**Report about conducted load test**

**Date:** 26-27th of June 2022

**Author:** Oleksandr Maksymenko

**Application:** BlogEngine.NET version 3.2

**Environment:** Host machine where script was run + Virtual Machine where blog is deployed

**Test Environment configuration (RAM, CPU etc.):**

|  |  |
| --- | --- |
| **RAM** | 4096 MB |
| **CPU** | 1 Core CPU (Intel(R) Core(TM) i7-8665U CPU @ 1.90GHz 2.11 GHz) |
| **System Type** | Windows 10 64-bit |

1. **Why such testing was conducted:** To determine general capacity of BlogEngine application, find saturation point
2. **Test script description:** Current script is combined script of 3 different role user scenarios: Anonymous, Admin and Editor.
   1. **Anonymous Script:** contains different user scenarios which are chosen with different probabilities (Open Home Page, Open Random Date, Open Predefined Date, Search Post by Name, Open large calendar, Open contacts). First three scenarios are opening random page with posts (in 50% cases). All scenarios (except Open Contacts) have Open Page script, which is used randomly in ~80% cases. Detailed description can be found on screenshots and in tables below.
   2. **Admin Script:** contains actions for Admin flow. Firstly “user” opens Home page, Logs In, after that starts loop which executes 10 times: Open Admin Page -> Open Users Page -> depending on current number of users -> Create User OR Delete User. After loop ends “user” will be logged off.
   3. **Editor Script:** contains actions for Editor flow. Firstly “user” opens Home page, Logs In, after that starts loop which executes 50 times: Open Predefined Date -> Open Random Page (in case editable post was not found on first page) -> Open Random Post -> Edit Post. After loop ends “user” will be logged off.

**Diagram

Description automatically generatedDiagram

Description automatically generatedDiagram

Description automatically generated**

**Comparison of Anonymous script step probabilities:**

|  |  |
| --- | --- |
| **Action** | **Probability** |
| Open Home Page | 15% |
| Open Random Date | 10% |
| Open Predefined Date | 30% |
| Search by Name | 30% |
| Open Large Calendar | 10% |
| Open Contacts | 5% |

|  |  |
| --- | --- |
| **Action:** Open Random Page | **Probability** |
| Yes | 50% |
| No | 50% |

|  |  |
| --- | --- |
| **Action:** Open Post | **Probability** |
| Yes | 80% |
| No | 20% |

|  |  |
| --- | --- |
| **Action:** Open First or Random Post | **Probability** |
| First | 35% |
| Random | 65% |

|  |  |
| --- | --- |
| **Action:** Send Comment | **Probability** |
| Yes | 20% |
| No | 80% |

Diagram

Description automatically generatedDiagram

Description automatically generated

1. **Tests:** 2 test runs.  
     
   **Test run preconditions:**

* 1000 pregenerated posts on predefined dates
* CSV file with 10 predefined dates
* CSV file with Admin credentials
* CSV file with Editor credentials

**Load Models:**

|  |  |  |
| --- | --- | --- |
|  | **Load Model 1** | **Load Model 2** |
| **Anonymous** Users | 500 | **450** |
| **Anonymous** Rump up time (s) | 600 | 600 |
| **Admin** Users | 2 | 2 |
| **Admin** Rump up time (s) | 10 | **300** |
| **Editor** Users | 10 | 10 |
| **Editor** Rump up time (s) | 120 | **600** |
| **Duration (s)** | 600 | 600 |

1. **Short summary on conducted tests:**

* During first test execution with Load Model 1 – application reached saturation point at 100 concurrent users, break point was reached at 330-340 users. Average throughput was 19 requests per second.
* For second test run load model was changed: number of active threads in Anonymous script was decreased from 500 to 450; Admin script rumpup changed from 10 to 300 seconds, which means we had only 1 Admin user for the first 5 minutes, and after that its number was 2; and Editor script rumpup was increased from 120 to 600 seconds. These changes led to changes in result metrics: saturation point was reached at 90 concurrent users, break point was at 360 users, average throughput remained nearly the same at 18 requests per second.

1. **Detailed test results:**

**5a. Load Model 1**

**Composite Graph** (Active Threads, Throughput, AVG response time)

Chart

Description automatically generated

**Active Threads, Total Throughput, Errors**

A screenshot of a computer

Description automatically generated with low confidence

**Average response time**

Chart, histogram

Description automatically generated

**Errors**

A picture containing text, computer, screenshot

Description automatically generated

**Aggregate report**

A picture containing text, computer, electronics

Description automatically generated

**CPU**

A screenshot of a computer

Description automatically generated with medium confidence

**Memory**

A picture containing graphical user interface

Description automatically generated

**Cache**

A screenshot of a computer

Description automatically generated with medium confidence

**Heap and Garbage Collector**

A screenshot of a computer

Description automatically generated with medium confidence

**Current connections and requests per sec**

A screenshot of a computer

Description automatically generated with medium confidence

**Network**

Graphical user interface

Description automatically generated

**5b. Load Model 2**

**Composite Graph** (Active Threads, Throughput, AVG response time)

Chart, histogram

Description automatically generated

**Active Threads, Total Throughput, Errors**

A screenshot of a computer

Description automatically generated with medium confidence

**Average response time**

Chart, histogram

Description automatically generated

**Errors**

A screenshot of a computer

Description automatically generated with medium confidence

**Aggregate Report**

A picture containing text, outdoor, electronics, computer

Description automatically generated

**CPU**

A picture containing chart

Description automatically generated

**Memory**

A picture containing graphical user interface

Description automatically generated

**Cache**

A screenshot of a computer

Description automatically generated with medium confidence

**Heap and Garbage Collector**

A screenshot of a computer

Description automatically generated with medium confidence

**Current connections and requests per sec**

A screenshot of a computer

Description automatically generated with medium confidence

**Application Restarts**

**A screenshot of a computer

Description automatically generated with medium confidence**

**Network**

Graphical user interface

Description automatically generated

1. **Conclusion:**

* After 2 test executions with different load models, we can conclude that Admin and Editor users (and their flow) are the hardest for application. When Blog has more logged in users at the same time – its response time is growing significantly for all, even anonymous, users.
* In first test run saturation point (100 users) was reached when application had 2 Admin users and 6-7 Editor users. In Load Model 2 test execution (saturation point = 90 users) it was reached with 1 Admin user and 3 Editor users, but number of Anonymous users was much higher than in Load Model 1 test run.
* Slowest transactions (in average response time) in both test runs (Load Model 1 and Load Model 2) were “Open Edit Post page” (editor script), “Open Admin Page” (admin script) and all anonymous script transactions.
* For both test runs we can observe switching of Garbage Collector source during execution which happens at the same time when application starts to throw errors. In run #1 it happens once (from Green to Yellow color on graph), at the same time we can observe drop of metrics related to Cache, and nearly the same moment number of Current Connection reached its peak at ~2000 and after that this number remained on plateau. For run #2 we can observe that switch happens twice: firstly from Yellow to Green, at that moment Application was restarted, and original instance of app started to work again (all Cache metrics grown up as it saved some data from original instance); but second time switch happened again at the moment when Cache metrics dropped, Application Restart didn’t happen and number of Current Connections again reached its plateau at ~2k.