Complex report

Blogengine v. 3.2.0.3

The main purpose of this document is to describe entire performance testing process of application. This report includes test strategy, test plan, main results from test runs, detailed analysis and conclusion.

Performance Testing Strategy

1. **Introduction**

The purpose of this document is to specifically describe how the performance requirements for Blogengine application will be tested and verified on test environment (System Under Test, SUT). This document describes the strategy of performance testing for the Blogengine. It consists of outlines for the following items:

* Items to be / not to be tested;
* Approach;
* NFR;
* Test environment;
* Testing tools;
* Entry and Exit criteria;
* Suspension and Resumption Criteria;
* Risks.

1. **Items to be tested**

Mainly, Blogengine application will be tested as server side.

|  |  |
| --- | --- |
| **#** | **Modules name** |
|  | Search |
|  | Login/Logout |
|  | Open post |
|  | Create post |
|  | Edit post |
|  | Create/delete user |
|  | Leave comment |

1. **Items not to be tested**

Functionality of the whole application

1. **Approach**
   1. **Test types assumed for conducting**
      1. **Smoke testing**

Should be performed every time when functionality of the application and the script need to be checked. Also, if needed can be used as warming up test before main testing step.

* + 1. **Capacity testing**

Should be performed to find the number of virtual users which the application support in stable state. The test can be performed as one of first main tests, and should be performed after significant changes in the application or its configuration.

* + 1. **Load test**

Load test is a kind of the most regular tests to check benchmark of the application and its components. Usually, is running after finding capacity.

* + 1. **Stress testing**

Stress testing supposed to run occasionally to check application’s stability under high load. Can be performed close to after code complete or by special request.

* + 1. **Scalability testing**

Can be performed once or more in order to get multiplier(s) for different number of front-end/other servers.

* + 1. **Volume testing**

Is to run with small/planned/huge amount of data with regular load to get indicators on application’s responsiveness/metrics change. Should be performed at least once or by special request.

* + 1. **Durable (Stability) testing**

Supposed long time running the test with the load lower than average. Should be performed occasionally after significant code changes or by special request to make sure the application’s responsiveness and key performance indicators do not change significantly after long time running, and to check on memory leak as well.

1. **Non-Functional requirements**

Main non-functional requirements (NFR) are not provided, so they should be based on results mentioned below.

1. **Environmental needs**
   1. **Test environment(s)**

Blogengine is running on Virtual machine, that has such characteristics:

OS: Virtual Box 6.1, Windows 10 Enterprise(64 bit)

Processor: Intel Core i7-8665U CPU @ 1,90GHz 2,11GHz, count of cores varies in scalability test (basically 1 CPU)

Storage: 127 GB (2TB for scalability test)

RAM: amount varies in scalability test (basically 2048)

* 1. **Testing tools**

For Performance testing such Load Generator (LG) as Jmeter v. 5.2.1 was used.

**Important note!:** Jmeter was run on the same virtual system as tested application, so gathered metrics depend on load, that make running Java application, as well.

1. **Test data**

To have whole cycle of performance testing test data for Blogengine should be:

1. Reusable.
2. Generated in necessary amount for different stage of testing at any time:
   1. Posts;
   2. Users;
   3. Other.
3. Cleanable.
4. **Performance Entry, Exit, Suspension and Resumption Criteria**

**8.1 Entry Criteria**

* Test plan is completed and approved.
* Test environment is updated with needed system configurations.
* Test data is completed and present on the performance testing environment in sufficient time to allow test scripts to be completed.
* Test scripts are created.
* All assigned resources are available to monitor the test.
  1. **Exit Criteria**
* All test scripts completed successfully
* No critical problems encountered
* All non-critical problems are logged
* All test logs are captured.
* Report based on testing is prepared and accepted.
  1. **Suspension Criteria**
* Not all test scripts completed
* Critical problems are encountered and logged
* Software/Hardware errors prevented the completion of the test
* Test data issues (no test user or wrong credentials, incomplete data)
* Significant changes in workflow of functionality of the application which require updates in the test plan or scripts/scenarios

**8.4 Resumption criteria**

Resumption will only occur when the problem(s) that caused the suspension have been resolved.

1. **Risks and contingencies**
   1. **Risks**

* Performance testing results can be essentially different even in case of minor difference in think times, arrival rate and test duration.
* During the execution of the tests, some major performance or functional problems due to changes in Blogengine configurations may occur and in that case it may be necessary to repeat the load test from the beginning.
* Performance testing tool is not capable of identically reproducing real life scenarios - so results could only be trusted as having limited reliability level.
* Network/systems latency issues.
* Environment’s unavailability.
* Some applications may be running at the same time as script, so they may produce extra load.
* Unexpected shutdown of the system.

Performance Test Plan

1. **Introduction**

The purpose of this document is to specify performance requirements and conditions for Blogengine application. The document will outline test objectives, test scenarios, test data, user roles and testing tasks. This document has clearly identified what the test deliverables will be, and what is deemed in and out of scope.

1. **Test description**
   1. **Test objectives**

* Find capacity of the application;
* Define regular load;
* Define low load and measure how long system can work with such load and how behave during all test;
* Analyze how scalable system is;
* Check how application behavior will be changed after switching from file system to database data source;
* Understand what amount and type of data system is able/not able to operate without degradation.
  1. **Test scenarios**

Anonymous scenario:

* + - * Main script;
      * Anonymous script;
      * Open post script.

| **Scenario** | **Script** | **Test Script Steps** | | | |
| --- | --- | --- | --- | --- | --- |
| Anonymous user scenario | Main script |  | Open Home page | | |
| x50 | Anonymous script | | |
| Anonymous script | Random | Open Home page | Open Random Page | Open Post Script |
| Open Random Date |
| Open Predefined Date |
| Search by Name | Open Post Script | |
| Open Large Calendar |
| Open Contacts |  | |
| Open post script | Random or first | Open Random Post | Add Comment | |
| Open First Post |

Admin scenario:

* + - * Admin script.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Scenario** | **Script** | **Test Script Steps** | | | | |
| Admin user scenario | Admin script | Open Home page | | | | |
| Login | | | | |
| x10 | Number of users | <=10 | Add user | |
| >10 and <=20 | Random | Add user |
| Delete user |
| >20 | Delete user | |
| Logout | | | | |

Editor scenario:

* + - * Editor script.

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | **Script** | **Test Script Steps** | |
| Editor user scenario | Editor script | Open Home page | |
| Login | |
| x50 | Open Predefined Date |
| Open Random Post |
| Edit |
| Logout | |

* 1. **Test data**

1. Posts should be generated using separate script by Editors users.

Note!: amount of generated posts varies from type of testing.(basically 1000).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Script** | **Test Script Steps** | | | |
| Post test data preparation | Open Home page | | | |
| Login | | | |
| Open Admin page | | | |
| x1000 | Number of posts | <=100 | Create post with predefined date |
| >100 | Create post with random date |

1. TBD: Automate process of creation Editor, Admin users. For now they are created manually.
   1. **Test users roles**
      * + admin, admin1 users are using Admin user role;
        + Editor, Editor1 are using Editor user role;
        + All anonymous users are using Anonymous user role.
2. **Non-Functional requirements**

As mentioned in Blogengine testing strategy non-functional requirements (NFR) are not provided, so they should be based on results of running scripts mentioned below.

1. **Suspension criteria and resumption requirements**

Suspension and resumption criterias described in Blogengine strategy document.

1. **Test deliverables**

Main expected test deliverables are:

1. Test plan.
2. Test strategy.
3. Basic test scripts and scenarios.
4. Baseline of main metrics: response times for the pages, system resources consuming etc.
5. Performance indicators needed for NFRs definition.
6. Analysis on gathered KPI`s.
7. **Testing tasks**
   * 1. Basic scripting
     2. Basic scenarios creation
     3. Setting up environment
     4. Setting up load generation tools
     5. Smoke testing
     6. Running set of performance testing:
        1. Capacity
        2. Regular load test (after definition of capacity)
        3. Volume
        4. Durable (stability)
        5. Scalability
     7. Testing with DB data source:
        1. Capacity
        2. Regular load test (after definition of capacity)
        3. Volume
        4. Durable (stability)
        5. Scalability
        6. Stress testing
8. **Test environment**

Test environment described in Blogengine strategy document.

1. **Risks**

Main risks are described in Blogengine strategy document.

Test results

**Capacity test**

**Load model**

|  |  |  |
| --- | --- | --- |
| **Data** | | **Number** |
| Users | Anonymous | 200 |
| Admin | 2 |
| Editor | 2 |
| Ramp-up period, seconds | Anonymous | 600 |
| Admin | 6 |
| Editor | 6 |
| Test duration, seconds | | 600 |
| Time for control action flow, ms | | 1000-2000 |
| Posts in Blog | | 1000 |

**Results**

|  |  |
| --- | --- |
| Comfort zone | Max throughput: 27 tps  response time: < 60 ms  < 75 users; |
| Zone of degradation | Max throughput: 37 tps  response time: 80 - 150 ms  75-104 users; |
| Capacity zone | Max throughput: < 44 tps  Response time: > 150 ms  > 104 users; |
| Capacity point | 104 users; |

**Regular load test**

**Load model**

|  |  |  |
| --- | --- | --- |
| **Data** | | **Number** |
| Users | Anonymous | 70 |
| Admin | 2 |
| Editor | 2 |
| Ramp-up period, seconds | Anonymous | 210 |
| Admin | 6 |
| Editor | 6 |
| Test duration, seconds | | 420 |
| Time for control action flow, ms | | 1000-2000 |
| Posts in Blog | | 1000 |

**Results**

|  |  |
| --- | --- |
| Max throughput | 28 tps |
| Max response time | 30-40 ms, up to 132 ms; |
| CPU | 50-80 % load; up to 100 %; |
| Memory | 85-90% |
| Test issues | Yes, 0.02% for all test |

**Scalability test**

**Load model**

|  |  |  |
| --- | --- | --- |
| **Data** | | **Number** |
| Users | Anonymous | 70 |
| Admin | 2 |
| Editor | 2 |
| Ramp-up period, seconds | Anonymous | 210 |
| Admin | 6 |
| Editor | 6 |
| Test duration, seconds | | 420 |
| Time for control action flow, ms | | 1000-2000 |
| Posts in Blog | | 1000 |

**Results**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | **1 CPU, 2048 RAM** | | **2 CPU, 2048 RAM** | | **3 CPU, 2048 RAM** | | **4 CPU, 2048 RAM** | |
| Maxthroughput | | 30 tps | | 29 tps | | 32 tps | | 30 tps | |
| Response time | | 11-28 ms, up to 44 ms | | 7-13 ms | | 7-10 ms | | 7-13 ms | |
| CPU | | 26-52 % load; up to 83 %; | | 15-30 % load; up to 62 %; | | 9-27 % load; up to 60 %; | | 7-25 % load; up to 47 %; | |
| Memory | | 86-90% | | 85-90% | | 79-88% | | 88-93% | |
| Test issues | | 0% | | 0.24% | | 0.13% | | 0.18 % | |
|  | **1 CPU, 2048 RAM** | | **1 CPU, 3072 RAM** | | **1 CPU, 4096 RAM** | | **1 CPU, 6144 RAM** | | **Multiplier** | |
| Maxthroughput | 31 tps | | 30 tps | | 29 tps | | 29 tps | | - | |
| Response time | 14-35 ms, up to 302 ms | | 13-26 ms, up to 442 ms | | 13-22 ms, up to 310 ms | | 8-20 ms, up to 270; | | ≈ 0.8 | |
| CPU | 32-53 % load; up to 100 %; | | 30-54 % load; up to 100 %; | | 30-53 % load; up to 87 %; | | 30-50 % load; up to 100 %; | | - | |
| Memory | 86-90% | | 69-73% | | 50-55% | | 42-44% | | ≈ 0.8 | |
| Test issues | 0% | | 0% | | 0% | | 0% | | - | |

**Volume test**

**Load model**

|  |  |  |
| --- | --- | --- |
| **Data** | | **Number** |
| Users | Anonymous | 70 |
| Admin | 2 |
| Editor | 2 |
| Ramp-up period, seconds | Anonymous | 210 |
| Admin | 6 |
| Editor | 6 |
| Test duration, seconds | | 420 |
| Time for control action flow, ms | | 1000-2000 |
| Posts in Blog | | 500,  1000,  2000,  5000,  2000 mixed |

**Results**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **100 posts** | **1000 posts** | **2000 posts** | **5000 posts** |
| Maxthroughput | 30 tps | 29 tps | 29 tps | 29 tps |
| Response time | 10-20 ms, up to 49 ms | 11-19 ms, up to 44 ms | 13-28 ms, up to 270 ms | 30-51 ms, up to 136 ms |
| CPU | 30-50% load; up to 81%; | 30-50% load; up to 79%; | 39-59% load, up to 100% | 50-73% load, up to 100% |
| Memory | 77-83% | 85-88% | 81-87% | 82-88% |
| Test issues | 0% | 0% | 0% | 0.10% |

|  |  |  |
| --- | --- | --- |
|  | **2000 pure posts** | **2000 mixed posts with photos** |
| Maxthroughput | 29 tps | 29 tps |
| Response time | 13-28 ms, up to 270 ms | 29-75 ms, up to 186 ms |
| CPU | 39-59% load, up to 100% | 48-77% load, up to 100% |
| Memory | 81-87% | 87-91% |
| Test issues | 0% | 0.05% |

