Effective Security Testing

Author: Jeyasekar Marimuthu

Jeyasekar.m@gmail.com

Abstract

The number of software vulnerabilities exposed in recent times and the magnitude of impacts they have on customers is the key reason why effective security testing is required for software development organizations.

When proper security measures are not taken, a company not only puts their reputation at stake, this also puts their customers and their customers' sensitive data at risk. The magnitude of a data breach will vary - from financial loss, to loss of life - depending on the software deployment. Data exploitation in any form through software vulnerabilities affects the credibility of system and the company.

Effective security testing is achieved by and by understanding security at deeper levels and adopting key security measures in the PLF (Project Life Cycle). Software experts write tests for each layer, then measured via manual and tool based audits.

It is important to plan measures targeting the prevention of security breaches. Some of the known practices to prevent security breaches in software include vulnerability scans, security assessment, penetration tests, and security audits. Then, when issues arise, it is important to deploy counter measures before the problem magnifies.

This paper discusses various forms of software vulnerabilities and the measures to be taken to prevent them. The paper also details out some of the key security tools recommended for identifying security flaws.

Biography

Jeyasekar Marimuthu is a lead software developer at Intel Security, Beaverton (formerly McAfee) with comprehensive experience in architecting and developing solutions, project execution, process improvement, cross product development, and client relations. His strengths are: understanding business requirements, evaluating risks, providing architecture and designs, and strategizing profitable execution. A skilled developer and technical leader, Jeyasekar communicates effectively with management, vendors, team members, and staff at all skill levels. He brings innovation with ideas and drives improvements in development and processes. He has 14 years of Development experience in Java, J2EE, database technologies, and B2B/ Enterprise-security products.

.

1. Introduction

Every organization that deals with sensitive data should expect exploits from all possible directions. Widely classifying, the attacks target external facing systems, hosted applications, client side applications, company networks and individuals. It is important to understand the various kinds of attacks that have occurred and the damages they have caused.

Exploits can happen through all sorts of medium, security testing is not a well-defined term and it can never be fool proof. How can companies be smart in investing in right areas and protect their core values?

It is impossible to prevent every possible attack and it is a daunting task for organizations to block every possible channel.

This paper focuses security testing on the exploits. If attacks are expected then the magnitude of impacts due to the exploit should be as minimal as possible. Let us see how.

2. Understand the organizational need

Companies should prioritize their important assets and start protecting them based on their importance. It is wise to minimize the impact of exploitation on critical systems before it happens. The difference between security testing and an advanced security testing is similar to black box vs white box testing. In black box testing, one would look for the valid and invalid functional tests. In white box testing, it is required to understand the internal perspective of the system to write and perform tests. It is difficult to quantify the level of system protection without understanding the applicable vulnerabilities and the possible damages they may cause.

The rules of thumb for advanced security testing:

1. Study the recent disasters and motives behind each attack.
2. Evaluate the damages of each targeted attack and understand the consequences after attack.
3. Pick the relevant exploits that are applicable to your domain or area of operations.
4. Study the consequences of exploits on assets, networks, company reputations, customer retention etc.
5. Prioritize the assets that need to be protected according to business and operation needs.
6. Make your investments on security products wisely.

Understand that not all security items can be implemented; some systems may have no countermeasures for attacks. Choose the one that is most important to you and address the vulnerabilities as effectively as possible.

3. Need for security testing: A quick overview

In order to implement the rules of thumb, an organization must study the type of vulnerabilities and possible attacks. Without having a deeper understanding, it will not be easy to choose the relevant items that are applicable to the system or organization.

