only help one customer at a time. People enter, buy coffee, and leave. A basic cup of coffee is served up quickly and more complex drinks take longer. This drives the rate at which they serve people and send them on their way. However, if the number of customers arriving exceeds the number of customers leaving, eventually the coffee shop will fill up and no one else can enter. The coffee shop is maxed out. The variables that determine the maximum number of customers in the shop at any one time are the length of time they spend there, the complexity of their drink order, and the number of workers serving them.*

With scalability testing, there are many variables that can impact system. In order to manage this variability and to drive consistency in comparison of test runs, we standardized on several aspects of the test.

Standardized Environment

We standardized on the hardware on which we run these performance and scalability tests. Our specifications were as follows:

Server Type	Dell PowerEdge R410
Operating System	Windows Server 2008 SP2 64 Bit
CPU	2.66 GHz Dual CPU, Quad Core, with Hyper-threading enabled
Memory	32 GB
Disk Drives	SAS 15K RPM Mirror raid

Figure 5:
Hardware specification of the benchmarking environment

This environment is isolated for exclusive use for performance and scale testing to ensure that external network events don't have an impact on the results.

Deployment Topology

Tableau Server is designed for scaling up and/or scaling out. Depending on your needs, the server provides a flexible scale to meet your enterprise needs.

Scale Up & Scale Out

Tableau is architected to scale up with more CPU cores and memory and/or scale out by adding more servers.

This architecture allows you to maximize the use of compute resources while giving you the ability scale massively. For more information on deployment topologies, best practices and considerations, please refer to the Tableau Server administration guide.

Figure 6 shows the deployment architecture for our scalability testing. The gateway or load balancer simply redirects request and doesn't realistically impact scaling. We scaled the workload using load generators driving different workload mixes.

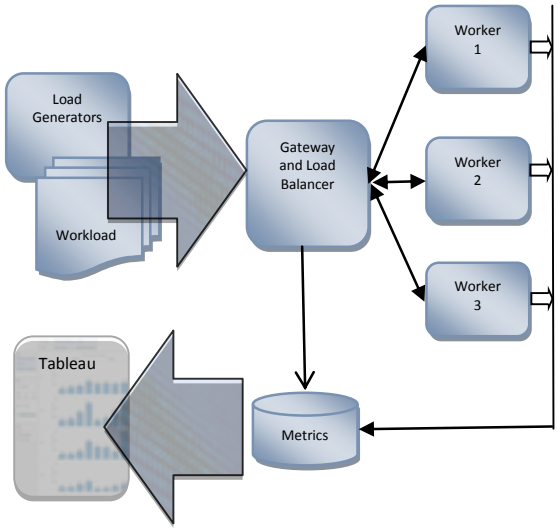

Figure 6:
Scalability testing, deployment

As we saturated each worker we continued to drive more load with additional workers being added to the deployment. During test execution, we collected system and performance metrics and analyzed the results using Tableau

For the scalability testing, we use a homegrown load and scale testing infrastructure that creates workload mixes and generates load with increasing number of users.

Real Workload Characteristics

For our tests, the workload is a combination of

workbooks loading and various user actions being executed against these workbooks.

The workload is characterized by the level of complexity of the workbook. Complexity is defined in terms of the number of zones in a dashboard, the number of dashboards, the number of sheets, the number of marks on visualization, quick filters and other factors.

For the purposes of the scale testing and to reflect real world scenarios, we identified several customer scenarios and workbooks that vary by complexity. We will identify and categorize them as below.

Moderate Workload

The moderate workload includes workbooks that have multiple dashboards with multiple moderately complex visualizations that have many interactions. This type of workbook is what we typically see in customer scenarios.

Complex Workload

The complex workload includes multiple zones with geo maps, large cross tabs, large numbers of marks per chart, over 40+ filters, drill down on maps, and dashboard filtering.

Workload Mix

Oftentimes, load tests are run by repeating the same user action many times. While this is a good stress test case for scalability, it is very rare that an executive (or a real user) will load the same report a million times over.

As noted in the introduction, we included two types of simulated users: read-only and interactors. “Read-only” users are users who primarily just load the report and view it. “Interactors” are users who load the report and perform various interactions with the report such as perform a selection, filter the view, change tabs etc.

In our tests, we simulate several test mixes of workloads across read-only users and interactors. The test mix we ran is described in Figure 8.

Test Case	Read Mix	Interact Mix
1	100%	0
2	80%	20%
3	40%	60%
4	0%	100%

Figure 8:
Test cases and workload mix

Measurement & Reporting

We measure a number for metrics to understand the system performance and scalability including an array of system metrics such as CPU, RAM, Disk/Network IO etc. and performance and scalability metrics such as response times, saturation points, number of users to determine where the server bends or breaks with increasing user load.

To understand the reports discussed in this white paper, let’s quickly review some definitions.

