International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 05 | May 2021 www.irjet.net p-ISSN: 2395-0072

Web Vulnerability Scanner

Rushabh Timbadia[1], Suraj Pawar[2], Devansh Sayani[3], Riken Shah[4], Bhavna Arora[5]

[1,2,3,4]Computer Engineering, Atharva College of Engineering, Mumbai, India
[5]Prof. Bhavna Arora, Dept. of Computer Engineering, Atharva college of Engineering, Malad, India

---***---

Abstract - *Now a day's the software development life cycle should incorporate the security features. Attacks like Input Validation Attack are most diverse attacks on the web application. Our main intention is to focuses on detection and prevention of Input Validation attacks like SQL Injection, Cross Site Scripting File Inclusions, OS Command Injection by incorporating security in software development life cycle. Cross site scripting (XSS), SQL Injection, File Inclusions, OS Command Injection are types of attacks on web pages and account as the unsafe vulnerability existed in web applications. Once the vulnerability is detected, the hijacker advances intended access of the authenticate user's web-browser and may perform session-hijacking, cookie-stealing, or various other malicious attacks. To prevent such attacks, it is important to implement security measures that block the third-party intrusion. Vulnerabilities of websites are exploited over the network through web request using GET and POST method. In this project, we are focusing on injection, detection and information about preventing these attacks. The unavailability and high price of automated scanners for detecting vulnerabilities in web applications which leads to defacement, hijacking, stealing data from server creates a security problem for all business as well as government people. So, making of an affordable scanner is important. In this project we will develop vulnerability scanner for finding vulnerabilities in web application and provide information about its remediation*

***Key Words*: Cross site scripting, Remote file inclusion, Local file inclusion, SQL injection, Command injection.**

1. INTRODUCTION

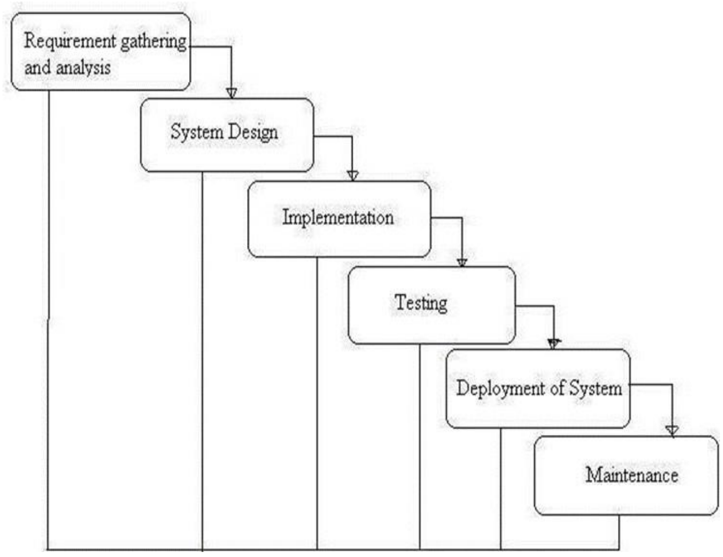
According to the current state of affairs, cyber risks are a major threat to small as well as big organizations and although big companies have the power the prevent and fight these, small companies and start-ups lack the financial as well as physical capacity to do so. So, by eliminating the high cost factor and stripping down and bringing the bare bones version of complex software used to identify and provide solution for loop-holes in a webpage of any given company, we aim to fill the gap and show the possibilities an affordable software can also provide.

1.1 NEED

Cyber risk is now at the center of the international agenda as high-profile breaches and hacking is raising fears that such attacks and other security breaches could endanger the global economy. Cybercrime is estimated to cost the global economy over US $400 billion every year, according to estimates by the Centre for Strategic and International Studies. In 2017, some 10,000 companies in the United States had their systems compromised by criminals, the Centre reports. So, there is a necessity for an automated software which will help in recognizing loop holes in web applications

1.2 BASIC CONCEPT

In Layman's terms, the basic concept is to create a software which does all the major tasks as done by much expensively priced software used by companies, as well as keeping the costs down. The software will scan full websites and find vulnerabilities as well as give information and solution on how to fix it.