**Stability test**

**Load model**

|  |  |  |
| --- | --- | --- |
| **Data** | | **Number** |
| Users | Anonymous | 30 |
| Admin | 2 |
| Editor | 2 |
| Ramp-up period, seconds | Anonymous | 90 |
| Admin | 6 |
| Editor | 6 |
| Test duration, seconds | | 21600 |
| Time for control action flow, ms | | 1000-2000 |
| Posts in Blog | | 1000 |

**Results**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Period of stability** 00:00:00-03:42:00 | **Period of down** 03:42:00-04:26:00 | **Period of recovery** 04:26:00-06:00:00 |
| Throughput | 13-15 tps | 8-10 tps | 13-15 tps; up to 23 tps |
| Response time | <100 ms - first half, 300-400 ms - second half | >1000 ms, up to 7300 ms | <200 ms, up to 400 ms |
| CPU | 20-40% load; up to 100%; | 95-100% | 20-50% load, up to 100% |
| Memory | 75-88% | 70-86% | 81-87% |
| Test issues | 0.04% | | |

**Testing with DB data source**

1. **Capacity test**

**Load model**

|  |  |  |
| --- | --- | --- |
| **Data** | | **Number** |
| Users | Anonymous | 200 |
| Admin | 2 |
| Editor | 2 |
| Ramp-up period, seconds | Anonymous | 600 |
| Admin | 6 |
| Editor | 6 |
| Test duration, seconds | | 600 |
| Time for control action flow, ms | | 1000-2000 |
| Posts in Blog | | 1000 |

**Results and comparing**

|  |  |  |
| --- | --- | --- |
|  | **File system** | **Database** |
| **Comfort zone** | Max throughput: 27 tps, response time: < 60 ms, < 75 users; | Max throughput: 35 tps , response time: < 43 ms, < 52 users |
| **Zone of degradation** | Max throughput: 37 tps  response time: 80 - 150 ms  75-104 users; | Max throughput: 46 tps  response time: 52 - 145 ms  52-96 users |
| **Frustration zone** | Max throughput: < 44 tps  Response time: > 150 ms  > 104 users; | Max throughput: < 48 tps  Response time: > 127 ms  > 96 users; |
| **Capacity point** | 104 users; | 96 users |

1. **Regular load test**

**Load model**

|  |  |  |
| --- | --- | --- |
| **Data** | | **Number** |
| Users | Anonymous | 64 |
| Admin | 2 |
| Editor | 2 |
| Ramp-up period, seconds | Anonymous | 192 |
| Admin | 6 |
| Editor | 6 |
| Test duration, seconds | | 384 |
| Time for control action flow, ms | | 1000-2000 |
| Posts in Blog | | 1000 |

**Results and comparing**

|  |  |  |
| --- | --- | --- |
|  | **File system** | **Database** |
| Maxthroughput | 28 tps | 28 tps |
| Response time | 30-40 ms, up to 132 ms | 12-30 ms, up to 115 ms |
| CPU | 50-80% load, up to 100% | 28-52% load, up to 91% |
| Memory | 85-90% | 79-84% |
| Test issues | 0.02% | 0% |

1. **Regular load test with maximum scalable configuration**

**Load model**

|  |  |  |
| --- | --- | --- |
| **Data** | | **Number** |
| Users | Anonymous | 64 |
| Admin | 2 |
| Editor | 2 |
| Ramp-up period, seconds | Anonymous | 192 |
| Admin | 6 |
| Editor | 6 |
| Test duration, seconds | | 384 |
| Time for control action flow, ms | | 1000-2000 |

**Results and comparing**

|  |  |  |
| --- | --- | --- |
| **1 CPU, 6144 MB RAM** | **File system** | **Database** |
| Maxthroughput | 29 tps | 27 tps |
| Response time | 8-20 ms, up to 270 ms | 8-24 ms, up to 607 ms |
| CPU | 30-50% load, up to 100% | 27-50% load, up to 100% |
| Memory | 42-44% | 52-56% |
| Test issues | 0% | 0% |

1. **Stress test**

**Load model**

|  |  |  |
| --- | --- | --- |
| **Data** | | **Number** |
| Users | Anonymous | 130 |
| Admin | 2 |
| Editor | 2 |
| Ramp-up period, seconds | Anonymous | 130 |
| Admin | 0 |
| Editor | 0 |
| Test duration, seconds | | 400 |
| Time for control action flow, ms | | 500-800 |

**Results**

|  |  |  |
| --- | --- | --- |
|  | **Comfort zone** | **Degradation zone** |
| Maxthroughput | 65 tps | 82 tps, down to 9 tps |
| Response time | <100 ms | 700-800 ms, up to 10000 ms |
| CPU | <100% | 100% |
| Memory | 90% | 88-92%, up to 95% |
| Test issues | Yes | No |

1. **Volume test**

**Load model**

|  |  |  |
| --- | --- | --- |
| **Data** | | **Number** |
| Users | Anonymous | 64 |
| Admin | 2 |
| Editor | 2 |
| Ramp-up period, seconds | Anonymous | 192 |
| Admin | 6 |
| Editor | 6 |
| Test duration, seconds | | 384 |
| Time for control action flow, ms | | 1000-2000 |
| Posts in Blog | | 2000,  2000 mixed |

**Results and comparing**

|  |  |  |
| --- | --- | --- |
| **2000 pure posts** | **File system** | **Database** |
| Maxthroughput | 29 tps | 27 tps |
| Response time | 13-28 ms, up to 270 ms | 10-24 ms, up to 300 ms |
| CPU | 39-59% load, up to 100% | 22-48% load, up to 81% |
| Memory | 81-87% | 82-86% |
| Test issues | 0% | 0% |

|  |  |  |
| --- | --- | --- |
| **2000 mixed posts (with 1MB image)** | **File system** | **Database** |
| Maxthroughput | 29 tps | 27 tps |
| Response time | 29-75 ms, up to 186 ms | 8-28 ms, up to 629 ms |
| CPU | 48-77% load, up to 100% | 21-50% load, up to 96% |
| Memory | 87-91% | 77-84% |
| Test issues | 0.05% | 0% |

Detailed analysis and Conclusion

**Capacity of Blogengine v. 3.2.0.3 application is 104 users.** After increasing this number CPU starts growing till 100% and don`t get down till the end of test. Also errors start occurring. Amount of transaction falling, and response time starts growing. So, capacity is reached when 104 users are using the site: 2 editors, 2 admins and 100 anonymous, and approximately 36 transactions at the same time are performed.

**System is not scalable by CPU.** Already from 2 CPU errors started occurring. Except of this, for 2 CPU and more system sometimes stops to respond, because CPU get overloaded by IIS Worker Process. Looks like IIS is not able to operate properly with transactions for 2 and more CPU.

**System is scalable by RAM** and it is possibly may be scaled even more (8Gb for the beginning) because there is no degradation. Throughput isn`t changed at all as well as CPU load, but response time starts falling from run to run. Memory load also get less with growing size of RAM, what is expected. What is perfect is that for all of runs no error occurred. Also was calculated multiplier and it is approximately 0.8.

**System is not able to operate amount of data > 2000 posts** and actions on it. Maximum throughput isn`t changing for different amount of posts, but response time as well as CPU load is changing. Already on 2000 posts run degradation was started. Results of 5000 posts run are much worse, both for response time and CPU load, but for failed transactions too. It`s 0.10 % for all run, while it`s 0% for all others.

**For system it`s harder to operate more heavy data(posts with 1MB image), performance is falling.** Maximum throughput isn`t changing for different collections of posts, but all other metrics are changing. We see degradation on 2000 mixed posts run: response time jumped up in more than two times, CPU load also grown up and some errors occurred.

**System can be used by 34 users with different roles for more than 3:30 hours**. Also we can say, that **system can recover then in 44 minutes and work stable, but with some losses in performance.** System works stable till 03:42:00. CPU did not get overloaded, throughput was also stable since all users started using the site. Issues occurs, but just couple on almost 4 hours of test. At approximately 03:42:00 system fell, response time was increased in several times, a lot of issue started occurring, CPU reached 100 % and didn`t get down a lot till 04:26:00. At approximately 04:26:00 system started recovery and in 15 minutes fully got stabilized. During these 15 minutes throughput was jumping from 7 to 23 tps, but then got normalized, CPU also fell till average 20-50 % load, issue stopped occurring a lot.

**Capacity of application, that using DB is lower**, that capacity of the system, that is using file system as data source. Till approximately 04:40 system is in comfort zone. In this period of time system reaches its capacity point, CPU reached 100% and don`t get down till the end of test. Amount of transaction falling, and response time starts growing. At this time approximately 96 users are using the site: 2 editors, 2 admins and 92 anonymous, and approximately 35 transactions at the same time are performed. Comparing with run of the same script using file system we can see, that in this run errors didn`t occur at all, but system reached its capacity earlier. Amount of maximum transactions is approximately the same.

**System with regular load, that is using database as data source works more stable and quickly** than system that is using file system. Response time for the system with database data source is lower, as well as CPU load and memory load. Max throughput is the same and it is good result, because load on system with DB is less, than on system, that using files as data source. Also we see, that test issues are not occurring at all, so looks like system is able to operate with defined amount of data at one time without problems.

**System with maximum scalable configuration(1 CPU and 6144 MB RAM) that is using DB works little bit worse but it is not critical a lot**. Response time is approximately the same as well as CPU load. Maximum amount of transactions is little bit less, but as already mentioned its only due to that fact, that load on system with DB is less, than on system, that using files as data source and sometimes system performed 28-29 tps as well. Memory load is bigger, but its only due to other running processes on the system. After killing them to make state of the system similar to state, when file system was checked, it got similar percentage. So, it varies from run to run. Test issues were not occurring in both cases. Possibly some increase in memory configuration will bring better results.

**System under stress load is able to handle with such load** on some way, because system didn`t stop working and continue to perform transactions. **It can normalize work after critical load moments.** On 01:35 system behavior of the system started degradation. CPU reached 100 % and didn`t fall a lot since this moment till the end of test. Response time started growing, amount of transactions continued growing but with jumps. On 03:04 we see the most critical moment for the system. Throughput fell in several times as well as response time increased in several times. After this this metrics came to the previous values and didn`t change till the end of test. Also we see that at the start of test some issues occurred, but according to the all test it is only 0.01% of them.

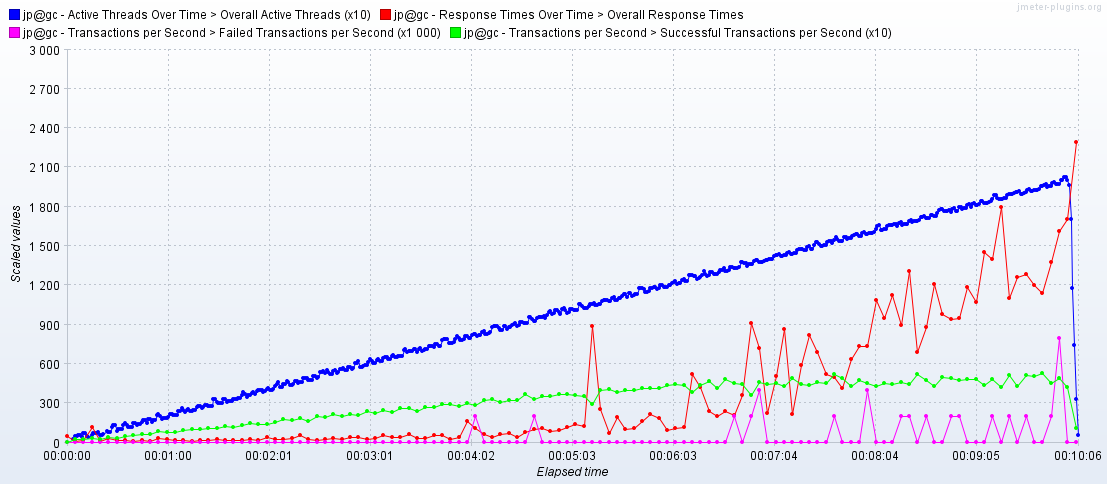
**System using file system data source can not well operate such amount of data(2000 posts), while system using DB is able.** Response time is less in two times than for system using DB, as well as CPU load and Memory. In both cases no issues occurred, but performance got higher.

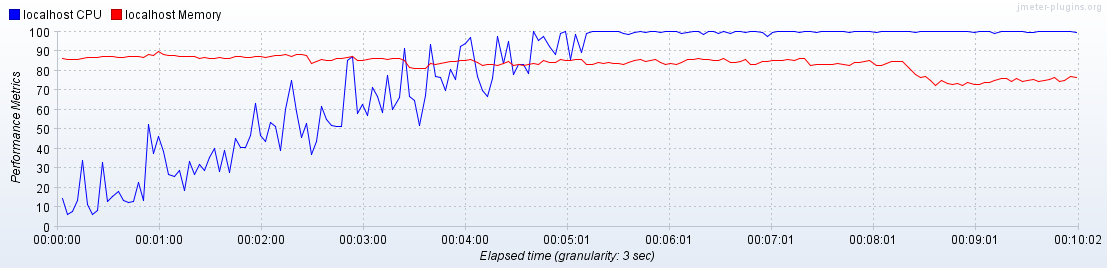
**Database can operate heavier file(posts with 1MB image) more quickly and effectively.** Response time is less in several times for file system data source, CPU load is also higher as well as memory. Throughput is lower for database, but it is acceptable, because as already said it vary from run to run and doesn`t relate to higher load.

