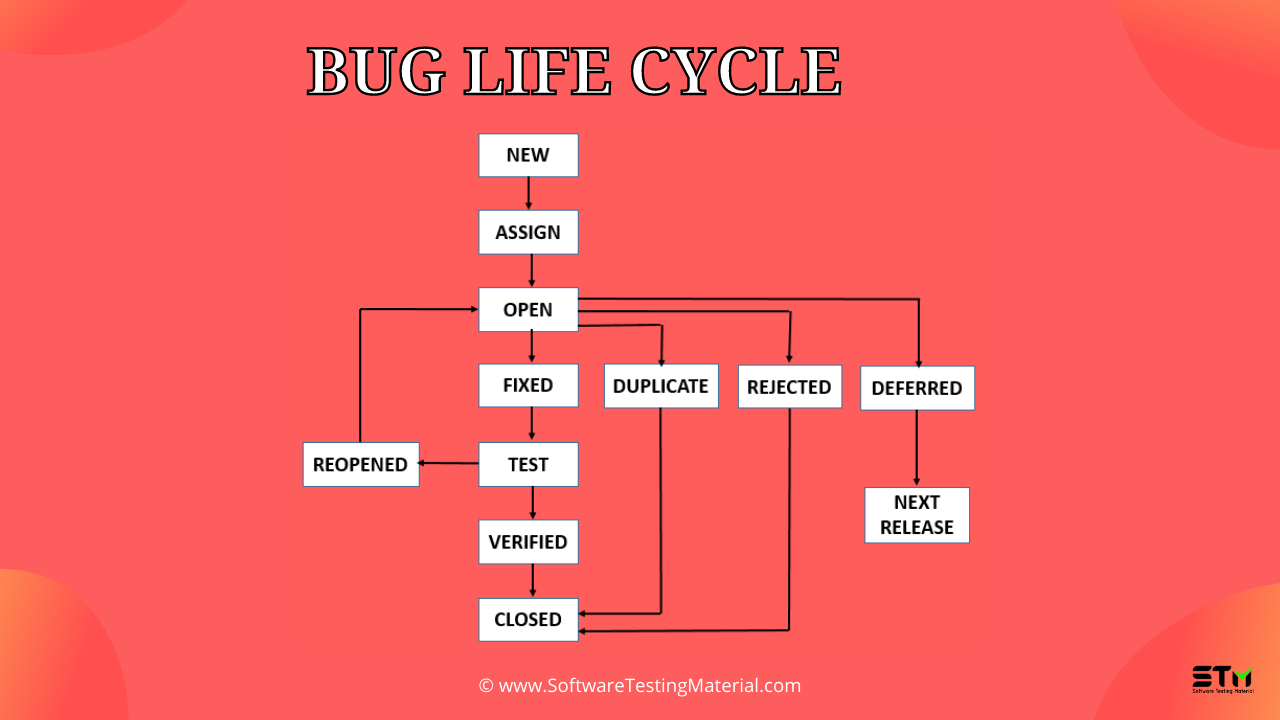
**Testing Concepts**

**Name : Nikita Ovhal**

**Batch no :2022-7669**

**Bug Life Cycle**

****

1. New

When a tester finds a new defect. He should provide a proper Defect document to the Development team to reproduce and fix the defect. In this state, the status of the defect posted by the tester is “New”

2. Assigned

Defects that are in the status of New will be approved (if valid) and assigned to the development team by Test Lead/Project Lead/Project Manager. Once the defect is assigned then the status of the bug changes to “Assigned”

3. Open

The development team starts analyzing and works on the defect fix

4. Fixed

When a developer makes the necessary code change and verifies the change, then the status of the bug will be changed as “Fixed” and the bug is passed to the testing team.

5. Test: If the status is “Test”, it means the defect is fixed and ready to do test whether it is fixed or not.

6. Verified

The tester re-tests the bug after it got fixed by the developer. If there is no bug detected in the software, then the bug is fixed and the status assigned is “verified.”

