## Software Testing Life Cycle

**Software Testing Life Cycle (STLC)** is a sequence of specific activities conducting during the testing process to ensure software quality goals are met. STLC involves both verification and validation activities. STLC consists of a series of activities carried out methodologically to help certify software product. (1)

**What is Entry and Exit criteria in STLC**

* **Entry Criteria**: Entry Criteria gives the prerequisite items that must be completed before testing can begin.
* **Exit Criteria**: Exit Criteria defines the items that must be completed before testing can be concluded.(1)

**STLC Phases:**

There are 6 major phases in every software testing life cycle model:

1. **Requirement Analysis**
2. **Test planning**
3. **Test Case Development**
4. **Environment Setup**
5. **Test Execution**
6. **Test closure**

A screenshot of a computer

Description automatically generated with low confidence

1. **Requirement Analysis**

During this phase of STLC, we are going to analyze and study the requirements. Requirement testing is also known as requirement analysis in which the test team studies the requirements from a testing point of view to identify testable requirements to understand requirements in detail. Requirements could be either functional or non-functional.

**Requirement analysis activities:**

* Identify type of tests to be performed
* Gather details about testing priorities and focus
* Prepare Requirement Traceability Matrix (RTM)
* Manually or Automation feasibility analysis

**Deliverables of requirement analysis:**

* Requirement Traceability Matrix (RTM)
* Manually or Automation feasibility report

1. **Testing Planning**

In this phase, we are going to test the plan strategy along with efforts and cost estimates for the project. Moreover, the resources, test environment, test limitations, and the testing schedule are also determined. The Test Plan gets prepared and finalized in the same phase.

**Main Activities in Testing Planning:**

* Preparation of test plan/strategy document for various types of testing
* Test tool selection
* Test effort estimation
* Resource planning and determining roles and responsibilities.
* Training requirement

**Deliverables of Test Planning:**

* [Test plan](https://www.guru99.com/what-everybody-ought-to-know-about-test-planing.html) /strategy document.
* [Effort estimation](https://www.guru99.com/an-expert-view-on-test-estimation.html) document

1. **Test Case Development**

This phase involves the creation, verification and rework of test case and test script after the test plan is finished. Initially, we are going to test data is identified then created and reviewed and then reworked based on the preconditions.

**Test Case Development Activities:**

* Create test cases and automation script if applicable
* Review and baseline test case and script
* Create test data if test environment is available

**Deliverables of Test Case Development:**

* Test cases/scripts
* Test data

1. **Test Environment Setup**

In this phase we are going to decide the software and hardware conditions under which a work product is testing. It is one of the critical aspects of the testing process and can be done in parallel with the Test Case Development Phase. Sometimes testing team may not be involved in this activity if the development team provides the test environment. Testing team is required to do a readiness check (smoke testing) of the given environment.

**Test Environment Setup Activities:**

* Understand the required architecture, environment set-up and prepare hardware and software requirement list for the Test Environment.
* Setup test Environment and test data.
* Perform smoke test on the build.

**Deliverables of Test Environment Setup:**

* Environment ready with test data set up
* Smoke Test Result

1. **Test Execution Phase**

In this phase is carried out by the testers in which testing of the software build is done based on test plans and test cases prepared. The process consists of test script execution, test script maintenance and bug reporting. If bugs are reported then it is reverted back to development team for correction and retesting will be performed.

**Test Execution Activities:**

* Execute tests as per plan
* Document test results, and log defects for failed cases
* Map defects to test cases in RTM
* Retest the[Defect](https://www.guru99.com/defect-management-process.html)fixes
* Track the defects to closure

**Deliverables of Test Execution:**

* Completed RTM with the execution status
* Test cases updated with results
* Defect reports

1. **Test Cycle Closure**

In this phase is completion of test execution which involves several activities like test completion reporting, collection of test completion matrices and test results. Testing team members meet, discuss and analyze testing artifacts to identify strategies that have to be implemented in future, taking lessons from current test cycle.

**Test Cycle Closure Activities:**

* Evaluate cycle completion criteria based on Time, Test coverage, Cost, Software, Critical Business Objectives and Quality
* Prepare test metrics based on the above parameters.
* Document the learning out of the project
* Prepare Test closure report
* Qualitative and quantitative reporting of quality of the work product to the customer.
* Test result analysis to find out the defect distribution by type and severity.

**Deliverables of Test Cycle Closure:**

* Test Closure report
* Test metrics

**STLC Phases along with Entry and Exit Criteria**

Graphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface, text, application

Description automatically generated

## Phases of Testing

**Following is the testing phase adopted by product firms:**

Graphical user interface, application, Word

Description automatically generated

**Pre-Alpha**: Software is a prototype. UI is complete. But not all features are completed. At this stage, software is not published

[**Alpha Testing**](https://www.geeksforgeeks.org/?p=294073) is a type of software testing performed to identify bugs before releasing the product to real users or to the public. Alpha Testing is one of the user acceptance tastings.

**Features:**

* Alpha testing involves both the white box and black box testing
* Alpha testing is performed by testers who are usually internal employees of the organization.
* Alpha testing is performed at developer’s site.
* Reliability and security testing are not checked in alpha testing.
* Alpha testing ensures the quality of the product before forwarding to beta testing.
* Alpha testing requires a testing environment or a lab.
* Alpha testing may require long execution cycle.
* Developers can immediately address the critical issues or fixes in alpha testing.
* In depth, functionality cannot be tested as software is still under development stage Sometimes developers and testers are dissatisfied with the results of alpha testing

[**Beta Testing**](https://www.geeksforgeeks.org/?p=294134) is performed by real users of the software application in a real environment. Beta testing is one of the types of User Acceptance Testing.

**Features:**

* Beta testing commonly uses black box testing.
* Beta testing is performed by clients who are not part of the organization.
* Beta testing is performed at end-user of the product.
* Reliability, security and robustness are checked during beta testing.
* Beta testing also concentrates on the quality of the product but collects users input on the product and ensures that the product is ready for real time users.
* Beta testing doesn’t require a testing environment or lab.
* Beta testing requires only a few weeks of execution.
* Most of the issues or feedback collected from beta testing will be implemented in future versions of the product

A screenshot of a computer

Description automatically generated with medium confidence

**Release Candidate (RC):** Based on the feedback of Beta Test, you make changes to the software and want to test out the bug fixes. At this stage,  you do not want to make radical changes in functionality but just check for bugs. RC is also put out to the public

**Release**: All works, software is released to the public.

# Software Testing

**Security Testing** is a type of [Software Testing](https://www.geeksforgeeks.org/software-testing-basics/) that uncovers vulnerabilities of the system and determines that the data and resources of the system are protected from possible intruders. It ensures that the software system and application are free from any threats or risks that can cause a loss. Security testing of any system is focused on finding all possible loopholes and weaknesses of the system which might result in the loss of information or repute of the organization.

## The goal of Security Testing The goal of security testing is to:

* To identify the threats in the system.
* To measure the potential vulnerabilities of the system.
* To help in detecting every possible security risk in the system.
* To help developers in fixing the security problems through coding.

## Principle of Security Testing Below are the six basic principles of security testing:

* Confidentiality
* Integrity
* Authentication
* Authorization
* Availability
* Non-repudiation

## Major Focus Areas in Security Testing:

* Network Security
* System Software Security
* Client-side Application Security
* Server-side Application Security

## Types of Security Testing:

1. Vulnerability Scanning:  
   Vulnerability scanning is performed with the help of automated software to scan a system to detect the known vulnerability patterns.
2. Security Scanning:  
   Security scanning is the identification of network and system weaknesses. Later on, it provides solutions for reducing these defects or risks. Security scanning can be carried out in both manual and automated ways.
3. Penetration Testing:  
   Penetration testing is the simulation of the attack from a malicious hacker. It includes an analysis of a particular system to examine for potential vulnerabilities from a malicious hacker that attempts to hack the system.
4. Risk Assessment:  
   In risk assessment testing security risks observed in the organization are analyzed. Risks are classified into three categories i.e. low, medium, and high. This testing endorses controls and measures to minimize the risk.
5. Security Auditing:  
   Security auditing is an internal inspection of applications and operating systems for security defects. An audit can also be carried out via line-by-line checking of code.
6. Ethical Hacking:  
   Ethical hacking is different from malicious hacking. The purpose of ethical hacking is to expose security flaws in the organization’s system.
7. Posture Assessment:  
   It combines security scanning, ethical hacking, and risk assessments to provide an overall security posture of an organization.

## Reference:

1.[STLC (Software Testing Life Cycle) Phases, Entry, Exit Criteria (guru99.com)](https://www.guru99.com/software-testing-life-cycle.html), <https://www.guru99.com/software-testing-life-cycle.html>

2. [An Introduction to Software Testing Life Cycle (STLC): Definition and Phases (sealights.io)](https://www.sealights.io/software-quality/an-introduction-to-software-testing-life-cycle-stlc-definition-and-phases/), <https://www.sealights.io/software-quality/an-introduction-to-software-testing-life-cycle-stlc-definition-and-phases/>

3. [Difference between Alpha and Beta Testing - GeeksforGeeks](https://www.geeksforgeeks.org/difference-between-alpha-and-beta-testing/), <https://www.geeksforgeeks.org/difference-between-alpha-and-beta-testing/>

4. [Alpha Testing Vs Beta Testing: What’s the Difference? (guru99.com)](https://www.guru99.com/alpha-beta-testing-demystified.html), <https://www.guru99.com/alpha-beta-testing-demystified.html>

5. [Software Testing | Security Testing - GeeksforGeeks](https://www.geeksforgeeks.org/software-testing-security-testing/?ref=lbp), https://www.geeksforgeeks.org/software-testing-security-testing/?ref=lbp

# What is Security Testing?

Security testing checks whether the software is vulnerable to cyber-attacks, and tests the impact of malicious or unexpected inputs on its operations. Security testing provides evidence that systems and information are safe and reliable, and that they do not accept unauthorized inputs.

Security testing is a type of non-functional testing. Unlike functional testing, which focuses on whether the software’s functions are working properly (“what” the software does), non-functional testing focuses on whether the application is designed and configured correctly (“how” it does it).

