Comparison of Server Response Capacities

By Cannon Collins

Abstract:

While there are a number of methods available to evaluate server performance, the focus of this analysis is directed towards quantity of queries as related to server response speed. The number of requests made to the server is hypothesized to be directly correlated to response time, the higher the number of requests the lower the speeds of response.

Introduction:

The point of this experiment was to “stress test” my server. As stated, I believe that the more requests there are, the slower the server will respond. By loading the server’s queue, it should max out response time by having the threads running to their full capacity. The tests will be run on different computers to ensure accurate data and readings, to show that it’s not just one computer’s capacity but that they are consistent on various computers with differing computing power. Having these as differences, we’ll be able to display similar data trends to show that the server is reacting the same. We will be sure to test the same data endpoint in all tests to keep the transfer size the same.

Related Work:

The related work to stress testing a server has been done many times. Many have tested server speeds to see how to improve performance. Apache is the most popular to test and have incredible speeds, however they do have top of the line computing performance and abilities. To compare properly would be like comparing a car that has wheels and an engine to get from point A to point B, versus a luxurious, land speed record setting car. To their core, they’re the same, but everywhere else is completely different.

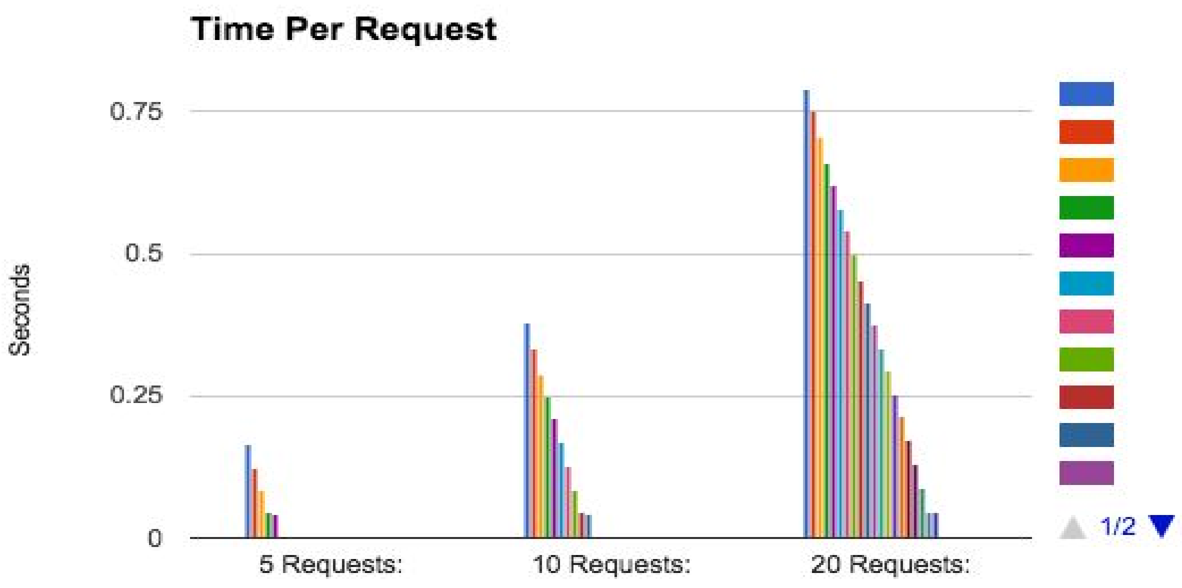
<u>Experimental Setup:</u>

How I've set up the experiment is really simple. The steps are:

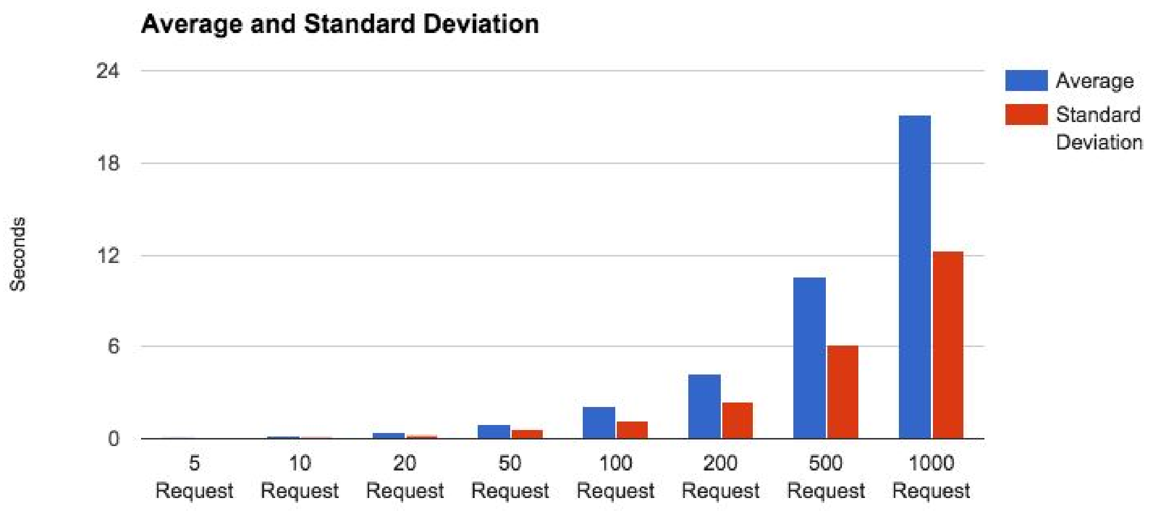
1. Start the server
2. Test the server using Apache Benchmark, testing on 10 requests, and 50 requests.
3. Test the server using your own version of Apache Benchmark that records the times and gives more informative data, such as my Lab 4. Testing on ranges from 5-1000 requests.
4. Compare the data to receive the results.

<u>Results:</u>

<u>My Client:</u>

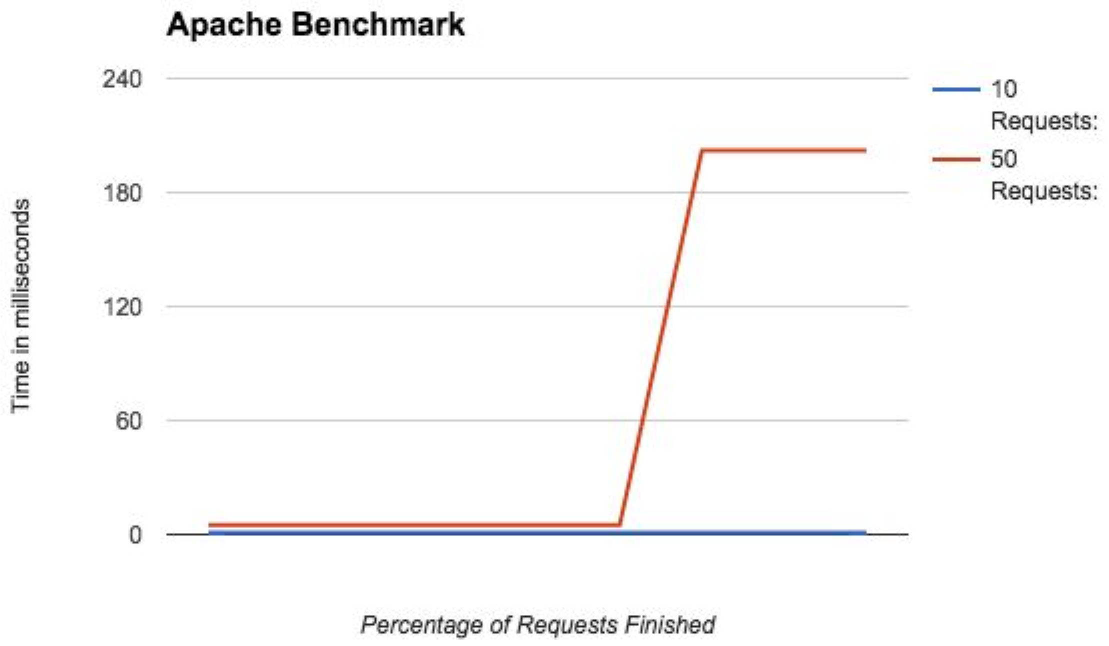


The graph above exhibits results from tests run on my server, which are supportive of my hypothesis. This data displays a trend of upward moving response times in conjunction with increased request loads. For example, when the server was hit with only 5 queries, the highest time per request was under .25 seconds, whereas when 20 queries came through, the time per request shot above .75 seconds before descending back down.