7. Closed

After verified the fix, if the bug is no longer exits then the status of the bug will be assigned as “Closed.”

8. Reopen

If the defect remains the same after the retest, then the tester posts the defect using the defect retesting document and changes the status to “Reopen”. Again the bug goes through the life cycle to be fixed.

9. Duplicate

If the defect is repeated twice or the defect corresponds to the same concept of the bug, the status is changed to “duplicate” by the development team.

10. Deferred

In some cases, the Project Manager/Lead may set the bug status as deferred.

• If the bug found during the end of the release and the bug is minor or not important to fix immediately.

• If the bug is not related to the current build.

• If it is expected to get fixed in the next release.

• The customer is thinking to change the requirement.

11. Rejected

If the system is working according to specifications and the bug is just due to some misinterpretation (such as referring to old requirements or extra features) then the Team lead or developers can mark such bugs as “Rejected”

**Levels of Testing**

**Level1: Unit Testing**

Unit testing is the first level of software testing, which is used to test if software modules are satisfying the given requirement or not. The first level of testing involves analyzing each unit or an individual component of the software application.

Unit testing is also the first level of functional testing. The primary purpose of executing unit testing is to validate unit components with their performance.

**Level2: Integration Testing**

The second level of software testing is the integration testing. The integration testing process comes after unit testing.It is mainly used to test the data flow from one module or component to other modules.

**Level3: System Testing**

The third level of software testing is system testing, which is used to test the software's functional and non-functional requirements. It is end-to-end testing where the testing environment is parallel to the production environment. In the third level of software testing, we will test the application as a whole system. To check the end-to-end flow of an application or the software as a user is known as System testing.

**Level4: Acceptance Testing**

The last and fourth level of software testing is acceptance testing, which is used to evaluate whether a specification or the requirements are met as per its delivery. The software has passed through three testing levels (Unit Testing, Integration Testing, System Testing). Some minor errors can still be identified when the end-user uses the system in the actual scenario.

In simple words, we can say that Acceptance testing is the squeezing of all the testing processes that are previously done.

* Alpha Testing: Testing or verifying product at organization.
* Beta Testing: Checking the product in customer’s place

**There are seven principles in software testing:**

1. Testing shows the presence of defects
2. Exhaustive testing is not possible
3. Early testing
4. Defect clustering
5. Pesticide paradox
6. Testing is context-dependent
7. Absence of errors fallacy

* **Testing shows the presence of defects**: The goal of software testing is to make the software fail. Software testing reduces the presence of defects. Software testing talks about the presence of defects and doesn’t talk about the absence of defects. Software testing can ensure that defects are present but it cannot prove that software is defect-free. Even multiple testing can never ensure that software is 100% bug-free. Testing can reduce the number of defects but not remove all defects.
* **Exhaustive testing is not possible:** It is the process of testing the functionality of the software in all possible inputs (valid or invalid) and pre-conditions is known as exhaustive testing. Exhaustive testing is impossible means the software can never test at every test case. It can test only some test cases and assume that the software is correct and it will produce the correct output in every test case. If the software will test every test case then it will take more cost, effort, etc., which is impractical.
* **Early testing**: To find the defect in the software, early test activity shall be started. The defect detected in the early phases of SDLC will be very less expensive. For better performance of software, software testing will start at the initial phase i.e. testing will perform at the requirement analysis phase.
* **Defect clustering:** In a project, a small number of modules can contain most of the defects. Pareto Principle to software testing state that 80% of software defect comes from 20% of modules. Grouping and combining similar type of defect together.
* **Pesticide paradox:** Repeating the same test cases, again and again, will not find new bugs. So it is necessary to review the test cases and add or update test cases to find new bugs.
* **Testing is context-dependent:** The testing approach depends on the context of the software developed. Different types of software need to perform different types of testing. For example, The testing of the e-commerce site is different from the testing of the Android application.
* **Absence of errors fallacy:** If a built software is 99% bug-free but it does not follow the user requirement then it is unusable. It is not only necessary that software is 99% bug-free but it is also mandatory to fulfill all the customer requirements.
* You should not say it is 100% bug free . you can say 99% bug free.