Security testing is structured around several key elements:

* **Assets—**things that need to be protected, such as software applications and computing infrastructure.
* **Threats and vulnerabilities**– activities that can cause damage to an asset, or weaknesses in one or more assets that can be exploited by attackers. Vulnerabilities can include unpatched operating systems or browsers, weak authentication, and the lack of basic security controls like firewalls.
* **Risk—**security testing aims to evaluate the risk that specific threats or vulnerabilities will cause a negative impact on the business. Risk is evaluated by identifying the severity of a threat or vulnerability, and the likelihood and impact of exploitation.
* **Remediation—**security testing is not just a passive evaluation of assets. It provides actionable guidance for remediating vulnerabilities discovered and can verify that vulnerabilities were successfully fixed

## Types Of Security Testing

### Vulnerability Scanning

Vulnerability scanning is performed by automated tools. It is used to identify known vulnerabilities in software components, evaluate vulnerabilities to identify the risk to the organization and assist with remediation.

### Penetration Testing (Ethical Hacking)

Penetration testing is the process of stimulating real-life cyber-attacks against an application, software, system, or network under safe conditions. It can help evaluate how existing security measures will measure up in a real attack. Most importantly, penetration testing can find unknown vulnerabilities, including zero-day threats and business logic vulnerabilities.

Penetration testing was traditionally done manually by a trusted and certified security professional known as an ethical hacker. The hacker works under an agreed scope, attempting to breach a company’s systems in a controlled manner, without causing damage. In recent years, automated penetration testing tools are helping organizations achieve similar benefits at lower cost and with higher testing frequency.

For example, NeuraLegion provides [Nex](https://www.neuralegion.com/products/pt-platform/nexploit/)Ploit, a penetration testing platform powered by artificial intelligence (AI). It automatically scans multiple layers of the IT environment and provides reports on vulnerabilities, including zero-day and complex business logic vulns.

**Learn more in our detailed guide to penetration testing (coming soon)**

### Web Application Security Testing

The goal of web application security testing is to determine whether a web application is vulnerable to attack. It covers a variety of automatic and manual techniques.

Web application penetration testing aims to gather information about a web application, discover system vulnerabilities or flaws, investigate the success of exploiting these flaws or vulnerabilities, and evaluate the risk of web application vulnerabilities.

The [Open Web Application Security Project (OWASP)](https://owasp.org/) is a community dedicated to discovering and reporting security vulnerabilities in web applications.

**Learn more in our detailed guide to**[**web application penetration testing**](https://www.neuralegion.com/blog/web-application-penetration-testing/)

### API Security Testing

API security testing helps identify vulnerabilities in application programming interfaces (APIs) and web services, and assists developers in remediating those vulnerabilities. APIs provide access to sensitive data, and attackers can use them as an entry point to internal systems. Testing APIs rigorously and regularly can protect them from unauthorized access and abuse.

APIs are especially vulnerable to threats like man in the middle (MiTM) attacks, in which attackers can eavesdrop on API communications and steal data or credentials, API injections, in which attackers can inject malicious code to internal systems, and denial of service (DoS), in which attackers flood APIs with fake traffic to deny service to legitimate users.

To mitigate these threats, an API must be verified to have strong authentication of user requests, authorization of users in accordance with the principle of least privilege, encryption of all communication using SSL/TLS, and sanitization of user inputs to prevent code injection and tampering.

**Learn more in our detailed guide to**[**API security testing**](https://www.neuralegion.com/blog/api-security/)

### Configuration Scanning

Security scanning, also known as configuration scanning, is the process of identifying misconfigurations of software, networks, and other computing systems. This type of scanning typically checks systems against a list of best practices, specified by research organizations or compliance standards.

Automated configuration scanning tools identify misconfigurations and provide a report with more details on each misconfiguration, with suggestions on how to resolve them.

### Security Audits

A security audit is a structured process for reviewing/auditing an application/software according to a defined standard. Audits usually involve reviews of code or architectures considering security requirements, analyzing security gaps, and assessing the security posture of hardware configurations, operating systems, and organizational practices. It also evaluates compliance with regulations and compliance standards.

### Risk Assessment

Risk assessment allows an organization to identify, analyze and classify the security risks faced by its business-critical assets. A risk assessment can help understand the most important threats to an organization are infrastructure and prioritize remediation of systems. It can also help with the long-term planning and budgeting of security investments.

### Security Posture Assessment

A security posture assessment combines security scans, ethical hacking, and risk assessment to identify not only the risks facing an organization but also its current security controls and how effective they are. It can identify gaps in the current security posture and recommend changes or improvements that will improve security for protected assets.

### Security Testing Tools

### Static Application Security Testing (SAST)

SAST tools assess the source code while at rest. The purpose of SAST is to identify exploitable flaws and provide a detailed report including findings and recommendations.

You can run SAST to detect issues in source code, to detect issues such as input validation, numerical errors, path traversals, and race conditions. SAST can also be used on compiled code, but this requires binary analyzers.

**Learn more in our detailed guide to**[**SAST**](https://www.neuralegion.com/blog/why-are-sast-solutions-not-the-best-option-for-ast/)

### Dynamic Application Security Testing (DAST)

DAST tools examine the application during runtime. The purpose of DAST is to detect exploitable flaws in the application while it is running, using a wide range of attacks.

A DAST tool often uses fuzzing to throw large volumes of known invalid errors and unexpected test cases at the application, trying to detect conditions during which the application can be exploited.

You can run DAST checks to check a wide range of components, including scripting, sessions, data injection, authentication, interfaces, responses, and requests.

**Learn more in our detailed guide to**[**DAST**](https://www.neuralegion.com/blog/dast-dynamic-application-security-testing/)

### Interactive Application Security Testing (IAST) and Hybrid Tools

IAST tools leverage both static and dynamic testing to create a hybrid testing process. The goal is to determine if known source code vulnerabilities are exploitable during runtime. IAST tools are often employed for the purpose of reducing the number of false positives.

An IAST tool combines various testing techniques to create multiple advanced attack scenarios, using pre-collected information about the data flow and application flow. Then, the tools recursively perform dynamic analysis.

Dynamic analysis cycles ensure that the IAST tool continues to learn more about the application, according to how the application responds to each test case. Depending on the capabilities of the solution, the tool may use the analysis to create new test cases to gain more insights into the application.

### Software Composition Analysis (SCA)

Software Configuration Analysis (SCA) is a technology used to manage and secure open-source components. Development teams can use SCA to quickly track and analyze the open-source components deployed in their projects.

SCA tools can detect all relevant components, libraries that support them, as well as direct and indirect dependencies. In each of these components, they can identify vulnerabilities and suggest remediation. The scanning process creates a Bill of Materials (BOM) that provides a complete list of the project’s software assets.

## Security Testing Best Practices

Here are a few best practices that can help you implement security testing and practice it successfully.

### Shift Security Testing Left

With the shift to DevSecOps – a closer collaboration between developers, security, and operations teams – organizations are adding security practices earlier in the development process. It is common to integrate security testing tools into the continuous integration / continuous delivery (CI/CD) cycle.

Shifting security testing left can help **developers** understand security issues and implement security best practices while the software is under development. It can also help **testers** find security issues early before the software goes into production. Finally, **operations and security** teams can use security testing in production to uncover issues and work with other teams to remediate them.

### Test Internal Interfaces, not Just APIs and UIs

Security testing commonly focuses on external threats, such as user inputs from publicly available web forms. However, it is increasingly common for attackers to exploit weaknesses in internal systems. You should use security testing to verify that there are secure interfaces between internal systems and that insider threats or compromised accounts cannot be used to escalate privileges. This moves your organization closer to a zero-trust security model.

### Automate and Test Often

While it is important to perform manual security testing, such as full penetration tests or security audits, organizations must automate security testing and perform it frequently—preferably with every change to applications or computing infrastructure.

Enterprise applications use many components that may require security updates or may no longer be supported by software vendors. Test business-critical systems often, give high priority to security issues that affect them and urgently devote resources to fixing them.

### Third-Party Components and Open-Source Security

Organizations must adopt security testing for third-party code used in their applications, especially open-source components.

It is unwise to trust commercial software, and equally important to test open-source components, which may require updates or may not be properly secured. You should scan and remediate third-party code just like you would your own, and prioritize updates, remediation, or replacement of unsecured components.

## Security Testing with NeuraLegion

NeuraLegion helps address the shortage of security personnel, enabling AppSec teams to provide governance for security testing, and enabling every developer to run their own security tests.

exploit empowers developers to incorporate an automated Dynamic Application Security Testing (DAST) solution into their unit testing process so they can resolve security concerns as part of their agile development process. NeurLegion’s DAST platform integrates into the SDLC fully and seamlessly:

* Test results are provided to the CISO and the security team, providing complete visibility into vulnerabilities found and remediated
* Tickets are automatically opened for developers in their bug tracking system so they can be fixed quickly

NexPloit can scan any target, whether Web Apps, APIs (REST/SOAP/GraphQL), and WebSocket’s to help enhance DevSecOps and achieve regulatory compliance with our real-time, false-positive free actionable reports of vulnerabilities. In addition, our ML-based DAST solution provides an automated solution to identify Business Logic Vulnerabilities.

[Security Testing: Types, Tools, and Best Practices - NeuraLegion](https://www.neuralegion.com/blog/security-testing/)

# How to Do Security Testing Manually: 12 Effective Ways

Cybersecurity attacks are becoming more prominent for businesses around the world. With evolving attacks, about [68% of business leaders](https://www.accenture.com/_acnmedia/PDF-96/Accenture-2019-Cost-of-Cybercrime-Study-Final.pdf#zoom=50) feel their cybersecurity risks are growing.

The need for security testing can no longer be overlooked.

While some companies rely on a handful of automated security testing tools and processes to maintain security compliance, others leverage both automated testing as well as manual security testing to ensure their software is thoroughly tested and secure.

There are many ways to do security testing manually to test the security posture of your application. Before we dive into them, let’s take a closer look at why you should do security testing manually.

## Why Should You Do Security Testing Manually?

Even with rapid improvements in automation technology, there are still many elements that need human attention to verify or to accurately determine potential web security vulnerabilities in an application.

Some potential vulnerabilities such as business logic issues or cryptographic issues, require a human to verify the vulnerability.

That’s why you need to do security testing manually.

Manual security testers often use a combination of handpicked security testing software and tools that are best suited to evaluate their application. These may include customized scripts and automated scanning tools.

Advanced techniques to do security testing manually involve precise test cases such as checking user controls, evaluating the encryption capabilities, and thorough analysis to discover the nested vulnerabilities within an application.

