Towards Effective Software Security through Testing Stage of SDLC

Vidyabhushan Anantrao Upadhye , Dr. Shashank D. Joshi
Department of Computer COE,Bharati Vidyapeeth University, Pune, India

Abstract: **Today software has become an integral part day to day life Keeping in view the daily increase in software security threats, developing secure software has become a necessity in the propose method most of the vulnerability are tested. Early detection of vulnerabilities in software while developing it and countering them in the software development cycle will save our time and energy spent on removing them after software release. In this paper we propose technique software security through Testing Stage of SDLC.**
Keyword: software development life cycle , vulnerabilities

1. INTRODUCTION

Testing is the process of executing a program or system with the target of finding bugs. It involves any activity aimed at evaluating a characteristic, capability of a program or system and significant that it meets its required results. Software bugs will almost forever exist in any software module with sensible size not because programmers are careless or reckless, but because the difficulty of software is generally stubborn and humans have only limited ability to manage complexity. It is also true that for any complicated systems, design defects can never be completely ruled out.

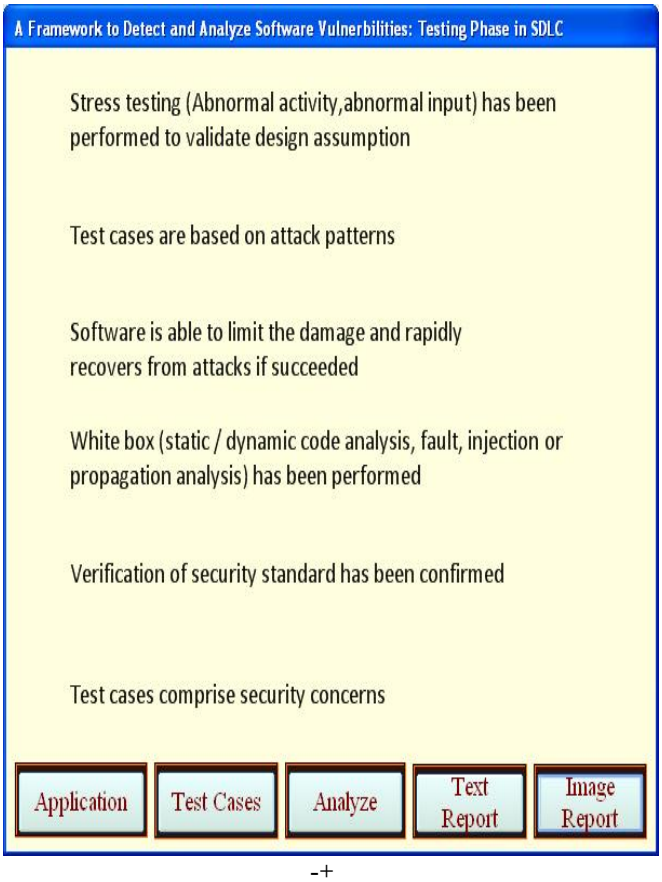
Software quality, reliability, consistency and security are tightly coupled. Flaws in software can be exploited by intruders to open security holes. With the development of the Internet, software security problems are becoming even more severe and very painful.

Many critical software applications and services need integrated security measures against malevolent attacks. The purpose of security testing of these systems include not only identifying but removing software flaws that may potentially lead to security violations, and validating the efficiency of security measures. Simulated security attacks can be performed to find vulnerabilities.

A vulnerability scanner is a application program that performs the analytical phase of a vulnerability analysis, also known as vulnerability assessment. Vulnerability analysis defines, identifies, and classifies the security holes and their vulnerabilities in a computer, server, network, or communications channel, infrastructure.

In addition, vulnerability analysis can forecast the effectiveness of proposed countermeasures, and evaluate how well they work after they are put into use. A vulnerability scanner relies on a database that contains all the information required to check a system for security holes in services and ports, anomalies in packet construction, and potential paths to credulous programs or scripts. Then the scanner tries to exploit each vulnerability that is discovered. This process is sometimes called ethical hacking.

