ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 8, Issue 8, August 2021

Automated Web Application Vulnerability Scanner

[1] Pranav Gadekar, [2] Samruddhi Kulkarni, [3] Shalaka Kulkarni, [4] Shruti More

[1][2][3][4] Dept. of Computer Engineering, Marathwada Mitra Mandal's College of Engineering, Pune, India

***Abstract*---- In recent times, use of web and web-based technologies have become more popular. The web applications are the most common interface for security-sensitive information and functionality available. As web applications are sources of sensitive data, they are prone to vast numbers of web-based attacks. The majority of these attacks happen because of vulnerabilities resulting from input validation problems. Although these vulnerabilities are easy to understand and mitigate, many web developers are unaware of these security aspects. Which results in more vulnerable web applications on the Internet. Among these, the most prominent vulnerabilities are SQL Injection and Cross Site Scripting (XSS). We implemented a system which will scan the web application for the most frequent vulnerabilities in an automated manner. Our system detects flaws in web applications and presents a comprehensive report.**

***Keywords*— SQL Injection, Cross Site Scripting, Web Application Testing, Security Scanner, Exploitation, Code Injection, Web Security, Machine Learning, Artificial Intelligence**

I. INTRODUCTION

As of January 2020, there have been over 1.74 billion websites on the web. Hackers attack every 39 seconds, on the average 2,244 times each day. This gives us the idea that many websites on the Internet are vulnerable to different attacks. As of the end of 2019, 42% of publicly facing websites are prone to SQL Injection and 19% to Cross Site Scripting attacks. A security researcher has earned a $25,000 bug bounty after finding a DOM-based Cross-Site Scripting (XSS) vulnerability in one of the most popular social media sites 'Facebook'. Another such attack, in August 2019, was on the famous coffee chain 'Starbucks' web services that created a way to access their critical database through the SQL Injection Vulnerability. From the above discussion, we can infer that Security plays an important role in developing websites. Unfortunately, web developers are not aware of these security aspects resulting in more vulnerable websites. Some of the most commonly occurring ones being SQLi, XSS, CSRF, Sensitive Data Exposure. So we are developing a system that will find these vulnerabilities in given web applications and report them to the user of the system. We are developing a system that will accept the target URL from the user.The system will then crawl the target URL in an Automated way using AI techniques and collect all the connected URLs. Then it will scan all collected URLs and it will test different payloads to exploit the vulnerabilities. Finally, a report will be generated which will contain the detected vulnerabilities and payloads used.

II. RELATED WORK

A. Various Vulnerability Scanner: A Survey

- