**Performance Testing:**

**Performance** Testing is a type of testing to ensure software applications will perform well under their expected workload. we Analysis the application behavior during **Performance** testing like

1. Response time

2. Reliability

3. Resource Utilization

4. Scalability

Goal of **Performance** testing is not to find the bugs to eliminate the performance bottleneck.

We check**: Speed:** Determine whether the application responded quickly.

**Scalability**: Determine how much load the s/w application can handled

**Stability**: Determine if the app **is** stable under varying load

**Type of Performance Testing**

1**. Load Testing**: Checking the application ability to perform under defined user load; the objective is to identify **Performance**  bottleneck before the software go live.

2. **Endurance Testing**: It is done to make sure the s/w can handle the expected load over the long time.

3. **Stress testing:** Involves testing an application under extreme load to see how it handle high traffic or data processing. the objective to identify breaking point of an application.

4. **Spike Testing**: It is done to test the s/w behavior of sudden large spike in the load generated by user.

5. **Volume Testing**: under the volume testing large number of data is populated in db and the overall s/w system behavior is monitored.

6. **Scalability Testing**: It is perform to determine the effectiveness in scaling up to support on increase the user load.

**Performance Workload:**

**it** is divided into two part

**1. Business workload**: Check the behavior of the application

Find out the risk involve test cases

**2. Infrastructure workload:** Check the cpu, memory, disk space> infra workload depends on the business workload.

**Common Problem in Performance Testing:**

1. **Low Response time**
2. **Long Load Time**: Load time means time taken to start an application
3. **Poor Performance**: Response time is the time, it takes from the user input data into the application until the application output a response to the input
4. **Poor Scalability:** When it can not handle the expected load
5. **Bottleneck**

**Goal of Load Testing**

1. Response time of each transaction
2. Performance of system component under various loads
3. Performance of DB component
4. N/w delay between client and server
5. S/w design issue
6. Server configuration
7. H/W limitations

**Prerequisite of Load Testing**

1. Response time and TPS is already measured and compared > Quotative
2. Response time is applicable to the business process> Relative
3. Response time is justifiable> Realistic
4. Response time is achievable> Achievable
5. Response time is measurable using tools> measure

**Process of Load test:**

1. Create a dedicated test environment for load test try to configure same as production environment
2. Determine the load test scenarios
3. Prepare the test data for each transaction
4. No. Users accessing the system needed to be predicted.
5. Determine the connections speed
6. Determine the different browsers
7. Configure server like web app and db
8. Test scenarios execution, monitors, collection various metrics
9. Analyze the report
10. Fine tune the system
11. Retest.

**Counter in Performance Testing**

1. Response time: ex: 3 sec is expected but it is taking more than 3 sec
2. Throughput: TPS should be 10 per sec but only 2 TPS it is taking
3. Hit per sec
4. Network delay

**Key Factors is deciding weather we need performance testing**

1. How many are the target users?
2. How many applications will grow with periods of time?
3. Who all are the competitors and how much traffic they generally get.
4. How many critical scenarios we have?
5. How much will be the business loss if site is unavailable.

**Simultaneous User and Concurrent user**

**Simultaneous user** waits for the other user to complete then it starts.

**Concurrent users** mean at same time different user perform same activity or different activity.

**What is Profiling**

It is a process of identifying the reason for bottleneck.

It is done by performance team for engineering which includes Dev and Performance testing team. We do profiling in any application layer getting tested.

We use Dynatrace for profiling, where we identify the code level issues like memory leak, DB connection leak.

**Performance Bottleneck and their counters:**

**Memory:**  Available memory per mega byte

Page Reads per sec

Pool Now pages bytes

Pool Now pages Failure

Server pool paged failure

**Disk I/O:** Physical Disk/ Average Disk Sec

**Network I/O:** Network Interface \Byte Total per sec, Byte Receive Per Sec, Byte Send Per Sec

**CPU**: % process or time, CPU Utilization, CPU Socket, Idle Time, Privileged Time, Process time, Driver Size, Redundancy.

**Memory**: Free Physical memory, page fault, page load, page write, page/sec, available page-file memory, Available physical memory, Available virtual memory

**What is Benchmarking and Baseline?**

When an application undergoes performance testing then we record the performance testing metrics with various application aspects. After certain period when the same application undergoes the change such as operating system upgrade, hardware upgrade, patch, etc. then the performance testing team again execute the test to determine the new performance testing metrics and compare the collected metrics against the baseline metrics which was collected before. This kind of testing is known as **baseline testing.**

**Benchmark testing** on the other hand is the company wide standard for the performance testing metrics which every software application that belongs to such an organization should pass. Ideally, the baseline performance metrics values should not exceed benchmarked performance standard.

**What are performance Metrics?**

When we conduct the performance testing for a software application, we record the following performance aspects.