An ideal vulnerability scanner has many capabilities for example Preservation of an up-to-date database of vulnerabilities. Detection of authentic vulnerabilities without an excessive number of false positives. Ability to perform trend analyses and provide clear reports of the results. Recommendations for countermeasures to eliminate discovered vulnerabilities. If security holes are detected by a vulnerability scanner, a vulnerability disclosure may be required. The person or organization that discovers the vulnerability, or a responsible industry body such as the Computer Emergency Readiness Team (CERT), may make the disclosure, sometimes after alerting the vendor and allowing them a certain amount of time to remedy or moderate the problem.



-+

Fig.1 UI for Vulnerabilities Detection Tool before process

Fig 1 Shows UI for Detection of Vulnerabilities in software application before process starts.After scanning predefined Vulnerabilities the tool presents report summery as shown in Fig.2.

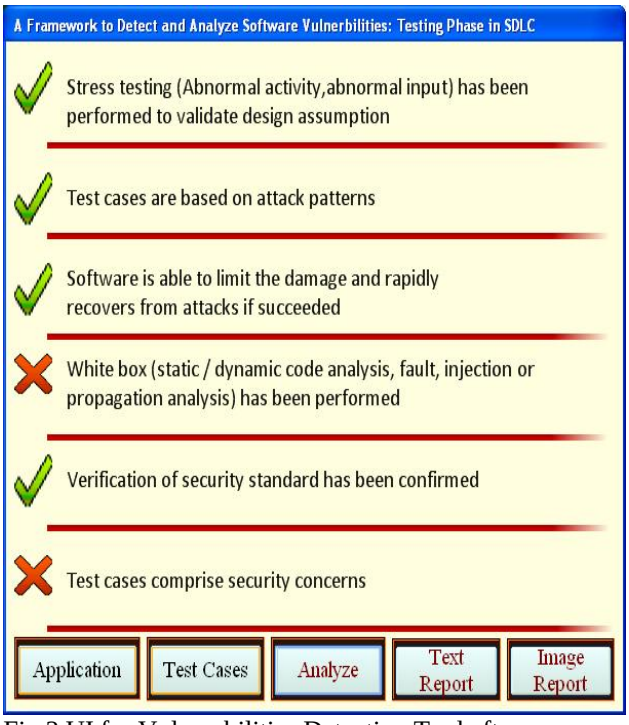


Fig.2 UI for Vulnerabilities Detection Tool after process

Stress testing in IT industry hardware as well as software sectors means testing of software for its effectiveness in giving consistent or satisfactory performance under extreme and unfavorable conditions such as heavy network traffic, heavy processes load. working under maximum requests for resource utilization of the peripheral or in the system.

In other words, stress testing helps find out the level of robustness and consistent or satisfactory performance even when the limits for normal operation for the system (software/hardware) is crossed. Most important use of stress testing is found in testing of software and hardware that are supposed to be operating in critical or real time situation. Such as a website will always be online and the server hosting the website must be able to handle the traffic in all possible ways (even if the traffic increases manifold), a mission critical software or hardware that works in real time scenario etc. Stress testing in connection with websites or certain software is considered to be an effective process of determining the limit, at which the system/software/hardware/website shows robustness, is always available to perform its task, effectively manages the load than the normal scenario and even shows effective error management under extreme conditions.

RELATED APPROACHES

The methodology aims to minimize vulnerabilities in software under expansion. To this aim, each time, output of each phase of SDLC is fed to the security checklist where it is verified whether the output fulfills fundamentals for security of the phase. If yes, the phase is declared as secure. If not, the output is properly analyzed for the detection of the vulnerabilities [1].

We are developing a process for systematic and continuous development of software security throughout the software lifecycle, that is suitable for industrial adoption, and focuses on preventing vulnerabilities in all phases of software development [2].