Attachments

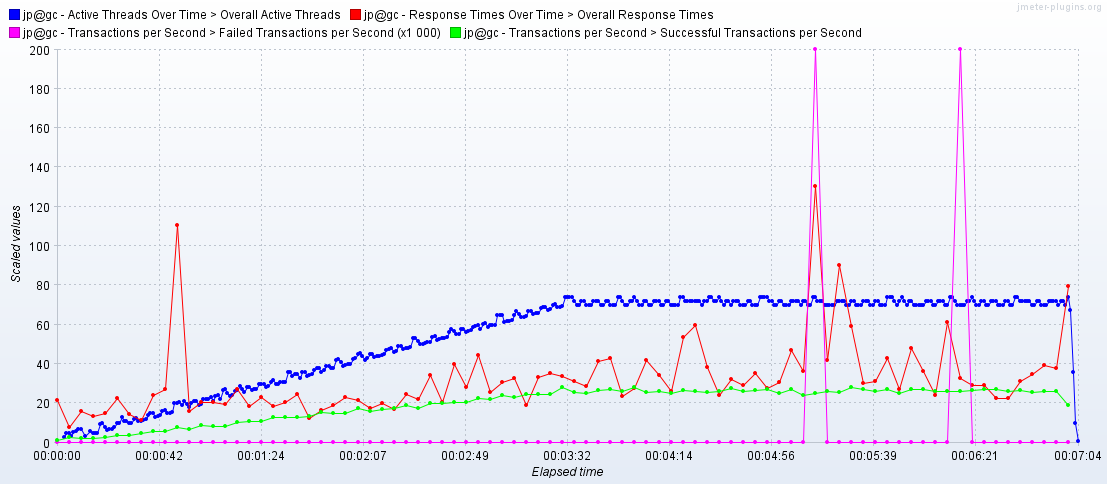
Graphics

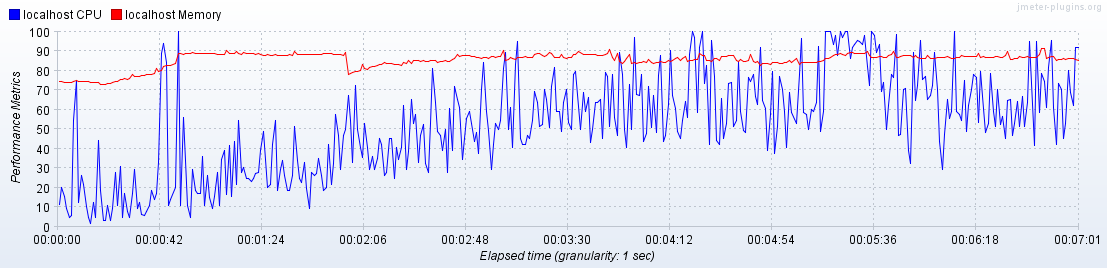
**Capacity test**





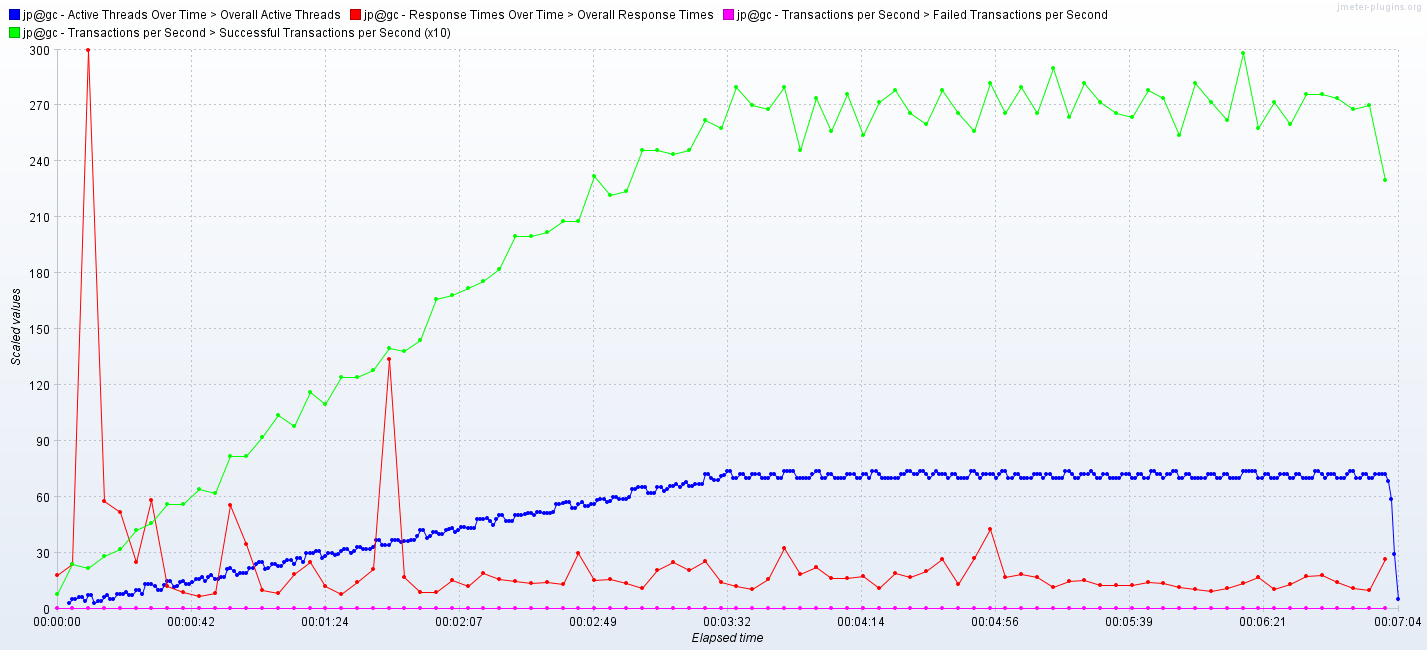
**Regular load test**

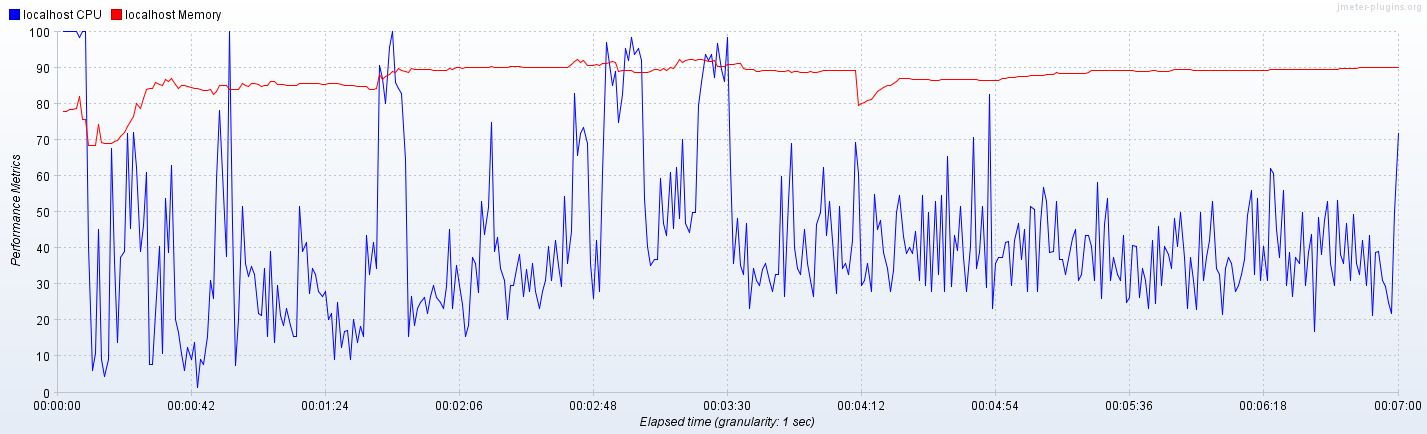




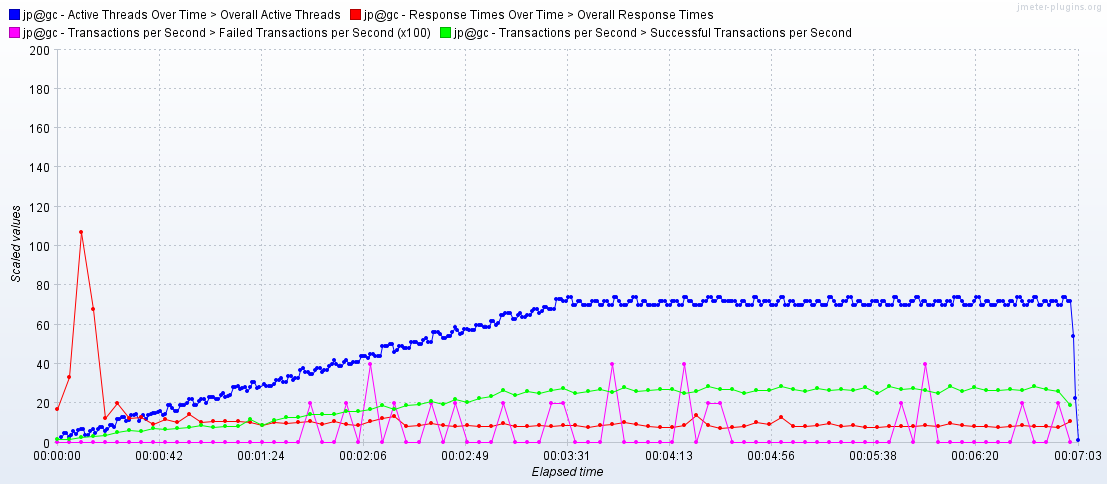
**Scalability test**

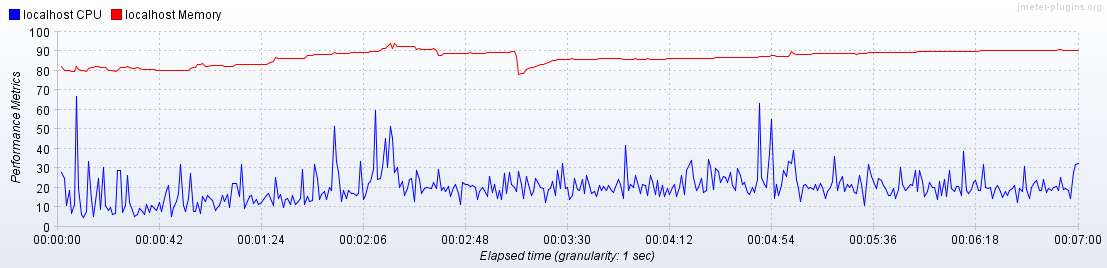
**CPU: 1, RAM 2048 MB**



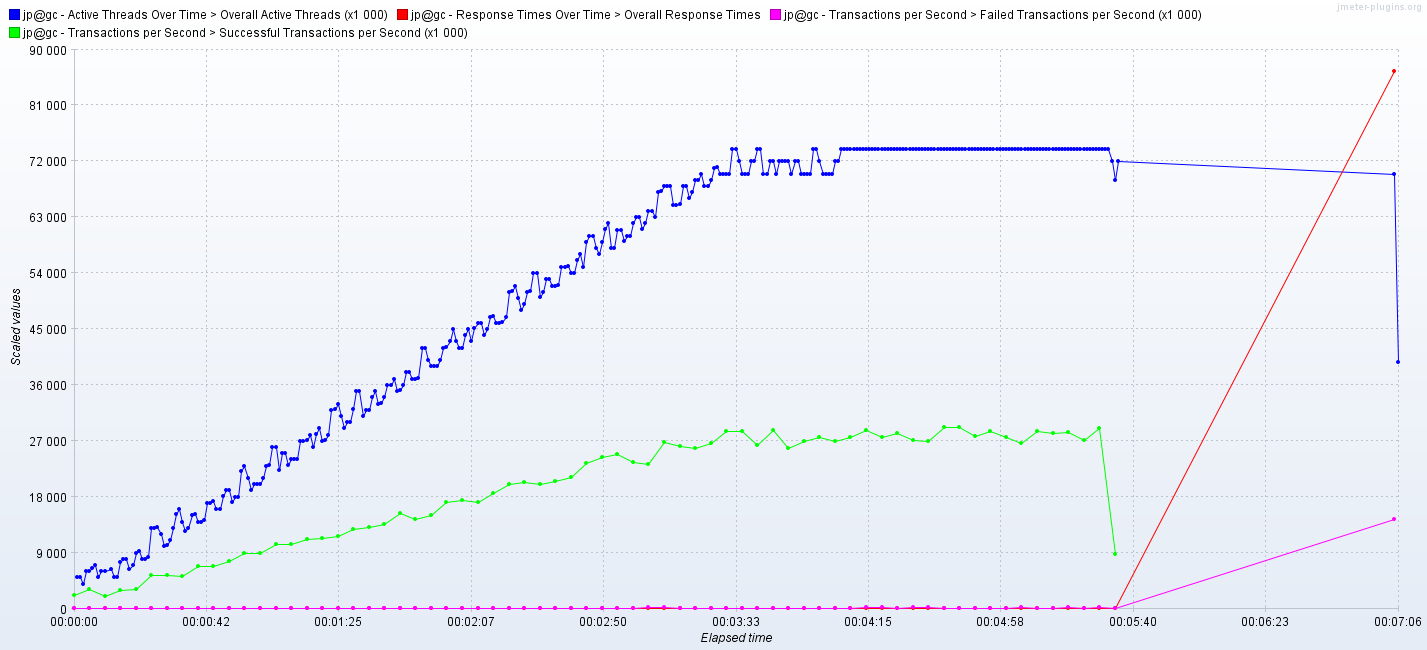


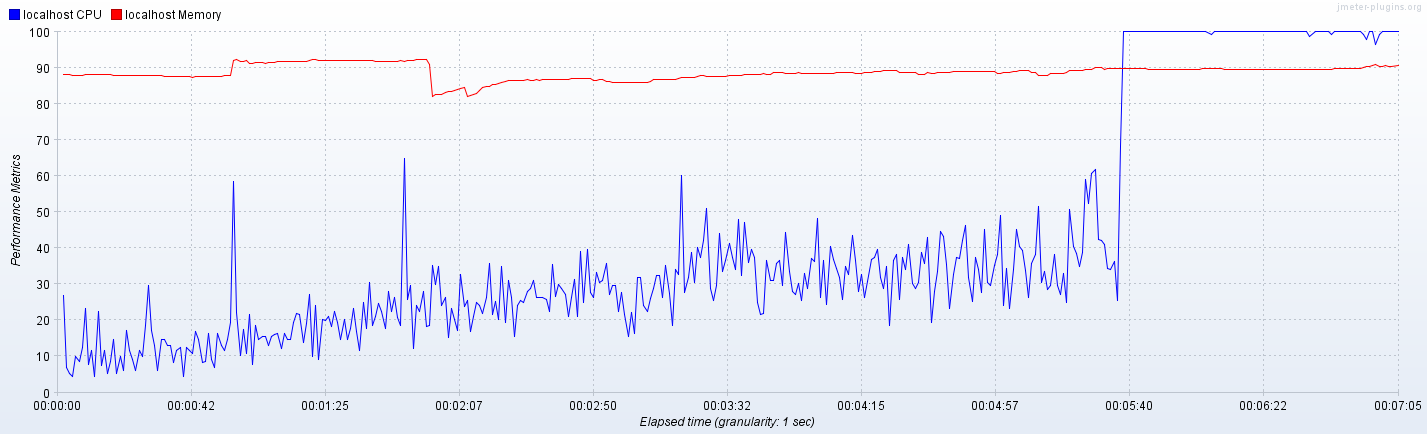
