Load Test Report

Date: 7/27/2016

Test from: virginia

Query URL: http://2016pressablevip.reviewsignal.com/

Started at: Wed Jul 27 2016, 05:01:36 -04:00
Finished at: Wed Jul 27 2016, 05:02:36 -04:00
Test link: https://www.blitz.io/to#/play

Analysis

This rush generated 129,866 successful hits in 60 seconds and we transferred 2.30 GB of data in and out of your app. The average hit rate of 2,164/second translates to about 187,007,040 hits/day.

The average response time was 132 ms.

You've got bigger problems, though: 0.01% of the users during this rush experienced timeouts or errors!

Response Times

- Fastest: 131 ms
- Slowest: 139 ms
- Average: 132 ms

Test Configuration

- Region: virginia
- Duration: 60 seconds
- Load: 1-5000 users
- Avg. Hits: 2,164 /sec
- Transferred: 19.00MB
- Received: 2,340.02MB

Hits

This rush generated 129,866 successful hits. The number of hits includes all the responses listed below. For example, if you only want HTTP 200 OK responses to count as Hits, then you can specify --status 200 in your rush.

<table>
<thead>
<tr>
<th>Code</th>
<th>Type</th>
<th>Description</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>HTTP</td>
<td>OK</td>
<td>129866</td>
</tr>
</tbody>
</table>

Errors

The first error happened at 37.5 seconds into the test when the number of concurrent users was at 3121. Errors are usually caused by resource exhaustion issues, like running out of file descriptors or the connection pool size being too small (for SQL databases).

<table>
<thead>
<tr>
<th>Code</th>
<th>Type</th>
<th>Description</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>TCP</td>
<td>Connection timeout</td>
<td>13</td>
</tr>
</tbody>
</table>

Timeouts

The first timeout happened at 27.5 seconds into the test when the number of concurrent users was at 2286. Looks like you've been rushing with a timeout of 1000 ms. Timeouts tend to increase with concurrency if you have lock contention of sorts. You might want to think about in-memory caching using redis, memcached or varnish to return stale data for a period of time and asynchronously refresh this data.
The max response time was: **138 ms @ 201 users**

The max hit rate was: **4,289 hits per second**