Doing security testing manually doesn’t imply that you cannot use automation. Rather, security experts can leverage automation technology to find patterns or other clues that might uncover important information about the application’s vulnerabilities.

The primary goal of manual security testing is to discover weaknesses and [potential vulnerabilities in an application](https://cypressdatadefense.com/blog/web-application-vulnerabilities/) that might not be understood or revealed completely by automated security testing alone.

Regardless of the number of automated testing software and tools one might use, it is critical to manually analyze software behavior to ensure its integrity, confidentiality, and availability principles are not being violated.

## Techniques to Help You Do Security Testing Manually

You can do security testing manually when any weakness in the application security needs a real, human judgment call. There is an array of manual security testing techniques that can help you assess your applications and systems to ensure they are secure.

Here are some of the most effective and efficient ways how to do security testing manually:

### 1. Monitor Access Control Management

Be it a web application or a computer, access control is a critical aspect that helps protect your application security or system from being exploited by attackers or insider threats.

Access control management can be categorized into two parts:

* Authentication - Who are you?
* Authorization - What can you do and what information do you have access to?

For instance, an employee should only have access to information that is required to perform his/her job.

By implementing access control, you can ensure that only authorized users can access data or a system.

In order to manually test this, the tester should create several user accounts with different roles.

Then the tester should attempt to access applications or systems by using these accounts and verify that every user account has access only to its own forms, screens, accounts, menus, and modules. The tester can then test requests made by one user/role in the session of a different user/role.

If the tester is able to login to an application with a disabled account, he/she can document the application security issue.

What’s more?

A user with restricted or lower access privileges should not be able to gain access to sensitive information or high privilege data.

You should also manually test for password quality rules, default logins, password recovery, password changes, web security question/answer, logout functionality, etc.

Similarly, authorization tests should also include a test for horizontal access control problems, missing authorization, path reversal, etc.

### 2. Dynamic Analysis (Penetration Testing)

[Penetration testing](https://www.cypressdatadefense.com/security-assessments/application-security-testing/web-application/dynamic-penetration-testing-reporting/), or a pen test, is a software testing technique that uses controlled cyber-attacks to target a running system to determine vulnerabilities that could be exploited by attackers.

Manual penetration testing of a running system consists of the following steps:

* **Data Collection -** The first step of conducting manual penetration testing is collecting data such as table names, databases, information about third-party plugins, software configurations, etc. It can either be done manually or by using testing tools (such as webpage source code analysis) that are freely available online.
* **Vulnerability Assessment -** Once the data is collected, the software penetration testing team evaluates it to determine security risks or vulnerabilities that could put the system at risk of a security attack.
* **Launch Simulated Attacks -** The penetration testing team launches controlled attacks on the target system to explore more vulnerabilities and understand how they can prevent attacks.
* **Report Preparation -** After the system has been targeted and assessed completely for potential vulnerabilities, the software testing team creates a report that outlines the discoveries of the test, and the measures required to protect the system.

This is the process you need to follow when you want to do penetration testing manually to enhance the security of a system.

### 3. Static Analysis (Static Code Analysis)

Another popular method of manual security testing is static code analysis. It is usually performed as a part of white-box testing, also known as a Code Review, and carried out to highlight potential vulnerabilities within the “static” (non-running) source code.

[Static code analysis uses](https://www.cypressdatadefense.com/security-assessments/application-security-testing/web-application/static-analysis/) techniques such as data flow analysis and taint analysis to determine vulnerabilities associated with a system.

It is conducted by manual testers who understand the operating environment the application is running in and the users that use the application. These testers know the overall purpose of the application as well as the purpose of individual functions.

They apply this knowledge to static analysis tools that examine the source code, documentation, and even the executables, to find vulnerabilities without actually running the code.

Static analysis tools vary greatly in purpose and scope, ranging from code styling enforcement to compiler-level checks for logical errors and much more.

Put simply, static code analysis helps you maintain secure code without having to actually run the code.

### 4. Check Server Access Controls

Web applications have multiple user access points that provide enough access to fulfill users’ requests, but they must maintain security to avoid data breaches or attacks.

How can testers check server access controls?

Testers should ensure that all intra-network and inter-network access points to the application are by expected machines (IPs), applications, and users and that all access is strictly controlled.

To verify if an open access point is sufficiently restricted, the tester should try to access these points from various machines having both untrusted and trusted IP addresses.

Additionally, a variety of real-time transactions should be performed in bulk to check the application’s performance under load conditions.

While doing security testing manually, the tester should also check if the open access points in the application allow specific actions by the users in a secure way.

For instance, the tester may upload a file exceeding the maximum permitted file size, try to upload a restricted file type, or download data from a restricted site to check if the application is allowing such actions.

The goal of checking server access controls is to ensure that while users are able to use the application, the application is secure from potential attacks.

### 5. Ingress/Egress/Entry Points

Testers often check ingress and egress network points to ensure that no unauthorized networks can send traffic or information to the host network and vice-versa.

What are ingress and egress points?

Ingress traffic consists of all the network traffic and data communications originating from external networks that are directed towards a node in the host network. On the other hand, egress traffic consists of all traffic originating from within the network and targeted towards an external network.

These entry points in a network can be easily checked via manual security testing methods such as trying to send data from a restricted network to the host network and check if it is allowing the traffic and accepting data.

A tester may even send sensitive data or confidential information from the host network to an authorized external network to check if the egress points are secured.

Ingress and egress filtering allows networks to interact with one another while maintaining security standards and restricting the sharing of sensitive data to unauthorized networks.

### 6. Session Management

When you do security testing manually, you should perform session management tests to check if the application is handling sessions properly.

To ensure that your application has proper session management, check the session expiration after a particular idle time, session termination after login and log out, session termination after maximum lifetime, check for session duration and session cookie scope, etc.

### 7. Password Management

One of the most productive security testing techniques that you can use while doing testing manually is password management. This refers to the various methods used to discover passwords and access user accounts or systems.

How can you test password management?

If the web application or system does not enforce stringent password policies, (for example, with numerics, special characters, or passphrases), it may be quite easy to brute force passwords and access the account.

Additionally, passwords that are not stored in an encrypted format are more vulnerable to being stolen and used directly. Attackers may use different methods to steal the information stored in the database such as SQL Injection.

Even if passwords are stored in a hashed format, once they are retrieved, they can be cracked using password cracking tools such as Brutus, RainbowCrack, or by manually guessing username/password combinations.

### 8. Brute-Force Attacks

Another way on how to do security testing manually is by using brute-force attacks.

Brute-force attacks rely on guessing different combinations of a targeted password until the correct password is discovered.

Attackers use brute-force attacks to gain access to sensitive information such as personal identification numbers, passphrases, passwords, or usernames to carry out identity theft, redirect domains to sites with malicious content, or other malicious activities.

This method is also widely used by application security testers to [test application security](https://cypressdatadefense.com/blog/application-security-best-practices/), and more specifically, evaluate the strength of the application’s encryption.

For instance, a tester should attempt to login to accounts with invalid passwords, and ideally, the system should block the user after a limited number of failed multiple login attempts.

Moreover, if the login attempts are made from an unknown device or suspicious network, the application should ask for multiple-factor authentication which might consist of one-time passwords sent to the verified email address or contact number of the user, or a security question set by the user.

### 9. SQL Injection (SQLi)

SQL Injection is a code injection technique used to inject malicious SQL statements into an application to modify or extract data stored in databases.

It is one of the most dangerous, frequent, and oldest web application vulnerabilities. It can affect any web application that uses SQL databases such as Oracle, SQL Server, MySQL, or others.

How can you prevent SQL Injection attacks?

Manual testers check the SQL injection entry points to identify if it can be exploited by a SQL injection attack. They identify and test the database code in which direct MySQL queries are performed on the database by accepting certain user inputs.

For instance, the application should be able to accept a single quote (‘) in an input field. But if the application throws a database error to the tester, it means that the user input has been inserted in some query to the database and it has been executed.

The SQL query error message shown on the browser may lead the attacker to crash the entire application or help them to extract data like usernames, passwords, credit card numbers, etc.

### 10. Cross-Site Scripting (XSS)

In addition to SQL Injection attacks, testers also check the web application for Cross-Site Scripting (i.e XSS) in manual security testing. It is a client-side injection attack where the attacker aims to execute malicious scripts in the victim’s browser.

These malicious scripts can perform a variety of functions such as send the victim’s login credentials or session token to the attacker, log their keystrokes, or perform arbitrary actions on behalf of the victim.

During manual testing, testers must ensure that the input fields do not trust unvalidated user input, and must properly encode the output of these fields if they are included in a server response.

Moreover, the primary way to protect your application from XSS injection attack is by applying proper input and output encoding.

### 11. URL Manipulation

URL manipulation is another technique through which attackers exploit applications. It is the process of modifying the parameters of a Uniform Resource Locator (URL) for malicious purposes by an attacker.

How can you protect your application from URL manipulation?

Manual testers should verify whether the application allows sensitive information in the query string. These types of attacks occur when the application uses the HTTP GET method to transfer information between the server and the client.

When a URL-based input is given to an application, it passes this information through the parameters in the query string. The tester may change a parameter value in the query string to verify whether the server accepts that value.

User information is passed through HTTP GET requests to the server to fetch data or make requests. If the tester is able to manipulate input variables passed through this GET request to the server, they can get access to unauthorized information.

### 12. Specify High-Risk Functions

Businesses deal with a lot of data on an everyday basis. There are thousands of business functionalities that require file upload/download, giving user access privilege to employees, sharing data with third-party contractors, and many other activities that may have potential vulnerabilities.

You need to identify high-risk functions to ensure that better security measures are implemented for particular activities such as restricting unwanted or malicious file uploads/downloads.

If your application deals with any sensitive data, you should manually check the application for injection vulnerabilities, password guessing, buffer overflows, insecure cryptographic storage, etc.

## Use These Ways to Do Security Testing Manually

While automated security testing has ample benefits, it is not enough to ensure that an application is completely secure.

Businesses must conduct manual security tests to ensure that there are no potential weaknesses or vulnerabilities in an application that could be exploited by an attacker.

By conducting proper security tests manually, companies can detect business flaws and injection vulnerabilities that might not be clear from automated security tests.

Ready to get started? You can use the effective manual security testing techniques above while doing security testing manually.