Types of attack/Exploits:

- Social engineering
- Session hijacking
- Network mapping
- Fuzzing

- Zero day attack
- SQL Injection
- Password Cracking
- DNS Poisoning
- ARP Poisoning
- Wardriving

Type of vulnerabilities:

- Arbitrary Code Execution
- Buffer Overflow
- Code Injection
- Heap Spraying
- Web Exploitation (client-side)
- Cross-site scripting
- HTTP header injection
- HTTP Request Smuggling
- Web Exploitation (server-side)
- DNS Rebinding
- Clickjacking
- CSRF

A short summary of 2013 disasters:

- Social media giants Facebook, LinkedIn, among others, get hacked repeatedly
- Nearly 40 million Target customers' credit and debit card numbers were stolen in the midst of the holiday shopping rush.
- Hacker group anonymous targets Twitter accounts.
- Adobe breach snowballs into multi-network security risk.
- System bug exposes 6 million Facebook users' personal data in yearlong breach.
- Upwards of 50 million LivingSocial user emails and passwords are stolen.
- Evernote resets about 50 million account passwords after data breach.
- The U.S. Department of Homeland Security finally corrected a four-year error in the software it uses to process employees' background checks.
- Federal Reserve Bank website hacked by Anonymous.

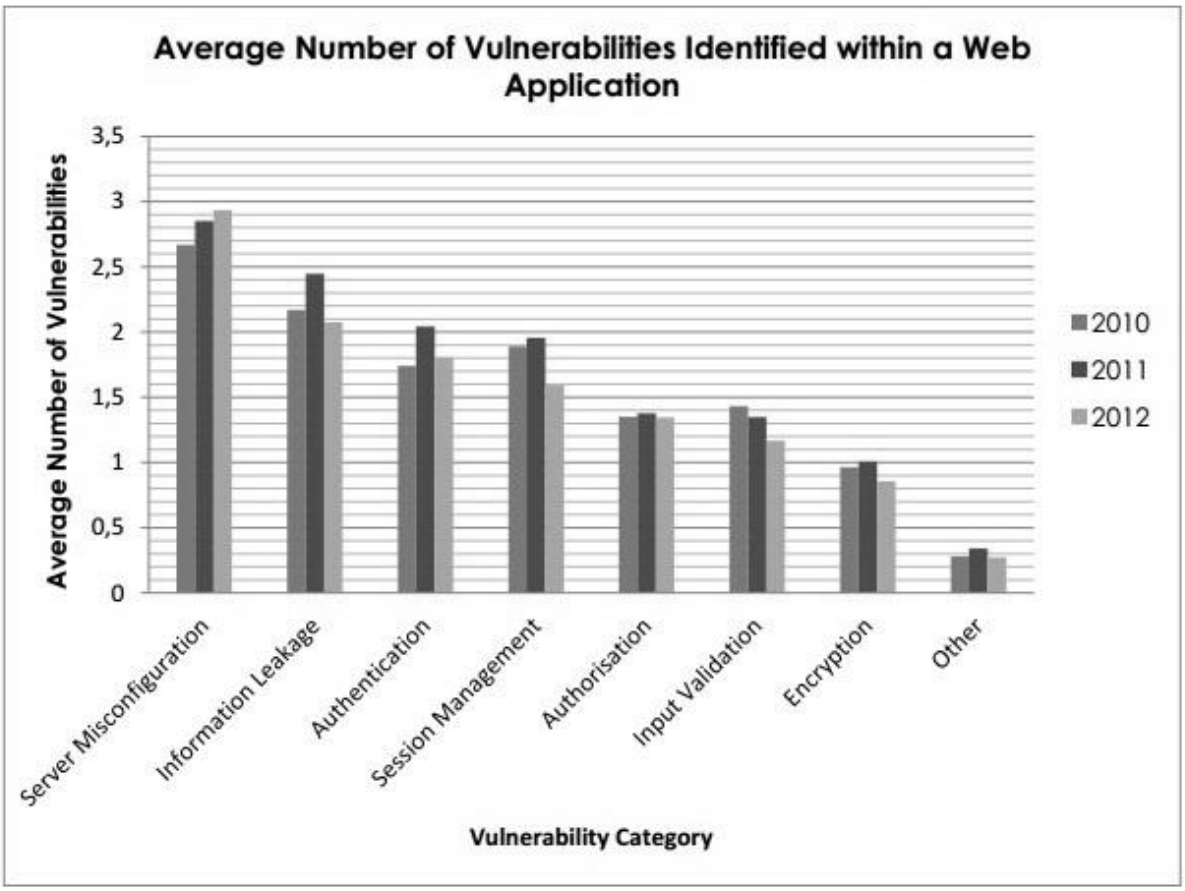
These breaches fall into 4 broad categories.

