EPA CA2 – X00129654 – Report

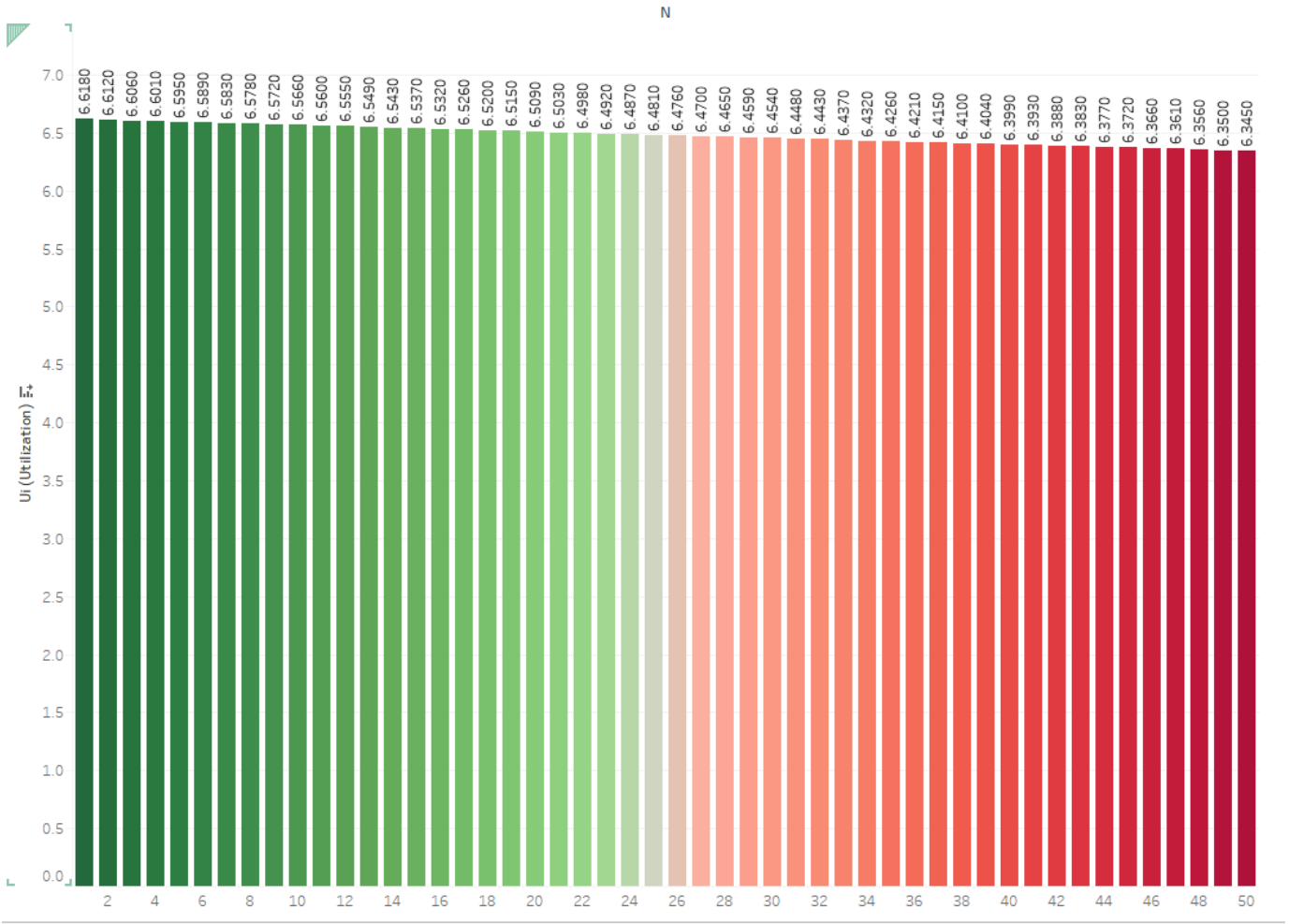
Plot Discussion

After gathering data for the load test bash script and loading the data into Tableau, I created 4 graphs to display my results. The first graph shows the utilization (Ui vs. N) and this was calculated using the following formula:

```
[Idle] / 10
```

This graph illustrates that the utilization of the CPU ranged between 6.6 and 6.3 for each iteration of the load test, and a clear pattern emerged as the value decreased after each iteration, as seen below.

