

Abstract

Performance testing is the process of determining the speed or effectiveness of a computer, network, software program or device. Qualitative attributes such as reliability, scalability and interoperability may also be evaluated. Performance testing is often done in conjunction with stress testing.

The goal of performance testing is not to find bugs, but to eliminate bottlenecks and establish a baseline for future regression testing. To conduct performance testing is to engage in a carefully controlled process of measurement and analysis. Ideally, the software under test is already stable enough so that this process can proceed smoothly.

The process can compare two or more devices or programs in terms of parameters such as speed, data transfer rate, bandwidth, throughput, efficiency or reliability.

Determining the most suitable performance testing technique can prove to be quite challenging especially with limited testing tools and available resources. However, successfully identifying and implementing an appropriate performance testing technique can be equally rewarding leading to an application developed which meets the stringent requirement specifications and exceeds customer needs and expectations.

Effective performance testing can quickly identify the nature or location of a software-related performance problem.

The Problem

With the ever growing need for quick access to data, and with the large volumes of data at our disposal, the need for high performance systems and applications has become increasingly important.

Valuable time can be wasted due to poorly designed hardware layout or badly developed application. Slow data transfer rate may be inherent in hardware but can also result from software-related problems, such as:

- Too many applications running at the same time.
- A corrupted file in a Web browser.
- A security exploit.
- Heavy-handed antivirus software.

Performance testing of an application under development aims to thoroughly address a multitude of performance affecting factors to result in an enhanced user experience when using an application.

Various techniques, depending on the application type, project duration, available performance testing tools and resources i.e., budget and time constraints are taken into consideration when devising the most suitable performance testing strategy.

Core Performance Testing Activities

1. Identify Test Environment

2. Identify Performance Acceptance Criteria

3. Plan and Design Tests

4. Configure Test Environment

5. Implement Test Design

6. Execute Tests

7. Analyze, Report, and Retest

Understanding Insecure Communication Security Testing

Performance testing is a type of testing intended to determine the responsiveness, throughput, reliability, and/or scalability of a system under a given workload. Performance testing is commonly conducted to accomplish the following:

Understanding Insecure Communication Security Testing

Performance testing is a type of testing intended to determine the responsiveness, throughput, reliability, and/or scalability of a system under a given workload. Performance testing is commonly conducted to accomplish the following:

- Evaluate against performance criteria.
- Compare performance characteristics of multiple systems or system configurations.
- Find the source of performance problems.

Holistic overview of Performance Testing Performance, Load and Stress Testing

Performance testing determines speed, scalability and reliability which is mostly concerned with response times, throughout and resource utilization levels.

Load testing, a sub-category of performance testing is concerned with testing the application under workloads and load volumes anticipated during actual operations.

When the load placed on the system is raised beyond normal usage patterns, in order to test the system's response at unusually high or peak loads, it is known as stress testing. The load is usually so great that error conditions are the expected result, although no clear boundary exists when an activity ceases to be a load test and becomes a stress test.

Stress testing is aimed towards determining performance characteristics when the application is subject to conditions beyond those anticipated during actual operations. Stressful conditions may include limited memory, insufficient disk space, or server failure. Stress tests are designed to determine when and how an application will fail, and what indicators will give warning of an approaching failure.

Baselines

Baselines are used to determine whether performance is improving or declining and to find variations across different builds. Baselines can be created at the application, component or system level. It is important to note that baselines are metrics and can include a broad set of key performance indicators i.e., response times, processor capacity, memory usage, network usage etc.

Performance Requirements

Performance requirements (PR) are necessary for system design and development. If there are no written performance requirements, it just

means that they exist in heads of stakeholders, but nobody bothered to write them down and made sure that everybody agrees with them. Then PR will be input for performance testing (where they will be validated) as well as capacity is planning and production monitoring (SLA - Service Level Agreement).

For Performance Goals

These are criteria that the team wants to meet before the application release date, with some of the criteria being negotiable, thus performance goals are somewhat flexible in comparison to performance requirements. Another commonly used term which equates to performance requirements is performance thresholds which indicate maximum values for identified metrics.

Smoke Test

To check an application can perform under normal load an initial run of performance test takes place.

Stress Test

Stress testing tries to break the system under test by overwhelming its resources or by taking resources away from it (in which case it is sometimes called **negative testing**). The main purpose behind this madness is to make sure that the system fails and recovers gracefully -- this quality is known as **recoverability**.

Capacity Test

A capacity test is designed to determine your server's ultimate failure point. This is done to plan for future growth, such as an increased user base or increased volume of data. Capacity testing helps you to identify a scaling strategy in order to determine whether you should scale up or scale out.

Endurance Test

Endurance testing is a subset of load testing, focused on determining or validating the performance characteristics of the product under test when subjected to workload models and load volumes anticipated during production operations over an extended period of time. Metrics such as Mean Time between Failure (MTBF) and Mean Time to Failure (MTTF) are valuable outcomes of endurance testing.

Spike Test

Spike testing is a subset of stress testing, focused on determining or validating the performance characteristics of the product under test when subjected to workload models and load volumes that repeatedly increase beyond anticipated production operations for short periods of time.

Solution

Depending on the nature of the project, and the given times frames along with the availability of resources and test tools available, it is advisable to determine the most suitable performance testing strategy to adopt to successfully complete performance testing within time and budget constraints. The underlying philosophy in agile performance testing is to remain efficient by assigning different things to different teams whilst maintaining flexibility throughout the project. Whilst on the other hand, structured performance testing (CMMI approach) is frequently used when heavyweight or safety-critical applications are being developed, or when software is subject to regulatory standards. Not only is this more challenging, but it is also an extremely robust approach to adopt for projects.

Conclusion

Security testing has gained unparalleled significance over the years, and will continue to rise up the popularity chart given the frequent advancements that we have come to notice and appreciate in the world of technology. Moreover, the importance of data integrity and confidentiality have begun to play a significant role, as have the client demands increased in terms of having access to secure software which delivers without compromising sensitive information. Our aim is to perform thorough security testing to highlight shortcomings and weaknesses in applications to enable developers to build more reliable, and dependable applications in future.