Cyber Attack Detection and Implementation of Prevention Methods For Web Application

Aishwarya Bhalme
Computer Engineering
MIT Academy of Engineering
Pune, India
avbhalme@mitaoe.ac.in

Akash Pawar
Electrical Engineering
MIT Academy of Engineering
Pune, India
adpawar@mitaoe.ac.in

Aditi Borkar
Computer Engineering
MIT Academy of Engineering
Pune, India
asborkar@mitaoe.ac.in

Mr.Pranav Shriram
Computer Engineering
MIT Academy of Engineering
Pune, India
prshriram@it.mitaoe.ac.in

***Abstract*— The internet and web applications are the only things that run the modern world. Today, the biggest concern facing businesses is web security. It is seen as serving as the fundamental framework for the global data society. Security breaches can happen to web applications. Web security is merely protecting a layer of a web application from attacks by attackers or unauthorized users. A large number of problems with web based applications are mostly the result of incorrect client input. The various facets of web security are covered in this paper, along with its flaws. This paper also discusses the key components of web security strategies, including encryption, authentication, passwords, and integrity. Additionally described in detail are the attack methods and the anatomy of a web based application attack. This paper explores a number of methods for detection and prevention of vulnerabilities in the web application. This research suggests a more effective method for reducing this category of web vulnerabilities. Additionally, it offers the finest defence against the a for mentioned threats.**

Keywords— Vulnerability, SQLi, LFI/RCE, XSS and Privilege escalation.

I. INTRODUCTION

You can use a web browser to access and browse applications on your device known as websites. In essence, it has two main elements in the background. web server, and database. A web server is a computer that handles, saves, and sends online pages that are requested by clients or end users. A database, on the other hand, is a systematic method of data storage. SQL and NO-SQL databases are the two different categories of databases. Using a SQL database, we can perform actions like inserting, deleting, and updating data using the structured query language, or sql. Complex data is stored using NO-SQL databases. We should secure the web application against unauthorized actions or web attacks because a web database may include sensitive data belonging to the client or end user. The term "attack" alludes to the idea that if an attacker or unauthorized person gains access to the database, he or she might change the data or carry out illicit actions, which is harmful to the clients' or customers' dignity. Basically, there are two types of attacks: 1. Server-side attack, 2. Client-side attack.

Server side can be done immediately on the targeted machine and doesn't need user interaction. After selecting the intended website, the attacker targets the webserver and collects the required data. This information may include the operating system that the target website runs on, the applications that are installed on the server machine, the services that are currently in use, and the port that is assigned to each of those services. Even while most services running on the distant server are designed to permit remote access, if they are not setup properly, an attacker may be able to take advantage of this situation and access these servers. Which poses a greater threat. In other words, an attacker has the ability to scan the IP address and obtain a list of every service that is active on the server. An attacker can then learn about any vulnerabilities related to each of the services by simply searching for them on the internet.

Gathering data on the victim is crucial because these attacks must be initiated by the victim. The kind of information the attacker is looking for includes, for instance, knowing who their friends are, the networks and websites they use, the websites and applications they trust, and so on. Client-side attacks target the victim, not the software or operating system, in contrast to server side attacks, which target both. Attackers gather data like the targeted machine's ip address, domain name, and technologies used behind the web application, such as the programming language used to create the website, port allocations such as which port is open, services installed on the targeted machine, and databases used in the backend to store the website data. Utilizing tools like Maltego, Zenmap, Nexpose, Knock, and others will help with this. Additionally, attackers utilize websites like "Whois lookup" to gather contact details, Ip addresses, server locations, server types, etc., and "Netcraft Site Report" to learn about the technologies employed by the machine that is being attacked. An attacker can access the machine without actually hacking it using the information acquired through these websites.

To introduce most common web application vulnerabilities which attackers prefer they are[1]:

A. File upload vulnerabilities:

978-1-6654-9291-1/22/$31.00 ©2022 IEEE

Any web application that allows users to upload any kind of file presents a vulnerability that an attacker can take advantage of by uploading a malicious file to the server.

B. Code Execution Vulnerability:

This vulnerability allows an attacker to use the input boxes to send any command to a web server.