**CPU: 2, RAM 2048 MB**



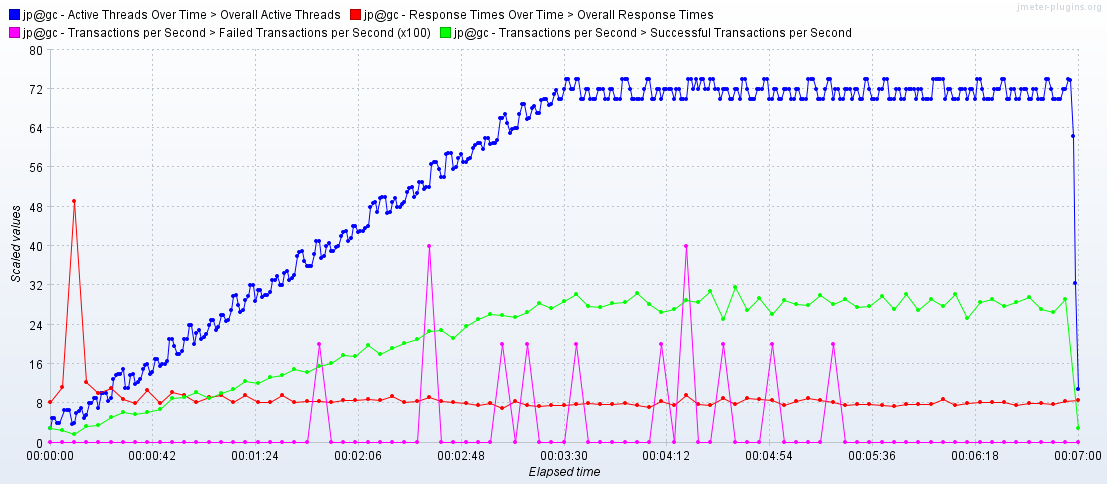


**Next results show, that system got overloaded since some time from start of the test:**

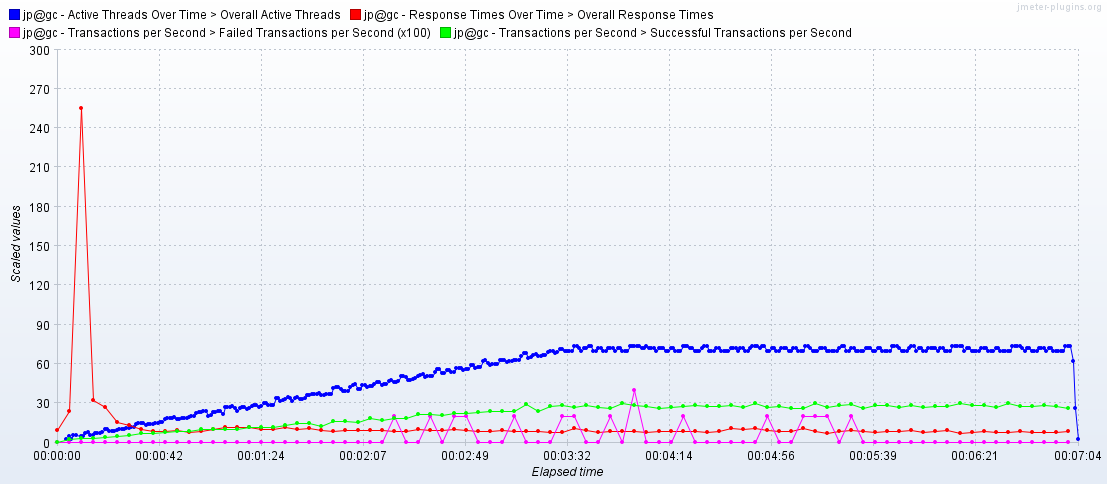


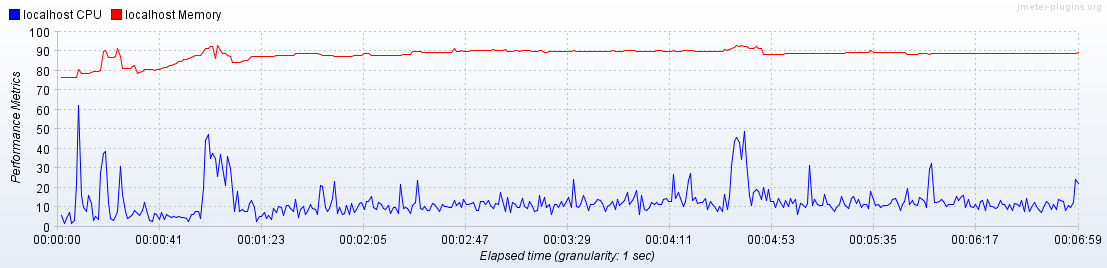


**CPU: 3, RAM:** **2048MB**

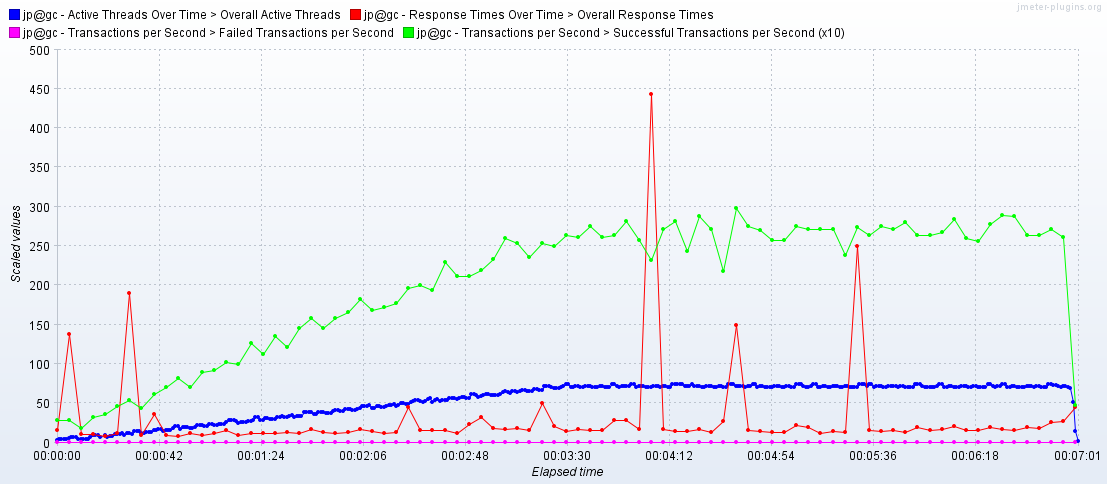


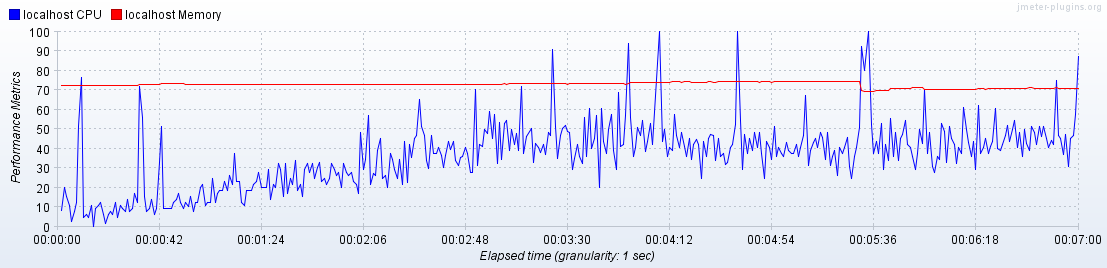
**CPU: 4, RAM:** **2048MB**



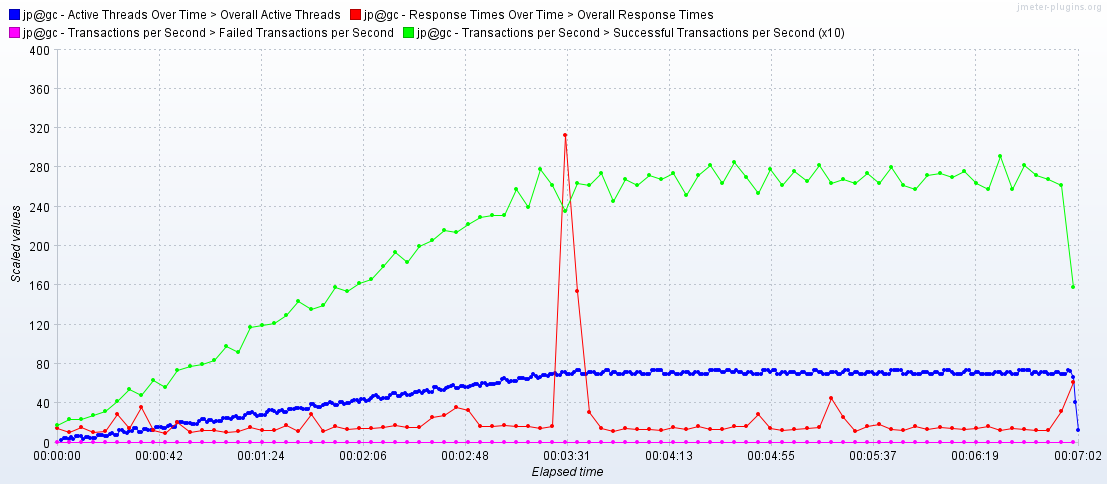


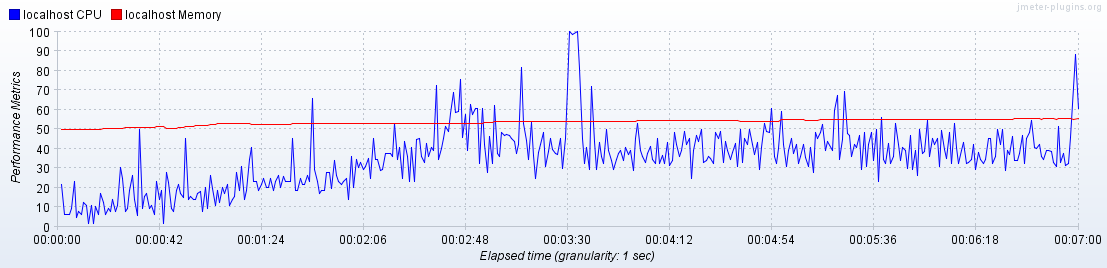