## Reference

[How to Do Security Testing Manually: 12 Effective Ways | Cypress Data Defense](https://www.cypressdatadefense.com/blog/how-to-do-security-testing-manually/)

# What is Security Testing? Types with Example

## What is Security Testing?

**Security Testing** is a type of Software Testing that uncovers vulnerabilities, threats, risks in a software application and prevents malicious attacks from intruders. The purpose of Security Tests is to identify all possible loopholes and weaknesses of the software system which might result in a loss of information, revenue, repute at the hands of the employees or outsiders of the Organization.

## Why Security Testing is Important?

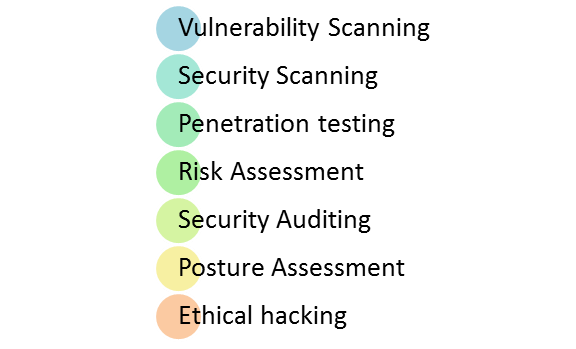
The main goal of **Security Testing** is to identify the threats in the system and measure its potential vulnerabilities, so the threats can be encountered and the system does not stop functioning or can not be exploited. It also helps in detecting all possible security risks in the system and helps developers to fix the problems through coding.

In this tutorial, you will learn-

* [What is Security Testing?](https://www.guru99.com/what-is-security-testing.html#1)
* [Types of Security Testing](https://www.guru99.com/what-is-security-testing.html#2)
* [How to do Security Testing](https://www.guru99.com/what-is-security-testing.html#3)
* [Example Test Scenarios for Security Testing](https://www.guru99.com/what-is-security-testing.html#4)
* [Methodologies/ Approach / Techniques for Security Testing](https://www.guru99.com/what-is-security-testing.html#5)
* [Security Testing Roles](https://www.guru99.com/what-is-security-testing.html#6)
* [Security Testing Tool](https://www.guru99.com/what-is-security-testing.html#61)
* [Myths and Facts of Security Testing](https://www.guru99.com/what-is-security-testing.html#7)

## Types of Security Testing:

There are seven main types of security testing as per Open Source Security Testing methodology manual. They are explained as follows:



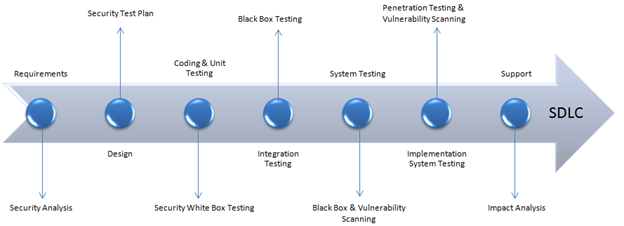
* **Vulnerability Scanning**: This is done through automated software to scan a system against known vulnerability signatures.
* **Security Scanning:** It involves identifying network and system weaknesses, and later provides solutions for reducing these risks. This scanning can be performed for both Manual and Automated scanning.
* **Penetration testing**: This kind of testing simulates an attack from a malicious hacker. This testing involves analysis of a particular system to check for potential vulnerabilities to an external hacking attempt.
* **Risk Assessment:** This testing involves analysis of security risks observed in the organization. Risks are classified as Low, Medium and High. This testing recommends controls and measures to reduce the risk.
* **Security Auditing:** This is an internal inspection of Applications and Operating systems for security flaws. An audit can also be done via line-by-line inspection of code
* **Ethical hacking:** It’s hacking an Organization Software system. Unlike malicious hackers, who steal for their own gains, the intent is to expose security flaws in the system.
* **Posture Assessment:** This combines Security scanning,[Ethical Hacking](https://www.guru99.com/ethical-hacking-tutorials.html)and Risk Assessments to show an overall security posture of an organization.



## How to do Security Testing

It is always agreed, that cost will be more if we postpone security testing after software implementation phase or after deployment. So, it is necessary to involve security testing in the SDLC life cycle in the earlier phases.

Let’s look into the corresponding Security processes to be adopted for every phase in SDLC



|  |  |
| --- | --- |
| **SDLC Phases** | **Security Processes** |
| **Requirements** | Security analysis for requirements and check abuse/misuse cases |
| **Design** | Security risks analysis for designing. Development of[Test Plan](https://www.guru99.com/what-everybody-ought-to-know-about-test-planing.html)including security tests |
| **Coding and Unit Testing** | Static and Dynamic Testing and Security [White Box Testing](https://www.guru99.com/white-box-testing.html) |
| **Integration Testing** | [Black Box Testing](https://www.guru99.com/black-box-testing.html) |
| **System Testing** | Black Box Testing and Vulnerability scanning |
| **Implementation** | [Penetration Testing](https://www.guru99.com/learn-penetration-testing.html), Vulnerability Scanning |
| **Support** | Impact analysis of Patches |

The test plan should include

* Security-related test cases or scenarios
* Test Data related to security testing
* Test Tools required for security testing
* Analysis of various tests outputs from different security tools

## Example Test Scenarios for Security Testing:

Sample Test scenarios to give you a glimpse of security test cases –

* A password should be in encrypted format
* Application or System should not allow invalid users
* Check cookies and session time for application
* For financial sites, the Browser back button should not work.

## Methodologies/ Approach / Techniques for Security Testing

In security testing, different methodologies are followed, and they are as follows:

* **Tiger Box**: This hacking is usually done on a laptop which has a collection of OSs and hacking tools. This testing helps penetration testers and security testers to conduct vulnerabilities assessment and attacks.
* [**Black Box**](https://www.guru99.com/black-box-testing.html): Tester is authorized to do testing on everything about the network topology and the technology.
* **Grey Box**: Partial information is given to the tester about the system, and it is a hybrid of white and black box models.

## Security Testing Roles

* Hackers – Access computer system or network without authorization
* Crackers – Break into the systems to steal or destroy data
* Ethical Hacker – Performs most of the breaking activities but with permission from the owner
* Script Kiddies or packet monkeys – Inexperienced Hackers with programming language skill

## Security Testing Tool

### 1) [Intruder](https://bit.ly/2NZWxEr)

[Icon

Description automatically generated](https://bit.ly/2NZWxEr)

[Intruder](https://bit.ly/2NZWxEr) is an enterprise-grade vulnerability scanner that is easy to use. It runs over 10,000 high-quality security checks across your IT infrastructure, which include, but are not limited to: configuration weaknesses, application weaknesses (such as SQL injection & cross-site scripting) and missing patches. Providing intelligently prioritised results as well as proactive scans for the latest threats, Intruder helps save time and keeps businesses of all sizes safe from hackers.

**Features:**

* AWS, Azure and Google Cloud connectors
* Perimeter-specific results to reduce your external attack surface
* High-quality reporting
* Slack, Microsoft Teams, Jira, Zapier integrations
* API integration with your CI/CD pipeline

### 2) [Acunetix](https://bit.ly/3qH5T77)

Intuitive and easy to use, [Acunetix](https://bit.ly/3qH5T77) by Invicti helps small to medium-sized organizations ensure their web applications are secure from costly data breaches. It does so by detecting a wide range of web security issues and helping security and development professionals act fast to resolve them.

[A picture containing text, clipart, tableware

Description automatically generated](https://bit.ly/3qH5T77)

**Features:**

* Advanced scanning for 7,000+ web vulnerabilities, including OWASP Top 10 such as SQLi and XSS
* Automated web asset discovery for identifying abandoned or forgotten websites
* Advanced crawler for the most complex web applications, incl. multi-form and password-protected areas
* Combined interactive and dynamic application security testing to discover vulnerabilities other tools miss
* Proof of exploit provided for many types of vulnerabilities
* DevOps automation through integrations with popular issue tracking and CI/CD tools
* Compliance reporting for regulatory standards, such as PCI DSS, NIST, HIPAA, ISO 27001, and more.

### 3) Owasp

The Open Web Application Security Project ([OWASP](https://bit.ly/2P4SruD)) is a worldwide non-profit organization focused on improving the security of software. The project has multiple tools to pen test various software environments and protocols. Flagship tools of the project include

1. [Zed Attack Proxy](https://bit.ly/2z6aki2) (ZAP – an integrated penetration testing tool)
2. [OWASP Dependency Check](https://bit.ly/2TN7fNu) (it scans for project dependencies and checks against know vulnerabilities)
3. [OWASP Web Testing Environment Project](https://bit.ly/2P4SruD) (collection of security tools and documentation)

### 4) WireShark

[Wireshark](https://bit.ly/2TMN561) is a network analysis tool previously known as Ethereal. It captures packet in real time and display them in human readable format. Basically, it is a network packet analyzer- which provides the minute details about your network protocols, decryption, packet information, etc. It is an open source and can be used on Linux, Windows, OS X, Solaris, NetBSD, FreeBSD and many other systems. The information that is retrieved via this tool can be viewed through a GUI or the TTY mode TShark Utility.

### 5) W3af

[w3af](https://bit.ly/2P5Qrm7) is a web application attack and audit framework. It has three types of plugins; discovery, audit and attack that communicate with each other for any vulnerabilities in site, for example a discovery plugin in w3af looks for different url’s to test for vulnerabilities and forward it to the audit plugin which then uses these URL’s to search for vulnerabilities.

## Myths and Facts of Security testing:

Let’s talk about an interesting topic on Myths and facts of security testing:

**Myth #1** We don’t need a security policy as we have a small business

Fact: Everyone and every company need a security policy

**Myth #2**There is no return on investment in security testing

Fact: Security Testing can point out areas for improvement that can improve efficiency and reduce downtime, enabling maximum throughput.

**Myth #3**: Only way to secure is to unplug it.

Fact: The only and the best way to secure an organization is to find “Perfect Security”. Perfect security can be achieved by performing a posture assessment and compare with business, legal and industry justifications.

**Myth #4**: The Internet isn’t safe. I will purchase software or hardware to safeguard the system and save the business.

Fact: One of the biggest problems is to purchase software and hardware for security. Instead, the organization should understand security first and then apply it.

## Conclusion:

Security testing is the most important testing for an application and checks whether confidential data stays confidential. In this type of testing, tester plays a role of the attacker and play around the system to find security-related bugs. Security Testing is very important in Software Engineering to protect data.

