Web Analyzer: Detecting and Preventing Intrusions in Multitier Web Applications

Pallavi Deshmane
ME student, Dept. of Computer Engineering
Shah & Anchor Kutchhi Engineering College, Chembur, Mumbai, Maharashtra, India

Sonali Bhutad
Professor, Dept. of Computer Engineering
Shah & Anchor Kutchhi Engineering College, Chembur, Mumbai, Maharashtra, India

Abstract - Nowadays there is an increasing dependency on web applications, ranging from individuals to large organizations. Almost everything is stored, available or traded on the web. Therefore more customer data going online by adapting to online banking or fund transfer practices, users accounts and other information have become vulnerable to fraud and other attacks. In this paper, we use web analyzer which provides security to multitier web application. Web analyzer provides security at two layers that is securing the data storage at the back end as well as the front end applications. Web analyzer, which monitors web as well as database request to detect attacks in multitier web application.

Keywords – Intrusion detection, Intrusion prevention, SQL Injection attack. XSS attack

I. INTRODUCTION

A. Overview-

Web applications are the backbone of today's business and are enormously widespread. Nearly all information systems and business applications (e-commerce, banking, transportation, web mail, blogs, etc.) are now available as web-based database applications. They need to be universally accessed by clients, employees and partners around the world as online trading is becoming more and more ubiquitous in the global economy. These web applications, which can be used from anywhere, also become so widely exposed that any existing security vulnerability will most probably be uncovered and exploited by hackers. Hence, the security of web applications is a major concern and is receiving more and more attention from the research community.

Web applications are highly exposed to attacks such as cross site scripting attack, SQL injection attack from anywhere in the world, which can be conducted by using widely available and simple tools like a web browser. SQL Injection is a type of web application security vulnerability in which an attacker is able to submit a database SQL command which is executed by a web application, exposing the back-end database. A SQL Injection attack can occur when a web application utilizes user-supplied data without proper validation or encoding as part of a command or query. The specially crafted user data tricks the application into executing unintended commands or changing data[4].

B. System Architecture-

Figure 1. describes the system architecture. This figure depicts a multitier client -server model. Where in clients can send request to the server which passes through the web analyzer. The web analyzer analyses the request and forwards it to the web server. The webserver isolates the user traffic by providing virtualized environment to every user by providing a separate session id. So that attacker will not be able to compromise other user sessions. The web analyzer employed on database provides security to the database. The web analyzer is set up at the webserver as well as the database for enhanced security protection mechanism.

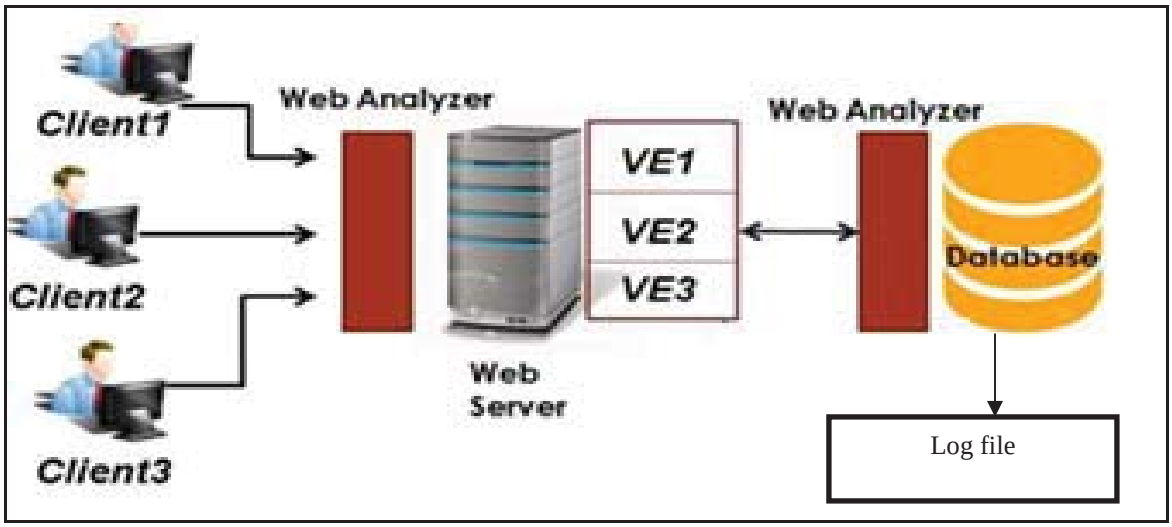

Figure 1. System Architecture

Types of attacks on multitier web application

An attacker can perform various types of attacks on web applications such as privilege escalation attack, SQL injection attack, cross site scripting attack.