An ideal vulnerability scanner has capabilities such as: Preservation of an up-to-date database of vulnerabilities. Detection of authentic vulnerabilities without an excessive number of false positives. Ability to perform trend analyses and provide clear reports of the results. Recommendations for countermeasures to eliminate discovered vulnerabilities. If security holes are detected by a vulnerability scanner, a vulnerability disclosure may be required. The person or organization that discovers the vulnerability, or a responsible industry body such as the Computer Emergency Readiness Team (CERT), may make the disclosure, sometimes after alerting the vendor and allowing them a certain amount of time to remedy or moderate the problem.

IMPORTANCE OF STRESS TESTING:

Stress testing is considered to be important because of following reasons:

1. Almost 90% of the software (systems) are developed with an supposition that they will be operating under usual scenario. And even if it is considered that the limit of normal operating conditions will be crossed, it is not significantly as high as it really could be.
2. The cost or effect of a very imperative significant software, system and website failure under intense conditions in real time can be huge (or may be catastrophic for the organization or entity owning the software/system).
3. It is always better to be prepared for extreme conditions rather than letting the system/software/web services crash, when the limit of normal, proper operation is crossed.
4. Testing carried out by the developer of the system/software/website may not be sufficient to help expose conditions which will lead to crash of the system/software when it is actually submitted to the operating environment.
5. It's not always possible to reveal possible problems or bugs in a system/software, unless it is subjected to such type of testing

CONCLUSIONS:

In this paper we are presented on overview of system designed to help software development originations to prevent Vulnerabilities in the software they developed. Testing is reasonably expensive. Vulnerabilities discovery Tool is very good, flexible technique to cut down cost and time. Testing efficiency and usefulness is the criteria for large coverage-based testing techniques. Present tool well tested on Banking domain .

REFERENCES

1. Davis, G. McGraw, "Software Security," *IEEE Security & Privacy*, vol. 2, no. 2, 2004, pp. 80–83.
2. G. Hoglund and G. McGraw, *Exploiting Software*, Addison- Wesley, 2004.
3. Jones, C., Software Assessments, Benchmarks, and Best Practices. 2000, Reading, MA: Addison-Wesley. 659.
4. Jacquith, A., The Security of Applications: Not All Are Created Equal. 2002, @Stake Research. p. 12.
5. SANS Institute, The Twenty Most Critical Internet Security Vulnerabilities (Updated) ~ The Experts Consensus Version 6.01. 2005.
6. Bishop, M. and S. Engle. The Software Assurance CBK and University Curricula. in Proceedings of the 10th Colloquium for Information Systems Security Education. 2006. University of Maryland, Adelphi, MD
7. Conklin, W.A. Bottom-Up meets Top-Down: A new Paradigm for Software Engineering Instruction. in Proceedings of the 10th Colloquium for Information Systems Security Education. 2006. University of Maryland, University Collage, Adelphi, MD
8. Conklin, W.A., Personal Communication: Discussion on secure coding practices with graduate students and former students, W.A. Conklin, Editor. 2005: San Antonio, TX.
9. Bloom, B.S., Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain. 1956, New York: David McKay Co Inc.
10. Howard, M., D. LeBlanc, and J. Viega, 19 Deadly Sins of Software Security 2005: McGraw-Hill Osborne Media. 304.
11. Howard, M. and D.C. LeBlanc, Writing Secure Code. Second Edition ed. 2002: Microsoft Press. 650.
12. Institute, S.E., Build Security In. 2006, Strategic Initiatives Branch of the National Cyber Security Division (NCSD) of the Department of Homeland Security (DHS) https://buildsecurityin.uscert. gov/portal/.
13. Schneier, B., Applied Cryptography: Protocols, Algorithms, and Source Code in C. 1995: Wiley.784.
14. Schneier, B., Secrets and Lies : Digital Security in a Networked World 2004: Wiley. 448.