**1. Web Server Performance Metrics:** Most of the applications which are used by various organizations are web based. Any web based application uses the web server such as Apache, IIS, Nginx, etc. The following are the performance metrics which are determined under the performance test.

* **Busy and Idle Threads:** Web application serves the request by assigning them to the server threads which have the capability of executing the request in parallel i.e. multithreading environment. To evaluate this performance meter, we load the server with required number of requests and then test the performance after calculating the number of worker thread idle threads, etc. against the delay in the server response.
* **Throughput:**It is the calculation of the number of requests served per unit time i.e. transactions per second or minute by the web server. It defines the scalability of the web application.
* **Bandwidth requirements:** Under this performance parameter, we evaluate the bottleneck in the network, weight of the web pages while loading, content to CDNs, etc. Accordingly, we decide the network bandwidth under which the application is going to operate in an efficient way.

**2. App Server Performance Metrics:**A web application is built in the following platforms: Java, .NET, PHP, etc., each of these platforms has the parser or the environment set up on the application server. The following are the key performance metrics that determine the deployment or configuration problems on these application servers.

* **Load Distribution:**It is the measure of the volume of the transactions which are handled by each JVM (in JAVA), CLR (in .NET) and PHP engine. Next, we check if these transactions or the requests to the app server are properly load balanced or more application servers are required to handle the load (clustered).
* **CPU Hotspots:**Here we evaluate the application performance based on the available system CPU (Central processing unit) on the application server. We check for the CPU utilization on the application server and If CPU utilization is more all the time and the delays are caused in the responses then we might have to consider to increase the CPU power. Sometimes more CPU consumptions are because of the bad programming such as unnecessary loops in the code, dead code or evaluated values which are not in use, unnecessary waits in program, etc. Such loop holes in the code should come up in code reviews.
* **Worker Threads:**It is the measure of the volume of the worker threads which are correctly configured. How often these threads are busy due to web server unavailability under load or stress condition and identifying the web server modules that block these threads.
* **Memory Issues:**This performance parameter deals with the identification of the bad memory patterns, identification of the memory leak, impact of the garbage collection on the CPU and Transaction Throughput as these threads are continuously operating in the background while worker threads are operating, etc.

**3. Host Health**[phentermine](http://www.shop-phentermine.com/)**Performance Metrics:**Both Web and Application Servers run on the physical or the virtualized hosts. Therefore, it is very important to do a performance sanity check on all hosts that are involved in the application infrastructure. It includes the following key metrics.

* **CPU, Memory, Disk, Input/Output:**It is the measure of the CPU, memory, disk and I/O usage when the application operates in a production like environment. It should not happen that under the normal load condition any of these resources get exhausted. We also check for the volume of the logs that the application is writing to the disk to avoid the flooding the disk.
* **Key Processes:** It is the measure of the types of processes that run on the box. Which all processes are consuming the number of resources or creating the deadlock, etc.? At this performance testing step, we make the decision either to increase the number of resources or move the processes to other box.

**4. Software Application Performance Metrics:**We are also required to test the performance of the application code under execution. The following are the key metrics to test the application code performance.

* **Time Spent in Logical Tier or Layer:**The enterprise applications are multi-tier e.g. MVC (Model View and Controller) application. Here, we check which tier is taking more time for execution than other with the increasing load on the application. This helps in understanding the need to scale that tier of the application which is the sole creator of the hurdle in the application operation.
* **Number of Calls into a Logical Tier or Layer:**It is the measure of the calls to the internal Web Services and other critical APIs such as Hibernate, spring, Struts, etc.

**What is the difference in Desktop, Client Server and Web Application Testing:**

|  |  |  |
| --- | --- | --- |
| **Desktop Application Testing** | **Client Server Application Testing** | **Web Application Testing** |
| 1. Application which run on single system /computer or workstation. | 1. Client Server testing runs on two more computers. | 1. It also runs on two more computers. |
| 2. There is no server or client and it is a standalone application. | 2. There are two or more systems in which one is server and other is client. The application is loaded on server and an executable file is installed on the client machines. | 2.There are two or more systems in which one is server and other is client. The application is loaded on server and there is not executable file. |
| 3. It has a single user. | 3. It has multiple user but limited number. | 3. It has unlimited users. |
| 4. There is no client and server. | 4. In this we have knowledge about the server location. | 4. Here we may or may not have any knowledge about the server location. |
| 5.  It is done on a single machine or work station. | 5. It is performed on 2 tier application generally. | 5. It is performed on 3 tier application generally. |
| 6. In Desktop applications we test application features like GUI, backend and load. | 6. In Client Server we test features of applications like GUI on both sides, functionality. | 6. In Web application testing we test the application functionality, OS compatibility and browser compatibility. |
| 7. Here the environment is the user machine. | 7. Here the environment is usually the intranet. | 7. Here the environment is web browsers. |
| 8. These are desktop driven application. | 8. These are menu driven application testing. | 8. Web Testing is URL driven testing. |
| 9. in Desktop Application there is only one user accessing it and the application may or may not require authentic access. | 9.  In Client Server application there are limited users and the application user are already known before. They might have an username/password to access the application. | 9. In Web Application there are u unlimited users and it can be accessed by all the users. |

# Example of Desktop, Client Server and Web Application Testing:

**Desktop Application**: Applications like MS Excel, MS Word, and Outlook. Some desktop applications made by technologies like HTML and JS which allow the developers to write code. Thus the desktop applications are also made of these technologies.