- **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) application that the system depends on.
- **Client-side application security**: This ensures the client such as browser cannot be manipulated.
- **Server-side application security**: This makes sure that the server code and its technologies are robust enough to fend off any intrusion.

Web application findings summary 2013:

Server-side application security is something test professionals need to pay more attention to. The Web Application Vulnerability chart below shows the category of breaches.

Category	**Description**
Server configuration	Insecure server configuration settings that result in security
Information leakage	Information leaked by the application that could be used by an attacker to help mount an attack
Authentication weaknesses	Issues related to the application's authentication mechanism that could be exploited by an unauthenticated attacker to gain or assist in the gaining of authenticated access
Session management weaknesses	Session management issues that could allow an attacker to hijack or assist in the hijacking of other users' sessions
Authorization weaknesses	Issues concerning access controls that could allow an attacker to perform either horizontal or vertical privilege escalation
Input validation weaknesses	Issues created by weaknesses in input validation processes.
Encryption vulnerabilities	Issues that concern the confidentiality of data during transport and in storage
Other	Any other issues identified that do not fit into the categories listed above



4. Effective tests

The previous section highlighted some of the key vulnerabilities and exploits. Based on the facts presented, prepare the applicability score against each of them and see what tops your findings. Pick those top 3 or 5 items from each vulnerability and exploit category and identify the tools and man-power required to perform those tests.

A sample Security Assessment sheet would look as follows.

Type of vulnerability	Applicable?	Estimated damage (in $)	Impact on Business sustainability
Buffer overflow	Yes	2 Million	Medium
Code injection	Yes	2 Million	Medium
Cross Site scripting	Yes	10 Million	High

The sample Security Assessment table helps to prioritize the items to focus. The next big question is how to determine whether the vulnerability is applicable to your organization or not. Companies like Intel Security offer various vulnerability detector tools that can detect the vulnerabilities easily. There are even manual ways to identify the flaws in the system.

According to ISACA (Information Systems Audit and Control Association), large-scale, covert penetration tests can be an effective tool for governments, private companies, and other national and international organizations to assess the security of their critical resources. According to the US National Institute of Standards and Technology, the purpose of covert security testing is to “examine the damage or impact an adversary can cause,” rather than to identify specific vulnerabilities.

The penetration test team should target to find the following vulnerabilities.

The majority of the vulnerabilities exploited could be categorized as follows:

- Weak and/or unchanged default passwords
- Default system and application configurations
- Failure to patch known vulnerabilities and use secure configurations enterprise-wide
- Failure to consistently apply the least-privilege access control model
- Failure to use secure coding
- Lack of security awareness by system users

5. Tools that can detect specific vulnerabilities

There are many tools on the market that detects specific vulnerabilities. Some of these are listed below.

5.1 Testing for DOM XSS

- DOMinator Pro - https://dominator.mindedsecurity.com

5.2 Testing AJAX

- OWASP Sprajax Project

5.3 Testing for SQL Injection

- OWASP SQLiX
- Sqlninja: a SQL Server Injection & Takeover Tool - *http://sqlninja.sourceforge.net*
- Bernardo Damele A. G.: sqlmap, automatic SQL injection tool - *http://sqlmap.org/*
- Absinthe 1.1 (formerly SQLSqueal) - *http://sourceforge.net/projects/absinthe/*
- SQLInjector - Uses inference techniques to extract data and determine the backend database server. *http://www.databasesecurity.com/sql-injector.htm*
- Bsqlbf-v2: A Perl script allows extraction of data from Blind SQL Injections - *http://code.google.com/p/bsqlbf-v2/*
- Pangolin: An automatic SQL injection penetration testing tool - *http://www.darknet.org.uk/2009/05/pangolin-automatic-sql-injection-tool/*
- Antonio Parata: Dump Files by sql inference on Mysql - SqlDumper - *http://www.ruizata.com/*
- Multiple DBMS Sql Injection tool - SQL Power Injector - *http://www.sqlpowerinjector.com/*
- MySql Blind Injection Bruteforcing, Reversing.org - sqlbftools - *http://packetstormsecurity.org/files/43795/sqlbftools-1.2.tar.gz.html*