**Black Box Testing:**

Black Box Testing is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is also known as Behavioral Testing.



Black Box Testing Types

The following are the several categories of black box testing:

* Functional Testing
* Regression Testing
* Nonfunctional Testing (NFT)

**Functional Testing:** This black box testing type is related to the functional requirements of a system; it is done by software testers.

**Regression Testing:** It ensures that the newly added code is compatible with the existing code. In other words, a new software update has no impact on the functionality of the software. This is carried out after a system maintenance operation and upgrades.

**Nonfunctional Testing:** Nonfunctional testing is also known as NFT. This testing is not functional testing of software. It focuses on the software’s performance, usability, and scalability.

Black Box Testing Techniques

* **Equivalence Class Testing:** It is used to minimize the number of possible test cases to an optimum level while maintains reasonable test coverage.
* **Boundary Value Testing**: Boundary value testing is focused on the values at boundaries. This technique determines whether a certain range of values are acceptable by the system or not. It is very useful in reducing the number of test cases. It is most suitable for the systems where an input is within certain ranges.
* **Decision Table Testing:** A decision table puts causes and their effects in a matrix. There is a unique combination in each column.

Advantages of Black Box Testing:

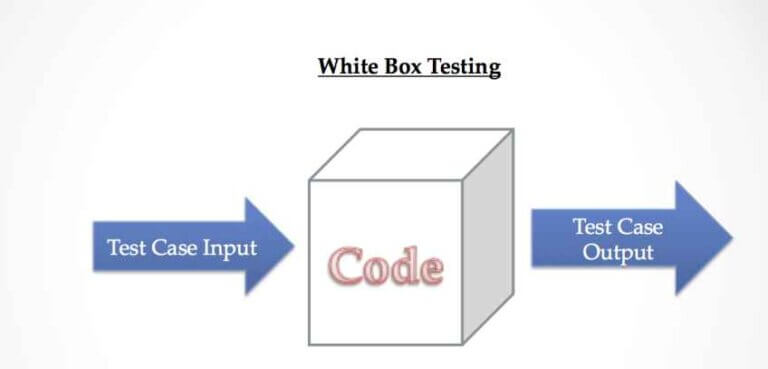
* The tester does not need to have more functional knowledge or programming skills to implement the Black Box Testing.
* It is efficient for implementing the tests in the larger system.
* Tests are executed from the user’s or client’s point of view.
* Test cases are easily reproducible.
* It is used in finding the ambiguity and contradictions in the functional specifications.

Disadvantages of Black Box Testing:

* There is a possibility of repeating the same tests while implementing the testing process.
* Without clear functional specifications, test cases are difficult to implement.
* It is difficult to execute the test cases because of complex inputs at different stages of testing.
* Sometimes, the reason for the test failure cannot be detected.
* Some programs in the application are not tested.
* It does not reveal the errors in the control structure.
* Working with a large sample space of inputs can be exhaustive and consumes a lot of time.

**White Box Testing**

White Box Testing is a testing technique in which software’s internal structure, design, and coding are tested to verify input-output flow and improve design, usability, and security. In white box testing, code is visible to testers, so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing, and Glass box testing.



**Working process of white box testing:**

* **Input:** Requirements, Functional specifications, design documents, source code.
* **Processing:** Performing risk analysis for guiding through the entire process.
* **Proper test planning:** Designing test cases so as to cover entire code. Execute rinse-repeat until error-free software is reached. Also, the results are communicated.
* **Output:** Preparing final report of the entire testing process.

Following are important WhiteBox Testing Techniques:

* Statement Coverage
* Decision Coverage
* Branch Coverage
* Condition Coverage
* Multiple Condition Coverage
* Finite State Machine Coverage
* Path Coverage
* Control flow testing
* Data flow testing

Advantages:

* White box testing is very thorough as the entire code and structures are tested.
* It results in the optimization of code removing error and helps in removing extra lines of code.
* It can start at an earlier stage as it doesn’t require any interface as in case of black box testing.
* Easy to automate.

