WEB APPLICATIONS PERFORMANCE TEST ENVIRONMENT PREPARATION

INTRODUCTION

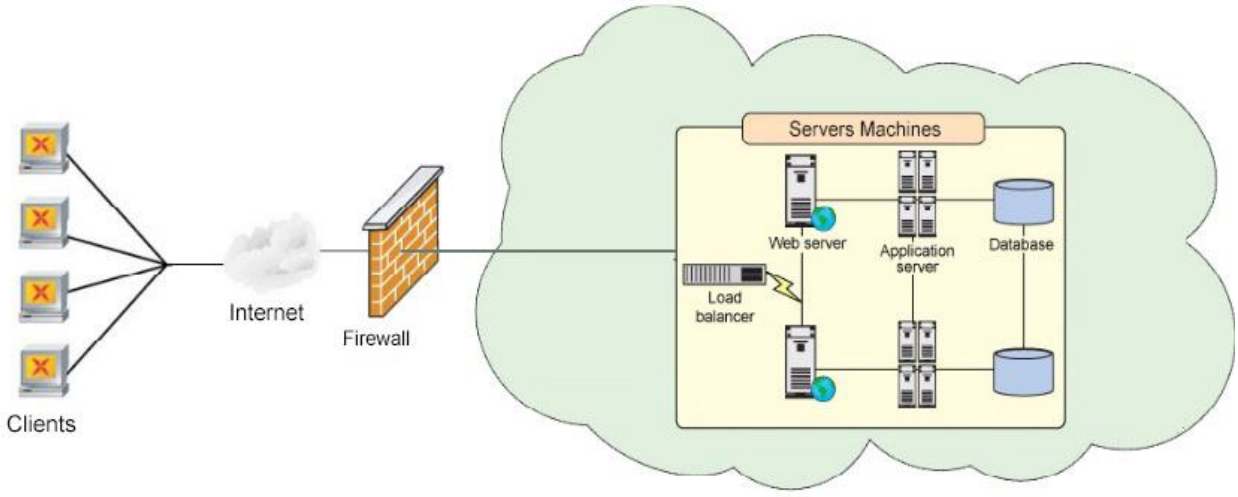
Performance testing is a complex activity where dozens of factors contribute to its success and effective usage of all those factors is necessary to get the accurate performance testing results. Executing the performance test on production similar system and availability of all the required resources to successfully run the test is very essential. Creating the exact replica of the production system is undoubtedly the most important part of the whole testing activity as even the minor differences between the test environment and production system can completely invalidate the test results. Any application performance test results immensely depend upon the test environment configurations.

Therefore, in order to accurately prepare the test environment, the whole application infrastructure should be thoroughly analyzed. Network, development and performance testing teams mutually devise the required test environment based on the application infrastructure. All the differences between the test and production system are completely analyzed and different methods are applied to overcome their impacts on the test results. Moreover, the required resources needed to successfully execute the performance test are also analyzed by the performance testing team to make sure that they conduct application performance testing perfectly.

In this paper, we will discuss the challenges faced while the preparation of accurate test environment, tips for preparing accurate test environment, different test environment options, test environment validation checklist and some troubleshooting techniques.

CHALLENGES IN PERFORMANCE TEST ENVIRONMENT PREPARATION

Performance testing is conducted to check the behavior of the application under test (AUT) on different load conditions. Performance test results are always dependent on the test environment. In an ideal situation, the performance test environment should be an exact replica of the AUT production environment. But in reality it's not so simple to achieve this challenging goal. Setting up the performance testing environment is usually a challenging task which could take weeks or even months to complete it due to the following reasons:



SERVERS INFRASTRUCTURE

Replicating the servers' content and their architecture is the most demanding task in performance test preparation process due the cost and complexity attached with these. Replicating a number of physical servers at each application tier in test environment is a real challenge.

NETWORK INFRASTRUCTURE

Deploying the testing servers on production servers' locations with their bandwidth and connectivity is always a tough task. Preparing the servers network infrastructure is although possible with some efforts but deploying the test servers on production servers' locations is a hard thing to do.

NUMBER OF APPLICATION TIERS

It's not all about replicating all the production servers in test environment but a number of application tiers also influence the test results. Test environment should have the exact number of application tiers as the production environment to achieve the accurate results which is also a challenge.

DATABASE SIZE

Database size also impacts the test results. Database server performance plays a vital role in application performance and the test environment database size should be the same as its being used in the production environment. A database with different size will not be able to generate accurate test results.

