International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 25 Issue 6 – MAY 2018.

IMMUNITY TOWARDS DATA INJECTION ATTACK AND DETECTION IN WEB APPS

T. M. Padmapriya [#1] and P. Sivagama Sundari [*2]

[#] *M.E - II Year,Dept of CSE, Adhiparasakthi Engineering College, Melmaruvathur.India*

[*] *Assistant Professor/CSE, Adhiparasakthi Engineering College, Melmaruvathur.India*

***Abstract*— Web applications are typically developed with hard time constraints and are often deployed with security vulnerabilities. Automatic web vulnerability scanners can help to locate these vulnerabilities and are popular tools among developers of web applications. Their purpose is to stress the application from the attacker's point of view by issuing a huge amount of interaction within it. Two of the most widely spread and dangerous vulnerabilities in web applications are SQL injection and cross site scripting (XSS), because of the damage they may cause to the victim business. The most common types of software faults are injected in the web application code which is then checked by the scanners. The results are compared by analyzing coverage of vulnerability detection and false positives. Added to this it checks whether the domain or URL gives is present in any of the domain blacklist sites. It also provides I Category- the category classification for the given domain. So this gives the analyst an advantage of having the Domain rating for the given domain with a lookup on top blacklist providing sites and also the Category Classification for the given domain. Three leading commercial scanning tools are evaluated and the results show that in general the coverage is low and the percentage of false positives is very high.**

***Index Terms*—Web application, web vulnerability, Cross Site Scripting**

I. Introduction

Information security, sometimes shortened to InfoSec, is the practice of defending information from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction. It is a general term that can be used regardless of the form the data may take (e.g. electronic, physical).

Sometimes referred to as computer security, Information Technology security is information security applied to technology (most often some form of computer system). It is worthwhile to note that a computer does not necessarily mean a home desktop. A computer is any device with a processor and some memory. Such devices can range from non-networked standalone devices as simple as calculators, to networked mobile computing devices such as smart phones and tablet computers. IT security specialists are almost always found in any major enterprise/establishment due to the nature and value of the data within larger businesses. They are responsible for keeping all of the technology within the company secure from malicious cyber-attacks that often attempt to breach into critical private information or gain control of the internal systems.

The act of ensuring that data is not lost when critical issues arise. These issues include but are not limited to: natural disasters, computer / server malfunction, physical theft, or any other instance where data has the potential of being lost. Since most information is stored on computers in our modern era, information assurance is typically dealt with by IT security specialists. One of the most common methods of providing information assurance is to have an off-site backup of the data in case one of the mentioned issues arises.

The CIA triad of confidentiality, integrity, and availability is at the heart of information security. (The members of the classic InfoSec triad - confidentiality, integrity and availability are interchangeably referred to in the literature as security attributes, properties, security goals, fundamental aspects, information criteria, critical information characteristics and basic building blocks).

II. RELATED WORKS

A. Generalization Algorithm for detecting Data Injection Attack

A genetic algorithm is a search heuristic which simulates the natural selection process. This heuristic (sometimes known as a metaheuristic) is usually used to come up with suitable solutions that can address search and optimisation-related issues. Genetic algorithms are founded on the evolutionary notions of natural selection and genetics. Thus, they signify an intelligent manipulation of a random search deployed to address optimisation issues. The elementary genetic algorithm steps are converted into a pseudo code.

B. Genetic Algorithm for cross site scripting

A genetic algorithm is a search heuristic which simulates the natural selection process. This heuristic (sometimes known as a meta heuristic) is usually used to come up with suitable solutions that can address search and optimisation-related issues. Genetic algorithms are founded on the evolutionary notions of natural selection and genetics. Thus, they signify an intelligent manipulation of a random search deployed to address optimisation issues. The elementary genetic algorithm steps are converted into a pseudo code.

III. SYSTEM ARCHITECTURE

The architecture involves validating the user name and password and registering the email address and getting authentication to access various websites. It scans the URL and sub URL in web Application for vulnerabilities. It identifies the SQL injection and XSS attacks and dynamically generates the report and send to registered mail in PDF format.

