A Review on Web Security Mechanism Performance Evaluation Using Vulnerability and Attack Injection

M.S. Patole[1], Sagar D. Kothimbire[2]

[1]Professor, Department of Computer Engineering, RMD Sinhgad School of Engineering, University of Pune, India

[2]Department of Computer Engineering, RMD Sinhgad School of Engineering, University of Pune, India

Abstract: ***In this method a tool is designed to evaluate the performance of web security mechanisms. The system is based on the idea that injecting realistic vulnerabilities in a web application and attacking them automatically. This can be used to support the assessment of existing security mechanisms and tools in custom setup scenarios. To provide real to life results, the stated vulnerability and attack injection methodology relies on the study of a large number of vulnerabilities in real web applications. In this method a tool is designed for automation of entire process called Vulnerability and Attack injection Tool. This paper provides a short Review on the technique to evaluate performance of web security mechanism.***

Keywords: Security, fault injection, internet applications, review and evaluation, vulnerability, sql probe.

1. Introduction

There is an increasing dependency on web applications, ranging from individual applications to large organizations. Almost everything is stored, available or traded on the web. Web applications can be personal websites, blogs, news, social networks, web mails, bank agencies, forums, e-commerce applications, etc. The omnipresence of web applications in our way of life and in our economy is so important that it makes them a natural target for malicious minds that want to exploit this new streak.

Vulnerability (Attack surface): Is a weakness which allows attacker to reduce a system's information assurance. The System is based on the idea that injecting realistic vulnerabilities in a web application and attacking them automatically. It can be used to support the assessment of existing security mechanisms and tools in custom setup scenarios. To provide true to life results, system relies on the study of a large number of vulnerabilities in real web applications. The system describes the implementation of the Vulnerability & Attack Injector Tool (VAIT) that allows the automation of the entire process. VAIT evaluate security mechanisms and to point out not only their weaknesses but also ways for their improvement. Static analysis and dynamic analysis of web application is done in order to gather information about it. In Vulnerabilty and Attack Injection Tool the main focus is on two popular web application vulnerabilities

1. SQL injection (SQLi)
2. Cross site Scripting(XSS)

This allows attacker to change SQL commands that are sent to the database(SQLi)
Or
Through the input of HTML and Scripting Language.

2. Need of System

Web applications should be constantly audited and protected by security tools during their lifetime. In practical situations, there is a need for new ways to effectively test existing web application security mechanisms in order to evaluate and improve them.

3. Literature Review

Fault injection techniques have traditionally been used to inject physical (i.e., hardware) faults. Initial fault injection techniques used hardware-based approaches such as pin-level injection. The increasing complexity of systems has lead to the replacement of hardware-based techniques by software implemented fault injection (SWIFI).

Neves et al. proposed an Attack Injector Tool (AJECT) to support the discovery of vulnerabilities in network servers, specifically IMAP servers. To attack the target system they used predefined test classes of attacks and some sort of **fuzzing**. The industry uses **fuzzing** and **mutation testing** to automate penetration testing of web applications.

Some of the best known of such tools are HP WebInspect, IBM Watchfire AppScan, Acunetix web application security scanner and Web-Sphinx. These tools still have many problems related to the high number of undetected vulnerabilities and high percentage of false positives, as shown by several studies.

4. Vulnerability and Attack Injection Concept

In this section we present the methodology for testing security mechanisms in the context of web applications. The methodology is based on the injection of realistic vulnerabilities and the subsequent controlled exploit of those vulnerabilities in order to attack the system. There are many types of vulnerabilities affecting web applications ,but XSS & SQLi are top of the list accounting 32% vulnerabilities So, we focus on these two important vulnerabilities **SQLi** & **XSS**

1) **SQLi**:- SQLi attack consist of tweaking(Damaging) input fields of web application to alter the Query sent to Backend Database. This allows attacker to retrieve sensible data or alter database records
2) XSS: XSS attack consists of injecting HTML and/or other scripting code (Usually JavaScript) in a vulnerable webpage.
   The attacker is able to change some of its functions, allowing him to take advantage of users visiting that webpage. Both SQLi & XSS vulnerabilities result from poorly coded application that do not properly check their inputs

