ISSN No. 0976-5697

Volume 4, No. 11, Nov-Dec 2013

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

An approach for Dynamic Software Security Testing of Web based Applications

Pradeep Muruganandam
Software Industry Professional, Freelance Researcher,India
pradeepm1991@gmail.com

***Abstract*—** In today's data centric world coupled with enormous power of software and Internet, the need for software security is very essential. Various new attacks are formulated and all the attacks need to be countered to secure data from malicious attacks. Software Security testing serves the same purpose. Web applications are usually the target of Attacks like SQL Injection, Cross Site Scripting or Denial of Service (DOS) [1]. All these attacks compromise on data safety, integrity and confidentiality. Several methods need to be handled for preventing these types of attacks. This paper details one such mechanism which tries to weed out the threat from the various web based threats to the software. This paper details a method which is based on IP based scanning of accounts, Java Script usage in an efficient way and also the efficient implementation of service providing algorithm to be employed on web services to weed out inactive accounts which are idle but logged in. [2]

***Keywords*—** Attack, Threat, Vulnerability, Authorization, Brute Force, Security, Safety, Confidentiality, Scanning.,

I. INTRODUCTION

As software systems are handling enormously large quantities of highly valuable and critical, real time information, software security has turned out to become an issue of utmost importance to companies. Security testing has recently moved beyond the realm of network port scanning to include probing-software's behavior as a critical aspect of system behavior [3]. With advancement of technology and widespread availability of knowledge in Internet regarding security loopholes exploitation methods, the attacks on software's and websites have increased tremendously. This makes the task of software testing really hard [4]. One important task is to test how the system behaves to handle any attacks at the design stage itself. Software security testing should aim to test the design models, in addition to the implementation. It needs to ensure that the designed system can protect itself and the data from attacks with the help of mitigation. We refer to an attack as an exploitation of a vulnerability to realize a threat, where vulnerabilities are actual security weakness or flaws that make a system susceptible to an attack, threats are potential attacks that violate security policies or goals. And mitigations are security features or assurance techniques for mitigating specific threats.

II. EASE OF USE

A. *Types of Software:*

Broadly, the software could be classified on the basis of deployment type as either Web Based software or Desktop Based Applications. Web based application use services deployed on web servers and is more prone to attacks from outsiders. They have more vulnerabilities as they are hosted on servers belonging to other service providers. On the other hand the desktop applications are less prone to attack by outsiders using hacking techniques.

B. *Maintaining The Integrity Of Data:*

The data that is used by the software should be protected from malicious attacks and it is not possible without filtering the access granted to the users. Data should also be consistent and all the changes in data should be reflected correctly. Granting access by using credentials limits the people who can access the software. It is the preliminary level of security to the software.

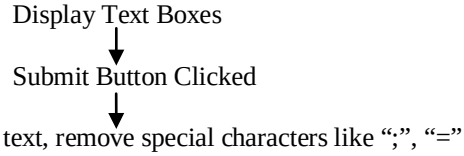
III. COMMON ATTACKS

SQL injection attacks pose a serious security threat to Web applications: they allow attackers to obtain unrestricted access to the databases underlying the applications and to the potentially sensitive information these databases contain. There are also attacks like the Denial of Service, Cross Site Scripting, and Request Flooding. All these attacks point to an outsider needing access to the service in order to gain access to data [5].

Software designed by companies should incorporate various mechanisms to thwart these attacks. They should be able to fend off the malicious offenders who do these activities.

A. *SQL Injection Attacks:*

At present to address this problem, we have techniques, such as hand-crafted checking, automatic filter, SQL command coding and special propose API. A rigorous and systematic application of these techniques is an effective solution for preventing SQL injection vulnerabilities. Identification and repairing of SQL injection in test phase of applications may reduce and even eradicate security threats which are introduced by SQL injection, improve security of applications. It is helpful to enhance database security by finding SQL injection vulnerabilities, automatically generating tests cases and making simulation attack to an application.



B. *Automate SQL injection testing:*

In the early days of SQL injection attacks, testing manually was the only way to determine if the software

systems, databases or applications were vulnerable to the SQL injection threat. Manual testing is a long and tedious process, and even then, there is no guarantee that we will be able to find each and every vulnerability. But nowadays there are several automated tools available to carry out simulated SQL injection attacks on your own databases to see how the systems respond to the threat. It's a two-step process to test your own systems for SQL injection vulnerabilities in an automated way [5].

a. Scan for vulnerabilities:

Scan the site with a Web application vulnerability scanner to see if any input filtering or other SQL injection-specific holes exist. Commercial tools such as N-Stealth Security Scanner, Acunetix Ltd.'s Web Vulnerability Scanner and SPI Dynamics Web Inspect can be used for the purpose. Free tools like Wikto can often find these vulnerabilities as well.

b. Begin SQL injection:

Once we determine if the target system is vulnerable to SQL injection or not, the next step would be to carry out the simulated SQL injection process. It is not necessary to injecting actual data or attempt to drop database tables, both of which are bad for the database. A tool for automating the actual SQL injection process is SPI Dynamics' SQL Injector. More number of tests should be performed if basic SQL injection simulation doesn't return any results. These tools are capable of querying and extracting data quickly in an automated manner, easily dumping the large tables in very short time.

C. Cross Site Scripting(CSS):

The three conditions for Cross-Site Scripting are: A Web application accepts user input, the input is used to create dynamic content and that the input is insufficiently validated. Scripting: Web Browsers can execute commands Embedded in HTML page and Supports different languages (JavaScript, VBScript, ActiveX, etc.). The most prominent scripting language used is JavaScript.