### You Might Like:

* [**What is Use Case Testing? Technique, Examples**](https://www.guru99.com/use-case-testing.html)
* [**Penetration Testing Tutorial: What is PenTest?**](https://www.guru99.com/learn-penetration-testing.html)
* [**What is Cloud Testing? SaaS Testing Tutorial**](https://www.guru99.com/cloud-testing-tutorial-with-saas-testing-primer.html)
* [**What is Volume Testing? Learn with Examples**](https://www.guru99.com/volume-testing.html)
* [**What is Pilot Testing? Definition, Meaning, Examples**](https://www.guru99.com/pilot-testing.html)

[What is Security Testing? Types with Example (guru99.com)](https://www.guru99.com/what-is-security-testing.html)

# Application Security Testing

## What is Application Security Testing?

Application security testing (AST) is the process of making applications more resistant to security threats, by identifying security weaknesses and vulnerabilities in source code.

AST started as a manual process. Today, due to the growing modularity of enterprise software, the huge number of open-source components, and the large number of known vulnerabilities and threat vectors, AST must be automated. Most organizations use a combination of several application security tools.

## Static Application Security Testing (SAST)

SAST tools use a [white box testing](https://www.imperva.com/learn/application-security/white-box-testing/) approach, in which testers inspect the inner workings of an application. SAST inspects static source code and reports on security weaknesses.

Static testing tools can be applied to non-compiled code to find issues like syntax errors, math errors, input validation issues, invalid or [insecure references](https://www.imperva.com/learn/application-security/cyber-security/). They can also run on compiled code using binary and byte-code analyzers.

## Dynamic Application Security Testing (DAST)

DAST tools take a [black box testing](https://www.imperva.com/learn/application-security/black-box-testing/) approach. They execute code and inspect it in runtime, detecting issues that may represent [security vulnerabilities](https://www.imperva.com/learn/application-security/cve-cvss-vulnerability/). This can include issues with query strings, requests and responses, the use of scripts, memory leakage, cookie and session handling, authentication, execution of third-party components, data injection, and DOM injection.

DAST tools can be used to conduct large-scale scans simulating a large number of unexpected or [malicious](https://www.imperva.com/learn/application-security/malware-detection-and-removal/) test cases and reporting on the application’s response.

## Interactive Application Security Testing (IAST)

IAST tools are the evolution of SAST and DAST tools—combining the two approaches to detect a wider range of security weaknesses. Like DAST tools, IAST tools run dynamically and inspect software during runtime. However, they are run from within the application server, allowing them to inspect compiled source code like IAST tools do.

IAST tools can provide valuable information about the root cause of vulnerabilities and the specific lines of code that are affected, making remediation much easier. They can analyze source code, data flow, configuration and third-party libraries, and are suitable for API testing.

## Mobile Application Security Testing (MAST)

MAST tools combine static analysis, dynamic analysis and investigation of forensic data generated by mobile applications. They can test for security vulnerabilities like [SAST, DAST and IAST](https://www.imperva.com/learn/application-security/sast-iast-dast/), and in addition address mobile-specific issues like jailbreaking, malicious wifi networks, and data leakage from mobile devices.

## Software Composition Analysis (SCA)

SCA tools help organizations conduct an inventory of third-party commercial and open source components used within their software. Enterprise applications can use thousands of third-party components, which may contain security vulnerabilities. SCA helps understand which components and versions are actually being used, identify the most severe security vulnerabilities affecting those components, and understand the easiest way to remediate them.

## Runtime Application Self-Protection (RASP)

RASP tools evolved from SAST, DAST and IAST. They are able to analyze application traffic and [user behavior at runtime](https://www.imperva.com/learn/data-security/ueba-user-and-entity-behavior-analytics/), to detect and prevent [cyber threats](https://www.imperva.com/learn/application-security/cyber-attack/).

Like the previous generation of tools, RASP has visibility into application source code and can analyze weaknesses and vulnerabilities. It goes one step further by identifying that security weaknesses have been exploited and providing active protection by terminating the session or issuing an alert.

RASP tools integrate with applications and analyze traffic at runtime, and can not only detect and warn about vulnerabilities, but actually prevent attacks. Having this type of in-depth inspection and protection at runtime makes SAST, DAST and IAST much less important, making it possible to detect and prevent security issues without costly development work.

## Application Security Testing Best Practices

### Shift security testing left

New organizational practices like DevSecOps are emphasizing the need to integrate security into every stage of the software development lifecycle. AST tools can:

* Help developers understand security concerns and enforce security best practices at the development stage.
* Help testers identify security issues early before software ships to production.
* Advanced tools like RASP can identify and block vulnerabilities in source code in production.

### Test internal interfaces, not just APIs and UIs

It is natural to focus application security testing on external threats, such as user inputs submitted via web forms or public API requests. However, it is even more common to see attackers exploit weak authentication or vulnerabilities on internal systems, once already inside the security perimeter. AST should be leveraged to test those inputs, connections and integrations between internal systems are secure.

### Test often

New vulnerabilities are discovered every day, and enterprise applications use thousands of components, any of which could go end of life (EOL) or require a security update. It is essential to test critical systems as often as possible, prioritize issues focusing on business critical systems and high-impact threats, and allocate resources to remediate them fast.

### Third-party code security

Organizations should employ AST practices to any third-party code they use in their applications. Never “trust” that a component from a third party, whether commercial or open source, is secure. Scan third-party code just like you scan your own. If you discover severe issues, apply patches, consult vendors, create your own fix or consider switching components.

**See how Imperva RASP can help you with Application Security Testing.**

[Request demo](javascript:openModal('modalid3533',%20'/learn/banner/virtual/request-demo/',%20'Personal%20Demo%20Request%20|%20Imperva');)[Learn more](https://www.imperva.com/products/runtime-application-self-protection-rasp/)

## Imperva RASP Solutions

Imperva provides RASP capabilities, as part of its application security platform. [Imperva RASP](https://www.imperva.com/products/runtime-application-self-protection-rasp/) keeps applications protected and provides essential feedback for eliminating any additional risks. It requires no changes to code and integrates easily with existing applications and DevOps processes, protecting you from both known and [zero-day attacks](https://www.imperva.com/learn/application-security/zero-day-exploit/).

In addition, Imperva provides multi-layered protection to make sure websites and applications are available, easily accessible and safe. These application security solutions include:

* [**DDoS Protection**](https://www.imperva.com/products/ddos-protection-services/)—maintain uptime in all situations. Prevent any type of DDoS attack, of any size, from preventing access to your website and network infrastructure.
* [**CDN**](https://www.imperva.com/learn/performance/what-is-cdn-how-it-works/)—enhance website performance and reduce bandwidth costs with a CDN designed for developers. Cache static resources at the edge while accelerating APIs and dynamic websites.
* [**Cloud WAF**](https://www.imperva.com/products/cloud-waf/)—permit legitimate traffic and prevent bad traffic. Safeguard your applications at the edge with an enterprise‑class cloud WAF.
* [**Gateway WAF**](https://www.imperva.com/products/on-premises-waf/)—keep applications and APIs inside your network safe with Imperva Gateway WAF.
* [**Attack analytics**](https://www.imperva.com/products/attack-analytics/)—mitigate and respond to real security threats efficiently and accurately with actionable intelligence across all your layers of defense.
* [**Account takeover protection**](https://www.imperva.com/products/ato-account-takeover-protection/)—uses an intent-based detection process to identify and defends against attempts to take over users’ accounts for malicious purposes.
* [**API security**](https://www.imperva.com/learn/application-security/web-api-security/)—protects APIs by ensuring only desired traffic can access your API endpoint, as well as detecting and blocking exploits of vulnerabilities.
* [**Advanced bot protection**](https://www.imperva.com/products/advanced-bot-protection-management/)—analyzes your bot traffic to pinpoint anomalies, identifies bad bot behavior and validates it via challenge mechanisms that do not impact user traffic.

[What is Application Security Testing (AST) | Tools & Best Practices | Imperva](https://www.imperva.com/learn/application-security/application-security-testing/)

## What is Security Testing?

Security testing is a ***Non-Functional Testing*** process to determine that the security mechanism of an information system protects data and maintains functionality as intended. It is done to check whether the application or the product is secured or not. It checks whether there is any information leakage in encrypting the application or using a wide range of software, hardware, and firewall, etc. It ensures that no one can hack the system and login to the application without any authorization.

***ISTQB Definition : Security Testing is a type of software testing that intends to uncover vulnerabilities of the system and determine that its data and resources are protected from possible intruders.***

***Principle of Security Testing : Confidentiality, Integrity, Authentication, Availability, Authorization, and Non-Repudiation.***

There are four main focus areas to be considered in security testing (Especially for web sites/applications):

* ***Network security***: This involves looking for vulnerabilities in the network infrastructure (resources and policies).
* ***System software security***: This involves assessing weaknesses in the various software (operating system, database system, and other software) the application depends on.
* ***Client-side application security***: This deals with ensuring that the client (browser or any such tool) cannot be manipulated.
* ***Server-side application security***: This involves making sure that the server code and its technologies are robust enough to fend off any intrusion.

### ***Why Security Testing?***

* Loss of customer trust.
* Disturbance to your online system leads to revenue impact.
* Website downtime leads to time loss and expenditures in recovering from damage.
* Cost associated with securing web applications against future attacks.
* Related legal implications and fees for having lax security measures in place.

### ***Security Threats***

1. ***Privilege Elevation***: When a hacker has an account on a system and uses it to increase his system privileges to a higher level.
2. ***SQL Injection***: Malicious SQL statements are inserted into an entry field for execution to get critical information from the server database.
3. ***Unauthorized Data Access*** : Gaining unauthorized access to data within an application from servers or on a network.
4. ***URL Manipulation*** : Application uses HTTP GET method to pass information between the client and the server and attackers manipulating the website URL query strings & capture of the important information.
5. ***Denial Of Service*** : (DoS) attack is an explicit attempt to make a machine or network resource unavailable to its legitimate users.
6. ***Cross-Site Scripting*** (XSS) : XSS enables attackers to inject client-side script into Web pages viewed by other users and trick a user into clicking on that URL.
7. ***Data Manipulation*** : Hacker changes data used by a website in order to gain some advantage or to embarrass the website’s owners.
8. ***Identity Spoofing*** : Hacker uses the credentials of a legitimate user or device to launch attacks against network hosts.