2. REVIEW OF LITERATURE

Paper Name	Summary
"An Effective Method for Preventing SQL Injection Attack and Session Hijacking" [1]	Most of the transaction information or the customer information is stored in the backend databases for these web applications. One of the vulnerabilities of these web applications is SQL (Structured Query Language) injection attack. Also, the web application sessions are prone to session hijacking attack, if the adversary can get hold of the session id. Considering that there are various tools available to retrieve session/HTTP cookies, this makes web applications very vulnerable session hijacking attacks. Though there are many ways proposed to defend the databases against SQL injection attacks, there is no sure shot way to prevent these SQL injection attacks. This project proposes an efficient technique for the prevention of SQL injection attack and session hijacking. The hashing technique is used for implementing the prevention these attacks.
"A second-order SQL injection detection method" [2]	This Second-order SQL injection is a serious threat to Web application and it is more difficult to detect than first-order SQL injection. The attack payload of second-order SQL injection is from untrusted user input and stored in database or filesystem he SQL statement submitted by web application is usually dynamically assembled by a trusted constant string in the program and untrusted user input, and the DBMS in unable to distinguish the trusted and untrusted part of a SQL statement. The paper presents a method of detecting second-order SQL injection attacks based on ISR Instruction Set Randomization. The method randomize the trusted SQL keywords contained in Web applications to dynamically build new SQL instruction sets, and add a proxy server before DBMS, the proxy detects whether the received SQL instruction contains standard SQL keywords to find attack behavior. Experimental results show that this system can effectively detect second-order SQL injection attack and has low processing cost.
"A sound framework for dynamic prevention of Local File Inclusion" [3]	VisAR Web applications take an important role in remote access over the Internet. These applications have many capabilities such as database access, file read/write, calculations as well as desktop applications but run in web browsers environments. As desktop applications, web applications can be exploited but with different techniques. One of the major known vulnerabilities of the web applications is Local File Inclusion. Inclusion in web applications is similar to library imports in desktop applications where a developer can include former developed codes. If an attacker includes his/her libraries, he/she can run his/her malicious code. Current research makes a brief survey of static and dynamic code analysis and suggests a framework for dynamically preventing malicious file inclusions by attackers. It is discussed that this framework prevents local file inclusions even if the developer has exploitable source code. The language PHP is used for describing the vulnerability and prevention framework.
"Prevention of Website Attack Based on Remote File Inclusion-A survey" [4]	This Web applications take an important role in remote access over the Internet. These applications have many capabilities such as database access, file read/write, calculations as well as desktop applications but run in web browsers environments. As desktop applications, web applications can be exploited but with different techniques. One of the major known vulnerabilities of the web applications is Local File Inclusion. Inclusion in web applications is similar to library imports in desktop applications where a developer can include former developed codes. If an attacker includes his/her libraries, he/she can run his/her malicious code.Current research makes a brief survey of static and dynamic code analysis and suggests a framework for dynamically preventing malicious file inclusions by attackers. It is discussed that this framework prevents local file inclusions even if the developer has exploitable source code. The language PHP is used for describing the vulnerability and prevention framework.
"An automated vulnerability scanner for injection attack based on injection point" [5]	We implemented a system that automated scanned the injection attack vulnerabilities. Our system was automatically analyses web sites with the aim of finding exploitable SQL injection and XSS vulnerabilities. White-box testing has not experienced widespread use for finding security flaws in web applications. An important reason is the limited detection capability of white box analysis tools, in particular due to heterogeneous programming environments and the complexity of applications that incorporate database, business logic, and user interface components.
"Design of efficient web vulnerability scanner" [6]	This paper elaborates existing web vulnerability detecting approaches with their advantages and disadvantages. We propose a clustering approach to efficiently detect the SQL Injection and Cross Site Scripting attacks. The objective is to improve detection efficiency of vulnerability scanner while maintaining low false positive and false negative rate. Although the scanner reduces the false positive rate, but it also ignores some true results as well. The scanner gets confused if a parameter is valid for one condition to be consider as a bug and another condition doesn't satisfy.
"Common Requirements for Web Application Vulnerability Scanners for the Internet of Things" [7]	This paper presents 3 common requirements for web application vulnerability scanners for theInternet of things devices, including browser's rendering engine support, false positive minimization, and device setting change minimization. These requirements have been drawn from the experience of the previous project, security vulnerabilities in residential gateways. An I-WAVS should minimize false positives by excluding uncommon vulnerabilities in IOTdevices. an I-WAVS should be able to set 'forbidden' links, buttons, or pages to minimize the device setting changes.
"A network-based vulnerability scanner for detecting SQLI attacks in webapplications" [8]	Vulnerability scanners has been proposed to deal with this, but none of them are able to detectSQLI completely, the existing tools have the accuracy ratio very less as well as they produce ahigh rate of false positive, apart from that all these tools take much time to scan. The scanningspeed of their scanners are extremely slow. Focusing on one vulnerability at a time leaves other bugs from detection.
"Testing and Comparing Web Vulnerability Scanning Tools for SQL Injection and XSS Attacks" [9]	A method to evaluate and benchmark automatic web vulnerability scanners using software fault injection techniques. 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.
"Research and design on Web application vulnerability scanning service" [10]	The paper focuses on a research among the existing Web application scanners firstly. Then we selected W3af (Web Application Attack and Audit Framework) as a basic platform for transformation, and by customizing scanning modules and scripts.
"Non-Detrimental Web Application Security Scanning" [11]	In this paper we introduce a testing methodology that allows for harmless auditing, define three testing modes—heavy, relaxed, and safe modes, and report our results from two experiments. In the first, we compared the coverage and side effects of the three scanning modes using 5 real-world Web applications chosen from the 38 found vulnerable in a previous static verification effort. In the second, we used the relaxed mode to conduct a 48-hour test involving 1120 random websites, of which 55 were found to be vulnerable. This test was limited to a search for XSS vulnerabilities using the relaxed mode.