Disadvantages:

* Main disadvantage is that it is very expensive.
* Redesign of code and rewriting code needs test cases to be written again.
* Testers are required to have in-depth knowledge of the code and programming language as opposed to black box testing.
* Missing functionalities cannot be detected as the code that exists is tested.
* Very complex and at times not realistic.
  1. Read a , b, c
  2. If(a>= b && a>= c);
  3. Print a;
  4. Else if(b >=a && b>=c)
  5. Print b;
  6. Else
  7. Print c;

Data flow testing :

3 , 2 ,6

Path = 1,6,7

Output= 6

Control flow testing:

A

Print b is largest

Print c is largest

If a>c

If a>b

1

Read a ,b ,c

B

2

False C True

If b>c

3 4

D E

Print a is largest

5 6 7 8

F G H

10 11

9

I

Branch coverage testing:

For true :A1-B2-C4-E8-H11-I

For false: A1-B2-3D-5F-9I

Branch coverage=No. of paths

Branch coverage = 2

Statement coverage:

Input 7 , 9 ,4

Statement coverage = 3/7\*100

300/7 = 42%

Decision coverage testing:

Decision coverage = ½ \*100

Decision coverage=50%

**Gray Box Testing:**

Gray box testing is a software testing technique to test a software product or application with partial knowledge of internal structure of the application. The purpose of grey box testing is to search and identify the defects due to improper code structure or improper use of applications.

Gray Box Testing is a software testing method, which is a combination of both White Box Testing and Black Box Testing method.

It is primarily used in integration testing and penetration testing.

Techniques used for Grey box Testing are-

**Matrix Testing**

This testing technique comes under Grey Box testing. It defines all the used variables of a particular program. In any program, variable are the elements through which values can travel inside the program. It should be as per requirement otherwise, it will reduce the readability of the program and speed of the software. Matrix technique is a method to remove unused and uninitialized variables by identifying used variables from the program.

**Orthogonal Array Testing**

The purpose of this testing is to cover maximum code with minimum test cases. Test cases are designed in a way that can cover maximum code as well as GUI functions with a smaller number of test cases.

**Pattern Testing**

Pattern testing is applicable to such type of software that is developed by following the same pattern of previous software. In these type of software possibility to occur the same type of defects. Pattern testing determines reasons of the failure so they can be fixed in the next software.

**Difference Between Verification and Validation**

|  |  |
| --- | --- |
| **VERIFICATION** | **VALIDATION** |
| We check whether we are developing the product right | We check whether the developed right product |
| Verification is also known as static testing. | Validation is also known as dynamic testing. |
| Verification includes different methods like Inspections, Reviews, and Walkthroughs. | Validation includes testing like functional testing, system testing, integration, and User acceptance testing. |
| Verification is done before the validation testing. | After verification testing, validation testing takes place |
| The execution of code does not happen in the verification testing. | In validation testing, the execution of code happens. |
| Quality assurance comes under verification testing. | Quality control comes under validation testing. |

**Difference Between Black box and White Box Testing**

|  |  |
| --- | --- |
| **Black Box Testing** | **White Box Testing** |
| It is a testing approach which is used to test the software without the knowledge of the internal structure of program or application. | It is a testing approach in which internal structure is known to the tester. |
| Programming knowledge is not needed to perform Black Box testing. | Programming knowledge is needed to perform White Box testing. |
| It is less exhaustive and time-consuming. | Exhaustive and time-consuming method. |
| Black Box Testing is also known as functional testing, data-driven testing, and closed-box testing. | It is also known as structural testing, clear box testing, code-based testing, and transparent testing. |
| It is done at higher levels of testing that are system testing and acceptance testing. | It is done at lower levels of testing that are unit testing and integration testing. |
| It is mainly performed by the software testers. | It is mainly performed by developers and tester |

**Test Scenario:**

A Test Scenario is defined as any functionality that can be tested. It is also called Test Condition or Test Possibility. As a tester, you should put yourself in the end user’s shoes and figure out the real-world scenarios and use cases of the Application Under Test.