### ***Security Testing Techniques***

1. ***SQL Injection*** : Entering a single quote (‘ ) in any text-box should be rejected by the application. Instead, if the tester encounters a database error, it means that the user input is inserted in some query and that is executed by the application.
2. ***Cross Site Scripting*** (XSS): The tester should additionally check the web application for XSS (Cross-site scripting). Any HTML e.g. <HTML> or any script e.g. <SCRIPT> should not be accepted by the application.
3. ***Ethical Hacking*** : This helps identify potential threats on a computer or network. An ethical hacker attempts to bypass the system security and search for any vulnerability that could be exploited by malicious hackers aka Black hats.
4. ***Password Cracking*** : Hackers can use a password cracking tools to crack passwords. Until a web application enforces a complex password (long password with a combination of numbers, letters, and special characters), it is easy to crack.
5. ***Penetration Testing*** : A penetration test is an attack on a computer system with the intention of finding security loopholes, potentially gaining access to it, its functionality and data.
6. ***Risk Assessment*** : This is a process of assessing and deciding on the risk involved with the type of loss and the possibility of vulnerability occurrence.
7. ***Security Auditing*** : A security audit is a systematic evaluation of the security of a company’s information system by measuring how well it conforms to a set of established criteria.
8. ***Security Scanning*** : This is a program that communicates with a web application through the web front-end in order to identify potential security vulnerabilities in the web application, OS and Networks.

[Security Testing (toolsqa.com)](https://www.toolsqa.com/software-testing/security-testing/)

# **What is Security testing in software testing?**

* It is a type of non-functional testing.
* Security testing is basically a type of [**software testing**](http://tryqa.com/what-is-a-software-testing/) that’s done to check whether the application or the product is secured or not. It checks to see if the application is vulnerable to attacks, if anyone hack the system or login to the application without any authorization.
* It is a type of non-functional testing.
* Security testing is basically a type of [**software testing**](http://tryqa.com/what-is-a-software-testing/) that’s done to check whether the application or the product is secured or not. It checks to see if the application is vulnerable to attacks, if anyone hack the system or login to the application without any authorization.

# **What are Security testing tools in software testing?**

There are many tools that protect systems from external attack. Like firewall, this is very important for any system.

**Security testing tools**can be used to test security of the system by trying to break it or by hacking it**.**The attacks may focus on the network, the support software, the application code or the underlying database.

Features or characteristics of security testing tools are:

* To identify viruses;
* To detect intrusions such as denial of service attacks;
* To simulate various types of external attacks;
* Probing for open ports or other externally visible points of attack;
* To identify weaknesses in password files and passwords;
* To do the security checks during operation, e.g. for checking integrity of files, and intrusion detection, e.g. checking results of test attacks.

Popular security testing tools

* Zed Attack Proxy
* Aircrack-ng
* Metasploit
* ZMap
* SOAtest
* Nmap
* Jtest
* American fuzzy lop
* AddressSanitizer
* SQLmap
* Wireshark
* Wapiti
* Vega
* W3af
* Skipfish
* Ratproxy
* Wfuzz
* Grendel Scan
* Arachni
* Grabber

# **What is Compatibility testing in software testing?**

* It is a type of non-functional testing.
* Compatibility testing is a type of [**software testing**](http://tryqa.com/what-is-a-software-testing/) used to ensure compatibility of the system/application/website built with various other objects such as other web browsers, hardware platforms, users (in case if it’s very specific type of requirement, such as a user who speaks and can read only a particular language), operating systems etc. This type of testing helps find out how well a system performs in a particular environment that includes hardware, network, operating system and other software etc.
* It is basically the testing of the application or the product built with the computing environment.
* It tests whether the application or the software product built is compatible with the hardware, operating system, database or other system software or not.

# **What is Endurance testing in software testing?**

Endurance testing is a non functional type of [**software testing**](http://tryqa.com/what-is-a-software-testing/).

* It is a type of non-functional testing.
* It is also known as Soak testing.
* Endurance testing involves testing a system with a significant load extended over a significant period of time, to discover how the system behaves under sustained use. For example, in software testing, a system may behave exactly as expected when tested for 1 hour but when the same system is tested for 3 hours, problems such as memory leaks cause the system to fail or behave randomly.
* The goal is to discover how the system behaves under sustained use. That is, to ensure that the throughput and/or response times after some long period of sustained activity are as good or better than at the beginning of the test.
* It is basically used to check the memory leaks.

# **Mobile Application Security Testing Approaches – Beginners Guide**

In this article we take a high level look at some of the Mobile Application Security Testing approaches. Test engineers need to use the most effective, fast and understandable environment for conducting the testing process and also to supplement it with a flexible infrastructure to support the implementation of full-scale testing in a situation where the frequency of updating the number of mobile devices and digital technologies is rapidly increasing.

## Mobile Apps Industry – Opportunity and Demand

As per the CTT company, detailed report of 2016 implies: the profit increase of the mobile sector to 530 billion US dollars is expected at the end of 2020.

However, the rapid growth in demand for mobile gadgets involves not only analyzing how to use them but also requires significant financial costs to conduct an optimized procedure for testing the mobile apps.

According to ABI Research, the number of sales of tools, which allow manual analyzing the performance of mobile applications, has increased from 200 million to 850 million from 2012 to 2017.

It speaks volumes that the mobile development sphere with subsequent performance verification of its individual components is quite a demanded procedure, which makes a sense to consider the mobile applications testing processes more detailed, in particular, such a direction as the security check of the developed application.

## Mobile Application Testing Process

Virtually all of the studies, which are conducted during the [**mobile application testing**](http://tryqa.com/mobile-application-development-and-testing-checklist/) process, are directly or indirectly related to such actions:

* [**White box**](http://tryqa.com/what-is-white-box-or-structure-based-or-structural-testing-techniques/) (test object whose features are known to the tester) + unit testing;
* Black box (test object whose device is unknown before the process of testing) + testing the visual performance of the mobile application;
* [**Verification**](http://tryqa.com/what-is-verification-in-software-testing-or-what-is-software-verification/) of the established validation of the quality of service parameters (QoS, Quality of Service);
* [**Usability testing**](http://tryqa.com/what-is-usability-testing-in-software-and-its-benifits-to-end-user/) of the utility;
* Opportunities to connect the automation infrastructure for testing mobile technologies;
* Work with security systems.

If we consider the term “testing mobile programs in the security perspective” we can distinguish the fact that this term covers the verification of not only the developed application and the work with testing utilities specially created for the operation of another application, as well as a set of work to test mobile web applications.

In general terms,**a mobile product testing is a set of test actions that are conducted in a strictly established sequence and guarantee the performance of all the functions declared in the technical documentation, fully meet the quality standards and also have the characteristic system features:**mobility, usability, connectivity, logical consistency and, of course, **security**!

### **Specified requirements**

Checking the mobile application efficiency is very different from the tests that are carried out when examining the classic software functionality. After all, in the first case, several original and unique requirements must be applied at once.

First of all, due to the fact that the mobile application must be correctly executed under any conditions at any time.

It is very important that the mobile product functions correctly on various platforms which use completely different OSs, the ratio of the screen size, as well as the technological characteristics of computing capacities, and, of course, the maximum indicators of the battery life.

Any mobile application should be “mobile” in everything: from the methods (channels) of information input to maintaining work with other multimedia systems and services.

So, the mobile product should work optimally in any heterogeneous network environment considering the trend of mobile services in a fairly wide range of wireless networks.

### **Core objectives and ongoing activities**

It can be identified such groups of objectives and activities for optimal [**software testing services**](https://testmatick.com/our-qa-services/) when taking into account the requirements for testing applications:

* [**Checking the functional part**](http://tryqa.com/what-is-functional-testing-testing-of-functions-in-software/) and system behavior allows QA to understand how the functions inside the system interact, connected APIs, external systems, as well as user interface complexes
* QoS checking means a work on testing the load on the system, checking its stability, throughput, and scalability
* Verification of interoperability is an analysis of the ability of the system to interact with various mobile devices and wireless networks
* Checking the degree of confidentiality and security means carrying out measures to verify the client authentication logic, device security, work session security, potential penetration capabilities, compliance with common security rules and the level of client data confidentiality

## Types of mobile application security testing

The most common and important elements of conducting mobile security are the verification of such systems and components:

* Potentially unsafe storage of information
* Low data transmission protection threshold
* Weak authentication and authorization level
* Unsafe management and session management

## Potentially unsafe storage of information

This problem has several important “sub-problems” as:

* Hardcoded and forgotten
* Incorrect files permissions
* SD Storage
* Logs

Let’s consider each component separately.

## Hardcoded and forgotten

For example, we have an application for the Android operating system. As is well known, it comes in the form of a special apk file (original archive which contains resource and executable files).

Sometimes it’s enough just to go into resource files and you can find these elements which in theory should not be in the supplied product:Graphical user interface, text, application

Description automatically generated

### **1. Incorrect file permissions**

Here the main problem is that, as is known, any application has its own original UID – a specialized number (ID) by which it can be found on the network. It uses a fairly common principle – one application – one path. However, sometimes, the system needs to get its personalized access to the files in order to output some functions to the console. Then a completely different path is indicated and that is not good!

The example:Text

Description automatically generated

### **2. SD Storage**

It’s no secret that the user data, that is on the SD card, can easily be used by other applications. They do not have such an intra-system protection.

Everything would be fine if the applications load simple photos or video on a memory card. Not at this time. There are utilities that “are able” to record to external media cookies or special authorization tokens. So, it is precisely this logic that should be avoided of, if possible.

### **3. Logs**

Web developers try to use the process of logging (the logic is automatically recorded in the file environment of the smartphone) during a mobile product development very often.

The same about Android, it could easily configure the intra-system files so that the installed on the device application could request the android.permission.READ\_LOGS function before the release of version 4.4.

The example:Text

Description automatically generated

Summing up, the following recommendations can be noted:

* Try not to store data on the memory card
* Be sure to turn off the logging
* Customize rights, taking into account that a potential customer can use a smartphone with Root rights or with a jailbreak
* View the configuration files of your utility for forgotten data

## Low data transmission protection threshold

This problem has several important “sub-problems” such as:

* No encryption was used
* Use of self-signed certificates

## Encryption

In order to avoid problems with the encryption area, it is recommended:

* Check mobile application traffic
* Use certificates that have been signed by trusted centers
* Verify and prescribe access rights if you use content providers (content provider provide access to files or databases for other applications)

## Weak authentication and authorization level

The given problem has several important “sub-problems” as:

* Anonymous work with the utility
* Use of clients with a low level of system privileges to obtain an accessible to all users’ information
* Weak and simple passwords

In this case protection methods should be as follows:

* Authentication in the application should be similar in the web version
* Local authentication must function through the cookie system after the client has been authorized in the server
* Anonymous access disable
* Introduction of the client rights verification
* Complex passwords
* Change of an anonymous customer to a registered one
* Switch between registered users
* Switch between clients with different access rights

An important recommendation for this case – the registration token should be removed on the server. All cookies should be invalid!

