# LOAD TEST REPORT

## DATE: 6/24/2014

## TEST FROM : VIRGINIA

**Query URL:** http://69.55.49.190/

**Started at:** Tue Jun 24 2014, 09:04:55 -04:00

**Finished at:** Tue Jun 24 2014, 09:04:55 -04:00

## ANALYSIS

This rush generated **105,194** successful hits in **60 seconds** and we transferred **894.43 MB** of data in and out of your app. The average hit rate of **1,753/second** translates to about **151,479,360** hits/day.

The average response time was **38 ms**.

You've got bigger problems, though: **3.70%** of the users during this **rush** experienced timeouts or errors!

### RESPONSE TIMES

FASTEST: 18 MS

SLOWEST: 302 MS

AVERAGE: 38 MS

### TEST CONFIGURATION

REGION: VIRGINIA

DURATION: 60 SECONDS

LOAD: 1800-2000 USERS

### OTHER STATS

AVG. HITS: 1,753 /SEC

DATA TRANSFERED: 894.43MB

HITS 96.30% (105194)

ERRORS 3.37% (3682)

TIMEOUTS 0.33% (360)

---

## HITS

This rush generated **105,194** 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.

| CODE | TYPE | DESCRIPTION | AMOUNT |
|---|---|---|---|
| 200 | HTTP | OK | 105194 |


HTTP 200 OK **100%** (105194)

---

## ERRORS

The first error happened at **2.5 seconds** into the test when the number of concurrent users was at **1808**. 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).

| CODE | TYPE | DESCRIPTION | AMOUNT |
|---|---|---|---|
| 17 | TCP | Connection reset | 1019 |
| 23 | TCP | Connection timeout | 2663 |


CONNECTION RESET **28%** (1019)

CONNECTION TIMEOUT **72%** (2663)