Steps :

Step 1: Read the Requirement Documents like BRS, SRS, FRS, of the System Under Test (SUT). You could also refer uses cases, books, manuals, etc. of the application to be tested.

Step 2: For each requirement, figure out possible users actions and objectives. Determine the technical aspects of the requirement. Ascertain possible scenarios of system abuse and evaluate users with hacker’s mindset.

Step 3: After reading the Requirements Document and doing your due Analysis, list out different test scenarios that verify each feature of the software.

Step 4: Once you have listed all possible Test Scenarios, a Traceability Matrix is created to verify that each & every requirement has a corresponding Test Scenario

Step 5: The scenarios created are reviewed by your supervisor. Later, they are also reviewed by other Stakeholders in the project.

**Test Case:**

A Test Case is a set of actions executed to verify a particular feature or functionality of your software application. A Test Case contains test steps, test data, precondition, and postcondition developed for a specific test scenario to verify any requirement. The test case includes specific variables or conditions, using which a testing engineer can compare expected and actual results to determine whether a software product is functioning as per the requirements of the customer.

**Test Basis**

Test basis is defined as the source of information or the document that is needed to write test cases and also for test analysis.

Test basis should be well defined and adequately structured so that one can easily identify test conditions from which test cases can be derived.

## Typical Test Basis:

* Requirement document
* Test Plan
* Codes Repository
* Business Requirement

**Test Suite:**

Test suite is a container that has a set of tests which helps testers in executing and reporting the test execution status. It can take any of the three states namely Active, Inprogress and completed.

A Test case can be added to multiple test suites and test plans. After creating a test plan, test suites are created which in turn can have any number of tests.

Test suites are created based on the cycle or based on the scope. It can contain any type of tests, viz - functional or Non-Functional.

FUNCTIONAL TESTINGS:

**1 .ASSERTION TESTING:**

An assertion is a Boolean expression. It is a concept of functional testing. It is used to test a logical expression. An assertion is true if the logical expression that is being tested is true and there are no bugs in the program. Assertion testing can be used at any particular stage of the program.

EX: 3<4=true, assertion true and logical expression true then their is no bugs.

7>10=false, assertion false and logical expression false then bugs are their.

**2. Gorilla Testing:**

Gorilla testing is a software testing technique that repeatedly applies inputs on a module to ensure it is functioning correctly and that there are no bugs.A module can be tested over a hundred times, and in the same manner. So, Gorilla Testing is also known as “Frustrating Testing” or fault tolerance testing. Gorilla testing is a manual testing procedure and is performed on selected modules of the software system with selected test cases. Gorilla testing is mainly used in the Unit testing part of the testing phase, which concerns testing independent units of source code.

**3. SANITY TESTING:**

Sanity testing is performed on stable builds and it is also known as a variant of regression testing. Sanity testing was performed when we are receiving software build (with minor code changes) from the development team. It is a checkpoint to assess if testing for the build can proceed or not.

In other words, we can say that sanity testing is performed to make sure that all the defects have been solved and no added issues come into the presence because of these modifications.

**4. MONKEY TSETING:**

Monkey Testing is a software testing technique in which the tester enters any random inputs into the software application without predefined test cases and checks the behavior of the software application, whether it crashes or not. The purpose of Monkey testing is to find the bugs and errors in the software application using experimental techniques.

**5. SMOKE TESTING:**

Smoke Testing is a software testing process that determines whether the deployed software build is stable or not. Smoke testing is a confirmation for QA team to proceed with further software testing. It consists of a minimal set of tests run on each build to test software functionalities. Smoke testing is also known as “Build Verification Testing” or “Confidence Testing.”

**6. EXPLORATOY TESTING:**

Exploratory testing? If requirement does not exist, then we do one round of exploratory testing. So, for this first, we will be exploring the application in all possible ways, understanding the flow of the application, preparing a test document and then testing the application, this approach is known as exploratory testing.