C. *Local and Remote File Inclusion Vulnerability*:

A website vulnerability known as "Local File Inclusion" enables an attacker to issue a command by using the URL. The attacker is now able to read any file from the target computer. A website vulnerability known as remote file inclusion enables an attacker to inject any file onto the target machine. This implies that an attacker can then run commands, payloads, and establish a reverse connection.

D. SQL Injection Vulnerability:

With the help of this vulnerability, an attacker can tamper with database queries that an application makes. The attacker is then able to read and write files on the server as well as access and manipulate any data from the database.

E. *Cross-Site Scripting (XSS) Vulnerability:*

If a website generates output using user input that has not been sanitized, it is vulnerable to XSS. An attacker can access all items that the web page has access to by exploiting this vulnerability.Compared to 2018, there were a third fewer vulnerabilities on average per online application. The average number of vulnerabilities per system was 22, with four of those being extremely serious. Five vulnerabilities have one with a high severity.

II. Literature Survey

A Study on Web Application Security as well as Detecting Security Vulnerabilities [2]. Basically, this paper expands on SQL injection and cross-site scripting. Additionally, it explains how these are important vulnerabilities. These kinds of vulnerabilities have an immediate impact on the environment of web servers, application servers, and web applications. Web applications are less secure as a result of harmful attacks since a hacker could damage the database's integrity by using malicious queries. This document explains how to defend against XSS and SQLi attacks. The document includes information on OWASP and how it may be used to research vulnerabilities. OWASP focused on identifying vulnerabilities such as information theft, worms, phishing, and information warfare. proposed framework and architecture for identifying security flaws. attacks like SQL Injection Attack, Url Injection Attack, Cross Site Scripting Attack (XSS), etc. are carried out by attackers.

Component Based Web Application Firewall for Analyzing as well as Defending SQL Injection Attack Vectors [3]. The authors of this work offer defences against SQL Injection attacks. According to OWASP, structured query language injection is a top-rated vulnerability. SQL-related vulnerabilities in the source code of the application allowed for injection attacks by attackers, which resulted in the loss of important data and jeopardised database security and user privacy. Top 10 weaknesses that are frequently exploited. These vulnerabilities include SQL injection flaws, cross-site scripting (XSS), session management, flawed authentication and authorization, flawed access control mechanism, security misconfiguration, exposed sensitive data, and cross-site request forgery.

Extenuating Web Vulnerability with a Detection as well as Protection Mechanism for a Secure Web Access [4]. Web applications are the backbone of the internet, allowing every person to conduct daily tasks including information collecting, e-commerce, and other tasks. According to a 2014 poll, 70% of web apps are vulnerable to hacking due to the fact that they may be accessed from anywhere in the world. Cross-site scripting, often known as XSS or DOM-based XSS, is a security risk where an attacker inserts malicious client code into a webpage. Mechanism for defending a method (like having policy based webapp, penetration testing for web services, etc). WSP, policy filter, WS Policy Management, and Malevolent Report are a few examples of such mechanisms.

Proxy for XSS Web Security Attack. Machine Learning for Web Vulnerability detection: Case of Cross Site Request Forgery [5]. The authors of this research have suggested using machine learning to find web vulnerabilities. The author suggests a way for utilising Machine Learning (ML) to find vulnerabilities in online applications. Web applications are enticing targets for hostile users (attackers) who are intent to inflict financial losses, get unauthorised access to personal information, or cause their victims disgrace. Ransomware prevention as well as mitigation Techniques [6]. The authors of this study have suggested both ransomware mitigation and prevention measures. A group of malicious software known as ransomware encrypts user files and other resources using encryption, then demands cryptocurrency in exchange for the decryption key.