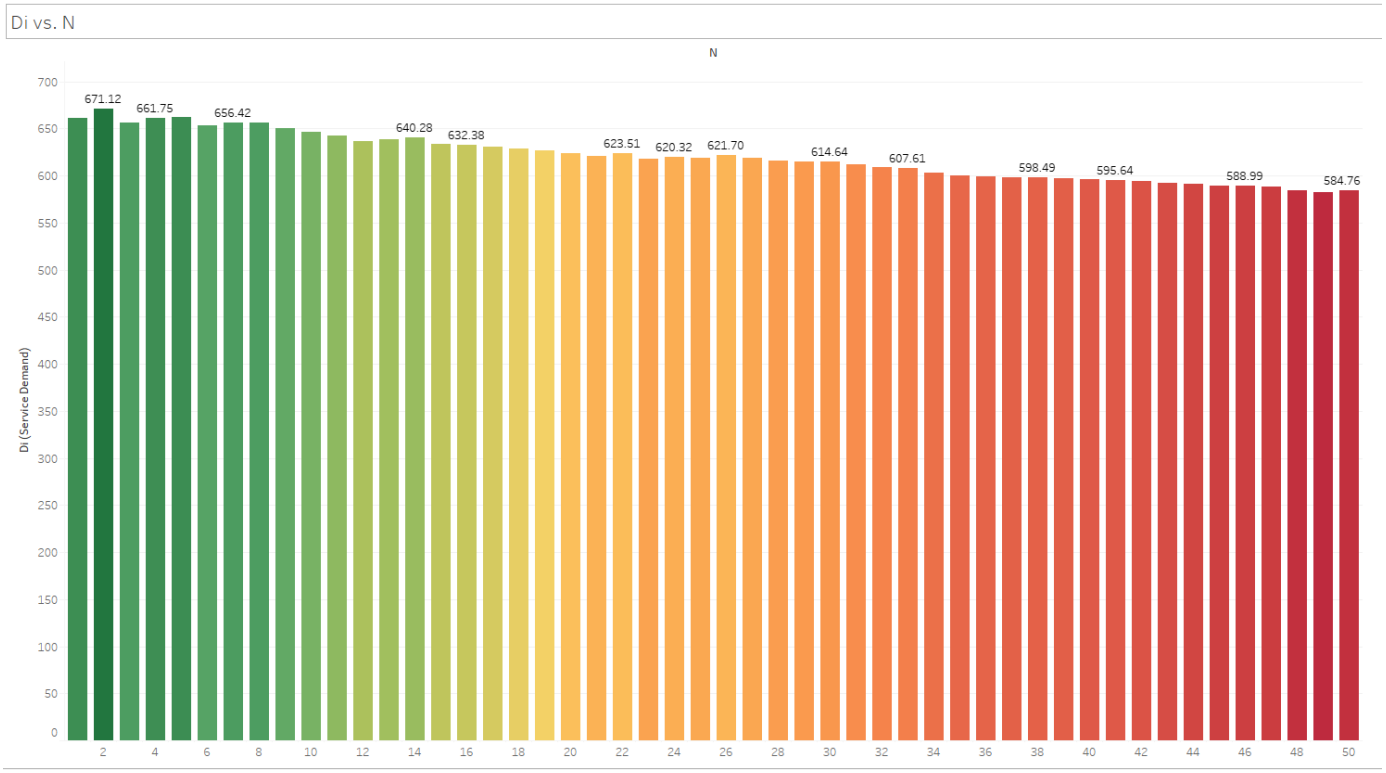
Ui vs. N



The second visualization I created was centered around working out the Service Demand (Di vs. N) and I used the following calculation: Di = Vi x Si = Ui / X0

```
([C0]/ 10) * ([Idle] / [N])
```

This showed a very consistent set of results yet again with another clear real pattern emerging that I could establish. The service demand ranged between 671.12 and 582.90, so the values were not very sparse.