**7 . MUTATION TESTING:**

Mutation Testing is a type of software testing in which certain statements of the source code are changed/ mutated to check if the test cases are able to find errors in source code. The goal of Mutation Testing is ensuring the quality of test cases in terms of robustness that it should fail the mutated source code.

**8 .BENCHMARK TESTING:**

A Benchmark in Performance Testing is a metric or a point of reference against which software products or services can be compared to assess the quality measures.

In other words, Benchmark means a set standard that helps to determine the quality of a software product or service.

The user experience cannot be quantified in numbers, but the time a user spends on a webpage due to good UI can be quantified.

Benchmark Testing is not a term related to just software testing, but it also deals with Hardware Testing

NON-FUNCTIONAL TESTINGS:

**1 . LOAD TSETING:**

Load Testing is a non-functional software testing process in which the performance of software application is tested under a specific expected load. It determines how the software application behaves while being accessed by multiple users simultaneously. The goal of Load Testing is to improve performance bottlenecks and to ensure stability and smooth functioning of software application before deployment.

Load Testing Tools:

1. Apache JMeter

2. WebLoad

3. NeoLoad

4. Load Ninja

5. HP Performance Tester

6. LoadUI Pro

7. LoadView

**2 .STRESS TESTING:**

Stress testing (sometimes called torture testing) is a form of deliberately intense or thorough testing used to determine the stability of a given system, critical infrastructure or entity. Stress testing involves testing the application under varying load. Extremely large numbers of concurrent users try to log into the application. Database linked to the website shuts down when the website tries to reach it from the front end. Data in added in extremely large quantity in the database. Stress Testing is a type of software testing that verifies stability & reliability of software application. The goal of Stress testing is measuring software on its robustness and error handling capabilities under extremely heavy load conditions and ensuring that software doesn’t crash under crunch situations.

**3. USABILITY TSETING:**

Usability testing refers to evaluating a product or service by testing it with representative users. Typically, during a test, participants will try to complete typical tasks while observers watch, listen and takes notes.

To check the usability or ease of using a software product. Checking the user-friendliness, efficiency, and accuracy of the application is known as Usability Testing.

Parameters:

Efficiency

Memorability

Accuracy

Learnability

Satisfaction

Errors

**4. PERFORMANCE TESTING:**

Performance testing is in general a testing practice performed to determine how a system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage.

Types of Performance Testing:

• Load

• Stress

• Spike

• Endurance

• Scalability

• Volume

**5. VOLUME TESTING:**

Volume Testing is a type of software testing which is carried out to test a software application with a certain amount of data. In volume testing a huge volume of data is acted upon the software. It is basically performed to analyze the performance of the system by increasing the volume of data in the database. Volume testing is performed to study the impact on response time and behavior of the system when the volume of data is increased in the database. Volume Testing is also known as Flood Testing.

**6. SCALABILITY TESTING:**

scalability testing, which comes under the non-functional testing of software testing. It is used to check an application's performance by increasing or decreasing the load in particular scales known as scalability testing. It is executed at a hardware, software, or database level.

**7 .SECURITY TESTING:**

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 cannot be exploited. It also helps in detecting all possible security risks in the system and helps developers to fix the problems through coding.

Types of security Vulnerability Scanning. ...

Security Scanning. ...

Penetration Testing. ...

Security Audit/ Review. ...

Ethical Hacking. ...

Risk Assessment. ...

Posture Assessment. ...

Authentication.

**Security Testing**

KeyTerms used in Security Testing

**URL Manipulation:**

One of the popular ways to hack a website is URL manipulation where in hackers manipulate website URL query strings and get access to confidential information. This usually takes place when the application makes use of HTTP GET to pass information between client & server. Information is passed via query string. The tester alters the query parameters to check if is accepted by the server. An URL Manipulation testing ensures that database records are not accessed neither other vital information of the website by unauthorized users.