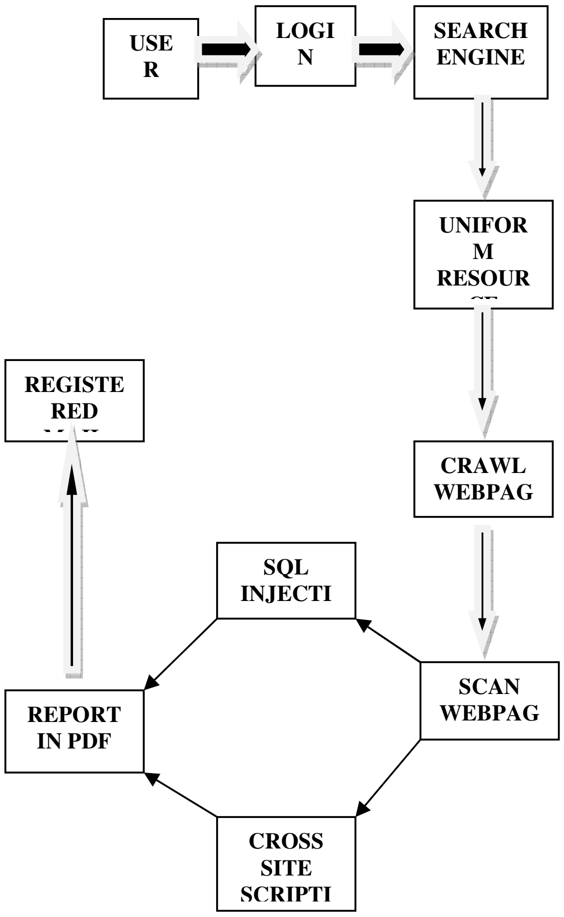

Figure 1.Architecture diagram

IV. SYSTEM MODULES

A. *Authentication*

Authentication is the process of determining whether someone or something is, in fact, who or what it is declared to be. In private and public computer networks (including the Internet), authentication is commonly done through the use of logon passwords. Knowledge of the password is assumed to guarantee that the user is authentic. Each user registers initially (or is registered by someone else), using an assigned or self-declared password. On each subsequent use, the user must know and use the previously declared password.

B. *Code Execution*

It is commonly used in arbitrary code execution vulnerability to describe a software bug that gives an attacker a way to execute arbitrary code. A program that is designed to exploit such vulnerability is called an arbitrary code execution exploit. Most of these vulnerabilities allow the execution of machine code and most exploits therefore inject and execute shell code to give an attacker an easy way to manually run arbitrary commands. The ability to trigger arbitrary code execution from one machine on another (especially via a wide-area network such as the Internet) is often referred to as remote code execution.

C. *Command Injection*

Command injection is an attack in which the goal is execution of arbitrary commands on the host operating system via a vulnerable application. Command injection attacks are possible when an application passes unsafe user supplied data (forms, cookies, HTTP headers etc.) to a system shell. In this attack, the attacker-supplied operating system commands are usually executed with the privileges of the vulnerable application. Command injection attacks are possible largely due to insufficient input validation.

D. *HTTP Header Injection*

HTTP header injection is a general class of web application security vulnerability which occurs when Hypertext Transfer Protocol (HTTP) headers are dynamically generated based on user input. Header injection in HTTP responses can allow for HTTP response splitting, Session fixation via the Set-Cookie header, cross-site scripting (XSS), and malicious redirect attacks via the location header.

E. *Reflected Cross Site Scripting*

Reflected Cross-site Scripting occur when an attacker injects browser executable code within a single HTTP response. The injected attack is not stored within the application itself; it is non-persistent and only impacts users who open a maliciously crafted link or third-party web page. Reflected XSS attacks are also known as non-persistent XSS attacks and, since the attack payload is delivered and executed via a single request and response, they are also referred to as first-order or type 1 XSS. When a web application is vulnerable to this type of attack, it will pass unvalidated input sent through requests back to the client. Commonly the attacker's code is written in the Javascript language, but other scripting languages are also used, e.g., ActionScript and VBScript. Attackers typically change the content of the page (e.g., download links).

F. *Stored Cross Site Scripting*

Stored XSS, also known as persistent XSS, is the more damaging. It occurs when a malicious script is injected directly into a vulnerable web application. The attacker who is inserting the malicious payloads can be served back to any other users unknowingly who will be browsing the pages of the web application and can also be executed later in their context. Therefore the end users do not require to click on a malicious link in order to run or execute the script, simply just needs to visit the affected page of the web application once in a browser.

G. *Unvalidated Redirects*

Unvalidated redirect vulnerabilities occur when an attacker is able to redirect a user to an untrusted site when the user visits a link located on a trusted website. This vulnerability is also often called Open Redirect. Some ways to detect are to look at the code for every place that utilizes a redirect. If there is no kind of whitelist for the URL being redirected, the site is probably vulnerable. Then crawl the site and save all pages that generate a redirect. If a parameter is changed, the site is most likely vulnerable. Next way is to manually look around and investigate all parameters that are suspected with redirects.

V. METHODOLOGY

A. *Pattern Matching Algorithm*

The SQL injection attack is possible by injecting specially crafted user inputs to the stored procedure. For prevention, the method proposed in this paper is dynamic semantic equivalence checking. For doing that the query structure that is being formed within the procedure is required. But, in case of stored procedures, getting query structure before actual execution is difficult. To manage this, we are constructing one additional procedure which is similar to the one being considered, but, with one additional output argument 'qry' for getting the dynamic query structure which is required for semantic equivalence checking.

For prevention, first execute this procedure with original arguments. Then the 'qry' variable will give the dynamic query structure that is being generated. For example, if the inputs given are ''1' or '1'='1'—' for uname and '' for password, then the result will be:

qry = select * from login where id='1' or '1'='1'-- and pass=

Now pass the original inputs and this query string to the above explained attack detection algorithm.