5. Vulnerability and Attack Injection Setup

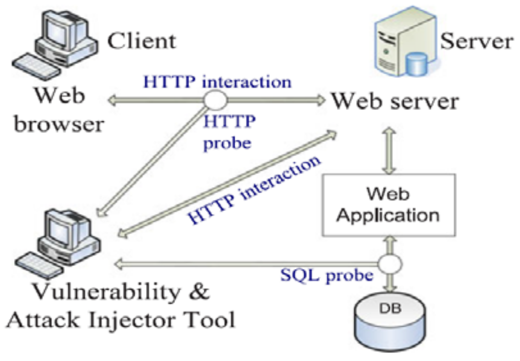

Figure 1: VAIT in typical setup

As shown in above fig., the attack injection uses **two external probes**, one for the **HTTP communication(HTTP PROBE)** and other for the **database communication(SQL PROBE)**. These probes monitor the HTTP and SQL data exchanged, and send a copy to be analyzed by the attack injection mechanism. This is a key aspect of the methodology to obtain the user interaction and the results produced by such interaction for analysis, so they can be used to prepare the attack

6. Vulnerability and Attack Injection Internal Components

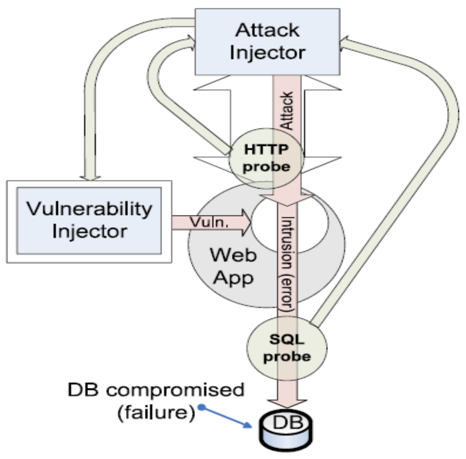

Figure 2: VAIT internal components

As shown in above fig. are the internal components of vulnerability and attack injection tool.

In this tool HTTP probe and SQL probe are two main components; these two components are used to monitor the communication between web browser and web server, web application and database respectively. These two probes are used to gather information related to various vulnerabilities like SQLi and/or XSS. In VAIT Vulnerability injector is used to inject vulnerabilities in to the web application ,after injecting the vulnerabilities the web applications security mechanisms performance is monitored and check for the errors occurred due to the injected vulnerabilities, if there is any error it will be reported to the vulnerability and attack injector tool through HTTP and SQL probes. Similarly the attack injector is used to inject the attacks in to the web application and its performance is observed and reported to the vulnerability and attack injection tool through various probes used in system (e.g. HTTP probe and SQL probe).

7. System Architecture of Vulnerability and Attack Injection Tool

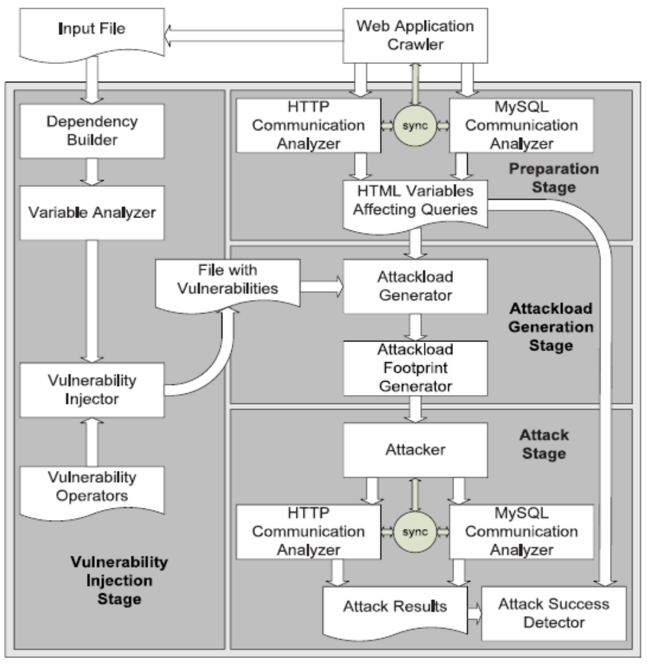
Input File
Web Application Crawler
Dependency Builder
HTTP Communication Analyzer
sync
MySQL Communication Analyzer
Preparation Stage
Variable Analyzer
HTML Variables Affecting Queries
File with Vulnerabilities
Attackload Generator
Attackload Generation Stage
Attackload Footprint Generator
Vulnerability Injector
Attacker
Attack Stage
Vulnerability Operators
HTTP Communication Analyzer
sync
MySQL Communication Analyzer
Vulnerability Injection Stage
Attack Results
Attack Success Detector