III. System Diagram

Scope, reconnaissance, vulnerability discovery, information analysis and planning, penetration testing, privilege escalation, result analysis, report, and clean up are the nine steps of the system diagram

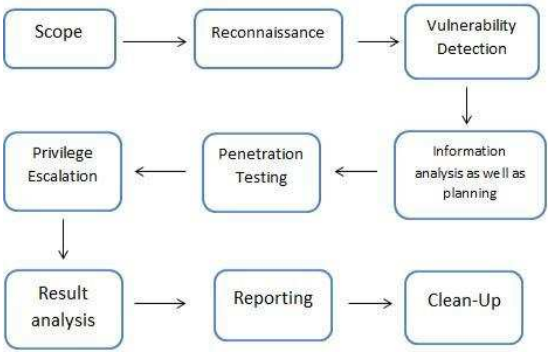

Fig 1. System diagram for Vulnerability assessment

Keep your Scope of this system is that includes all IT assets that are connect to organization network. Basically scope define what specify the system, network and application were all review as security assessment which states documentation what you reviewed. Reconnaissance is first phase of penetration testing.Tester is collecting information about system or target that information includes network topology, operating system and applications,user accounts and other applicable information.Vulnerability

detection means that identification of different type of vulnerability.Analysis as well as planning is using penetration testing techniques on system or target.Use penetration testing techniques on system or target. Authorized attack performed on computer system to finding the security.Privilege Escalation is escalating pre-define permission on system as well as trying to gain access of machine.With help of types of privilege escalation horizontal and vertical that we need to understand the type and how to protect against this in general.Analyzing result with help of given guideline and what are types of vulnerability and How to apply the prevention method.Report is final output of Vulnerability assessment and technical assessment as reference for teams.Clean up is job where apply all prevention method and do changes cleaning up the all flaw from target or system.

IV. METHODOLOGY

There are numerous methods for preventing web application vulnerabilities, but we have found that VAPT is most commonly used in many organizations because it's important for organizational security. Process of locating and describing vulnerabilities offers a way to identify and address security problem by ranking vulnerabilities before someone or something can exploit them. In this procedure, operating systems, software app and networks are examined to find instances of vulnerabilities, such as poor software design, unsecured authentication, etc. Vulnerability Assessment Process.

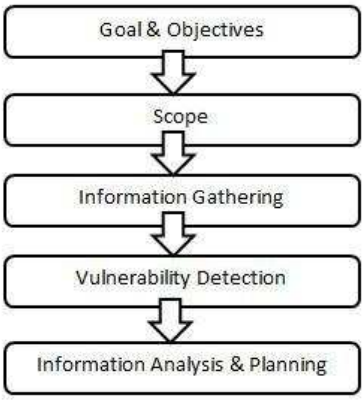

Fig 2. Phases of Vulnerability assessment

Goals for the vulnerability analysis should be defined.The scope of the assignment must be clearly established before the evaluation and test are administered.The three potential scopes are as follows: Black box testing is the process of testing an external network without having any prior knowledge of the inside network or systems. Grey box testing is the testing of either internal or external networks while having knowledge of both the system and the internal network. Black box testing and white box testing are both combined in it. White box testing involves testing within an internal network while having knowledge of both the internal network and the system. Likewise called internal testing.Getting a lot of data on the IT environment, including networks, IP addresses, operating system versions, etc. It is applicable to all three categories of scopes, including White Box, Grey Box, and Black Box testing.Vulnerability scanners are used in this step to scan the IT environment and find weaknesses. It will analyse the vulnerabilities found in order to create a strategy for breaking into both systems and networks.

Prevention method for Some of OWASP 10 vulnerability

A. File inclusion:

File Inclusion vulnerabilities regularly affect web programs that depend upon a scripting run time, as well as arise whilst a web software permits users to put up enter into files or add documents to server. They may be frequently located in poorly-written packages. Report Inclusion vulnerabilities permit an attacker to read as well as on occasion execute documents on victim server or, as is case with faraway record Inclusion There are two types of file inclusion firstly LFI(Local file inclusion) which defines it is attack used to trick application into exposing or going for walks files at server and secondly RCE(Remote code execution)which defines its main intention of attacker is to take advantage of referencing function inside goal application as well as to upload malware from a faraway URL, located on a one of a kind domain.