1) Privilege Escalation Attack

Suppose that the website is used by both regular users and administrators. Regular users will trigger a web request with the set of SQL queries while an administrator will trigger a web request with the set of admin level queries. An attacker can logs into the web server as a normal user, and tries to obtain administrators data by triggering an admin queries. This type of attack can never been detected by IDS, either it is web server IDS or database IDS, because both the requests and queries are permissible. But according to our mapping model, a database query doesn't match the request and therefore we can detect this type of attack [1].

2) SQL injection attack

In this type of attack, attackers can inject SQL commands into an SQL statement, via web page input.

Consider the following SQL query:

SELECT * FROM Users WHERE Username='$username' AND Password='$password'

In the example above the variable $username & $password contains user-supplied data. A similar query is generally used from the web application in order to authenticate a user. If the query returns a value it means that inside the database a user with that set of credentials exists, then the user is allowed to login to the system, otherwise access is denied. Suppose user enters

username= 1' or '1' ='1 and password= 1' or '1' ='1 then the generated query will be:

SELECT * FROM Users WHERE Username= '1' or '1' = '1' AND Password= '1' or '1' = '1'

This query returns a value (or a set of values) because the condition is always true (OR 1=1).

In this way the system has authenticated the user without knowing the username and password [2].

3) Cross site scripting attack

Cross Site scripting is one of the problems that has plagued a lot of websites. According to White Hat Security Top Ten more than 50% of the websites are vulnerable to cross site scripting [8]. Cross site scripting is nothing but injection of client side scripts into a website. These scripts can be HTML scripts or JavaScript scripts. Now the question would be how a person can inject scripts on a running page. This can easily be done using all the various ways a website is collecting inputs. Cross site scripting can be performed by passing scripts in form of:

Text Box (input controls)

Query Strings

Cookies

Session variables

Application variables

C. Types of XSS attack-

1) Non-persistent/Reflected XSS Attack

In Reflected XSS, the attacker's payload script has to be part of the request which is sent to the web server and reflected back in such a way that the HTTP response includes the payload from the HTTP request [5]. One of the ways to achieve reflected XSS attack through websites text box. For example consider a web page which takes username as input and in output it reflects the username back but attacker can inject java script along with html content into text box, which is shown in figure 2.

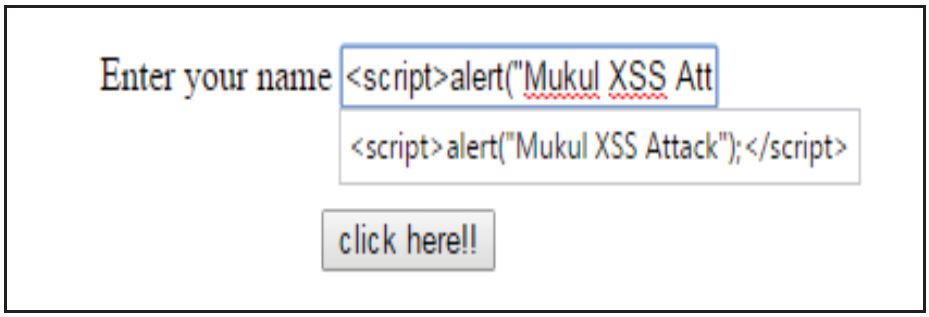

Figure 2. Java Script as Input

In a reflected XSS attack [3], the injected code is "reflected" off the web server such as in an error message which is shown in figure 3.

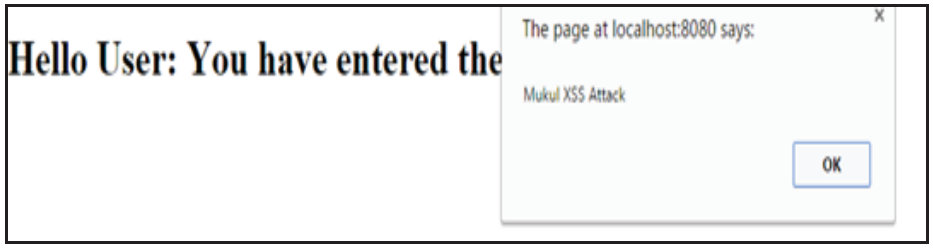

Figure 3. Reflected result due to Java Script

2) Persistent/Stored XSS attack

The most damaging type of XSS is Stored (Persistent) XSS. Because malicious scripts injected by attacker are stored permanently by server in databases or webpages and served back to other users as normal pages coming from trusted application and user is interpreting them without proper HTML sanitization. Various sinks like database, message forum, comment fields and visitor logs are used to store such malicious scripts permanently, and sent to user browser when request is made for particular content.