3. REPORT ON THE PRESENT INVESTIGATION

Currently, there are a good amount of software available for enterprise as well as small company oriented but the problem with each of them is that they get too technical and complex and are extremely expensive to purchase. Specially impacting small Indian companies and startups which can't afford them.

Software like Acunetix and Burp provide varied plans but their starting costs ranges from thousands to millions of dollars, which goes up each year. They provide features which are excess and not needed by many companies and also quite difficult to use. But due to not many alternative solutions and people not having much knowledge on the applications itself, it becomes harder for people to employ affordable options.

The current softwares available are mainly focused to provide solutions for Multi-National Companies with many employees and a good revenue in order to afford the costs. There are not many softwares which can provide the same vulnerability scanning at cheaper rates and in a more understandable, easy to use and interactive way.

4. OBJECTIVES

Our aim is to develop a system to detect the vulnerabilities like SQLi, Cross site scripting, Local and Remote file inclusion, Command injection in web applications and it will also provide information about remediation of vulnerable URL and its vulnerability.

- To find the vulnerabilities like SQLi, Cross Site Scripting, Local and Remote File Inclusion, Command Injection in web applications.
- To provide information about remediation of vulnerable URL and its vulnerability.
- To provide all these at affordable rates.

5. PROBLEM STATEMENT

There is unavailability of automated scanners for detecting vulnerabilities in web applications. And some scanners which are present are highly expensive. These leads to defacement, hijacking, stealing data from server. These creates a security problem for all business as well as government people. So, making of an affordable scanner is important. Scanner first crawl the webpages of a particular domain and scan each URL using different payloads. By using different algorithms, it finds out if a URL is vulnerable or not.

6. SCOPE

- The software will scan the full web application.
- The main focus will be on finding vulnerabilities on parameter-based URLs.
- It will show information about vulnerable URL and its vulnerability and gives the solution to fix it.

7. Feasibility

Operational Feasibility:
Operational Feasibility is the ability to utilize, support and perform the necessary tasks of a system. The proposed solution requires hardly any prior knowledge of the technologies involved for the end users and hence it is adaptable to use the application. The user will be able to navigate through the application using its interactive UI design and get used to the interface in no additional time.

Technical Feasibility:
Technical Feasibility, the proposed software & hardware requirements for our Application is well within the current boundary of technology. The motive of the project is affordable pricing as well as feasible software, which is kept in mind and hence our application will be made such that it can run on every Computer from this decade.

Economic Feasibility:
The scope and benefits of this project is well enough to outweigh and overcome the cost. Main agenda of our software is to offer similar vulnerability scanning techniques like that of a high- end enterprise solution at much affordable costs and an easier interactive GUI for the user.