"Cross-Site" means: Foreign script sent via server to client. An Attacker makes Web-Server deliver malicious script code and the malicious script is executed in Client's Web Browser. The purpose of these attacks is to steal access credentials, Denial-of-Service, modify Web pages and to execute any command at the client machine.

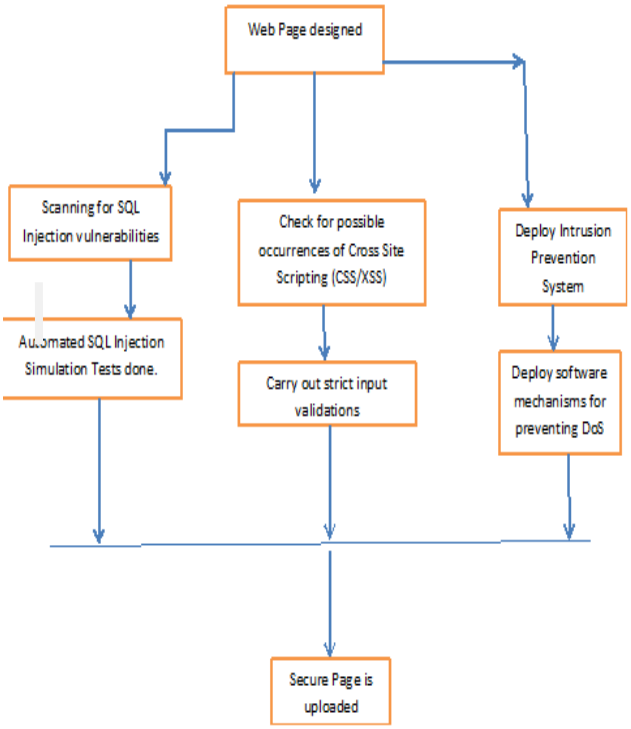
To prevent these attacks in machines, we need to incorporate a well-built input validation system. Check if the input is the exact type which is expected. Don't allow some random text to pass. Usage of regular expressions will serve the purpose.

D. Denial of Service(DoS):

A denial-of-service attack (DoS attack) is an attempt to make a machine or network resource unavailable to its intended user. It is carried out by malicious groups to make famous web services unavailable by occupying the hardware resources of the website completely.

Proper prevention mechanisms need to be in place for preventing such attacks which may cause complete outage of the web applications and may lead to monetary losses too. Some of the methods include Intrusion Prevention System, Firewalls, Clean Pipes., etc.

IV. AN UNIFIED APPROACH FOR TESTING APPLICATION SECURITY [6][7][8]



A. The SQL Injection Part:

a. It involves making use of the certain automated tools for generating security cases for simulating the attacks.

B. The Cross Site Scripting Part [9]:

a. It involves prevention of the script files which may try to get executed on the Client side even when executed from remote machines.
b. It has strict input validation methods to weed out the unnecessary inputs.

C. The DoS Part:

a. It involves usage of the special systems like Intrusion Prevention System, special hardware like Firewalls, Switches are used.
b. Also IP Scanning needs to be done, with IP Filters blocking the requests coming in from the same IP address for the same set of resources or access when it has already been granted the same once. This can help eliminate flooding of request packets. (Like in SYN Flood or other types) .

V. CONCLUSION

Placing more and more revenue for the purpose of ensuring the software security may not seem like a wise idea during development of many products. If it does not involve any significant data or any complex entity is controlled by it, then less emphasis on security testing can be understood. But in all other cases, we are in a need for extensive testing of the software in reference to finding the behaviour of the software. The above paper details an approach that takes care of the most significant attacks faced by the web applications in today's Internet centric world.

VI. REFERENCES

[1] Vulnerability(Computing), Wikipedia, The Free Encyclopedia. Wikimedia Foundation,Inc, http://en.wikipedia.org/wiki/Vulnerability_(computing)

[2] Stytz, Martin R., and Sheila B. Banks. "Dynamic software security testing."Security & Privacy, IEEE 4.3 (2006) (pp77-79).

[3] McGraw, Gary, and Bruce Potter. "Software security testing." IEEE Security and Privacy 2.5 (2004).(pp 81-85).

[4] Thompson, Herbert H. "Why security testing is hard." Security & Privacy, IEEE1.4 (2003). (pp. 83-86).

[5] Haixia, Yang, and Nan Zhihong. "A database security testing scheme of web application." Computer Science & Education, 2009. ICCSE'09. 4th International Conference on. IEEE, 2009. (pp. 953-955).

[6] He, K., Feng, Z., & Li, X. (2008, December) "An attack scenario based approach for software security testing at design stage". In IEEE Computer Science and Computational Technology, 2008. ISCSCT'08. International Symposium (Vol. 1, pp. 782-787). IEEE

[7] Buchler, M., Oudinet, J., & Pretschner, A. (2012, June) "Semi-automatic security testing of web applications from a secure model". In Software Security and Reliability (SERE), 2012 IEEE Sixth International Conference on (pp. 253-262). IEEE

[8] Yang, Y., Zhang, H., Pan, M., Yang, J., He, F., & Li, Z. (2009, November). "A Model-Based Fuzz Framework to the Security Testing of TCG Software Stack Implementations". In Multimedia Information Networking and Security, 2009. MINES'09. International Conference on (Vol. 1, pp. 149-152). IEEE.

[9] Cross Site Scripting, Wikipedia, The Free Encyclopedia. Wikimedia Foundation,Inc,http://en.wikipedia.org/wiki/Cross-site_scripting