**SQL Injection:**

One of the other common ways picked by hackers to steal the vital information from the web, the SQL Injection testing ensures all the databases are safe and protected. It is a type of testing that takes the advantages of the loopholes that make the hackers easily pass into the system by passing all possible SQL queries to hack it. They try to query the database using the SQL Injection statements to pull information and crash the system. Even the errors displayed while crashing the system will provide generous amount of important data to the hackers. So, SQL Injection testing is purposed to take care of the input fields like comments, text boxes etc. Special characters are either handled or skipped from the input.

**Vulnerability:**

This is to identify the weakest attributes in the system which might lend easy paths for the malicious software to be attached by unauthorized users. Vulnerability can occur due to bug in software, inaccurate software testing or presence of malicious code. This phase requires fixes, patches to prevent the compromised integrity by malware or hackers.

**Cross-Site Scripting (XSS)**

Cross-site scripting is a computer security vulnerability found in web applications. XSS enables attackers to inject client-side script into Web pages viewed by other users and trick a user into clicking on that URL. Once executed by the other user’s browser, this code could then perform actions such as completely changing the behavior of the website, stealing personal data, or performing actions on behalf of the user.

Types of XSS

* **Stored XSS** − Stored XSS also known as persistent XSS occurs when user input is stored on the target server such as database/message forum/comment field etc. Then the victim is able to retrieve the stored data from the web application.
* **Reflected XSS** − Reflected XSS also known as non-persistent XSS occurs when user input is immediately returned by a web application in an error message/search result or the input provided by the user as part of the request and without permanently storing the user provided data.
* **DOM Based XSS(**Document Object Model-based) − DOM Based XSS is a form of XSS when the source of the data is in the DOM, the sink is also in the DOM, and the data flow never leaves the browser.

**Security Testing Tools**

1. **ACUNETIX**

Acunetix is an end-to-end web application security scanner. This will give you a 360-degree view of the security of your organization. It is capable of detecting 6500 types of vulnerabilities like SQL injections, XSS, and Weak Passwords, etc. It makes use of advanced macro recording technology for scanning complex multi-level forms. The platform is intuitive and easy to use. You can schedule and prioritize full scans as well as incremental scans. It contains a built-in vulnerability management functionality. With the help of CI tools like Jenkins, new builds can be scanned automatically.

1. **INVICTI**

Invicti (formerly Netsparker) is a platform for all web application security testing requirements. This web vulnerability scanning solution has capabilities of vulnerability scanning, vulnerability assessment, and vulnerability management. Invicti is best for scanning precision and unique asset discovery technology. It can be integrated with popular issue management and CI/CD applications. Invicti provides proof of exploit on the identification of vulnerability to confirm that it is not a false positive. It has an advanced scanning engine, advanced crawling authentication features, and WAF integration functionality, etc. With this tool, you will get detailed scanned results with insights on vulnerability.

1. **NetSparker**

NetSparker acts as a one-stop shop for all the web security needs. Available as both hosted as well as self-hosted solution, this platform can be easily integrated completely in any type of test and dev environment. NetSparker has a trade-marked Proof-Based-Scanning technology that uses automation to identify vulnerabilities and verify false positives, thus eliminating the need for unnecessary investment of huge man hours.

1. **ImmuniWeb**

ImmuniWeb is a next-gen platform that employs Artificial Intelligence to enable security testing. This AI-enabled penetration testing platform offers holistic benefits package for security teams, developers, CISOs, as well as CIOs. Having a one-click virtual patching system, this platform assists continuous complaince monitoring. It boasts a proprietary Multilayer Application Security Testing technology and checks a website for compliance, server hardening, and privacy.

1. **SQLMap**

SQLMap is a penetration testing tool, powered by a detection engine for automating identification and exploitation of SQL injection flaws. Encompassing support for a broad spectrum of database management systems and SQL injection techniques, SQLMap automatically recognizes hash-based passwords and supports orchestration of a dictionary-based attack to crack them. With seven levels of verbosity support, it offers ETA support for each query and brings granularity and flexibility for both users’ switches and features. Its fingerprint and enumeration features are valuable in streamlining an effective penetration test run.