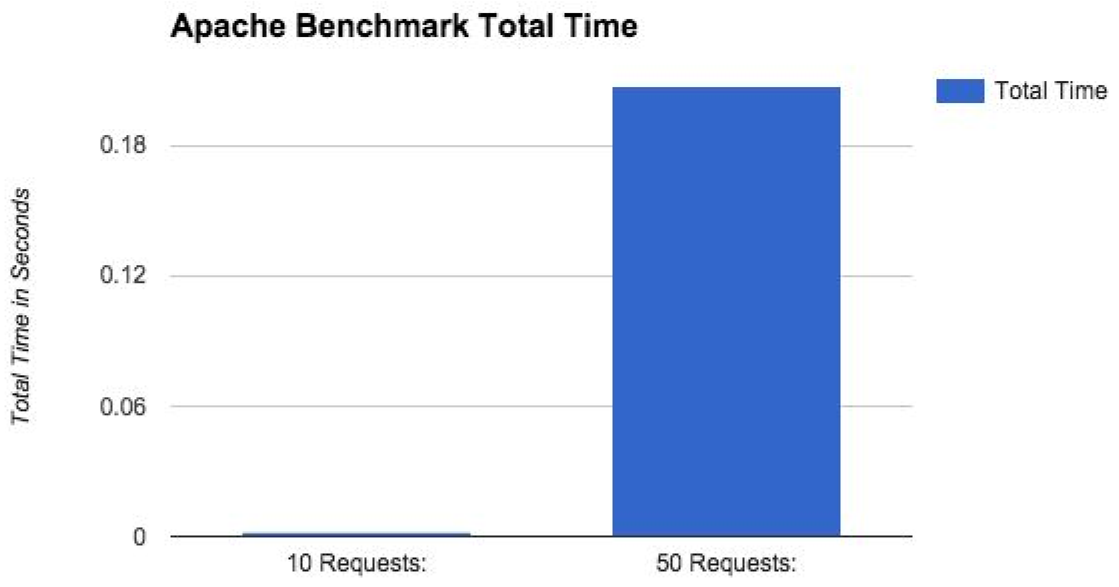


This set of data further displays the trend discussed above, simply extending test query quantities up to 1000 at a time. The average time per request increases with increasing number of requests sent to the server, with standard deviation staying relatively proportionate. Although each set of requests seems to catch up to speed with smaller query loads once numbers are comparable, the larger the initial number of requests sent the higher the response times spike, increasing overall averages accordingly.

Apache Client:



The Apache Benchmark was difficult to test. Apache has so much computing power, the results often differ in just a few milliseconds. However we were able to test in a white box fashion, meaning we knew information about the server. Knowing that my server has a queue of size 20, this means that when I run a test of 10 requests, they should all be able to be in the queue on the server and be taken care of. Whereas when we test it with more stress, 50 requests, the server is able to be close to the same speeds before ultimately slowing down as it nears the last 10% of the the requests.



This graph was made to show more data on the comparison of 10 to 50 requests using Apache Benchmark as the server testing tool. It shows the total time it took for Apache to do all tests. One can see that the 50 requests took much more time than the 10 requests. 50 requests took 0.207 seconds to complete and 10 requests only too .002 seconds to complete, concluding that 50 requests took 100x the time that 10 requests took.

Conclusion:

In conclusion, my hypothesis was correct. A server receiving more requests per second will be slower than one that receives fewer requests. As shown in the recordings, the requests would be almost identical towards the last set of requests, however with more responses the initial requests seemed to lag. Through our testing using my own client for stress testing the server as well as using the popular tool Apache Benchmark from Apache, the global leader in hosting web servers, we were able to see that running up the requests, slowed the response time and proved my hypothesis true.