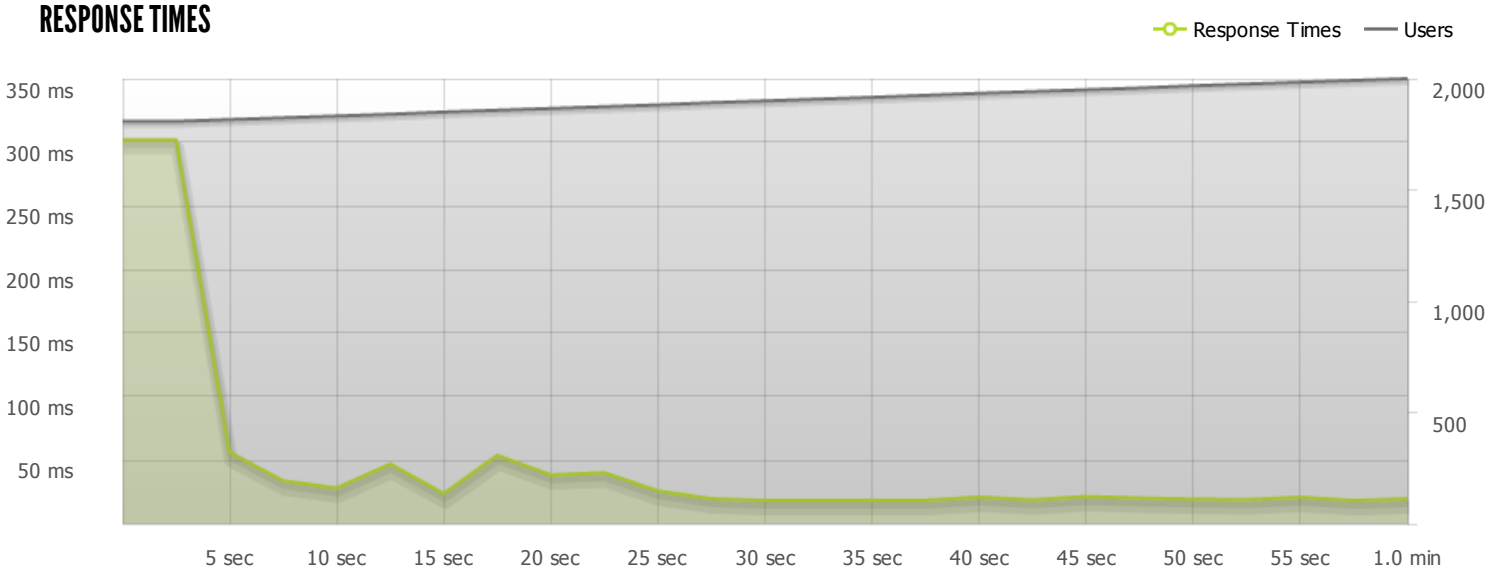
---

## TIMEOUTS

The first timeout happened at **2.5 seconds** into the test when the number of concurrent users was at **1808**. 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.

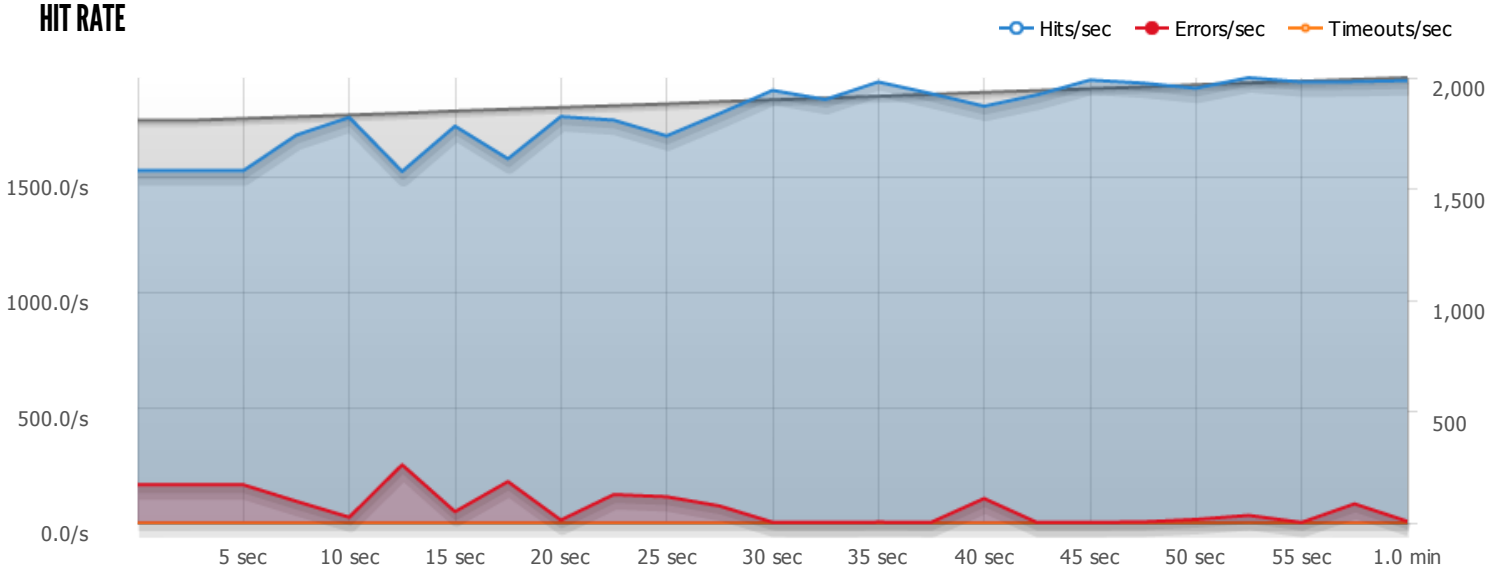
---

## RESPONSE TIMES




The max response time was: **302 ms @ 1808 users**

## HIT RATE




The max hit rate was: **1,932 hits per second**