LOAD INJECTION FROM DIFFERENT GEOGRAPHICAL LOCATIONS

As internet applications are accessed from different geographical locations, simulating users' locations is also very important to achieve proper test results. Although it depends upon the testing tool (i.e. whether it provides the facility to simulate users from different locations or not) but it's not very common in all the available performance testing tools.

IP SPOOFING IMPLEMENTATION

All performance testers experience this many times when they are unable to insert load on AUT from one machine with the same IP. There could be a lot of reasons for this technique such as:

1. Due to security purposes AUT doesn't allow multiple user requests from same IP. It only allows a certain number of user requests and all the subsequent requests from the same IP are blocked.
2. In few cases, load balancing is implemented based on the IP addresses. Users are distributed on different servers based on their IP addresses and if all incoming requests are from the same IP address, they will be directed to a single server and hence load balancing will never be observed.

Users' requests are delivered from different IP addresses in such situation by implementing the IP spoofing. IP spoofing is an advance technique and all the available performance testing tools don't support it. So before selecting the testing tool, it's important to analyze the tool's features and capabilities thoroughly.

TIPS FOR ACCURATE PERFORMANCE TEST ENVIRONMENT PREPARATION

Although there are various challenges (as discussed above) in accurately preparation of the performance test environment but some of these challenges can be overcome by following below mentioned practices.

COMPLETE KNOWLEDGE OF AUT PRODUCTION AND TEST ENVIRONMENT

Complete understanding of the AUT production environment is one of the most important and fundamental responsibilities of the performance testing engineer(s). The details of AUT production environment should be completely documented and understood in the initial stage of the performance testing. Performance testing engineer must know the AUT architecture details and ensure that the exact architecture is being implemented in the test environment. Test result always depends on the test environment and having a test environment different from production environment can result in a total waste of lot of time, effort and cost. So before starting the performance testing activity, performance test engineers must be aware of the AUT details like its servers machines details (load balancing, number of CPUs, CPU speed, RAM, disk, NIC capacity and network bandwidth etc.), load balancing policy and details of other system components.

TEST ENVIRONMENT ISOLATION

It's highly recommended that no other activity should be carried out on the performance test environment during the test execution. Performance test results can greatly vary and it's always difficult to analyze and reproduce performance bottlenecks in a test environment where other users are also interacting with the system. Moreover, application servers' performance is always affected under heavy load and it might not be able for real application users to complete their tasks successfully during the performance test execution.

NETWORK ISOLATION

Network bandwidth always plays a vital role in achieving accurate performance test results. You must ensure the availability of sufficient network bandwidth for your test to simulate user requests appropriately. User requests will start producing the timeout errors in case of lower network bandwidth and test results will not be valid. So in order to provide the maximum network bandwidth to your test environment, one solution is that you should isolate your test network from other users.

LOAD INJECTORS REQUIREMENTS

Performance testing is no longer done manually and today different tools are being used to produce the required load on the AUT. Almost every good performance testing tool provides a separate utility called "load injector" that could be installed on any machine and the test machine can be connected to it for creating the required amount of load. One thread or process is created for every virtual user from the load injector to insert load on application servers. Load Injector machines should have sufficient hardware resources to support the running users.

The amount of load that can be generated from one load injector depends on various factors like machine resources (RAM, CPU, Disk), network bandwidth, script complexity and think time etc. For AgileLoad Injector CPU and Memory requirements, check out the following link:

http://www.agileload.com/learning-center/test-setup-run/advanced-tutorials/heavy-load-test/how-to-setup-agileload-injectors

TEST DATA GENERATORS

Database reading, writing, deletion and updating are the most performance intrusive actions in an application. Moreover, a number of database records always have a great impact on performance test results. Reading a record from 1000 rows will be much faster than reading it from 10,000 database records. It's very likely that an application tested on lower database records might fail in the production environment. Therefore, it's one of the responsibilities of the performance engineers to make sure that the test environment system has an equal number of test records. But ironically in most cases, test bed data is less than the production data and the performance test engineers or DBAs have to generate the required test data to accurately test the system. Various tools are also available in markets that have the capability of generating test data. Database dumps of the production system can also be taken to generate the required amount of test data.

PROXY SERVERS REMOVAL FROM NETWORK PATH

Having a proxy server between the client and the web server can affect performance results. In case of having a proxy server in middle of client and web server, the proxy will serve the client with data in cache instead of sending requests to web server, which results in lower AUT response time than actual. The issue can be resolved either by bringing the web server in an isolated environment or by hitting directly to the web server by editing HOSTS file by including server IP address.