B. *Secure Hash Algorithm*

In cryptography, SHA-1 (Secure Hash Algorithm 1) is a cryptographic hash function designed by the United States National Security Agency and is a U.S. Federal Information Processing Standard published by the United States NIST. SHA-1 produces a 160-bit (20-byte) hash value known as a message digest. A SHA-1 hash value is typically rendered as a hexadecimal number, 40 digits long.

SHA-1 forms part of several widely used security applications and protocols, including TLS and SSL , PGP , SSH , S / MIME, and IPsec. Those applications can also use MD5; both MD5 and SHA-1 are descended from MD4. SHA-1 hashing is also used in distributed revision control systems like Git , Mercurial , and Monotone to identify revisions, and to detect data corruption or tampering.

C. *Salt Algorithm*

Salt is random data that is used as an additional input to a one-way function that "hashes" a password or passphrase. Salts are closely related to the concept of nonce. The primary function of salts is to defend against dictionary attacks or against its hashed equivalent, a pre-computed rainbow table attack.

Salts are used to safeguard passwords in storage. Historically a password was stored in plaintext on a system, but over time additional safeguards developed to protect a user's password against being read from the system. A salt is one of those methods.

A new salt is randomly generated for each password. In a typical setting, the salt and the password (or its version after Key stretching) are concatenated and processed with a cryptographic hash function, and the resulting output (but not the original password) is stored with the salt in a database. Hashing allows for later authentication without keeping and therefore risking the plaintext password in the event that the authentication data store is compromised.

Since salts do not have to be memorized by humans they can make the size of the rainbow table required for a successful attack prohibitively large without placing a burden on the users. Since salts are different in each case, they also protect commonly used passwords, or those who use the same password on several sites, by making all salted hash instances for the same password different from each other.

VI. CONCLUSION

In this process an automation tool which run at the time of developing the application, data injection is more secure prediction technique. The incorporation of the proposed algorithm into detection has provided more accurate results than existing oscillation monitoring schemes in the presence of data-injection attacks. The immunity of monitoring applications against intentional data injections has been enhanced. We also studied the different mechanisms through which SQL Injection Attacks can be introduced into an application and identified the techniques that are able to handle the mechanisms.

REFERENCES

[1] T. T. Kim and H. V. Poor, "Strategic protection against data-injection attacks on power grids," IEEE Trans. Smart Grid, vol. 2, no. 2, pp. 326–333, Jun. 2011.

[2] Guimarães, B. D., "Advanced SQL Injection to Operating System Full Control," Black Hat Europe, white paper, April 2009.

[3] Halde, J., "SQL Injection Analysis, Detection and Prevention," MSc Thesis, Department of Computer Science, San Jose State University, San Jose, CA, USA, 2008.

[4] Anirudh Anand, "https ://www. Owasp .org /index. php/Category: OWASP_ SQLiX _Project," 16 March 2014. [Online]. Available: https : // www .owasp. org /index. Php / Category : OWASP _ SQLiX _Project. [Accessed 10 June 2017].

[5] O. Kosut, L. Jia, R. J. Thomas, and L. Tong, "Malicious data attacks on the smart grid," IEEE Trans. Smart Grid, vol. 2, no. 4, pp. 645–658, Dec. 2011.

[6] S. Cui et al., "Coordinated data-injection attack and detection in the smart grid: A detailed look at enriching detection solutions," IEEE Signal Process. Mag., vol. 29, no. 5, pp. 106–115, Sep. 2012.

[7] L. Xie, Y. Mo, and B. Sinopoli, “Integrity data attacks in power market operations,” IEEE Trans. Smart Grid, vol. 2, no. 4, pp. 659–666, Dec. 2011.
[8] Z. Lijiu, Q. Gu, S. Peng and X. Chen, “D-WAV A Web Application Vulnerabilities Detection Tool Using Characteristics of Web Forms,” in Fifth International Conference on Software Engineering Advances (ICSEA), 2010, Nice, 2010.
[9] Lwin Khin Shar, Hee Beng Kuan Tan, "Predicting sql injection and cross site scripting vulnerabilities through mining input sanitization patterns", Information and Software Technology, vol. 55, no. 10, pp. 1767-1780, 2013.
[10] Lwin Khin Shar, Hee Beng Kuan Tan, Lionel C. Briand, "Mining sql injection and cross site scripting vulnerabilities using hybrid program analysis", Proceedings of the 2013 International Conference on Software Engineering, pp. 642-651, 2013.

P. SivagamaSundari received the **B.E.** degree in Computer science and Engineering from Adhiparasakthi Engineering College, Anna University, Chennai, India in 2000. Received **M.E.** degree in Computer Science and Engineering in Sri Krishna Engineering College, Anna University, Chennai, India in 2011. Her research interest includes Networks.

T.M. Padmapriya received the **B.E.** degree in Computer Science and Engineering from Shri Andal Alagar College of Engineering, Anna University Chennai, India, in 2008. Currently doing **M.E.** in Computer Science and Engineering in Adhiparasakthi Engineering College, Anna University, Chennai, India. Her research interest includes Networks and Information Security.