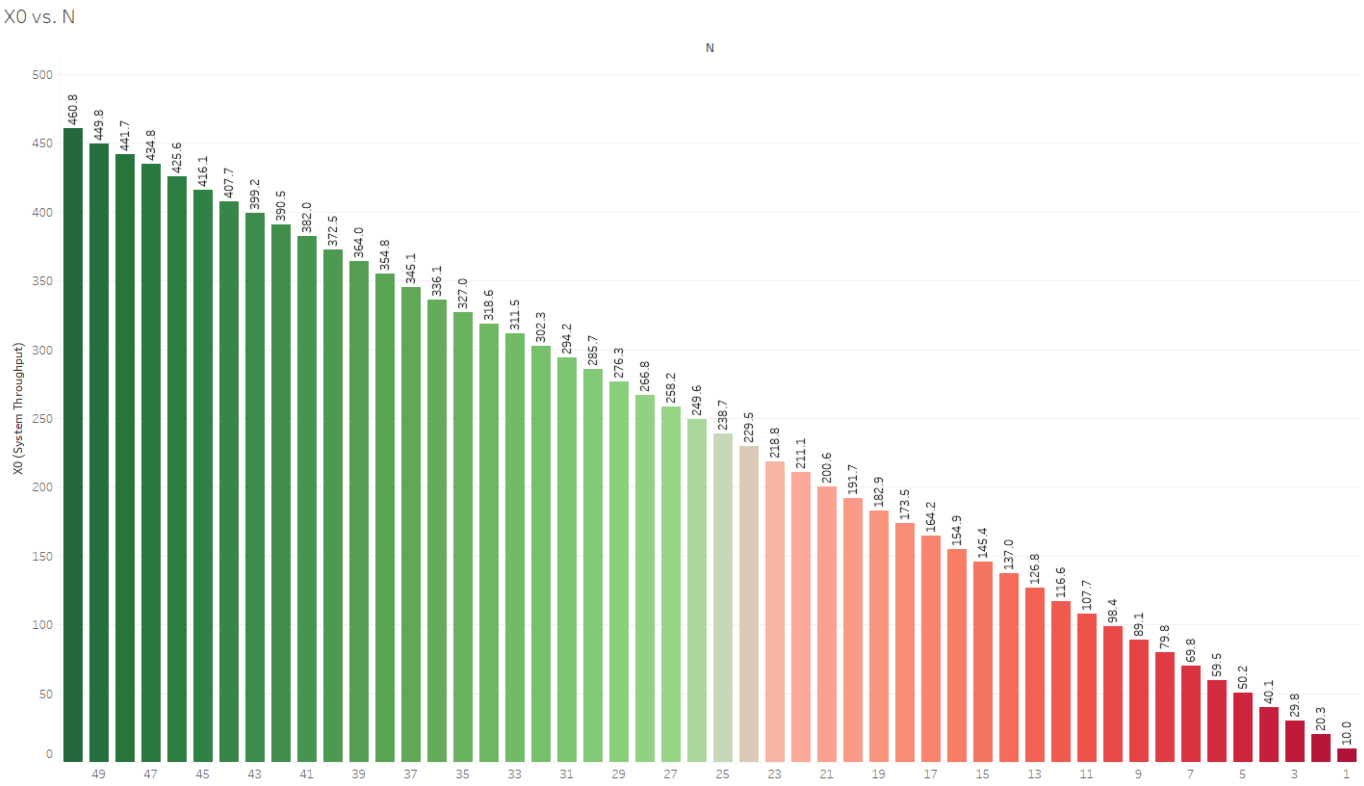


In my opinion, this was caused by the steady decreasing of the value of C0 causing a fixed sequence of values seeing the service gradually decrease with each iteration.

Moving on, the third graph I created is focused on calculating system throughput (X0 vs.N) and was worked out using the following formula: X0 = C0 / T

```
[C0] / 10
```

This graph shows similar findings that we saw with the service demand. This again is due to the consistent decrease of the C0 value as seen below.