Transaction

A transaction is a set of http(s) requests that constitute a user scenario. For example, a single transaction may include all actions associated with a user logging into the server, browsing for a specific view, loading the view, interacting with the view and logging off. Each one of these events could generate many http(s) requests either synchronously or asynchronously.

For an interactor, a transaction is completed when a user goes through all the interactions across all of the workbooks in the test experiment.

For example: If there are 400 interactions in workbook 1 and 650 interactions in work book 2, one user transaction is completed when the virtual user goes through 400 interactions for workbook1 and 650 interactions with workbook2.

Throughput

Throughput is the number of transactions per second (TPS). In other words, 5 TPS = 432,000 transactions in a 24 hour period.

Max Throughput

Max throughput is the number of transactions per second across all clients hitting the system when the system is in saturation (fully loaded).

Max Clients

A client is an end user driving the transactions. “Max clients” represents the total number of end users across the entire system when the system is in saturation (fully loaded).

In addition to these metrics, we measure many other metrics for performance and use cases such as page load, browser rendering, server rendering, caching, prior Tableau version comparisons, etc.

Now that we’ve seen the test execution framework and deployment and understand the metrics, let’s review the results.

Results & Reporting

As noted earlier, we ran tests across read-only users and interactors across moderate and complex workloads.

First, let’s review the 100% interactors scenario, followed by a more realistic scenario of 40% interactors and then a final scenario where we have 0% interactors. You should keep Little’s Law in mind as you read through this.

Scaling with 100% Interactors

For moderately complex workloads, at 100% interactivity, a single 8-core server with 8GB can scale up to 89 concurrent users. Moving to a conservative 10% concurrency rate—what we see at most customers—a single Tableau Server scales to 890 total users.

We also observed that as we add more workers (scale out), we are able to scale nearly linearly across concurrent users. Figure 9 shows the max clients (concurrent users) scaling with added workers.

Workers	Max Clients	Total Users
1 Worker	89	890
2 Workers	159	1590
3 Workers	222	2220

Figure 9:
Moderate workload, 100% interactors

For more complex workloads (Figure 10) and a 100% interactor workload, we observe that concurrency is slightly lower than in the moderately complex workload scenario. This is expected and adheres to Little’s Law. Again, the important observation is that the systems is scaling nearly linearly even for complex workloads.

Workers	Max Clients	Total Users
1 Worker	72	720
2 Workers	118.7	1187
3 Workers	163.9	1639

Figure 10:
Complex workload, 100% Interactors

Scaling with Typical Workload Mix

Based on our observation of customer environments, we find that a typical scenario includes a mix between read-only users and interactors.

Figure 11 shows the result when considering a mix of 40% interactors and 60% read-only users, based on a moderately complex workload.

We find that a typical workload scales from 980 total users on a single 8 core machine with 32 GB RAM to 3390 total users on a cluster of 3 machines across 24 cores and 96GB of RAM

Workers	Max Clients	Total Users
1 Worker	98	980
2 Workers	194	1940
3 Workers	339	3390

Figure 11:
Moderate workload, realistic workload

Figure 12 shows that same user mix—40% interactors and 60% read-only—for a complex workload. In this

scenario, the server scales from 1150 total users on a single 8 core machine with 32 GB RAM to 2660 users on a 24 core 3 node cluster with 96 GB of RAM.

Workers	Max Clients	Total Users
1 Worker	115	1150
2 Workers	180	1800
3 Workers	266	2660

Figure 12:
Complex workload, realistic workload

Scaling with 100% Readers

Lastly, all things remaining same, if the workload mix is updated to 100% read-only (and 0 interactors) we see scalability across workers represented in Figure 13.

Workers	Max Clients	Total Users
1 Worker	143	1430
2 Workers	177	1770
3 Workers	278	2780

Figure 13:
0% interactors moderate workload

For moderately complex workloads limited to read-only users, a single 8-core serve with 32 GB of RAM can scale up from 1430 total users to 2780 total users on a 24-core server cluster across 3 machines and 96GB of total RAM

The same user mix tested with complex workloads is represented in Figure 14. A single 8-core server with 32

GB of RAM can scale up to 2370 users and up to 2480 for a 3 worker 24-core cluster with 96GB of RAM.

Workers	Max Clients	Total Users
1 Worker	237	2370
2 Workers	218	2180
3 Workers	248	2480

Figure 14:
0% interactors, complex workload

You may be wondering why the 0% interactors/100% read-only users scenario results in much better numbers from the prior workload mixes—and even against prior versions of Tableau Server.

To understand this, you must understand some of the improvements Tableau Server 8 offers related to read-only workloads.

New in Tableau Server 8

Tableau Server 8 offers many improvements to system performance and scalability in common customer scenarios.