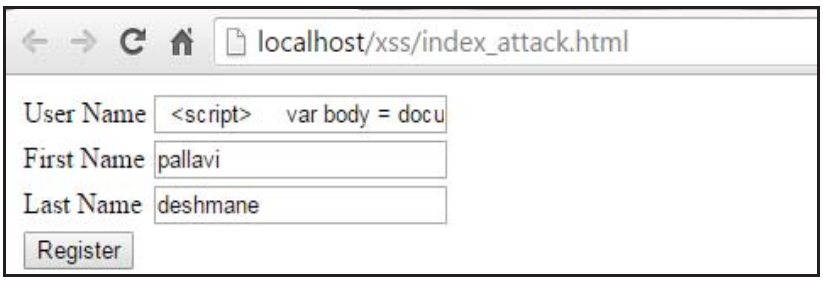

Figure 4. Java Script as Input

In figure 4. attacker has injected following java script in username field, that get stored in database <script> var body =document.getElementsByTagName ("body")[0]; body.innerHTML=""; body.style.backgroundImage = "url (http://wpcurve.com/wp-content/uploads /2013/08/ hacked.png)"; </script> [7].

Now every time when normal user will try to access the web page this script get executed automatically and instead of normal content this hacked image get delivered to the user as shown in figure 5., that is malicious content get delivered to the user.

Figure 5. Malicious content delivered to User

II. PROPOSED ALGORITHM

A. Model based architecture-

Mapping model will be built by considering various types of users and various types of operation to be performed by the users. Once the mapping model is built, it can be used to detect abnormal behaviors. Both the web request and the database queries within each session should be in accordance with the model. If there arises any request or query violating the normal model within the session, then the system will flag this activity as an attack. Suppose that the website is used by both regular users and administrators. Regular users will trigger a web request with the set of SQL queries while an administrator will trigger a web request with the set of admin level queries. An attacker logs into the web server as a normal user, and tries to obtain an administrators data by triggering an admin queries [1] then the system will flag this activity as an attack.

1) Design steps

User Authentication: The system detects whether the user is a valid user or an invalid user for the application and thereby enabling only valid users to utilize the application.

Application Intrusion Detection: System can detect the intrusion related to the features of the application wherein each type of user will be allocated a set of features identified by the mapping model created by the application for the user and thereby prevent invalid access operations to be executed.

Banking Operations: If user is identified as a valid user, then user can perform operations related to the application.

Database Intrusion Detection Module: System can detect intrusions occurring due to invalid structure of data which affects the backend systems of the applications. In order to facilitate detection of such invalid data structures system enables a second level of detection.

2) Attack Detection Technique

Privilege escalation attack: To detect privilege escalation attack the mapping model is used, which is created by the application for the user and thereby prevent invalid access operations to be executed.

SQL injection attack: The first step in this test is to understand when the application interacts with a DB Server in order to access some data. Typical examples of cases when an application needs to talk to a DB include:

a) Authentication forms: When authentication is performed using a web form, chances are that the user credentials are checked against a database that contains all usernames and passwords.

b) Search engines: The string submitted by the user could be used in a SQL query that extracts all relevant records from a database. To detect SQL injection attack regular expressions are used [7] to find unauthorized SQL statements in our code prior to executing the SQL statement. Regular expression is the use of a “string” to describe a or a class of features as “rules”, and then can verify the other group of target “string” is consistent with the characteristic rules. Regular expression is not affected by the development of environmental constraints. Java, .NET, and many other tools

are provided with regular expression support. A number of finite automata's can be integrated into a single regular expression.

Cross site scripting attack: Open web application security project (owasp) has its focus on improving the security of web applications. One project of the OWASP is called "ESAPI". It is the short form of "Enterprise security application programming interface". ESAPI is a toolkit, which provides a powerful set of input validation classes that not only validate user input but also filters user inputs. ESAPI filter is used to prevent XSS attack [6]. Thus, the ESAPI is available as a "Java", ".NET", "ASP", "PHP", or "JavaScript" implementation.

III. Experiment and Result

Figure 6. shows that if the number of sessions increases, the number of true positives increases. This leads to an effective system

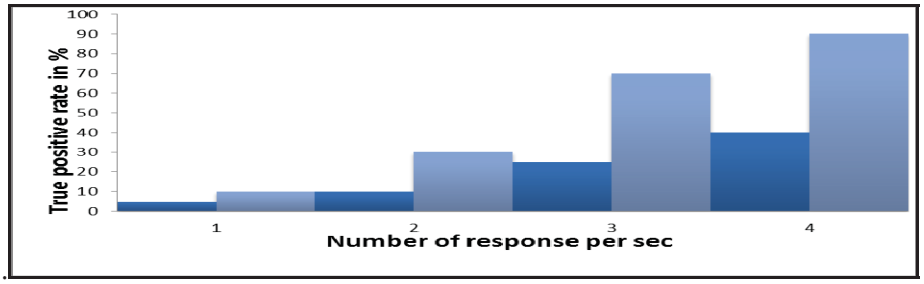

Figure 6. Nos. of sessions Vs. True Positive Rate

The false positive rates: It can be said to down perform the system's capacity to provide access to the users. As the number of sessions and operations increase, the number of false positive rate decreases as shown in figure 7.