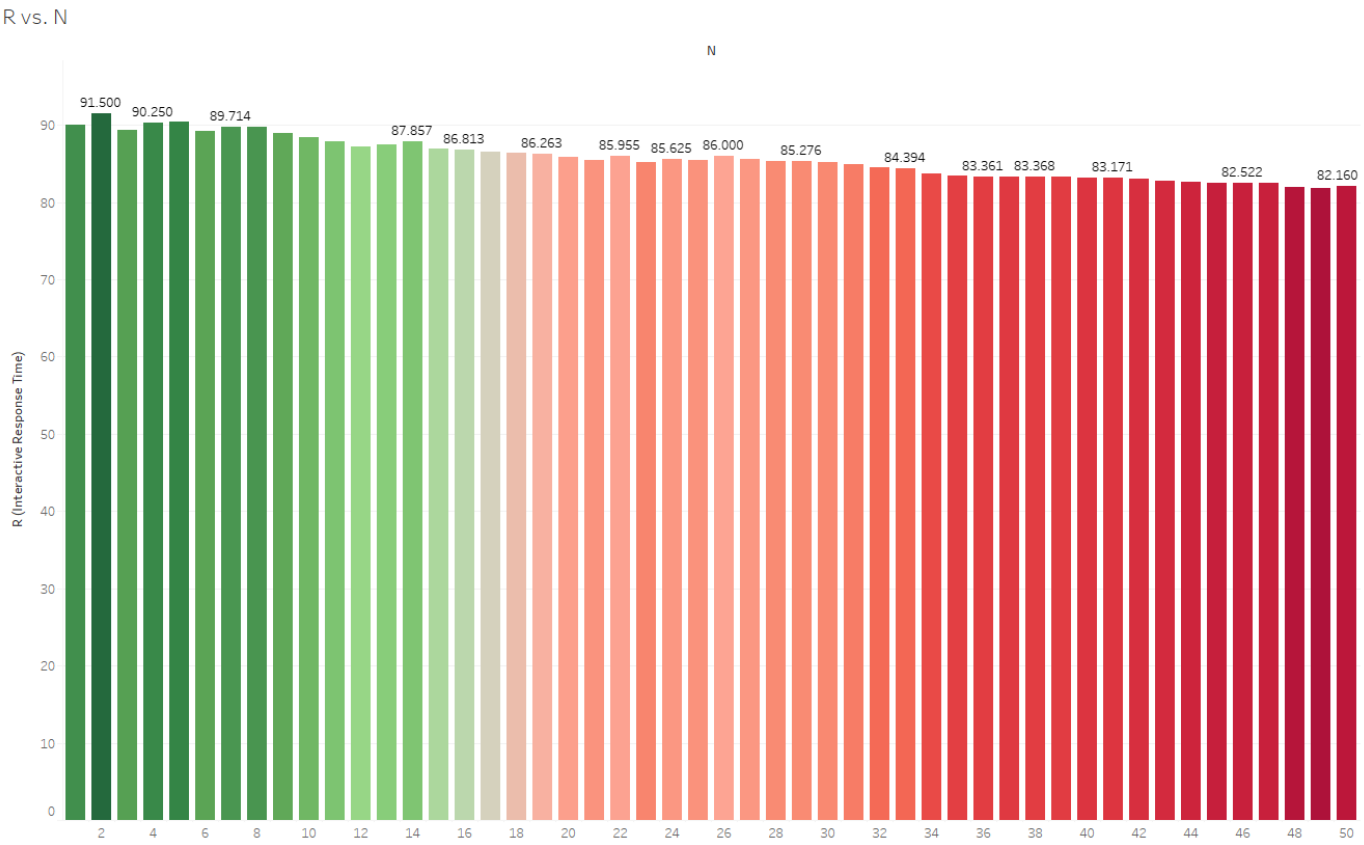


The findings ended up as they did due to the increased number of completions by the system with each iteration, causing higher amounts of throughput as the load test progressed.

The final visualization that I created was based around finding the Interactive Response Time (R) of the script and I used the following formula to work this out: R = (M / X0) – Z

```
([C0] / [N]) - 10
```

This shows that only the Interactive Response Time slowly decreases as the load test progresses. The Interactive Response Time ranges between 82.160 and 91.500 and its findings are consistent with that of the other visualizations we have seen already. The graph can be seen below:



Notes on the Script

When running this script from the command line the user must enter a number between 1 and 50 to run the script successfully. E.g: **./runtest.sh 50**. As the script is running through each iteration of the load test a message is displayed on the command line showing progress of the script. This is purely to keep the user aware of the progress of the load test.

```
[epauser@localhost EPA-CA2]$ ./runtest.sh 50
Iteration 1 of 50 completed.
Iteration 2 of 50 completed.
Iteration 3 of 50 completed.
Iteration 4 of 50 completed.
```

I ran the command **sudo yum install sysstat** to install sysstat on the Fedora VM. The script contains some input validation to ensure that the load test will only begin when a value between 1 and 50 is entered. The script also checks for the existence of both **stats.dat** and **findings.txt** before entering the for loop and will remove these files if they are found to ensure that each load test contains unique values each time it is run. The for loop iterates for as many times as the user has specified when they ran the file. The value **$1** is assigned to a variable called **m** which is used as the bounds for the loop as seen below:

```
# For loop to iterate through the loadtest $m times.
for (( l=1; l<=$m; l++ ))
```

A timeout of 10 seconds is set for each iteration of the loop and **mpstat** data is appended to the file **findings.txt** to be used later the display the value of **idle**. The current word count of **findings.txt** is stored in a variable called **c** which is then used to display the value of **C0** for each column in the script. Finally, the **awk** command is used to print the 12th column of data in the data.txt file which is the value for **idle**. This value is then appended to the results.dat file and a message is displayed stating that the iteration has completed. The value for the column **C0** is calculated by incrementing the **c** variable storing the word count of the **synthetic.dat** file. This allows us to keep track of the total number of completions recorded by the script after each iteration.

When the loop finishes, the contents of results.dat are displayed to the user in the format specified in the CA Notes, as seen below.

```
Contents of stats.dat
C0      N       idle
95      1       63.78
199     2       63.72
306     3       63.67
399     4       63.62
482     5       63.57
582     6       63.52
683     7       63.47
764     8       63.42
857     9       63.37
951     10      63.32
```

Notes on the VM

The bash script was written and run within a Fedora Virtual Machine on the Linux Operating System.

Virtual Machine Specs:

- Name: Fedora 28
- Type: Linux
- Version: Fedora (64-bit)
- Memory: 2505 MB
- CPUs: 2
- Exec Cap: 100%

Storage:

- Type: Normal (VMDK)
- Virtual Size: 17.61 GB
- Actual Size: 9.91 GB
- Details: Dynamically Allocated Storage

Github Repository

If you wish to view all of the code used during this CA it can be found here:
https://github.com/JakeWalsh69/EPA-CA2