5.4 Testing SSL

- Foundstone SSL Digger - *http://www.mcafee.com/us/downloads/free-tools/ssldigger.aspx*

5.5 Testing for Brute Force Password

- THC Hydra - *http://www.thc.org/thc-hydra/*
- John the Ripper - *http://www.openwall.com/john/*
- Brutus - *http://www.hoobie.net/brutus/*
- Medusa - *http://www.foofus.net/~jmk/medusa/medusa.html*
- Ncat - *http://nmap.org/ncat/*

5.6 Testing Buffer Overflow

- OllyDbg - A windows based debugger used for analyzing buffer overflow vulnerabilities - *http://www.ollydbg.de*
- Spike - A fuzzer framework that can be used to explore vulnerabilities and perform length testing *http://www.immunitysec.com/downloads/SPIKE2.9.tgz*
- Brute Force Binary Tester (BFB) - *http://bfbtester.sourceforge.net* A proactive binary checker
- Metasploit - *http://www.metasploit.com/* - A rapid exploit development and Testing frame work

5.7 Fuzzer

- OWASP WSFuzzer
- Wfuzz - *http://www.darknet.org.uk/2007/07/wfuzz-a-tool-for-bruteforcingfuzzing-web-applications/*

6. Best practices / Countermeasures

After detecting vulnerabilities, it is important to fix them by adopting some best practices and deploying countermeasures.

- Form a dedicated security audit team that can periodically conduct audits and report security issues.
- Make sure all systems in the organization are behind the firewall.
- Make sure all systems are patched periodically.
- Allow network traffic only via firewall.
- Make sure you have a well-defined security policy for your org and enforce the rules on each system via products like McAfee's ePolicy Orchestrator.
- Deploy countermeasures on all critical systems.
- Educate employees about phishing attacks through phone, emails and alert them to exercise caution.
- Make sure every project release is certified by your Security testing experts.

7. Qualities of Good Security Testers

A security tester is preferred to have thorough understanding on all types of attacks and their potential impacts on a system. They should analyze the SuD (System under development) in length and breadth and should be able to identify the areas that need to be tested against a bunch of security tests. A person with certification like CISSP/CompTIA Security+ would be a good fit. More importantly, she/he should follow the recent security breaches and should be able to map the breaches with the system they work on.

A security test plan capturing the areas and applicable tests should be rolled out as soon as the system design is ready. A security tester is expected to have mastery in one or more security tools that can detect and analyze potential weakness in the system. She/he shall additionally have white box testing abilities to conduct manual security audits on certain portion of code where tools have limited reach. She/he may urge organizations to treat security violations with high priority/important.

8. Management's Role

Having a great security tester without management support can be like having no tester at all. Management has the primary role in implementing good security testing. Management must balance coverage of the product(s), budget, and time required to implement quality measures.

Security testing needs to be treated as critical in risk management. When the breach happens, the amount of damage (loss of trust, reputation) that can occur from an uncaught vulnerability is much higher than any functional product defect.

It is wise to have one or more full time dedicated security analyst(s) working on projects. No project should be allowed to run without having a security test plan in place. A dedicated team of testers should have a role throughout the project life cycle. It is also good to have measures to evaluate the efficiency of security analyst and testers so that the quality of security testing constantly improves.

9. Conclusion

This paper has discussed the importance of conducting an effective security testing and the measures to identify the applicable areas for an organization. Various vulnerabilities and attacks discussed in the paper would serve as good reference points to get educated and to pick a right tool. Adopting the best practices and guidelines is vital to anybody who wants to keep the attackers and hackers at bay. Having all best practices followed and audited the system with relevant tools, the amount of damages that could have caused is significantly reduced. Despite all, keep in mind one thing. No system is fool proof. A continuous monitoring of security flaws only help for betterment of any system.

10. References

- http://softwaretestingfundamentals.com/security-testing/
- http://thinkprogress.org/security/2013/12/31/3108661/10-biggest-privacy-security-breaches-rocked-2013/
- http://www.isaca.org/Journal/Past-Issues/2012/Volume-2/Pages/Security-Through-Effective-Penetration-Testing.aspx
- https://tysonmax20042003.wordpress.com/tag/types-of-exploits/