In Tableau Server 8, we redesigned the entire VizQL pipeline, keeping performance and scale in mind. The VizQL pipeline is a critical architecture component of Tableau Desktop and Tableau Server.

Secure Shared Sessions

Secure shared sessions are another improvement to scalability for primarily read-only workloads.

In many situations, a majority of the users view or simply read a report and don't necessarily interact with it.

Tableau Server 8 has built in many optimizations for the exceptions in which a user does request interaction with a report. In these cases—as security permissions allow—the data in the session is cloned on demand.

This is a significant difference from Tableau Server 7, in which shareable data was always cloned. We found that cloning by default added more load to the server than necessary.

Cloning on demand allows interactors to work with their own copy of the data while avoiding interference with the other users who are simply reading the original report.

This architecture change has provided two important benefits: Read-only users experience better performance and scale and memory consumption on the server scales on demand.

Figure 16:
Max TPS in versions 7 (blue) and 8 (orange)

Figure 16 shows a comparison of the maximum throughput per second (saturation point) across version 7(blue line) and version 8(orange line).

For a moderate workload in Tableau Server 7 we observed a max TPS of ~4.

In version 8, we observe a max TPS of 26—more than 6x improvement

For complex workloads, we notice the TPS shoot from ~3TPS to 30TPS a 10x improvement.

Browser vs. Server Rendering

In version 8, we introduced browser rendering capabilities. This allows the users' modern browsers to do some "heavy lifting" for the server.

For example, the browser can now perform much of the compute-intensive work for layout.

In version 8, the decision to render using the browser or to render on the server is a smart choice made at run time. The choice is dependent on variables such as the number of marks on the report and other variables.

For example, a high number of marks on a report will cause Tableau Server 8 to default to using server-side rendering.

For desktops—where appropriate graphics acceleration hardware is available—Tableau 8 now uses OpenGL to allow for very fast rendering of large data sets in the desktop.

Additional Improvements

Version 8 has many more improvements across the board. We have reduced the size of extracts by more than 40% in many cases, parallelized work in the VizQL pipeline, optimized queries, made layout improvements added a performance recorder capability for self-service, and more.

Tableau Server 8 scales Nearly Linearly across Workloads, Scenarios and Users

Comparing versions 7 and 8 of Tableau Server makes it clear that we have holistically improved scalability for the scenarios and workloads we used in our testing.

As seen in Figure 17, comparing maximum clients at saturated loads across 1, 2, 3 workers for version 8 versus version 7, demonstrates some significant improvements. Even the worst results for version 8 are on par with version 7.

In the figure below, the orange line shows max clients for Tableau 8 and the blue line shows max clients for Tableau 7. The horizontal axis shows the number of Tableau workers being used and the vertical grouping from bottom to top shows 100% interactors, 40% interactors and 0% interactor workload mixes.

Let's review this bottom to top looking from left to right as we follow first the moderate workload and then the complex.

For 100% interactor workload, the bottom swim lane, we notice that version 8 is improved over version 7 for moderately complex workloads and continues to scale linearly and for complex workloads, it has stayed the same for our experiments.

For the 40% interactor workload, 60% of the users are read-only and 40% are interacting let's focus on the middle swim lane. We notice improvements in version 8 across both moderate and complex workloads due to secure shared sessions and other improvements.

Last but not least, the top swim lane shows the 0% interactor workload (i.e. 100% read-only users) across moderate and complex workloads. In both cases, we observe that there is a significant improvement in scale owing primarily to the advancements in secure shared session technologies.

Scalability Considerations

The most common questions we see on the topic of scalability are centered on how to determine system configuration for a specific number of users or how to optimize an existing system for a given user base size.