It is also possible to note such useful reference as for how to use user session in time slice, namely:

* No more than 15 minutes for an application with a high level of security
* About 30 minutes for applying an average level of security
* More than 1 hour for all others

In the end, it can be identified as a group of problems which prevent developers from creating the safest and most secure applications and [**software testing companies**](https://testmatick.com/) to test them:

* **Test environment** – As many reviews of developers say, the current mobile test environments are still very expensive and not understandable. Creating an optimized environment to test multiple applications on one site turns into a very long and complex process and constant operational updates of web platforms only aggravate the current situation. Ideally, the testing environment should consist of such components: unified support for different web platforms, a wide range of configurations taking into account the functions of the operational deployment, as well as parameters for the diversification of the required configurations;
* **Automation** – It is the processes of the security automated testing of the mobile utility that cause the greatest questions because there is no generally accepted standardization of the test environment for carrying out the corresponding checking The creation of emulating clouds, a set of commands for creating, deploying and managing applications that allow you to work with the internal filling of any developed application can be the way out of this situation.

As a result, it turns out that it is the testing environment for the security of mobile products that is the most vulnerable and up-to-date direction that should be developed everywhere and supplemented with new verification parameters and testing in order to minimize the number of bugs that are allowed in the product development.

[What are Security testing tools in software testing? (tryqa.com)](http://tryqa.com/what-is-security-tools-in-software-testing/)

# Security Testing (A Complete Guide)

**How to Test Application Security – Web and Desktop Application Security Testing Techniques**

**Need for Security Testing**

The software industry has achieved solid recognition in this age. In recent decades, however, the cyber-world seems to be an even more dominating and driving force which is shaping up the new forms of almost every business.

Web-based ERP systems used today are the best evidence that IT has revolutionized our beloved global village. These days, websites are not only meant for publicity or marketing but they have evolved into stronger tools to cater to complete business

## A Complete Security Testing Guide

Web-based Payroll systems, Shopping Malls, Banking, and Stock Trade applications are not only being used by organizations but are also being sold as products today.

This means that online applications have gained the trust of customers and users regarding their vital feature named SECURITY. No doubt, that security factor is of primary value for desktop applications too.

However, when we talk about the web, the importance of security increases exponentially. If an online system cannot protect the transaction data, then no one will ever think of using it. Security is neither a word in search of its definition yet, nor a subtle concept. However, we would like to list some compliments on security.

**Examples of Security flaws in an application**

* A Student Management System is insecure if the Admission branch can edit the data of the ‘Exam’ branch.
* An ERP system is not secure if a DEO (data entry operator) can generate ‘Reports’.
* An online Shopping Mall has no security if the customer’s Credit Card Details are not encrypted.
* A custom software possesses inadequate security if an SQL query retrieves actual passwords of its users.

**Security – Meaning**

**“Security means that authorized access is granted to protected data and unauthorized access is restricted”.**

So, it has two major aspects – first is the protection of data and the second one is access to that data. Moreover, whether the application is desktop or web-based, security revolves around the two aforementioned aspects.

**Let’s have an overview of security aspects for both desktop and web-based software applications.**

## Desktop And Web Security Testing

A desktop application should be secure not only regarding its access but also with respect to the organization and storage of its data.

Similarly, web applications demand, even more, security with respect to its access, along with data protection. A web developer should make the application immune to SQL Injections, Brute Force Attacks and XSS (cross-site scripting). Similarly, if the web application facilitates remote access points then these must be secure too.

Also, keep in mind that Brute Force Attack is not only related to web applications, but the desktop software is also vulnerable to this.

I hope this foreword is enough and now let me come to the point. Kindly accept my apology if you have so far thought that you are reading about the subject of this article. Though I have briefly explained software security and its major concerns, my topic is “Security Testing”.

**Recommended Reading=>**[**Web Application Security Testing**](https://www.softwaretestinghelp.com/security-testing-of-web-applications/)

I will now explain how the features of security are implemented in software applications and how these should be tested. My focus will be on what’s and how’s of security testing, not on security.

### Recommended Security Testing Tools

**#1) Netsparker**

[](https://www.netsparker.com/netsparker-web-application-security-issues/?utm_medium=3rdparty&utm_source=softwaretestinghelp&utm_campaign=web-app-sec&utm_content=banner)

[**Netsparker**](https://www.netsparker.com/netsparker-web-application-security-issues/?utm_medium=3rdparty&amp;utm_source=softwaretestinghelp&amp;utm_campaign=web-app-sec&amp;utm_content=banner) is a web application security testing solution with the capabilities of automatic crawling and scanning for all types of legacy & modern web applications such as HTML5, Web 2.0, and Single Page Applications. It makes use of Proof-Based Scanning Technology and scalable scanning agents.

It gives you complete visibility even though you have a large number of assets to manage. It has many more functionalities like team management and vulnerability management. It can be integrated into CI/CD platforms like Jenkins, TeamCity, or Bamboo.

**=>**[**Try the best Netsparker Security Test Tool**](https://www.netsparker.com/netsparker-web-application-security-issues/?utm_medium=3rdparty&utm_source=softwaretestinghelp&utm_campaign=web-app-sec&utm_content=banner)

**#2)** **Indusface WAS Free Website Malware Check**

[](https://www.softwaretestinghelp.com/Indusface-Security-Testing-Technique)

**Indusface WAS**provides both manual penetration testing bundled with its own automated web application vulnerability scanner that detects and reports vulnerabilities based on OWASP top 10 and also includes a Website reputation check of links, malware, and defacement checks of the website in every scan.

**=>**[**Run a Quick Website Scan for Free**](https://www.softwaretestinghelp.com/Indusface-Security-Testing-Technique)

**=>**[**Contact us**](https://www.softwaretestinghelp.com/contact/)**to suggest a listing here.**

## List Of Top 8 Security Testing Techniques

### #1) Access to Application

Whether it is a desktop application or a website, access security is implemented by **“Roles and Rights Management”.** It is often done implicitly while covering functionality.

**For Example,** in a Hospital Management System, a receptionist is least concerned about the laboratory tests as his job is to just register the patients and schedule their appointments with doctors.

So, all the menus, forms and screens related to lab tests will not be available to the Role of ‘Receptionist’. Hence, the proper implementation of roles and rights will guarantee the security of access.

**How to Test:** In order to test this, thorough testing of all roles and rights should be performed.

The tester should create several user accounts with different as well as multiple roles. He should then be able to use the application with the help of these accounts and should verify that every role has access to its own modules, screens, forms, and menus only. If the tester finds any conflict, then he should log a security issue with complete confidence.

**This can also be understood as authentication and authorization testing which is very beautifully depicted in the below image:**

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2011/09/Access-to-Application.jpg)

So, basically, you need to test about ‘who you are’ and ‘what you can do’ for distinct users.

Some of the authentication tests include a test for password quality rules, test for default logins, test for password recovery, test captcha, test for logout functionality, test for password change, test for security question/answer, etc.

Similarly, some of the authorization tests include a test for path traversal, test for missing authorization, test for horizontal access control problems, etc.

### #2) Data Protection

There are three aspects of data security. The first one is that **a user can view or utilize only the data which he is supposed to use**. This is also ensured by roles and rights

**For Example,** TSR (telesales representative) of a company can view the data of available stock, but cannot see how much raw material was purchased for production.

So, this aspect of security testing is already explained above. The second aspect of data protection is related to **how that data is stored in the DB**.

**Further reading =>>**[**What is Database Security Testing**](https://www.softwaretestinghelp.com/database-security-testing/)

All sensitive data must be encrypted to make it secure. Encryption should be strong, especially for sensitive data like passwords of user accounts, credit card numbers or other business-critical information.

The third and the last aspect is an extension of this second aspect. Proper security measures must be adopted when the flow of sensitive or business-critical data occurs. Whether this data floats between different modules of the same application or is transmitted to different applications, it must be encrypted to keep it safe.

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2011/09/Data-Protection.jpg)

**How to Test Data Protection:** The tester should query the database for ‘passwords’ of the user account, billing information of clients, other business-critical and sensitive data, should verify that all such data is saved in encrypted form in the DB.

Similarly, he must verify that the data is transmitted between different forms or screens after proper encryption only. Moreover, the tester should ensure that the encrypted data is properly decrypted at the destination. Special attention should be paid to different ‘submit’ actions.

The tester must verify that when the information is being transmitted between the client and server, it is not displayed in the address bar of a web browser in an understandable format. If any of these verifications fail, then the application definitely has a security flaw.

The tester should also check for proper use of salting (appending an extra secret value to the end input like password and thus making it stronger and more difficult to be cracked).

Insecure randomness should also be tested as it is a kind of vulnerability. Another way to test data protection is to check for weak algorithm usage.

**For example,** since HTTP is a clear text protocol, if sensitive data like user credentials are transmitted via HTTP, then it is a threat to application security. Instead of HTTP, sensitive data should be transferred via HTTPS (secured through SSL and TLS tunnels).

However, HTTPS increases the attack surface and thus it should be tested that the server configurations are proper and certificate validity is ensured.

### #3) Brute-Force Attack

Brute Force Attack is mostly done by some software tools. The concept is that by using a valid user ID, the s**oftware attempts to guess the associated password by trying to log in again and again.**

A simple example of security against such an attack is account suspension for a short period of time, as all mailing applications like Yahoo, Gmail and Hotmail do. If a specific number of consecutive attempts (mostly 3) fail to log in successfully, then that account is blocked for some time (30 minutes to 24 hrs).

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2011/09/Brute-Force-Attack.jpg)

**How to test Brute-Force Attack:** The tester must verify that some mechanism of account suspension is available and is working accurately. (S)He must attempt to login with invalid user IDs and Passwords alternatively to make sure that the software application blocks the account if continuous attempts are made to login with invalid credentials.

If the application is doing so, then it is secure against brute-force attack. Otherwise, this security vulnerability must be reported by the tester.