Fig 3. Methods used in LFI and RCE

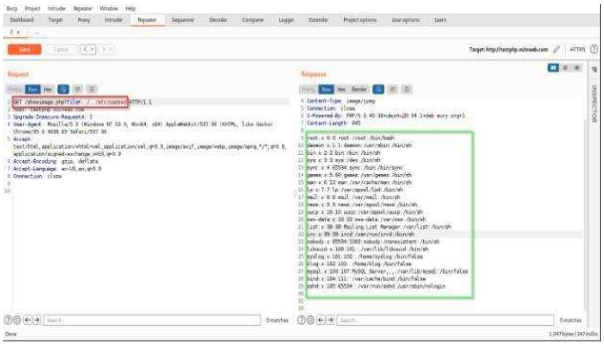

Fig 4. Proof of LFI which disclose the data

Remediation method:

- It is considerably safer to keep a white-list of permitted filenames and use a related identifier (rather than the actual name) to access files when parsing user-supplied filenames.
- Any request with an incorrect identification can thus be categorically denied.
- A local/remote file inclusion occurs when user input is supplied to PHP functions like include once, require once, require once, fopen, and readfile without being

properly checked. Consequently, you should never include files straight from variables that the user can change.

B. SQL injection (SQLI):

Using the online security flaw known as SQL injection, an attacker can alter database queries made by an application. In most circumstances, it gives an attacker access to data that they otherwise would not have. data belonging to other users or any other information to which the application is permitted to hold access.an attacker can update or remove this data, permanently altering the content or functionality of the application.Risk associated with this vulnerability includes the ability to edit, delete, and bypass authentication using data login credentials such username and password.

Remediation methods:

- *Input validation method:*

Whitelist structured data using regular expressions (such as name, age, income, survey response, zip code) make sure to use a fixed set of values and rigorous input validation.

- *Parameterized queries:*

Using parameterized queries, a SQL statement can be pre-compiled so that you can give parameters when deploying the statement. This technique enables the database to discriminate between input data and code.

- *Stored procedure:*

the developer must logically combine one or more SQL statements to produce an execution plan. Statements are automatically parameterized by subsequent executions.

- *Escaping:*

Use character-escaping functions for all user-supplied data available by each database management system (DBMS). This is done to avoid the DBMS from mistaking it for a SQL query supplied by the developer. For example, use mysql_real_escape_string() in PHP.

- *Avoiding administrative privileges:*

By not connecting your application to the database with a root account, you can prevent giving yourself administrative rights. If you don't, attackers may be able to access the entire system, so only do it if it's really necessary.

- *Use Web Application firewall*

C. Cross Site Scripting (XSS):

Attacks such as Cross-Site Scripting (XSS) include inserting malicious code into normally safe and reliable websites. XSS attacks take place when an attacker sends malicious code, typically in the form of a browser side script, to various end users through an online application. These attacks are made possible by vulnerabilities that are relatively common and can happen anywhere a web application uses user input in the output it produces without verifying or encoding it.

Remediation method:

- *Use escape techniques:*

To stop XSS attacks, a suitable escaping strategy can be used as the first line of security; examples include HTML escape, JavaScript escape, CSS escape, and URL escape. depending on where the untrusted string has to be inserted, it can be utilised within an HTML document.

- *Use a library to clean your HTML:*

You can't escape or encode user input if it must contain HTML because doing so would break valid tags. Use a reliable and validated library to parse and sanitise HTML in certain situations. Choose a library according on the programming language you're using; examples are Python Bleach, HtmlSanitizer, Sanitizer Helper, and DOMPurify.

- *Avoid JavaScript URLs:*

When utilised in URL DOM locations like anchor tag HREF attributes or iframe src locations, untrusted URLs that contain JavaScript protocol will execute JS code. Verify all untrusted URLs to make sure they only contain secure protocols like HTTPS.

- *Use HTTPOnly cookie flag:*

To ensure that cookie is not accessible via client-side JavaScript, we must set an HTTPOnly flag for cookies. This will help to mitigate impact of XSS vulnerability.

- *Implement Content Security Policy:*

Content Security Policy is a browser-side feature that enables you to set source whitelists for client-side resources of your web application, such as JavaScript, CSS, pictures, etc.

- *Use appropriate response headers:*

In HTTP responses that aren't designed to contain HTML or JavaScript, XContent-Type-Options headers can be used to avoid XSS.

- *Use tools made for purpose:*

It could be difficult to manually audit code to check for XSS vulnerabilities. Tools like BurpSuite, Acunetix, IBM Rational AppScan, and many more can be used to discover these kinds of vulnerabilities in the code.