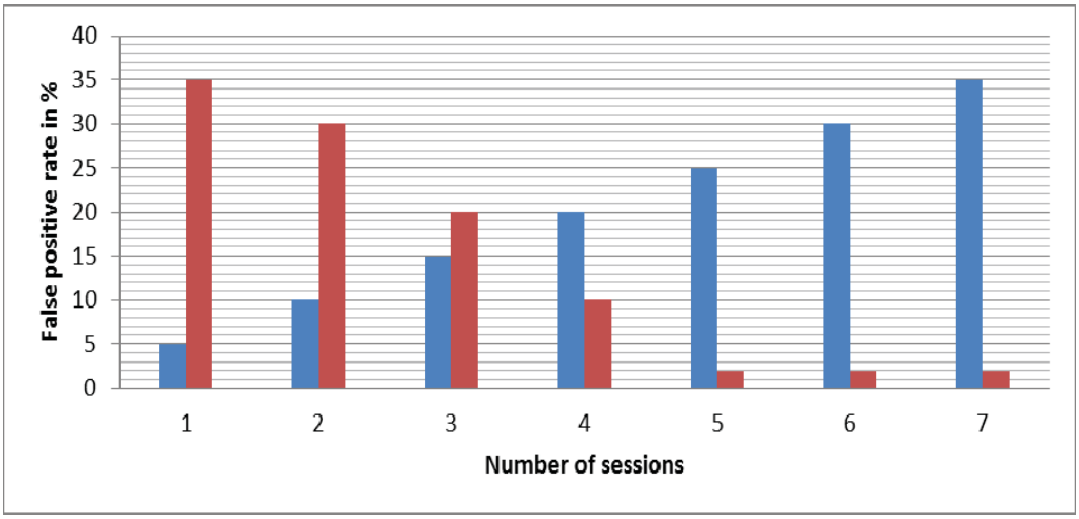

Figure 7. No. of sessions Vs. False Positive Rate

Table -1 shows the comparison table in which the evaluation of the system's performance measured against various other tools. The web analyzer is the most efficient tool for detecting various attacks.

Table -1 Comparison Table

Operation	Green SQL	Double Guard	Web Analyzer
SQL injection Attack	YES	YES	YES
Direct DB Attack	NO	YES	YES
XSS Attack	NO	NO	YES
Privilege Escalation Attack	NO	YES	YES

IV.CONCLUSION

The mapping model is used for detecting the anomalous behavior of the multitier web applications both at the front end as well as the backend data. The container based architecture provides session ID for different user sessions, which is used to isolate information flow of each user session. The web analyzer helps to protect the system against various attacks. The System can be extended to detect number of attacks. System can be integrated in the cloud. The attacks like the SQL injection can be done using the database queries with injection to the database server. Hence there is a need to protect these systems in the cloud storage, which is an important aspect of the quality of service.

REFERENCES

[1] Meixing Le, Angelos Stavrou, Brent Byung Hoon Kang," Double Guard: Detecting Intrusions in Multitier Web Applications", IEEE transactions on dependable and secure computing, vol. 9, no. 4, July/august 2012.

[2] Sruthy Manmadhan, Manesh T "A Method of Detecting SQL injection Attack to secure web applications" International Journal of Distributed and Parallel Systems (IJDPS) Vol.3, No.6, November 2012

[3] Engin Kirda, Christopher Kruegel, Giovanni Vigna, and Nenad Jovanovic "Noxes: A ClientSide Solution for Mitigating CrossSite Scripting Attacks" SAC'06 April 2327,2006, Dijon, France Copyright 2006 ACM 1595931082/06/0004

[4] William G.J. Halfond, Jeremy Viegas, and Alessandro Orso "A Classification of SQL Injection Attacks and Countermeasures" Proceedings of the IEEE, 2006

[5] https://www.acunetix.com/websitesecurity/xss/

[6] Marcus Niemietz "JavaScript-based ESAPI: An In-Depth Overview"

[7] http://larrysteinle.com/2011/02/20/use-regular-expressions-to-detect-sql-code-injection/

[8] http://wpcurve.com/wp-content/uploads/2013/08/hacked.png

[9] http://www.codeproject.com/Articles/573458/An-Absolute-Beginners-Tutorial-on-Cross-Site-Scrip