Paul Romer

CSD370

Module 7.2 Assignment

Software Testing

1. Attack Surface Analysis

Description: Attack surface analysis involves identifying and mapping all the ways an attacker might use or interface with a system.

Tools Available: Microsoft Attack Surface Analyzer

Expected Results: Documentation that includes a map of the system, indicating what users directly and indirectly interact with for an understanding of the attack surface.

This Order: This is first because, to reliably test a program, you must understand the attack surface.

2. Scanning

Description: Scanning is when you use automated tools to identify characteristics and known vulnerabilities of a system. This can be done at the network, application, and code levels.

Tools Available: Nmap (network), Burp Suite (application), SonarQube(static analysis), OWASP ZAP (dynamic analysis)

Expected Results: A list of vulnerabilities.

This Order: This is second because it's easier to take action to mitigate the list of vulnerabilities from scanning when you have a map of the system and understand the attack surfaces.

3. Fuzzing

Description: Fuzz testing is used to test input validation. It uses brute force to try a large variety of inputs to discover vulnerabilities and issues.

Tools Available: Synopsys Fuzzing Test Suite, Code Intelligence Fuzz, Beyond Security beSTORM

Expected Results: Logs that include any detection of crashes, memory corruption, unexpected exceptions and execution.

This Order: Fuzzing is forth because after an understanding of the system from attack surface analysis and scanning, it's time to start directly testing the functionality through stress-testing like fuzzing.

4. Simulation Testing

Description: Simulation testing is testing a system by recreating realistic scenarios. This is done to ensure that the system functions as expected under typical loads and in realistic conditions.

Tools Available: Chaos Monkey

Expected Results: Logs and data that provide insights into how the system reacts to real-world scenarios.

This Order: Simulation testing goes a step further in directly testing the software and is a natural progression once fuzzing is done. A cool example of real-world simulation testing that I found was done by Tiger Beetle, a financial database company. The video demonstration is source 5.

5. Penetration Testing

Description: Penetration testing is when a security team actively tests software, with a specific goal. It is often a manual process that can include using tools to automate certain vulnerability finding. There are 4 steps, reconnaissance, attack and exploitation, removal of evidence, and reporting.

Tools Available: Metasploit, Cobalt Strike

Expected Results: Report including where the system is vulnerable and how access was gained if applicable.

This Order: After the previous tests have identified known vulnerabilities and those have been remedied, pen testing is a great way to deliberately test the hardened system to find any additional vulnerabilities that were missed.

6. Testing for Failure

Description: Testing for failure test how the system handles failures. This type of testing includes how the system handles component outages, network errors, and load testing. This is done to ensure that these failures are handled gracefully.

Tools Available: Automation Scripts

Expected Results: Logs and monitoring confirm continued operation of the system.

This Order: Once the system has been tested and is secure, testing for failure adds a level of polish to ensure that the system is reliable and resilient.

7. Regression Testing

Description: Regression testing ensures that changes, from any previous tests or feature updates, didn't break functionality or introduce new vulnerabilities.

Tools Available: CI/CD tests – Jenkins, GitHub Actions. Selenium.

Expected Results: All tests pass confirming system function.

This Order: Regression testing is done frequently throughout the SDLC. I've listed it last as a final test to ensure functionality of the system.

Sources

1. CISSLP Exam Guide, 3rd. Edition - Chapter 11.
2. https://www.genrocket.com/blog/synthetic-data-redefines-the-test-data-lifecycle/
3. https://owasp.org/www-community/Vulnerability_Scanning_Tools
4. https://netflix.github.io/chaosmonkey/
5. https://www.youtube.com/watch?v=Vch4BWUVzMM
6. https://www.metasploit.com
7. https://www.browserstack.com/guide/regression-testing
8. https://www.jenkins.io
9. https://github.com/features/actions