V. RESULT AND ANALYSIS

We had taken linux machine to check what are weakness of the machine like different type of vulnerability in application.Detected vulnerabilities are SQL injection, Local File Inclusion/Remote Code Execution and Cross Side Scripting. We had differentiate result into two part firstly without prevention method used what are vulnerability visible and how they are impacting as well as Secondly With prevention method used application how they are defending or preventing to malicious activity. Without Prevention method in simple login page which is simply hosted on linux machine which bypass by attacker using SQL injection payload and access the admin portal or extract sensitive from database,

Username: admin'OR'1=1

Password: admin'OR'1=1

second part is attacker can delete the data from database by using following SQLi payload (query),

Username : admin

Password :';DELETE FROM user1 WHERE '1=1

Next type is LFI/RCE that is local file inclusion/ Remote code execution which because of parameter open on database setting allow_url_fopen and and allow_url_include are on so we can able add and see the all file impact of this manipulation and leaking of secret information as shown in below figure

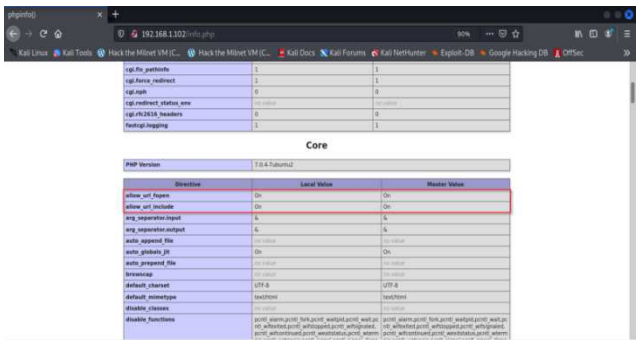

Fig 5. Parameter are open in configuration

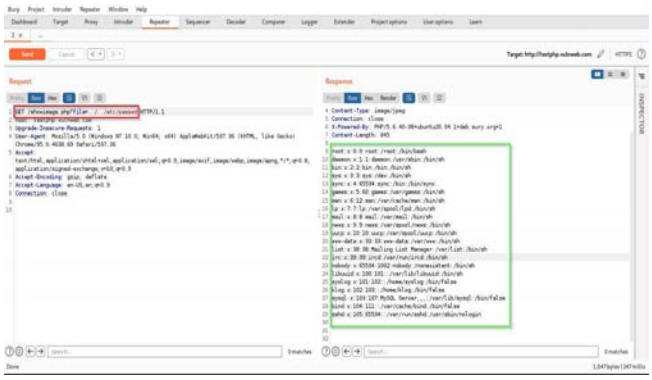

Fig 6. Leaking the secret information(/etc/passwd file)

Secondly part which because of unsanitize user input can done broken authentication.

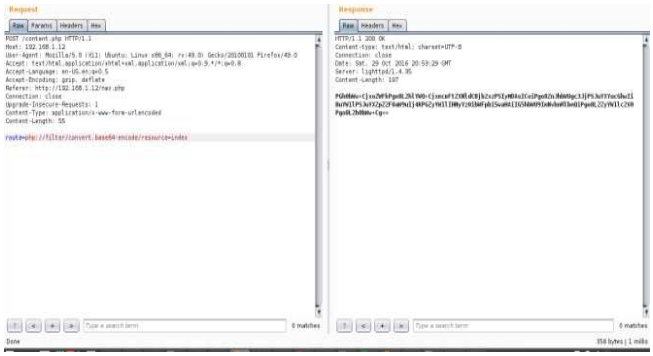

Fig 7. Information leaking (login credential)

Third part contain the route post parameter because of this with help of php flter getting course code and with help of this we add malicious file path and can run on application and get reverse shell Proof of XSS is running alert() with help of this we can read any data and attacker can access it With prevention method we can minimize the weakness of the application like fro preventing the SQLi attack using input validation method and parametrized query ,LFI/RCE vulnerability using Disable allow_url_include and allow_url_fopen in php.ini or .htaccess, Sanitize user input, not able get file /etc/passwd file, Use sanitizing and validation method for user input, running limited privilege application, remove execution statement from source code and fro XSS vulnerability by filtering the input and perform content security policy for application.