8. METHODOLOGY

8.1 Requirement gathering and analysis:

In this step of waterfall, we identify what are various requirements are need for our project such as software and hardware required, database, and interfaces.

8.2 System Design:

In this system design phase, we design the system which is easily understood for end user i.e. user friendly. We design some UML diagrams and data flow diagram to understand the system flow and system module and sequence of execution.

8.3 Implementation:

In this phase of our project we have successfully implemented different modules required for getting predicted outcome at the different module levels.

With inputs from system design, the system is first developed in small programs called units, which are then integrated in the next phase. Each unit is developed and tested for its usability which is then referred to as Unit Testing.

8.4 Testing:

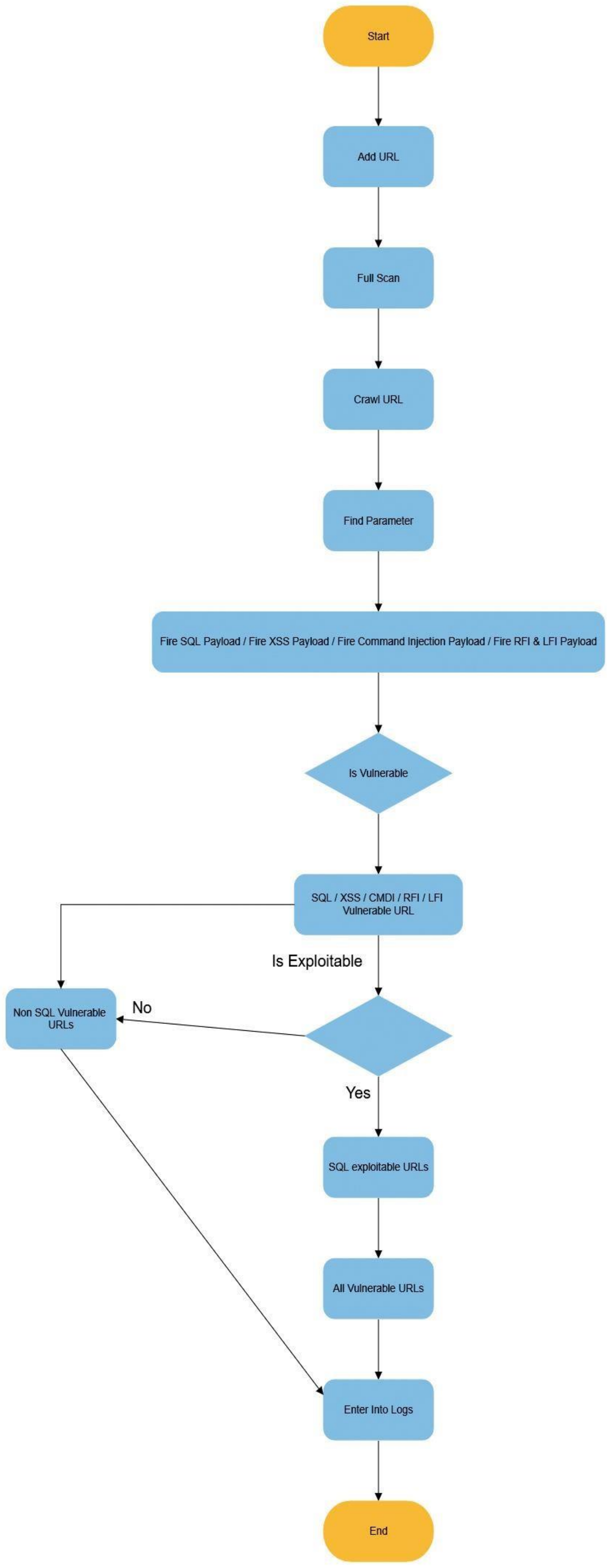
The different test cases are performed to test whether the project module is giving expected outcome in assumed time. All the units developed in the implementation phase are integrated into a system after testing of each unit. The entire system is then tested for faults and failures post integration.

8.5 Deployment of system:

Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.

8.6 Maintenance:

There are various issues which come up in the client environment. To fix those issues patches are released. Also, to reduce faults and failures of the product some better versions are released. Maintenance is responsible to deliver these changes in the customer environment. All these phases are cascaded to each other in which progress is seen as flowing steadily downwards like a waterfall through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". No model phases overlap during this.