**CPU: 1, RAM 3072 MB**



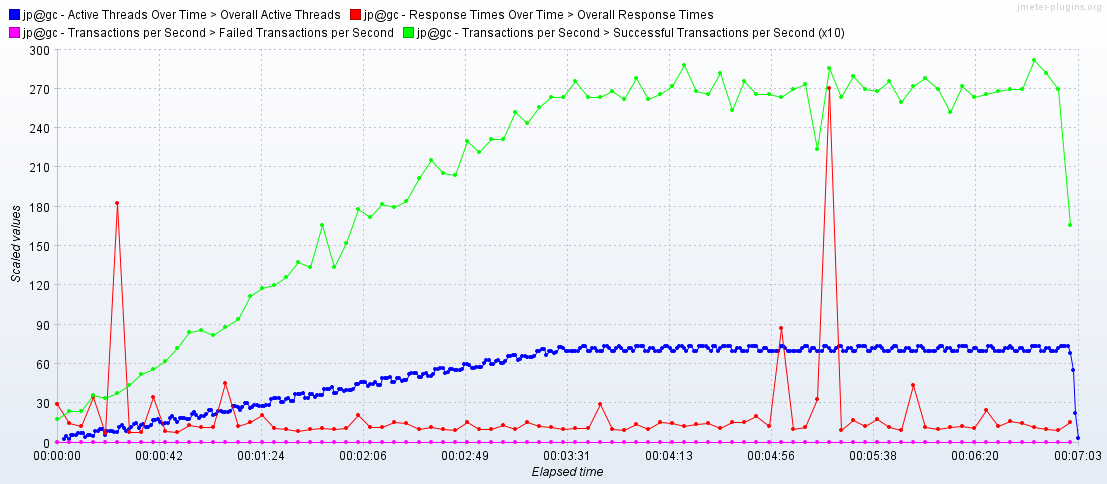


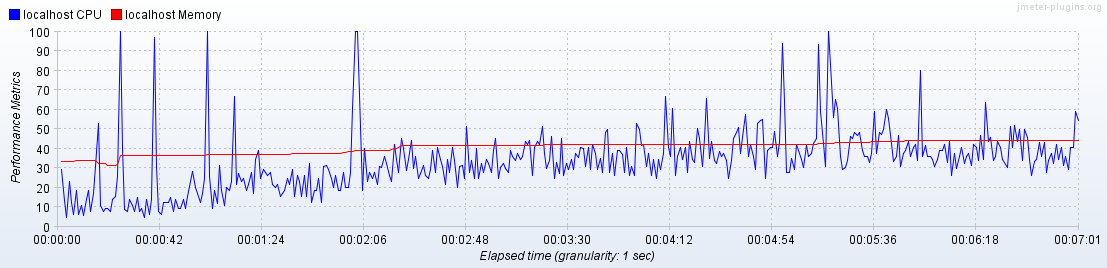
**CPU: 1, RAM 4096 MB**





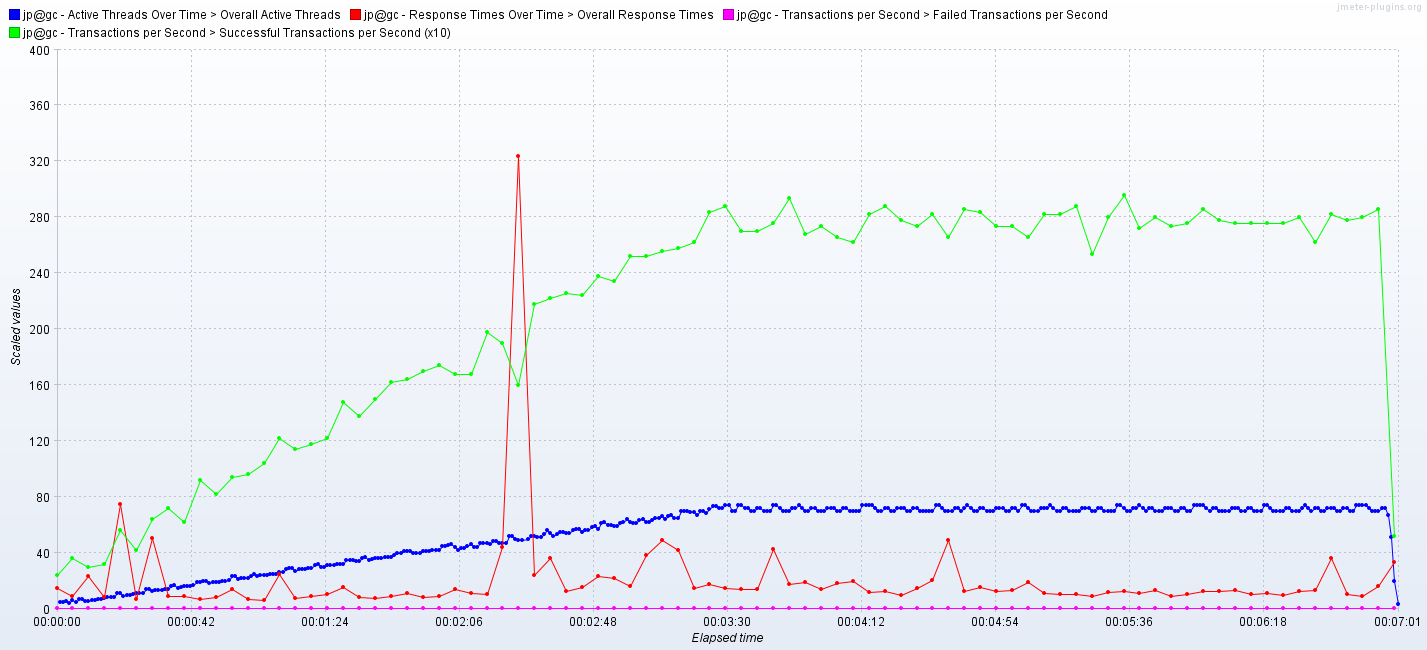
**CPU: 1, RAM 6144 MB**

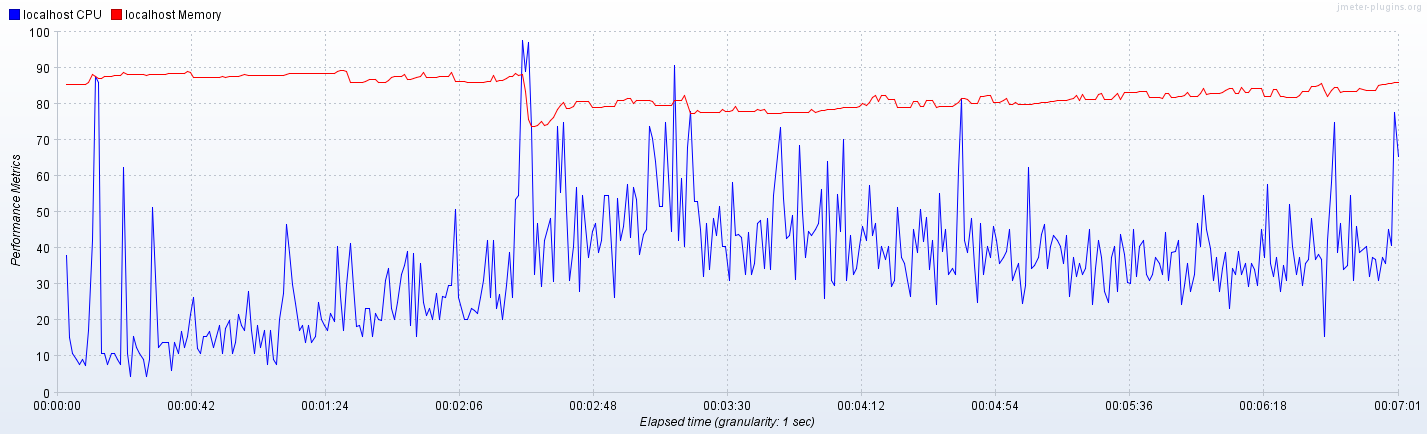




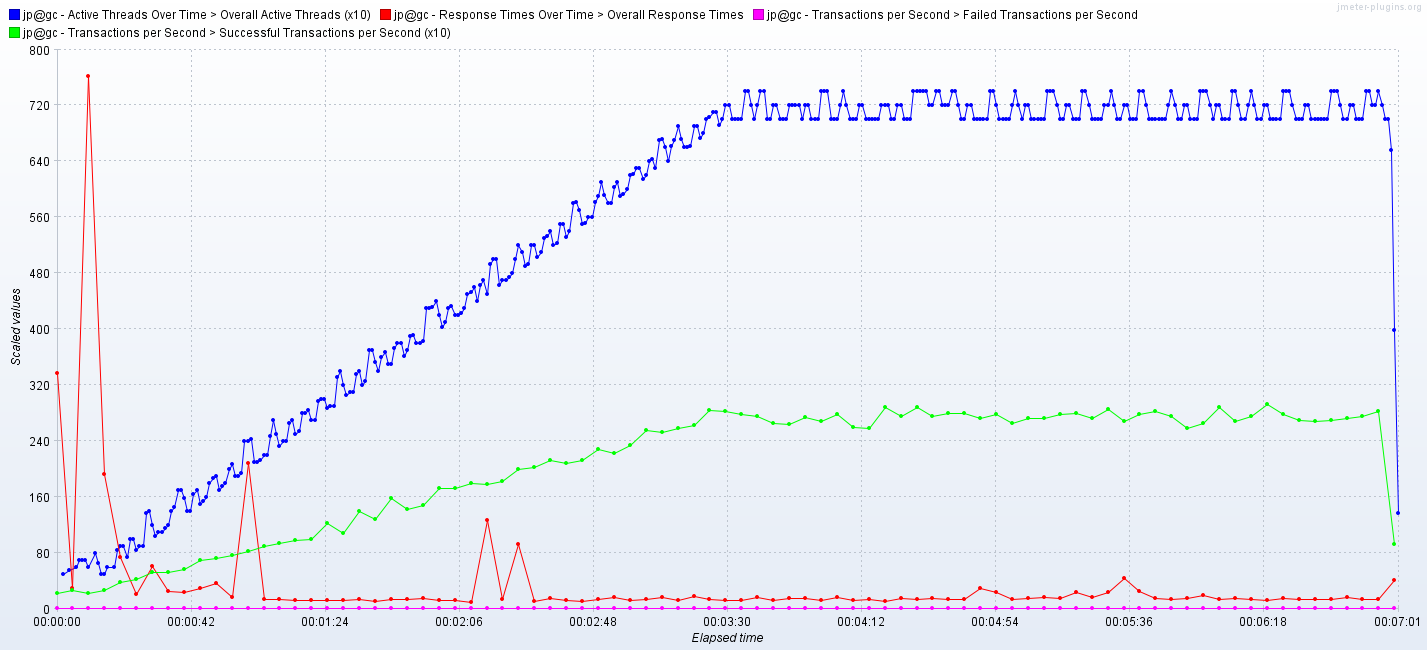
Volume testing

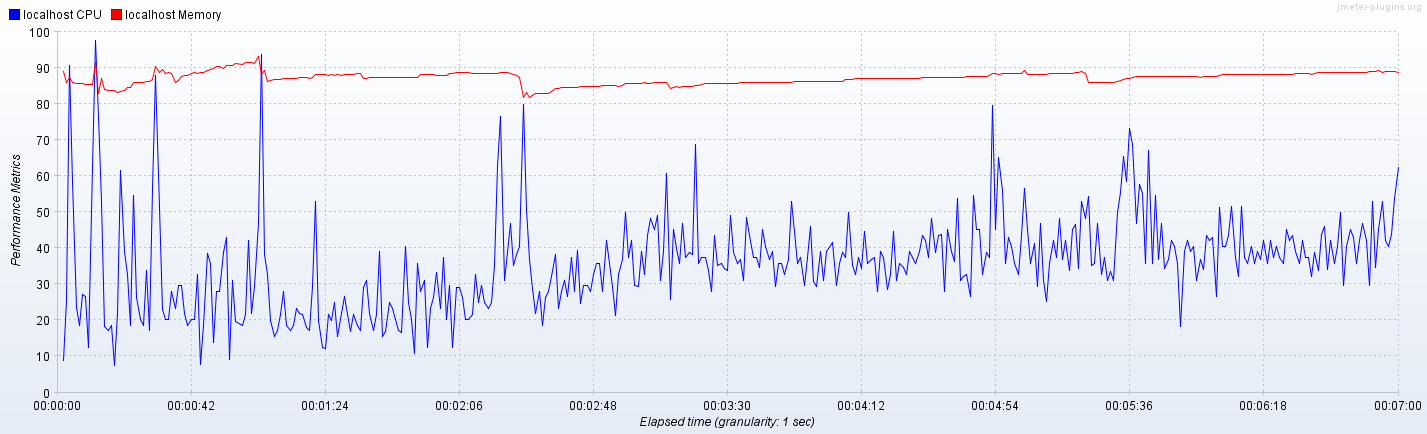
**100 posts**



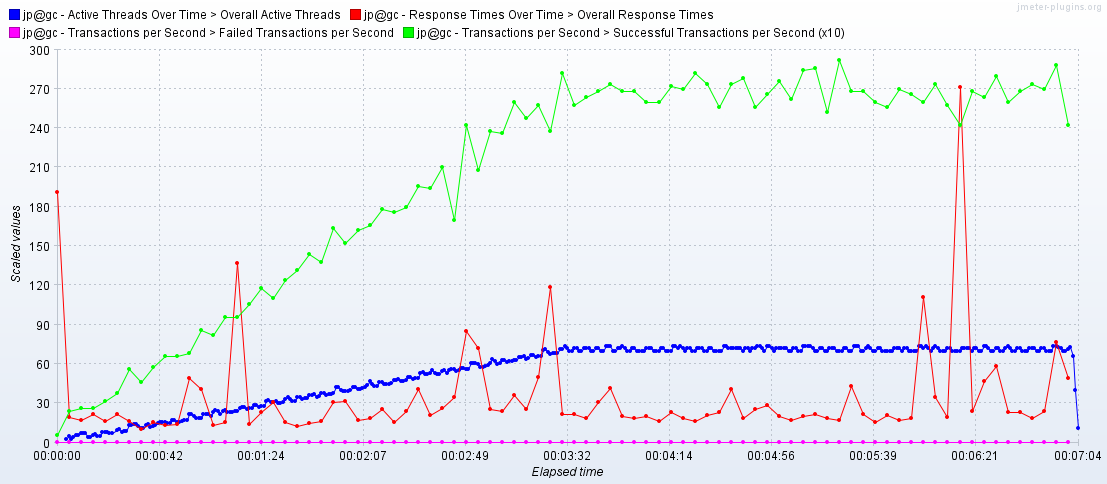


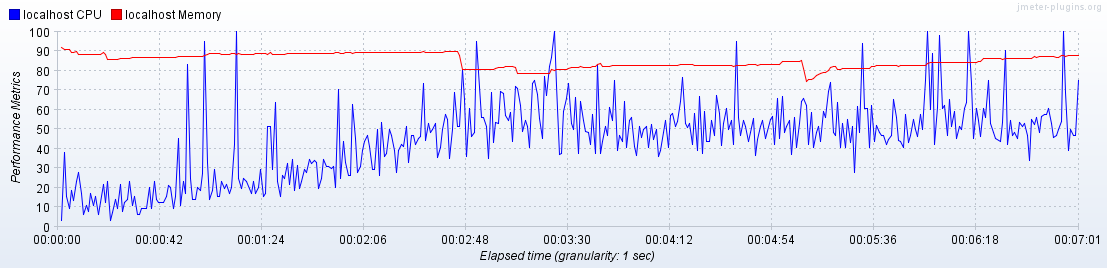