Figure 3: VAIT internal components

Above fig. gives the typical System Architecture of Vulnerability and attack injector tool. There are mainly four stages in Vulnerability and Attack Injector Tool. These stages are enlisted below:

- Preparation Stage
- Vulnerability Injection Stage
- AttackLoad Generation Stage
- Attack stage

These are main four stages of VAIT, these explained in detail in below explanation.

1] Preparation stage: In this preparation stage, the web application is interacted (crawled) executing all the functionalities that need to be tested (Fig. 2). Meanwhile, both HTTP and SQL communications are captured by the two probes and processed for later use. The interaction with the web application is always done from the client's point of view (the web browser). The outcome of this stage is the correlation of the input values, the HTTP variables that carry them and their respective source code files, and its use in the structure of the database queries sent to the back-end database (for SQLi) or displayed back to the web browser (for XSS).

Later on, in the attack stage, the malicious activity applied. Is based on tweaking the values of the variables, which correspond to the text fields, combo boxes, etc., discovered in this preparation stage.

2. Vulnerability Injection Stage: It is in this vulnerability injection stage that vulnerabilities are injected into the web application. For this purpose, it needs information about which input variables carry relevant information that can be used to execute attacks to the web application. This stage starts by analyzing the source code of the web application files searching for locations where vulnerabilities can be injected.

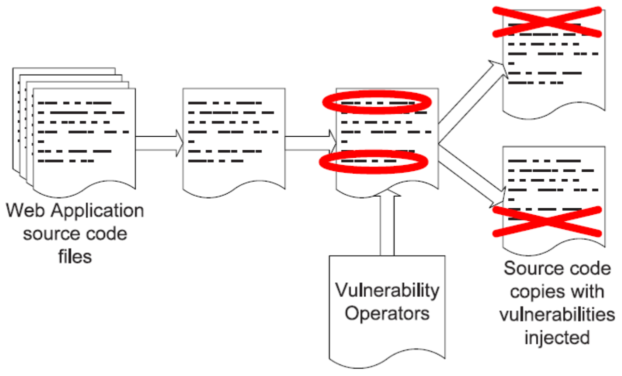
Web Application source code files
Vulnerability Operators
Source code copies with vulnerabilities injected

Figure 4: VAIT internal components

3. AttackLoad Generation Stage: After having the set of copies of the web application source code files with vulnerabilities injected we need to generate the collection of malicious interactions (attackloads) that will be used to attack each vulnerability. This is done in the attackload generation stage. The attackload is the malicious activity data needed to attack a given vulnerability. This data is built around the interaction patterns derived from the preparation stage, by tweaking the input values of the vulnerable variables.

4. Attack Stage: In the attack stage, the web application is, once again, interacted. However, this time it is a "malicious" interaction since it consists of a collection of attack payloads in order to exploit the vulnerabilities injected. The attack intends to alter the SQL query sent to the database server of the web application (for the case of SQLi attacks) or the HTML data sent back to the user (for the case of XSS attacks).

8. Attack Injection Scenarios

We envisage the following two scenarios as the most relevant utilizations of the proposed attack injection methodology and its VAIT tool:

1. Inline. The VAIT is executed while the security assurance mechanisms under evaluation are also being executed.
2. Offline. The VAIT is executed in advance to provide a set of realistic vulnerabilities for later use.

1] Inline Scenario: In the inline scenario, the VAIT can be used to evaluate tools and security assurance mechanisms, like IDS for databases, web application IDS, web application firewalls and reverse proxies. For example, when assessing an IDS for databases, the SQL probe should be placed before the IDS, so that the IDS is located between the SQL probe and the database (see Fig. 2 to locate the SQL probe and the database). During the attack stage, when the IDS inspects the SQL query sent to the database, the attack injector tool also monitors the output of the IDS to identify if the attack has been detected by the IDS or not. The entire process is performed automatically, without human intervention. The output of the VAIT also contains, in this case, the logs of the IDS detection. By analyzing the attacks that were not detected by the IDS, the security practitioner can gather some insights on the IDS weaknesses and, possibly, how the IDS could be improved.