9. APPLICATIONS

1. Use by security researchers to find security vulnerabilities in their domain.
2. In business or corporate websites to detect security vulnerabilities.
3. Used by individual ethical hackers to secure websites.

10. ADVANTAGES

1. Required less time
2. Increase Efficiency
3. Improve the accuracy.

11. CONCLUSIONS

This report discussed web application security principles and fundamental information that can help us to prevent web exploits in our system. The most exposed and least protected are considered to be the web applications thereafter vulnerable because the standards somehow are not focused on security but more into serving the need functionality.

Security threats are more common than before because the internet has become today's economy most valuable tool for everyone. So, there is indeed need to protect our resources, data and user privacy information. As technology move forward and brings new strategies, tools, models and methods to increase security levels, hackers will be part of this never end game.

The proposed system is developed to detect the vulnerabilities like SQLi, Cross Site Scripting, Local and Remote File Inclusion, Command Injection in web applications and it will also provide information about remediation of vulnerable URL and its vulnerability. Our formalization goes at the heart of the problem and captures seemingly different types of above- mentioned vulnerabilities. The simple and effective strategy is meant to be cost effective and is openly targeted toward large commercial applications. The results are satisfactory.

REFERENCES

[1] K. D'silva, J. Vanajakshi, K. N. Manjunath and S. Prabhu, "An effective method for preventing SQL injection attack and session hijacking," 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, 2017, pp. 697-701, doi: 10.1109/RTEICT.2017.8256687.

[2] C. Ping, "A second-order SQL injection detection method," 2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, 2017, pp. 1792-1796, doi: 10.1109/ITNEC.2017.8285104.

[3] M. S. Tajbakhsh and J. Bagherzadeh, "A sound framework for dynamic prevention of Local File Inclusion," 2015 7th Conference on Information and Knowledge Technology (IKT), Urmia, 2015, pp. 1-6, doi: 10.1109/IKT.2015.7288798.

[4] P.S.Sadaphule, Priyanka Kamble,Sanika Mehre,Utkarsha Dhande,Rashmi Savant" Prevention of Website Attack Based on Remote File Inclusion-A survey" International Journal of Advance Engineering and Research Development" e-ISSN : 2348-4470.

[5] J. Chen and C. Wu, "An automated vulnerability scanner for injection attack based on injection point," 2010 International Computer Symposium (ICS2010), Tainan, 2010, pp. 113 - 118, doi: 10.1109/COMPSYM.2010.5685537.

[6] S. Patil, N. Marathe and P. Padiya, "Design of efficient web vulnerability scanner," 2016 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, 2016, pp. 1-6, doi: 10.1109/INVENTIVE.2016.7824873.

[7] H. Y. Lee and Y. S. Park, "Common Requirements for Web Application Vulnerability Scanners for the Internet of Things," 2017 International Conference on Software Security and Assurance (ICSSA), Altoona, PA, 2017, pp. 111-111, doi: 10.1109/ICSSA.2017.31.

[8] A. Kumar Singh and S. Roy, "A network based vulnerability scanner for detecting SQLI attacks in web applications," 2012 1st International Conference on Recent Advances in Information Technology (RAIT), Dhanbad, 2012, pp. 585-590, doi: 10.1109/RAIT.2012.6194594.

[9] J. Fonseca, M. Vieira and H. Madeira, "Testing and Comparing Web Vulnerability Scanning Tools for SQL Injection and XSS Attacks," 13th Pacific Rim International Symposium on Dependable Computing (PRDC 2007), Melbourne, Qld., 2007, pp. 365-372, doi: 10.1109/PRDC.2007.55.

[10] W. Qianqian and L. Xiangjun, "Research and design on Web application vulnerability scanning service," 2014 IEEE 5th International Conference on Software Engineering and Service Science, Beijing, 2014, pp. 671-674, doi: 10.1109/ICSESS.2014.6933657.

[11] Yao-Wen Huang, Chung-Hung Tsai, D. T. Lee and Sy-Yen Kuo, "Non-detrimental Web application security scanning," 15th International Symposium on Software Reliability Engineering, Saint-Malo, Bretagne, 2004, pp. 219-230, doi: 10.1109/ISSRE.2004.25.