COMPLETE SERVERS ACCESS

Complete servers access during the test helps in identifying all server resources and bottlenecks root-causes. Moreover, with complete servers' access performance engineers will be in a better position to analyze the servers' performance monitors.

SIMULATE CLIENTS CLOSER TO WEB SERVER

Latency can be one of the major factors in application response time. Closer users requesting from less distance will receive less response time as compared to the users sitting on long distance. Moreover, less network issues will occur on simulating closer user requests.

WARM UP THE WEB SERVER BEFORE TESTING

It's very common that you will receive some additional response time in case the web server is not warmed up before starting the testing activity. So it's always advised to first manually run the performance testing scenarios to verify the scenarios functionality as well as to ensure that the web server is warmed up.

PERFORMANCE TEST ENVIRONMENT SETUP OPTIONS

Performance test environment should be exactly similar to the production environment to get 100%accurate performance results but as discussed earlier but it is not an easy thing to do. In the industry, various alternative strategies are practiced to setup the performance test environment as mentioned below:

- Performance Testing on Production Environment
- Performance Testing on Scaled Environment
- Performance Testing over the Cloud
- Performance Testing with Service Virtualization

PERFORMANCE TESTING ON PRODUCTION ENVIRONMENT

Although it's very risky and there are a lot of concerns involved in testing the AUT performance in the production environment but still it's the most commonly used performance test environment option. Ideally performance test environment should be an exact replica of the production system but it's not always possible due to its associated cost, time and challenges. Moreover, Performance testing on live application is performed to validate the test environment results as well. Following are some of the advantages and disadvantages of performance testing in production environment.

Advantages

- No need to reproduce the production site data set
- It helps in validating performance test results performed on test environment
- It reduces test infrastructure cost and time
- Application recovery process and its complexities are well known

Disadvantages

- Real application users will receive slower application and errors
- Difficult to identify the bottleneck root cause in presence of real application users
- Real users access might have to be blocked to properly achieve the performance test results
- In case of generating lots of data on production database, database may become very slow even after the test

PERFORMANCE TESTING ON SCALED ENVIRONMENT

As we know that the performance engineers have to face a lot of challenges while preparing the exact performance test environment as the production environment, therefore many times performance testing is conducted on smaller environments as compared to the complete production system environment. These scaled environment results are then extrapolated to reach out on a conclusion. Following are some of the advantages and disadvantages of this technique,

Advantages

- Cost effective and results can be mapped by using extrapolation techniques
- Easy to setup as it requires less infrastructure
- Easy to identify application bottlenecks and tune it on the scaled environment

Disadvantages

- It's difficult to find out performance issues past the scaled environment
- Application tolerance and capacity is reduced on scaled environment and more performance issues are revealed in production

PERFORMANCE TESTING OVER THE CLOUD

Cloud computing is becoming more and more famous and mature with the time and its usage has been increased exponentially. Performance engineers setup the copy of the production system in cloud and deploy load injectors on different geographical locations over the cloud to effectively perform the load test.

Advantages

- Cloud testing provides the flexibility of deploying the production system on discrete environment to conveniently test the application
- It's extremely simple to fix the defects and quickly configure the changes
- It reduces the test cost due to its convenient rental models
- It provides greater test control to simulate required user load and to identify and simulate the bottlenecks

Disadvantages

- Security and privacy of data is the biggest concern in cloud computing
- Cloud computing works on-line and completely depends on the network connection speed
- Complete dependency on Cloud Service Provider for quality of service
- Although cloud hosting is a lot cheaper in long run but its initial cost is usually higher than the traditional technologies
- Although it's a short-term issue due to the emerging technology and it's difficult to upgrade it without losing the data

PERFORMANCE TESTING WITH SERVICE VIRTUALIZATION

Service virtualization is a mode used to simulate the AUT specific components behavior which is not accessible to test the application completely. Through service virtualization, complete AUT is not emulated rather only specific and required components are emulated to fulfill the requirements. For example, instead of virtualization of the complete database and its dataset, application interaction with the database is monitored and only the required database behavior is emulated. Currently this method is being widely used for conducting performance testing in agile development where application evolves in iterations. The method has its following pros and cons:

Advantages

- It helps in emulating realistic performance for dependent application
- It provides access to AUT constrained components at convenient times
- It helps in testing application performance with different parameter settings
- It helps in simulating extreme loads on third party components without much additional costs
- It can test the AUT components performance which are not yet completely developed

Disadvantages

- Virtualized components performance vary greatly from live AUT components which yield incorrect performance results
- There isn't any guarantee that AUT performing as per requirements when tested it in virtualized environment will also performed similarly in production
- You can't virtualize all the complex and secure systems
- You can't emulate the production data in virtualized environment which can greatly affect the performance of AUT

RE-STORING DATA FOR RE-TEST

The proper testing and optimization of database operations (i.e. read, write, delete and update) is extremely vital for any application performance. Therefore in order to properly test any web application database, having all the production data in it is the thumb of rule. Although lots of cost, effort and resources are required for migrating all the production data but there isn't any other option for application real testing and getting the optimized database paths.

