# A Critical Look at our Infrastructure Journey

Elizabeth Chabot, Cassandra Seney, Rani Fields

## Abstract

When designing a visualization system, it is critical to balance the needs of any project with what is feasible in terms of set-up time, maintenance time, costs, and system performance. In this paper, we will analyze Tableau Server, Shiny, and Tableau Public by the metrics of cost, expected benefits, expected drawbacks, expected performance where applicable, and how a subset of these systems would operate if our sample data were to grow in terms of volume and variety.

## Introduction

When we analyze our visualization ecosystem in terms of visualization software and potential infrastructure solutions, we first need to define which traits we would like in our visualization system. First, our system must facilitate seamless coordination between team members. After all, if we were to work in a situation where multiple individuals are developing the same visualization, we need to have some collaboration mechanism in place. Second, we need some degree of familiarity in our software for when we create our dashboard. That is, we are looking for an easy to use and easy to develop dashboard. For this requirement, we specifically mandate either a Tableau-like interface or R capability. Finally, we require some system which allows us to demonstrate our visualizations to the broader public.

Next, we need to place emphasis on dashboard hosting. As every framework is different, each framework comes with its own unique requirements and caveats which need to be handled. We will be discussing important caveats on a per-product basis.

Ultimately, we aim to holistically analyze each potential solution, explain why we did or did not chose that specific solution or path, and finally discuss how our visualization infrastructure might change in the future as more data and data sources are on boarded.

Please note that we will not explore the larger world of dashboarding frameworks and products in this paper. Rather, we have created an express list of products: Tableau Server, Shiny, and Tableau Desktop/Public.

## Solution: Tableau Server

The first, primary, and by raw requirements, best solution we will be analyzing is the Tableau Server.

### Tableau Server Benefits

For our purposes, Tableau Server fits every functional requirement: it offers a unified way to view and combine workbooks, it uses a fairly familiar interface for every team member, and it provides a way to publically serve our visualizations.

### Tableau Server Negatives

While Tableau Server does offer a perfect fit for our requirements, Tableau Server has several complications which can prevent a proper deployment of the system. Specifically, issues regarding the hosting costs, licensing costs, and configuration can prevent Tableau Server from becoming a viable solution despite the clear at-face benefits it provides. It is worth noting that as of this paper, Tableau Server only operates on Windows 2012 and 2016 Server VMs, which is a requirement that can cause several complications in its own right.

### Tableau Server Expected Costs

If one were to decide that Tableau Server is worth the time investment, it could be useful to refer to the table of costs below:

*Table 1: Windows VM costs, 8/20/2017, all costs are 30-day approximates assuming the VM runs for x hours a day*

| Instance | Risks Associated | Cost (24-hr) | Cost (12-hr) | Cost (8-hr) |
|---|---|---|---|---|
| **t2.xlarge** | • Requires EBS for storage | $164.88 | $82.44 | $54.96 |
| **m4.xlarge** | • Requires EBS for storage<br>• Insufficient physical cores, unknown issues likely | $276.48 | $138.24 | $92.16 |
| **m4.2xlarge** | • Requires EBS for storage<br>• Requires strict active hours management to limit costs | $552.96 | $276.48 | $184.32 |

Ultimately, we opted for the t2.xlarge flavor. Please note that the minimum Tableau Server licensing cost is $125/mo.

### Final Decision: Do not use

Ultimately, issues stemming from the Windows 2012 virtual machine prevented Tableau Server from properly installing. Considering these issues, we were faced with two options: risk a major time sink and continue working on Tableau Server as our primary visualization solution, or assume Tableau Server to be unviable with regard to time investment and continue on to our next data visualization solution. We chose to abandon the Tableau Server option and continue with Shiny.

## Solution: Shiny

Shiny was our second solution after Tableau Server. While Shiny is not as much of a perfect fit as Tableau Server for a variety of reasons (discussed below), a pilot run resulted in Shiny up and

running within 15 minutes on a VM. Furthermore, Shiny appears to have a relatively light lowerbound footprint and does not require multiple threads.