**1000 posts**



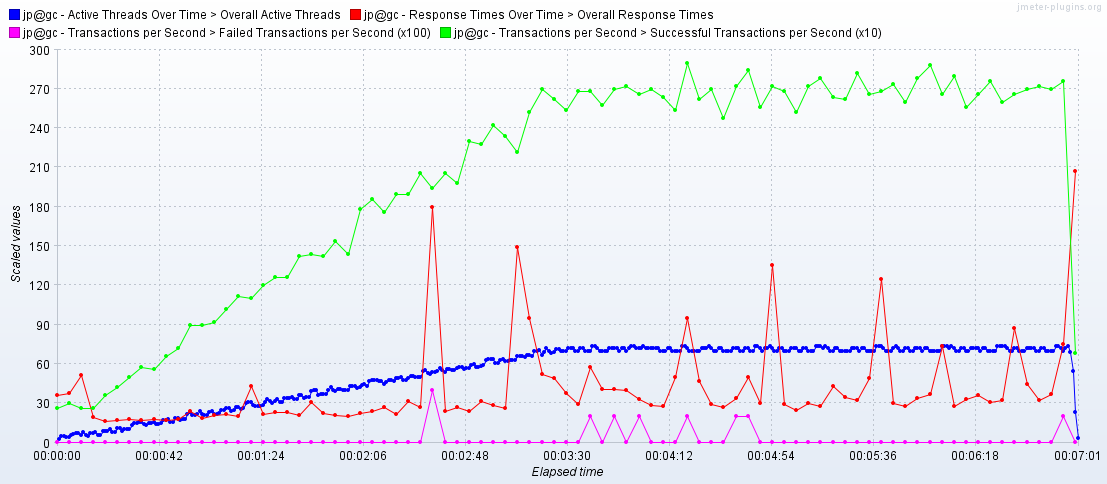


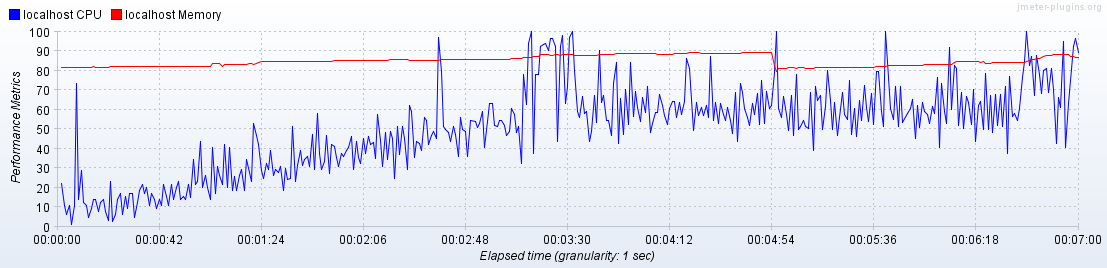
**2000 posts**



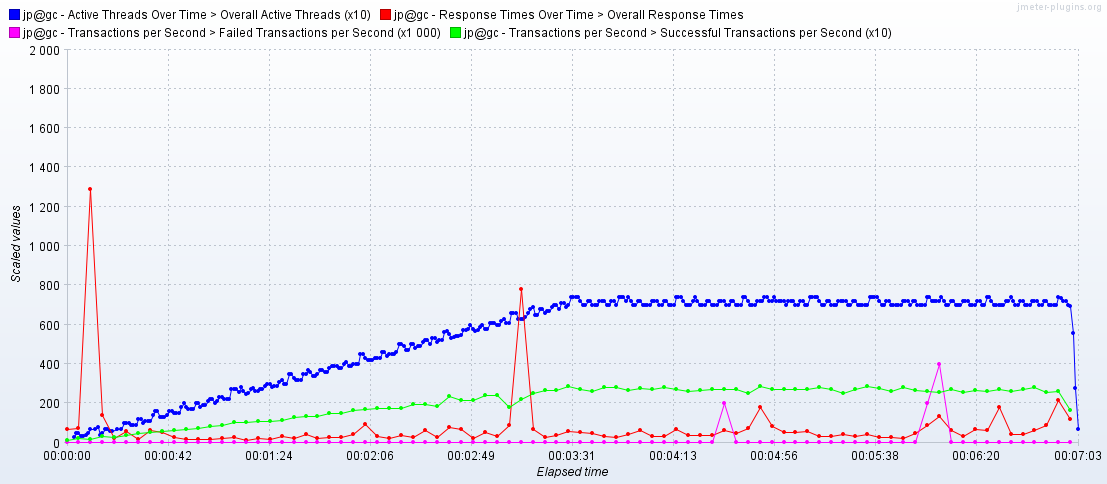


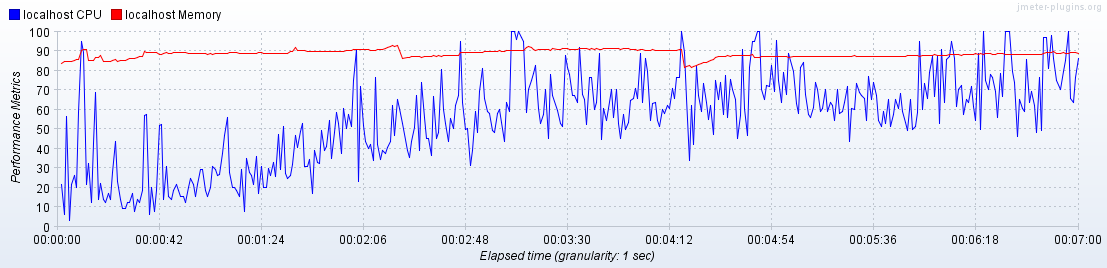
**5000 posts**



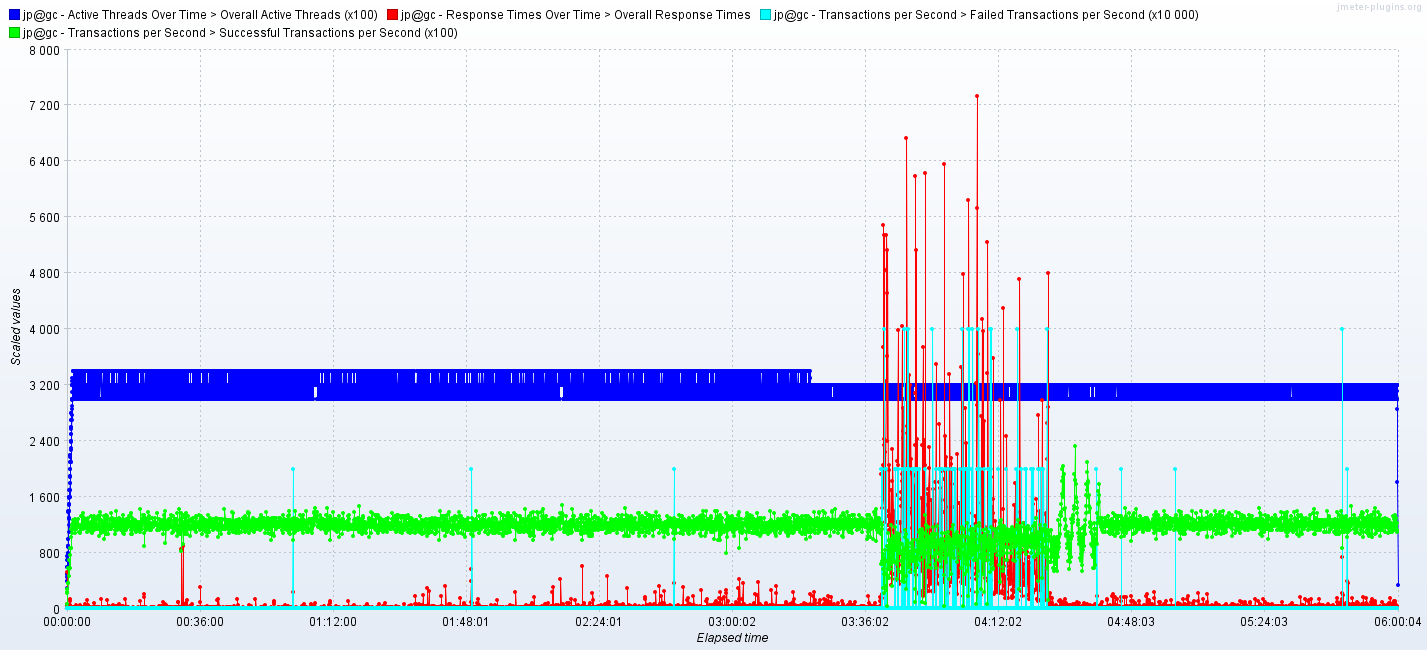


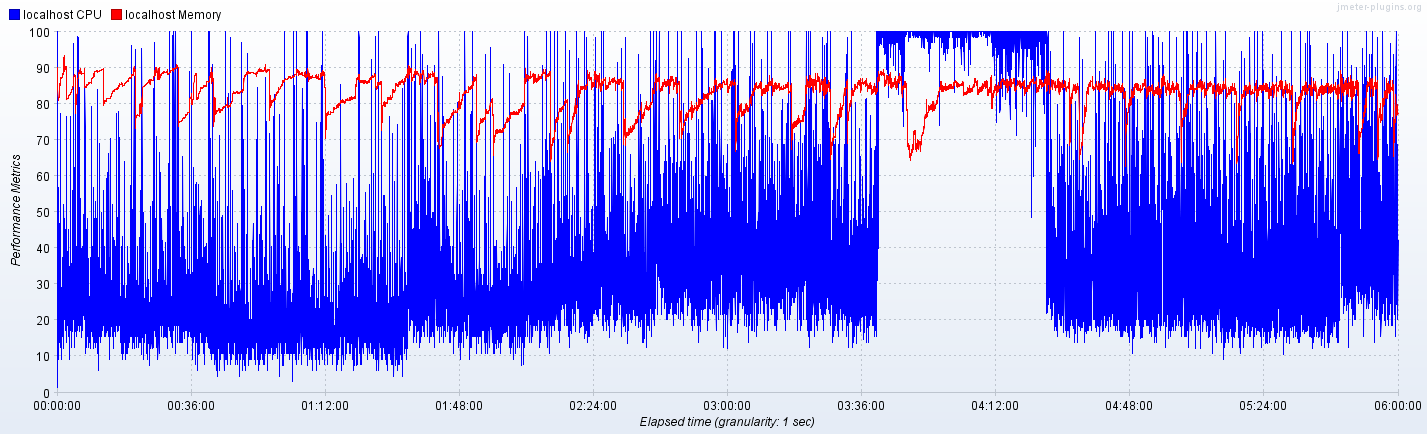
**2000 mixed posts**





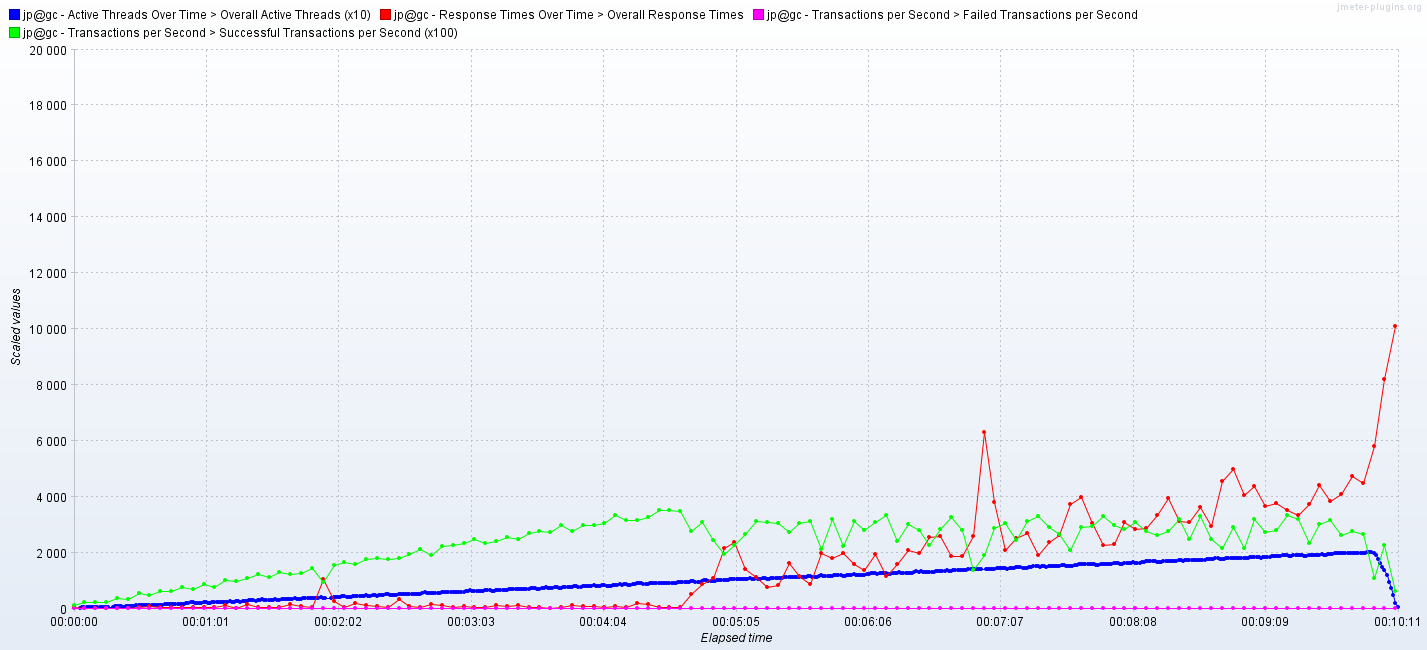
**Longevity test**

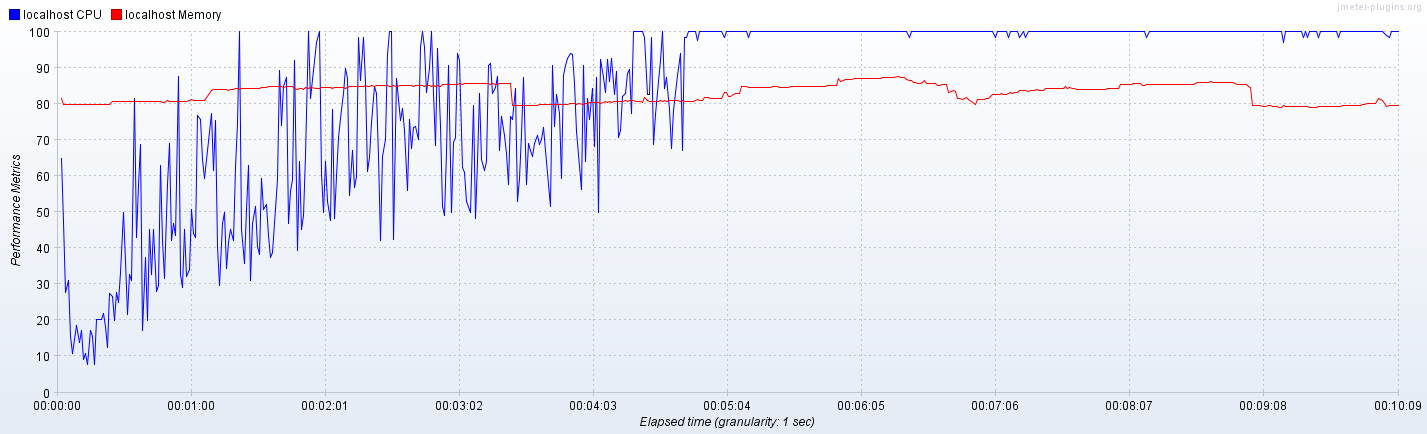




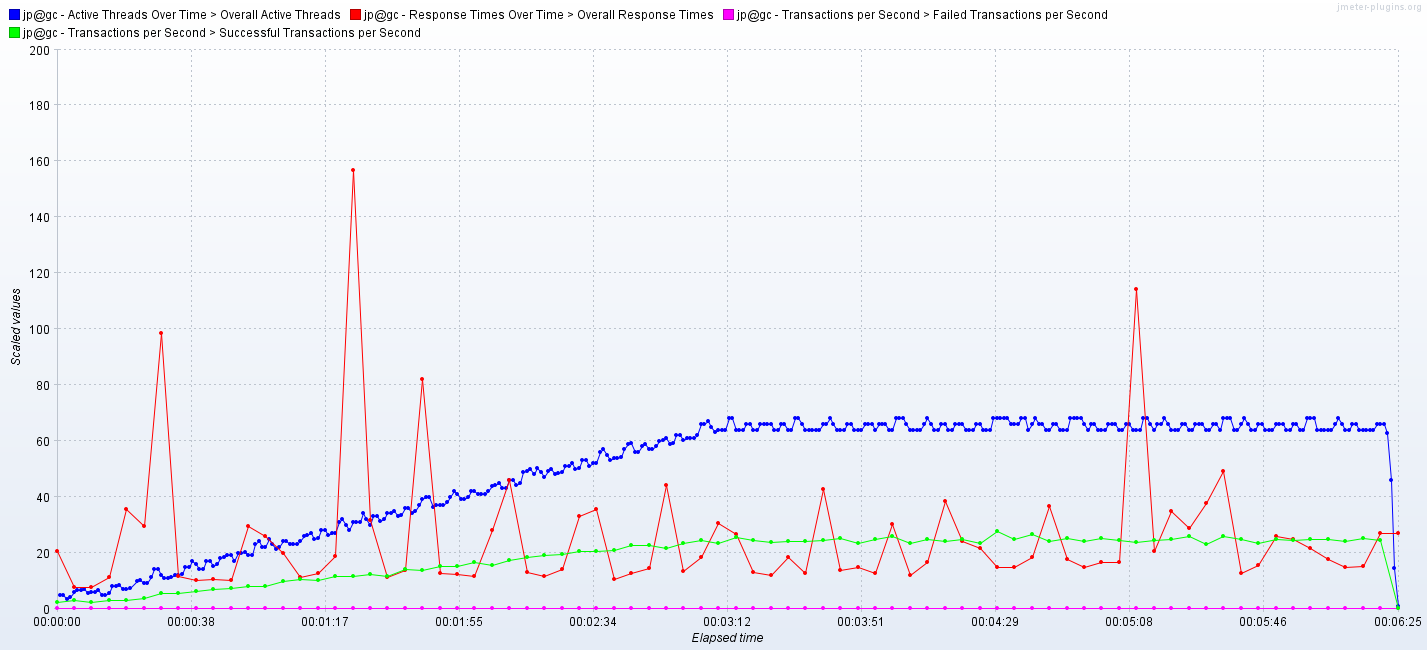