Testing for brute force can also be divided into two parts – black box testing and grey-box testing.

In Black box testing, the authentication method employed by the application is discovered and tested. Furthermore, the grey box testing is based on partial knowledge of password & account details and memory trade-off attacks.

Click [here](https://www.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004)#Gray_Box_testing_and_example) to explore the black box & grey box brute force testing along with examples.

**The above three security aspects should be taken into account for both web and desktop applications while the following points are related to web-based applications only.**

### #4) SQL Injection And XSS (Cross-Site Scripting)

### [SQL Injection](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2011/09/SQL-Injection.jpg)

Conceptually speaking, the theme of both these hacking attempts is similar, hence these are discussed together. In this approach, the **malicious script is used by hackers in order to manipulate a website**.

There are several ways to immune against such attempts. For all input fields on the website, field lengths should be defined small enough to restrict the input of any script

**For example,** the Last Name should have a field length of 30 instead of 255. There may be some input fields where large data input is necessary, for such fields proper validation of input should be performed prior to saving that data in the application.

Moreover, in such fields, any HTML tags or script tag input must be prohibited. In order to provoke XSS attacks, the application should discard script redirects from unknown or untrusted applications.

**How to**[**test SQL Injection**](https://www.softwaretestinghelp.com/sql-injection-%E2%80%93-how-to-test-application-for-sql-injection-attacks/)**and XSS:** Tester must ensure that maximum lengths of all input fields are defined and implemented. (S)He should also ensure that the defined length of input fields does not accommodate any script input as well as tag input. Both of these can be easily tested.

**For Example,** If 20 is the maximum length specified for the ‘Name’ field, and input string “<p>thequickbrownfoxjumpsoverthelazydog” can verify both these constraints.

It should also be verified by the tester that the application does not support anonymous access methods. If any of these vulnerabilities exist, then the application is in danger.

**Basically, SQL injection testing can be done through the following five ways:**

* Detection techniques
* Standard SQL injection techniques
* Fingerprint the database
* Exploitation Techniques
* SQL Injection Signature Invasion Techniques

Click [here](https://www.owasp.org/index.php/Testing_for_SQL_Injection_(OTG-INPVAL-005)#Alternative_Expression_of_.27or_1_.3D_1.27) to read in detail about the above ways to test SQL injection.

XSS is also a type of injection which injects malicious script into a website. Click [here](https://www.owasp.org/index.php/Testing_for_Cross_site_scripting) to explore in-depth about testing for XSS.

### #5) Service Access Points (Sealed and Secure Open)

### [Service Access Points (Sealed and Secure Open)](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2011/09/Service-Access-Points-Sealed-and-Secure-Open.jpg)

Today, businesses depend and collaborate with each other, the same holds good for applications especially websites. In such a case, both the collaborators should define and publish some access points for each other.

So far the scenario seems quite simple and straightforward but, for some web-based products like stock trading, things are not so simple and easy.

If there is a large target audience, then the access points should be open enough to facilitate all users, accommodating enough to fulfill all users’ requests and secure enough to cope with any security-trial.

**How to Test Service Access Points:** Let me explain it with the **example** of the stock trading web application; an investor (who wants to purchase the shares) should have access to current and historical data on stock prices. The user should be given the facility to download this historical data. This demands that the application should be open enough.

By accommodating and secure, I mean that the application should facilitate investors to trade freely (under the legislative regulations). They may purchase or sale 24/7 and the data of transactions must be immune to any hacking attack.

Moreover, a large number of users will be interacting with the application simultaneously, so the application should provide enough access points to entertain all the users.

In some cases, these **access points can be sealed for unwanted applications or people**. This depends on the business domain of the application and its users.

**For example,** a custom web-based Office Management System may recognize its users on the basis of IP Addresses and denies establishing a connection with all the other systems (applications) that do not fall in the range of valid IPs for that application.

The tester must ensure that all **inter-network and intra-network access** to the application is through trusted applications, machines (IPs) and users.

In order to verify that an open access point is secure enough, the tester must try to access it from different machines having both trusted and untrusted IP addresses.

Different sorts of real-time transactions should be tried in bulk to have good confidence in the application’s performance.  By doing so, the capacity of access points of the application will also be observed clearly.

The tester must ensure that the application entertains all communication requests from trusted IPs and applications only while all other requests are rejected.

Similarly, if the application has some open access point, then the tester should ensure that it allows (if required) uploading of data by users in a secure way. In this secure way, I mean about the file size limit, file type restriction and scanning of the uploaded file for viruses or other security threats.

This is how a tester can verify the security of an application with respect to its access points.

### #6) Session Management

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2011/09/Session-Management.jpg)

A web session is a sequence of HTTP requests and response transactions linked to the same user. Session management tests check how session management is handled in the web app.

You can test for session expiry after particular idle time, session termination after maximum lifetime, session termination after log out, check for session cookie scope and duration, testing if a single user can have multiple simultaneous sessions, etc.

### #7) Error handling

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2011/09/Error-handling.jpg)

**Testing for Error handling includes:**

**Check for error codes**: **For example,** test 408 request time-out, 400 bad requests, 404 not found, etc. To test this, you need to make certain requests on the page such that these error codes are returned.

The error code will be returned with a detailed message. This message should not contain any critical information that can be used for hacking purposes

**Check for stack traces**: It basically includes giving some exceptional input to the application such that the returned error message contains stack traces that have interesting information for hackers.

### #8) Specific Risky Functionalities

Mainly, the two risky functionalities are **payments** and **file uploads**. These functionalities should be tested very well. For file uploads, you need to primarily test if any unwanted or malicious file upload is restricted.

For payments, you need to primarily test for injection vulnerabilities, insecure cryptographic storage, buffer overflows, password guessing, etc.

**=>**[**Contact us**](https://www.softwaretestinghelp.com/contact/)**to suggest a listing here.**

To Get Started with  
Security Testing  
, I mainly recommend starting from scratch and I have written a blog post on Work2Code which can help anyone who wants to start “Penetration Testing” or “Security Testing”.

Security Testing & Mindset – Q&A With Santhosh Tuppad  
<https://work2code.com/application-security-testing-mindset-with-santhosh-tuppad/> [Text]

VIDEO: GETTING STARTED WITH SECURITY TESTING  
<https://work2code.com/video-getting-started-with-security-testing/> [Video]

I hope this information helps you to re-think about your passion and also get started if you decide to start “Penetration Testing”. Please write to me if you have any questions.

### Security Testing Test Scenarios

1. Check for SQL injection attacks.  
2. Secure pages should use the HTTPS protocol.  
3. Page crash should not reveal application or server info. The error page should be displayed for this.  
4. Escape special characters in the input.  
5. Error messages should not reveal any sensitive information.  
6. All credentials should be transferred over to an encrypted channel.  
7. Test password security and password policy enforcement.  
8. Check the application logout functionality.  
9. Check for Brute Force Attacks.  
10. Cookie information should be stored in encrypted format only.  
11. Check session cookie duration and session termination after timeout or logout.  
11. Session tokens should be transmitted over a secured channel.  
13. The password should not be stored in cookies.  
14. Test for Denial of Service attacks.  
15. Test for memory leakage.  
16. Test unauthorized application access by manipulating variable values in the browser address bar.  
17. Test file extension handling so that exe files are not uploaded or executed on the server.  
18. Sensitive fields like passwords and credit card information should not have to be autocomplete enabled.  
19. File upload functionality should use file type restrictions and also anti-virus for scanning uploaded files.  
20. Check if directory listing is prohibited.  
21. Passwords and other sensitive fields should be masked while typing.  
22. Check if forgot password functionality is secured with features like temporary password expiry after specified hours and security questions are asked before changing or requesting a new password.  
23. Verify CAPTCHA functionality.  
24. Check if important events are logged in log files.  
25. Check if access privileges are implemented correctly.

**Penetration Testing test cases** – I’ve listed around 41 test cases for Penetration Testing on [this page](https://www.softwaretestinghelp.com/penetration-testing-guide/).

I ‘d really like to thank ***Devanshu Lavaniya*** (Sr. QA Engineer working for I-link Infosoft) for helping me to prepare this comprehensive testing checklist.

I’ve tried to cover almost all standard test scenarios for Web and Desktop application functionality. I still know that this is not a complete checklist. Testers on different projects have their own testing checklist based on their experience.

[180+ Sample Test Cases for Testing Web and Desktop Applications - Comprehensive Software Testing Checklist (softwaretestinghelp.com)](https://www.softwaretestinghelp.com/sample-test-cases-testing-web-desktop-applications/#Security_Testing_Test_Scenarios)

## Test Cases for Security Testing

Try to directly access bookmarked web page without login to the system.

2. Verify that system should restrict you to download the file without sign in on the system.

3. Verify that previous accessed pages should not accessible after log out i.e. Sign out and then press the Back button to access the page accessed before.

4. Check the valid and invalid passwords, password rules say cannot be less than 6 characters, user id and password cannot be the same etc.

5. Verified that important i.e. sensitive information such as passwords, ID numbers, credit card numbers, etc should not get displayed in the input box when typing. They should be encrypted and in asterix format.

6 .Check Is bookmarking disabled on secure pages? Bookmarking Should be disabled on secure pages.

7. Check Is Right Click, View, Source disabled? Source code should not be visible to user.

8. Is there an alternative way to access secure pages for browsers under version 3.0, since SSL is not compatible with those browsers?

9. Check does your server lock out an individual who has tried to access your site multiple times with invalid login/password information?

10. Verify the timeout condition, after timeout user should not able to navigate through the site.

11. Check Are you prevented from doing direct searches by editing content in the URL?

12. Verify that relevant information should be written to the log files and that information should be traceable.

13. In SSL verify that the encryption is done correctly and check the integrity of the information.

14. Verify that restricted page should not be accessible by user after session time out.

15. ID / password authentication, the same account on different machines cannot log on at the same time. So at a time only one user can login to the system with a user id.

Advertisements

REPORT THIS AD

16. ID / password authentication methods entered the wrong password several times and check if the account gets locked.

17. Add or modify important information (passwords, ID numbers, credit card number, etc.). Check if it gets reflected immediately or caching the old values.

18. Verify that Error Message does not contain malicious info so that hacker will use this information to hack web site.

[Test Cases for Security Testing | tfortesting (wordpress.com)](https://tfortesting.wordpress.com/2012/10/08/test-cases-for-security-testing/)