The existing system for securing a web-based application is complicated so it has several reasons like the heterogeneity and complexity of web apps to the adoption of undisciplined scripting language offering dubious security guarantees and is not amenable to static analysis. through this perspective might miss important insights so by white-box techniques which require to access web application source code and other side black-box methods that operates only the HTTP traffic(request and response) and this is made using Burp Suite and OWASP ZAP tools. In Machine learning for Web Vulnerability detection: The case of cross-site request forgery journal paper describing web application vulnerability detection for CSRF using machine learning this system value of standard HTTP request headers such as referrer and origin indicating the page originating the request, X-Request cannot be set from the cross-site position using the RandomForestClassifier. In this proposed system it is not specifically for CSRF but is used to detect the most type of vulnerability like SQLi, XSS, LFI, etc using Vulnerability analysis and penetration testing process.

VI. Conclusion

In addition to understanding the key security functions and the characteristics that safe web applications should have, we have covered every aspect of web application development and classified the body of literature into three main categories. We are also talk about a few topics that still need to be consider. Different programming concepts and techniques are using to access a few out-of-the-box functionalities in web applications, which has importants security implications for our apps. Typically, our browsers, which expose programming concepts, are where our logic and important programs sit. As a result, it is simple for attackers to disrupt the server-side application state and intercept functionality. Because Vulnerability assessment is an open-source application that detects almost all vulnerabilities, is automated for scanning, and is simple to run regularly, it is a more straightforward method for vulnerability prevention. Most businesses and organizations use vulnerability assessment to secure their web applications. We implemented vulnerabilities like LFI and RFI, SQLi, XSS, and others. Through this, we gained knowledge of how to identify vulnerabilities, scan a target, and exploit any web application. The injection of malware on a computer or target that compromises confidential information like login credentials, company hidden files, etc. may be the fault of LFI and RFI. Secret data stored in databases, such as credit card information and login credentials, are released through SQL

injection and cross-site scripting (XSS). Therefore, we had to establish specific preventative methods in this project to prevent this are really dangerous attack.

REFERENCES

[1] https://owasp.org/www-project-top-ten/

[2] Fairoz Q. Kareem, Siddeeq Y. Ameen , Azar Abid Salih, Dindar Mikaeel Ahmed , Shakir Fattah Kak , Hajar Maseeh Yasin , Ibrahim Mahmood Ibrahim, Awder Mohammed Ahmed, Zryan Najat Rashid as well as Naaman Omar1; SQL Injection Attacks Prevention System Technology,2021

[3] V S Stency as well as N Mohanasundaram ; A Study on XSS Attacks: Intelligent Detection Methods,2021

[4] RenuVig .SubirHalder, MauroConti ,Aanshi Bhardwaj 2021 DirectDistributed denial of service attacks in cloud: State-ofthe-art of scientific as well as commercial solutions

[5] T.P.Latchoumia , Manoj Sahit Reddy,K.Balamurugan; European Journal of Molecular &Clinical Medicine;Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection as well as Prevention,2020.Clinical Medicine;Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection as well as Prevention,2020.

[6] mi Christiana Abikoye, Abdullahi Abubakar, Ahmed OluwakeHaruna Dokoro, Oluwatobi Noah Ak as well ase & Aderonke Anthonia Kayode ; A novel technique to prevent SQL injection as well as cross-site,2020.

[7] Y. deepak dembla, yogesh chaba1 , kuldeep yadav, mridul chaba, as well as ashish kumar; Advances in Mathematics: Scientific Journal 9 a novel as well as effic;ient technique for prevention of xss attacks using knapsack based cryptography, 2020.

[8] https://portswigger.net/web-security/access-controlhttps://www.acunetix.com/websitesecurity/sql-injection/

[9] https://www.ptsecurity.com/ww-en/analytics/web-vulnerabilities-2020/

[10] https://www.sqlshack.com/sql-injection-detection- as well as-prevention/

[11] Raniah Alsahafi: sql Injection Attacks: Detection as well as Prevention Techniques, international journal of scientific & technology research, volume 8, issue 01,2019.