LOAD TEST REPORT

DATE: 7/24/2014

TEST FROM: VIRGINIA

Query URL: http://wp-nexcess.reviewsignal.com/

Started at: Thu Jul 24 2014, 12:44:45 -04:00

Finished at: Thu Jul 24 2014, 12:44:45 -04:00

ANALYSIS

This rush generated 25,556 successful hits in 60 seconds and we transferred 258.42 MB of data in and out of your app. The average hit rate of 426/second translates to about 36,800,640 hits/day.

The average response time was 279 ms.

You've got bigger problems, though: 44.12% of the users during this rush experienced timeouts or errors!

**Response Times**

**Fastest:** 46 ms

**Slowest:** 396 ms

**Average:** 279 ms

**Test Configuration**

Region: VIRGINIA

**Other Stats**

Avg. Hits: 426/Sec

Data Transferred: 258.42 MB

| Hits | 55.88% (25556) |
| Errors | 33.91% (15509) |
| Timeouts | 10.20% (4666) |

**HITS**

This rush generated 25,556 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>25556</td>
</tr>
</tbody>
</table>
HTTP 200 OK 100% (25556)

ERRORS

The first error happened at 15 seconds into the test when the number of concurrent users was at 497. 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>15509</td>
</tr>
</tbody>
</table>

TIMEOUTS

The first timeout happened at 5 seconds into the test when the number of concurrent users was at 163. 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: **395 ms @ 1499 users**

The max hit rate was: **676 hits per second**