**Configuration test (DB data source)**

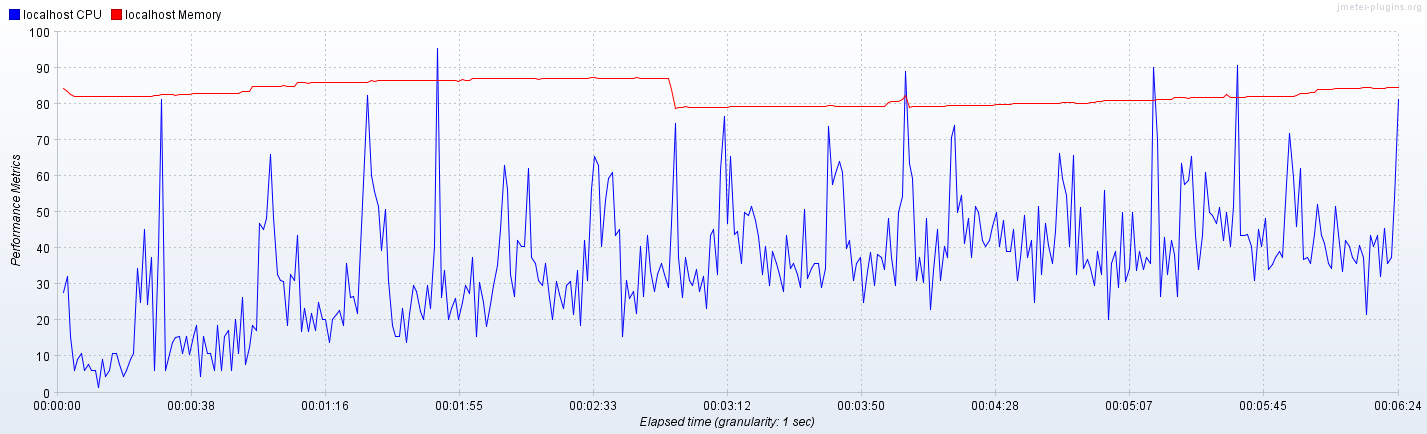
**Capacity test**





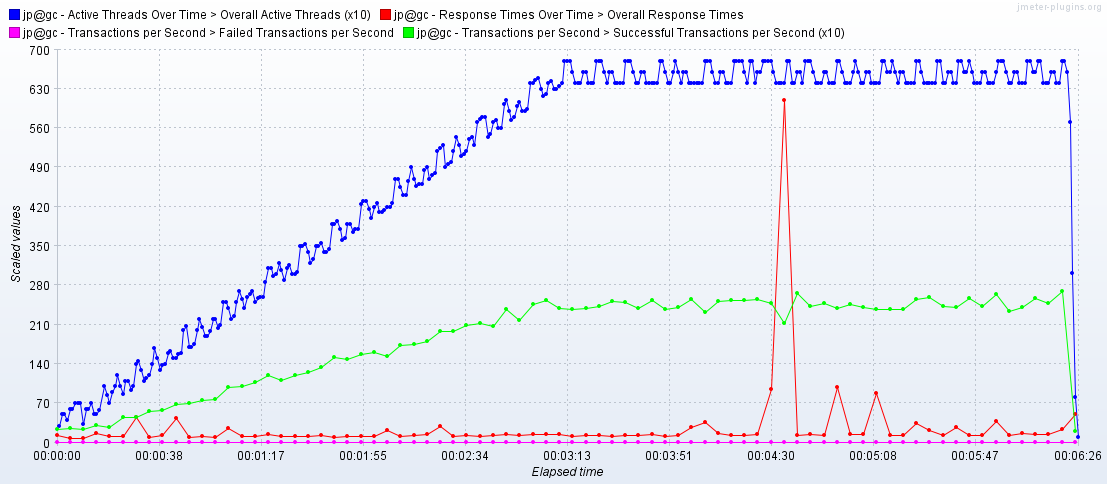
**Regular load test**

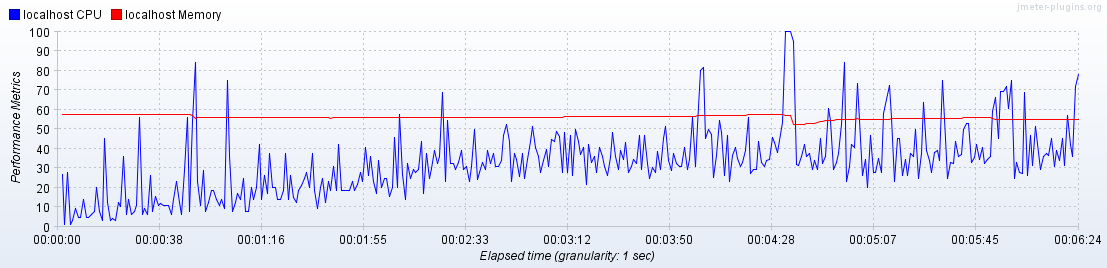




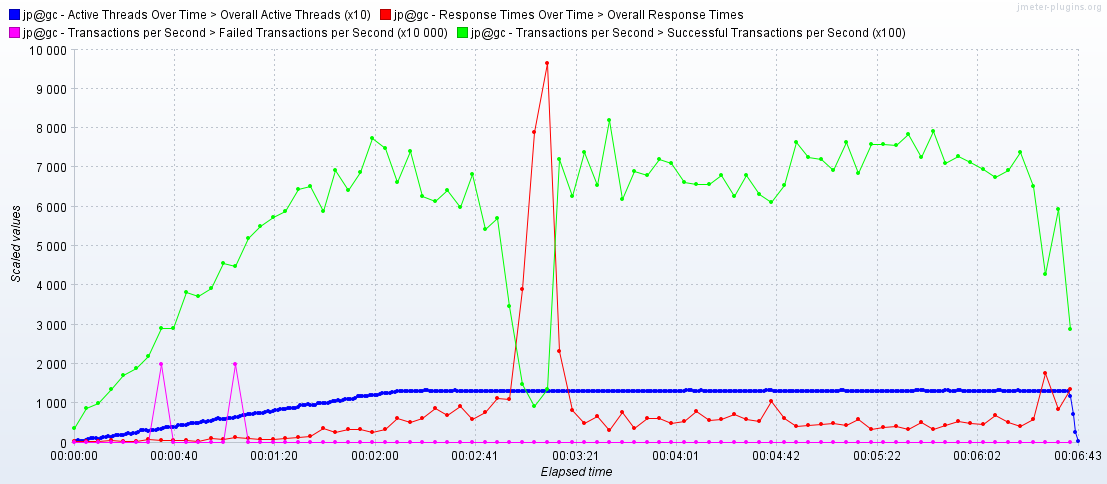
**Scalability test**

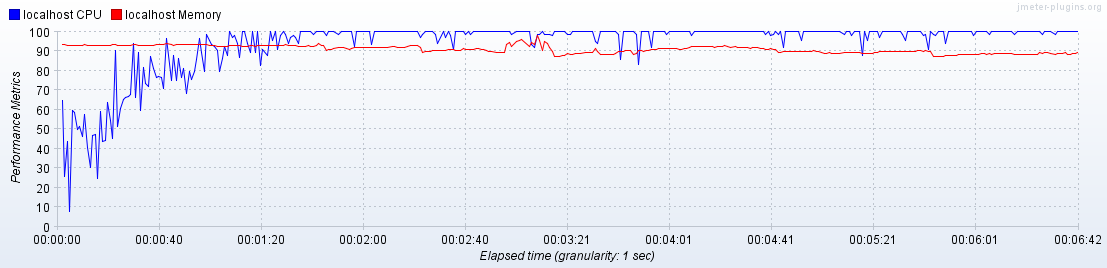
**1 CPU and 6144 MB RAM**





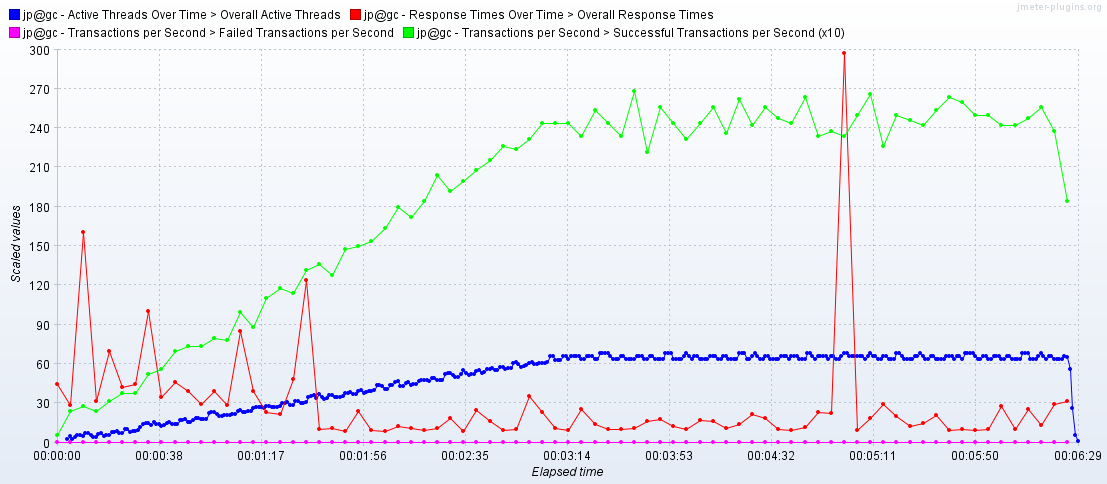
**Stress test**

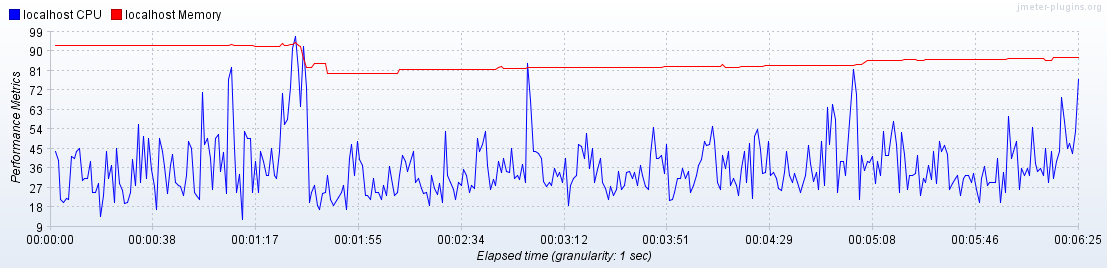