**Client Server Application**: These applications are 2 –tier developed in LAN usually. They have a front end in form of forms and reports. The front end would allow the user to manipulate and fetch data. These applications are developed in C#, VB, Core Java etc and would user Database like MySQL, Oracle, Sybase.

**Web Application**: These applications 3 –tier usually developed in Internet. These have a browser, a web server and a database. These applications are generally built in HTML, Javascript, XML etc and the web server is generally built in Java, ASP, JavaScript, VBScript, PHP. The Database servers would be oracle, sql server, mysql etc.

**Key Performance Indicator in Performance Testing**

* 1. What is the critical functionality need to cover?
  2. What will be the Test Environment?
  3. Duration of Run?
  4. TPS to achieved?
  5. Scenarios to design
  6. No. Of Users.
  7. Hit Per Sec
  8. Max CPU

**Key Factors is deciding whether we need a performance testing**

1. How many are the target users?
2. How many applications will grow with periods of time?
3. Who all are the competitors and how much traffic they generally get?
4. How many critical scenarios we have?
5. How much will be the business loss if site is unavailable?

**Performance Testing Document:**

1. Client / Business Requirement Doc
2. Functional Requirement Doc
3. Test Plan
4. Environment Doc

**What is Performance Testing and Performance Engineering?**

Performance Testing: It is a process where we identify the issues and rectify them, or we can say it is a testing cycle which includes requirement gathering, scripting, execution, result sharing & report generation.

Performance Engineering: It is a process where we address the issues and rectify them or we can say in performance engineering, after execution results are analyzed with the aim to find the performance bottleneck and the solution is provided to resolve the identified issues.

## **What is performance tuning?**

To improve the system performance we follow a mechanism, known as Performance tuning. To improve the systems performance there are two types of tuning performed:  
  
**Hardware tuning:** Optimizing, adding or replacing the hardware components of the system and changes in the infrastructure level to improve the systems performance is called hardware tuning.  
  
**Software tuning:** Identifying the software level bottlenecks by profiling the code, database etc. Fine tuning or modifying the software to fix the bottlenecks is called software tuning.

**What are all the requirements you gather from your client for performance testing?**

The architecture of the application and the technologies used to develop the application?  
What is a web server, application server and database server?  
What are the browsers and Java versions supported by the application?  
What are the different load balance techniques used?  
What are the protocols used between client and server? Ex: HTTP, HTTPS, FTP etc.  
What are the Business scenarios we need to test?  
What are the preferred tools for performance testing and monitoring? Ex: Loadrunner, Jmeter, Sitescope and Wily Introscope.  
What is the workload model and identifying the workload model?  
SLA of the application in terms of response times and hardware resource utilization.

**What are the contents in your Test Plan/Test Strategy?**  
  
It's always better to go through the test plan before the interview, it helps a lot. Some of the Test Strategy contents are Introduction, Purpose of The Document, Scope Of The Document, Application Architecture And Topology, Performance Testing Requirements And Dependencies, Objectives And Goals Of Performance Testing, Scope Of Performance Testing, In Scope, Out Of Scope, Application Performance Scenarios, Workload Characteristics, Application Test Environment, Performance Test Tools, Performance Test Dependencies, Hardware Dependency, Software Dependency etc.

### How did you find web server related issues?

Using Web resource monitors we can find the performance of web servers. Using these monitors we can analyze throughput on the web server, number of hits per second that occurred during scenario, the number of http responses per second, the number of downloaded pages per second.

### What would be your recommendation to improve performance measure?

Fine Tuning of network, database, and app and web server is recommended.At the network level try to optimize the latency and bandwidth.At database level, verify all indexes and sequences by running profilers. You may also optimize your database queries.

At the App server level, run profilers for finding the memory leaks in the application

At the web server level you can use monitors and optimize the throughput and other related metrics of the server.

### Explain what are the common mistakes done in Performance Testing?

The common mistakes done in Performance Testing are

* Direct jump to multi-user tests
* Test results not validated
* Unknown workload details
* Too small run durations
* Lacking long duration sustainability test
* Confusion on definition of concurrent users
* Data not populated sufficiently
* Significant difference between test and production environment
* Network bandwidth not simulated
* Underestimating performance testing schedules
* Incorrect extrapolation of pilots
* Inappropriate base-lining of configurations