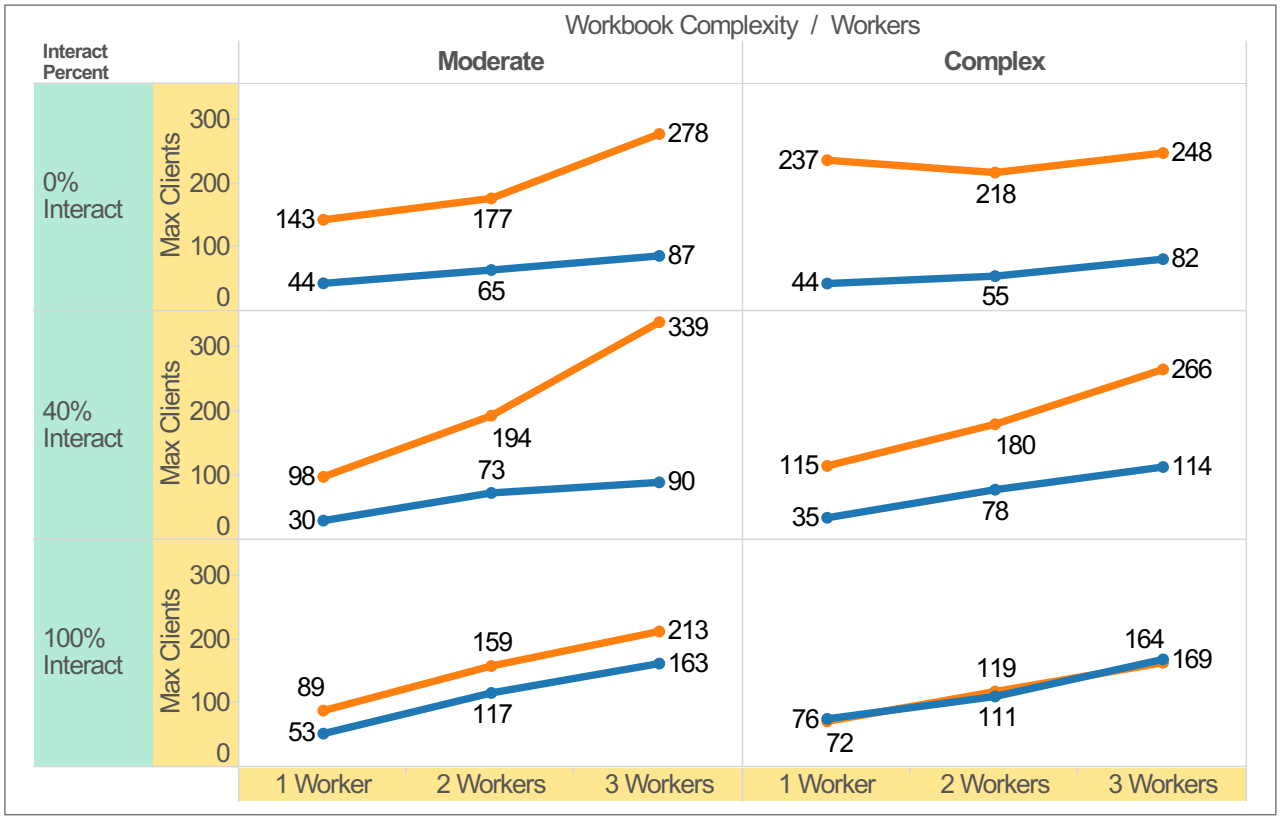

Figure 17:
v7 and v8 Max Client Comparison

Supporting Concurrent Users

Maximum throughput can be used to determine the number of concurrent users a given system can sustain. The number of concurrent users is a function of the workload (or throughput) that each client is executing and the amount of time over which that workload is executed.

Determining system size for a given user base means understanding how those users will use the system (level of complexity of interactions), how long they are likely to stay connected and how time elapses between their interactions.

For initial estimation purposes, feel free to use our test results. As noted before, the conservative concurrency rate is estimated to be 10% of the total number of all licensed users at any one point in time.

Best Practices for Optimization

In addition to a system that is optimally designed, here are best practices that can improve performance and reduce the average response time:

Use Tableau data extracts – If your database queries are slow, consider using extracts to increase query performance. Extracts store data in memory so that users can access the data without making direct requests to the database. Extracts can be easily filtered when users don't need the detail, significantly improving response time. The extract engine can be distributed to a local extract engine on a separate machine to maximize performance if necessary. In version 8, we have improved data extract compression for some types of data.

Schedule updates during off-peak times – Often, data sources are updated in real-time, but the users only need data daily or weekly. Scheduling extracts for off-peak hours can reduce peak-time load on both the database and on Tableau Server.

Avoid 'expensive' operations during peak times – Publishing, especially large files, are two very resource-consuming tasks. While it may be difficult to influence login behavior, it's often easy to influence publishing behavior.

Ask users to publish during off-peak hours, avoiding busy times like Monday mornings.

Cache views – As multiple users begin to access Tableau Server, the response time will initially increase due to the competition for shared resources.

With caching turned on, views from each request coming into the system will be cached. This means that the view will be rendered much more quickly when requested the next time.

In addition to the above, the Tableau Server online administration guide has specific deployment best practices and guidelines that will enable you to get the most out of your deployment.

Final Thoughts

The test results indicate a nearly linearly scalable system that scales out and scales up for varying types of workloads and deployment topologies.

Based on our testing, we were able to demonstrate that for a typical customer scenario with 40% of users interacting and 60% of users reading a moderately complex report, Tableau Server 8 scaled from 980 total users on a single 8 core server to 3390 total users on a 3 node cluster with 24 cores and 96 GB of RAM.

Your organization will likely differ from our test scenarios. Different architecture, workloads, user access patterns for that workload, and other aspects can inform the system scalability and performance. . These tests and the results are meant to be demonstrative of the ways in which Tableau Server can scale for larger enterprises and the expected number of users that a configuration could support.

Your results will vary and you should use these results as guidance for your own deployment.