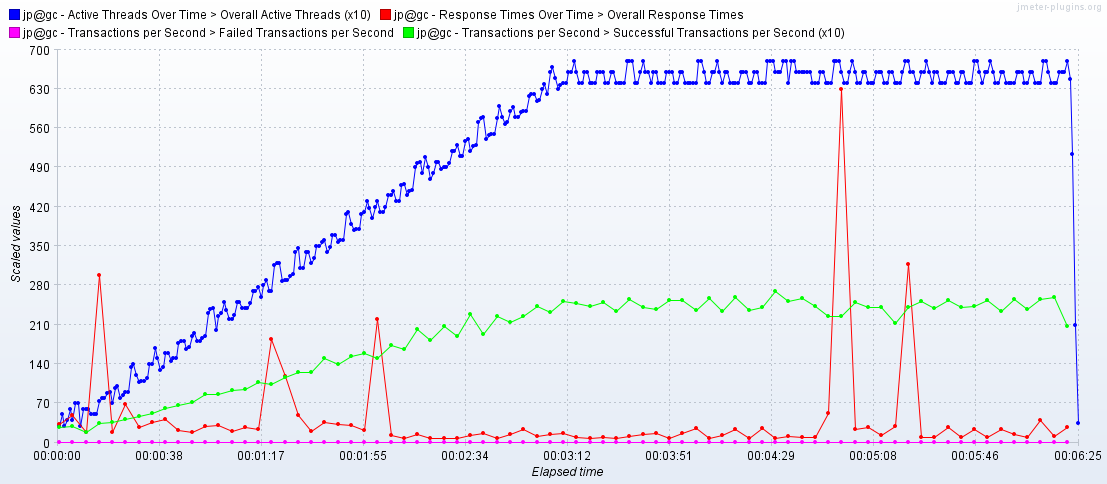
**Volume test**

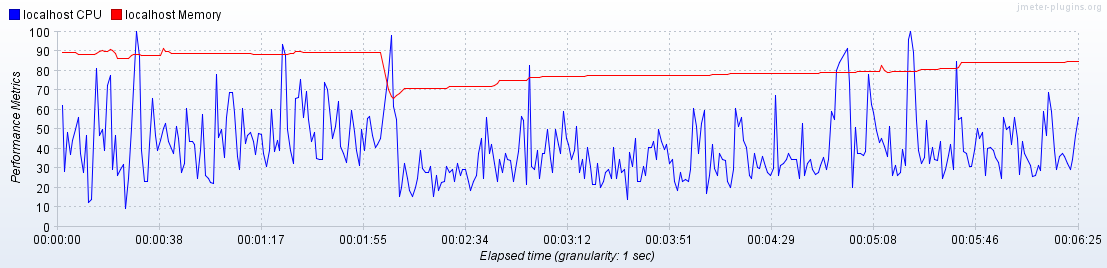
**2000 pure posts**





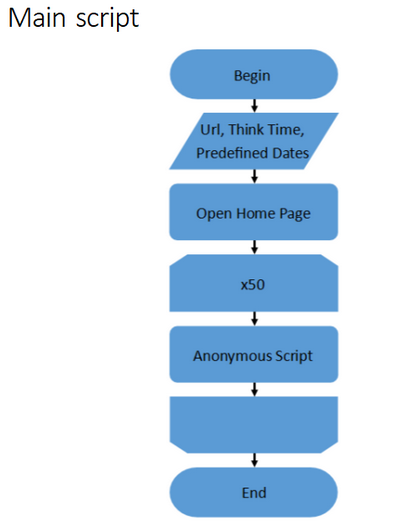
**2000 mixed posts**

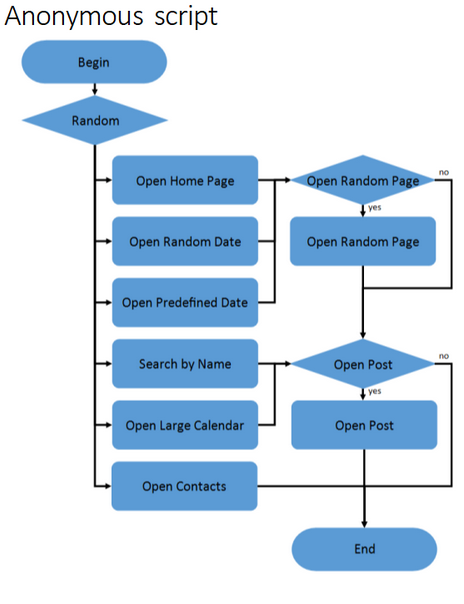


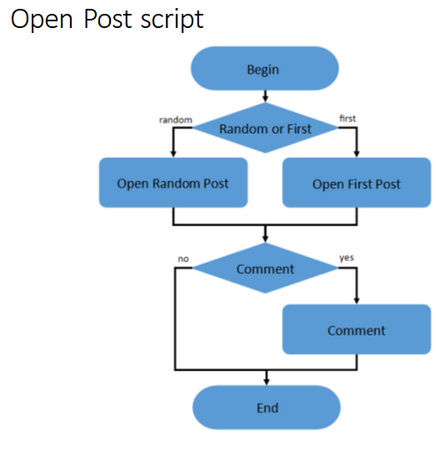


Scenarios:

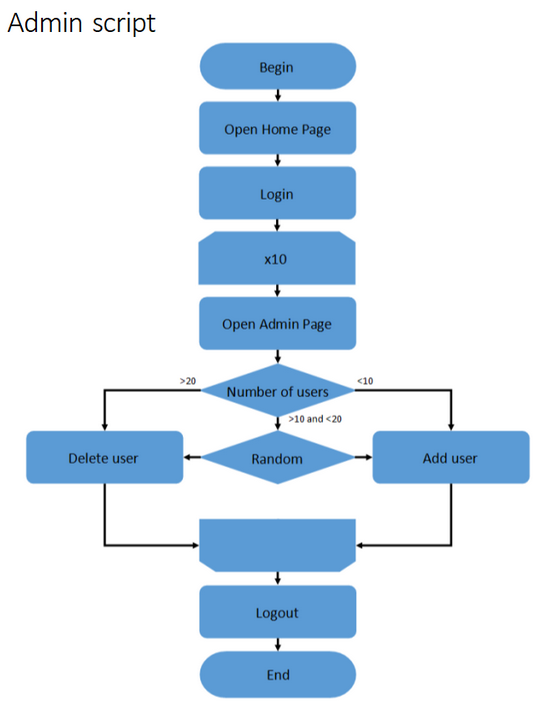
**Anonymous scenario:**



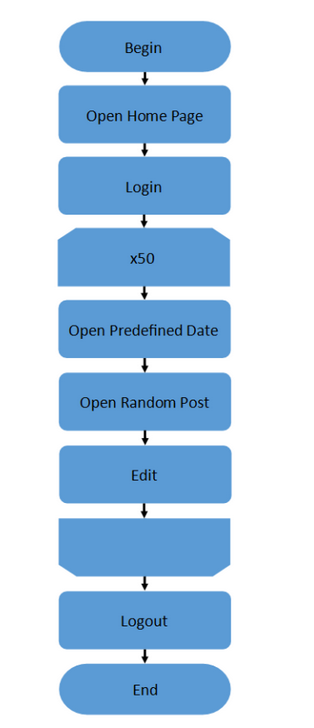




**Admin scenario:**



**Editor scenario:**



**Create test data(posts) scenario:**