### Shiny Benefits

While Shiny does not fit our requirements as perfectly as Tableau Server, it lacks a UI for generating dashboards after all, it does fit our requirements well enough. Likewise, Shiny offers several benefits on its own merit. It is easier to set up than Tableau Server, and it is cheaper to host than Tableau Server due to its low expected footprint. Finally, while Shiny does not operate on the workbook system used by Tableau, options are available for multiple users to develop different dashboards simultaneously and, since everything is code, the various dashboards can be merged into a final dashboard.

### Shiny Negatives

No dashboard is without its drawbacks. Very notably, Shiny does not offer any UI for creating dashboards. Thus, our ability to perform rapid prototyping for new dashboard components is highly limited. This results in two risks: onboarding new individuals is not completely straight-forward as it requires both R skills as well as basic knowledge of how Shiny works. Furthermore, in a worst-case scenario, the drawback of no UI can result in Shiny turning into a time sink.

### Shiny Expected Costs

Due to the small footprint of Shiny, dedicated hosting is likely not required. If performance issues are discovered, however, migrating specifically the Shiny instance to a new VM would be trivial due to Shiny's seamless set-up experience as well as the nature of how Shiny servers are designed- you simply would need to move your server and UI code to your new target host.

If a dedicated VM is required, in theory only a single vCPU is required. As such, it might be prudent to use a host with 1 vCPU and an unknown but medium to high amount of RAM.

### Final Decision: Do not use

Due to the time associated with creating a Shiny dashboard, the team opted to move on to the final back-up solution, a Tableau Public. For our purposes, we opted for Tableau Public rather than Tableau Desktop as Tableau Public allows the public to view the dashboard.

## Tableau Public: Chosen Solution

The final solution we looked at was Tableau Public. As Tableau Public is a Tableau product, we can leverage a UI very similar to Tableau Desktop. Furthermore, the fact that we have a UI available allows us to perform rapid prototyping as well as updating our UI, a trait which we found helpful.

### Tableau Public Benefits

The primary benefit of Tableau Public is that it offers the strongest benefits of Tableau Server with none of the cost. That is, Tableau Public allows for rapid prototyping via a UI as opposed to working with everything on a programming basis. Furthermore, as Tableau Public is divorced from the hosting equation, we do not need to worry about procuring any VMs other than VMs for hosting the database. Please note that we do not cover database hosting in this paper.

### Tableau Public Negatives

The primary drawback of Tableau Public is simple: it is not easily feasible to have multiple individuals working on the same Tableau Public project.

### Tableau Pubic Expected Costs

As hosting is not a factor for Tableau Public, we do not need to worry about expected costs.

### Final Decision: Use this solution

Ultimately, we opted to work with Tableau Public due to the strong benefits Tableau Public provides, most notably rapid, UI-based prototyping.

## Looking to the Future : As Data Evolves

As our project expands, we need to ensure our visualization infrastructure can handle any changes which might occur in data size as well as data variety. Assuming our project were to continue, we would likely onboard a variety of data sources which would provide us both finer-grained data as well as new data sources. These have been identified as the primary projects for how our data could change, that is, we expect changes in both volume and variety.

As our data grows more nuanced, access to a UI could prove useful to discovering new trends in our data. Due to this, if our data variety increases, Tableau variants pose substantial benefit due to Tableau's high accessibility and ease of creating dashboards.

If our data were to grow in volume, Tableau variants are likely able to bear the brunt up due to how Tableau handles data, especially in-memory data. If we reach a point, however, that the amount of memory Tableau consumes renders Tableau an unusable solution, we could opt to move to other solutions. For instance, we could find it useful to pre-compute some of our data to reduce the memory footprint.

## Conclusion

Ultimately, if we had more time, a Shiny server or, ideally, a Tableau Server solution would have been the most ideal solution. What ultimately killed both the Tableau Server and Shiny solutions in the end were their respective threats of becoming time sinks. If we were to continue down a more permanent route, it could still be prudent to invest in a Tableau Server installation. That said, due to the limited scope of this project, Tableau Public is sufficient.