2] Offline Scenario: In the offline scenario, the VAIT injects vulnerabilities into the web application and attacks them to check if they can be exploited or not. The outcome is the set of vulnerabilities that can, effectively, be attacked. They can then be used in a variety of situations, such as: to provide a testbed to train and evaluate security teams that are going to perform code review or penetration testing, to test static code analyzers, to estimate the number of vulnerabilities still present in the code, to evaluate web application vulnerability scanners, etc. It may also provide a ready to use testbed for web application security tools that can be integrated into assessment tools like the Moth and projects like the Stanford Security Bench, or in web applications installed in honeypots prepared to collect data about how hackers execute their attacks. This gathers insights on how hackers operates, what assets they want to attack and how they are using the vulnerabilities to attack other parts of the system. The offline scenario can also be applied to assess the quality of test cases developed for a given web application. For example, assuming that the test cases cover all the application functionalities in every situation, if the application code is changed (via vulnerability injection), the test cases should be able to discover that something is wrong. In situations where the test cases are not able to detect the modification, they should be improved and, maybe, the improvement can even uncover other unknown faulty situations. As an example, let us consider the assessment of web application vulnerability scanners, used to test for security problems in deployed web applications. These scanners perform black-box testing by interacting with the web application from the point of view of the attacker. In this scenario, the VAIT injects vulnerabilities and attacks them to see those that can be successfully attacked. These vulnerabilities are used, one by one, to assess the detection capability of the web application vulnerability scanner. This procedure can be used to obtain the percentage of vulnerabilities that the scanner cannot detect, and what are the most difficult types to detect. In this typical offline setup, the vulnerabilities can be injected one at a time (like in the case of vulnerability scanners) or multiple vulnerabilities at once (for the case of training security assurance teams, for example).

9. Conclusion

This paper provides review on Vulnerability and attack injection tool for performance evaluation of web security mechanism. Various studies have shown that proposed attack injection concept use effectively evaluate the web security mechanisms like IDS providing at the same time indications of what could be improved. By injecting vulnerabilities and attacking them automatically the VAIT could find weaknesses in the IDS. These results were very important in developing bug fixes. The VAIT was also used to evaluate two commercial and widely used web application vulnerability scanners, concerning their ability to detect SQLi vulnerabilities in web applications. These scanners were unable to detect most of the vulnerabilities injected, in spite of the fact that some of them seemed to easily be probed and confirmed by the scanners. The results clearly show that there is room for improvement in the SQLi detection capabilities of these scanners.

References

[1] Fonseca, J;Seixas,N.;Viera,M.;Madeira,H'' Analysis of Field Data on Web Security vulnerabilities''IEEE transaction on Dependable and secure Computing 2014
[2] Y.-W. Huang, S.-K. Huang, T.-P. Lin, and C.-H. Tsai, "Web Application Security Assessment by Fault Injection and Behavior Monitoring," Proc. Int'l Conf. World Wide Web, pp. 148-159, 2003.
[3] J. Fonseca, M. Vieira, and H. Madeira, "Detecting Malicious SQL," Proc. Conf. Trust, Privacy & Security in Digital Business, Sept. 2007.
[4] HP, download.hpsmartupdate.com/webinspect, Accessed 1 May 2013, Sept. 2013.
[5] IBM, www-03.ibm.com/software/products/us/en/appscan, Accessed 1 May 2013, Sept. 2013.
[6] Acunetix "Finding The Right Web Application Scanner; Why Black Box Scanning Is Not Enough," www.acunetix.com/websitesecurity/rightwvs.htm, Accessed 1 May 2013, 2009.

Author Profile

Prof. M. S. Patole received the B.E. and M.E. Degrees in Computer Engineering from University of Pune. She is working as Assistant Professor in Department of Computer Engineering, RMD Sinhgad School of Engineering Pune, India. She is having more than six years experience.

Sagar Kothimbire Reseach Scholar at RMD Sinhgad School of Engineering, University of Pune. He has received B.E. in Information Technology Engineering from University of Pune, Pune. Currently he is pursuing M.E. in Computer Engineering from RMD Sinhgad School of Engineering, Pune, University of Pune, Pune.