It is obvious that during performance testing you have to run multiple tests on the application to completely test all of its business scenarios and to identify and validate all the performance issues. For every test, database needs to be restored to its initial state. Furthermore, this data restoring process should be as quick as possible to save the idle time.

PERFORMANCE TEST ENVIRONMENT CHECKLIST

We all know about the importance of having test environment similar to the production system. Once we have setup the performance test environment, we can get an initial idea of the test environment state by comparing it with production environment based on the following factors:

- **Number of Servers:** Number of physical and virtual servers
- **Load Balancing Strategy:** The type of load balancing mechanism is in use
- **Hardware Resources:** CPUs count and type, RAM capacity, Number and type of NICs
- **Software Resources:** Standard application build apart from components of the AUT
- **Application Components:** Application components description which needs to be deployed on the server
- **External Links:** Links to third party application and other internal system components

TEST ENVIRONMENT VALIDATION

Even if we have setup the performance test environment and compared it with production system based on the above mentioned points, test validation doesn't end. It's always recommended to first run a few basic level performance scenarios to validate the test environment before moving to the detailed tests. Following are some of the points which should be considered while validating the performance test environment setup.

- Verify that whether the test environment is configured correctly as per the production environment or not
- Make sure all no other program is running on load generation machines to fully utilize their resources
- First run simple user scenarios for web server layer testing only. Run the script without think time so that they should completely utilize the web server resources. In case if user scenarios don't utilize 100% resources then there is a problem with Load generators capacity or network bandwidth
- Secondly run scenarios which are only reading the data from the database without think time. This should utilize 100% web server processing and failing to do so means that there are some issues with the network or the load generator

- Then run the mixed scenario involving simple user scenarios and database operations and check web server resources utilization
- Verify servers (web, application and database) performance monitors are configured to access the test environment

TROUBLE-SHOOTING PERFORMANCE TEST ENVIRONMENT

Many differences are commonly found between the test and production systems after the proper validation of test system. A differently configured performance test environment will produce invalid results which can greatly mislead all the stakeholders and the application itself can fail in production. There can be a dozens of reasons why your test environment is not producing the required results and some of them are as follows:

- **Load Injectors overloaded:** Check the load injector machines resource utilization. Quite often load injector machines consume more processor and memory and are unable to simulate the required number of virtual users. Run a small and simple test first and check the system resources consumption on these before running the detailed test.
- **Insufficient network bandwidth:** Network bandwidth plays a vital role when you are conducting the performance test over the WAN. Test results can greatly differ on the basis of available bandwidth. So make sure that sufficient network bandwidth is available for starting the test. Moreover, you need two network interface cards (NICs) when web server and database server are on different layers, one NIC will be facing the clients and the other one will be used for database communication.
- **Improper test data**: Improper test data can also create various issues in performance testing. It's highly possible that a variable is not parameterized and same value is being submitted to the database for every user, which can lead to low processor activity due to artificial locking.

CONCLUSION

Preparing the exact similar performance test environment as the production system is an essential part of the performance testing activity. Any reliable application performance test results can only be achieved by having a test environment that is just like the production system. Complete application infrastructure understanding helps in building the accurate test environment but it's a challenging activity due to the factors like complete servers and network infrastructure information, replicating all the application tiers, Database sizing, load generation from different geographical locations and simulating users from unique IP addresses (IP Spoofing) etc.

Various other options are also available (other than the replicating of the exact production system) like testing on production system, scaled system, testing by using service virtualization and cloud computing etc. with each having its own advantages and disadvantages. You can compare your prepared test environment with production system by using the checklist like number of servers, load balancing strategy, hardware and software resources, application components and the third party external links. It's also recommended to run a base line test before starting the actual application testing scenarios to validate the test environment. Moreover, different strategies such as Load Injectors overload, Insufficient Network bandwidth and improper test data should be verified to troubleshoot the test environment issues to achieve the required test environment.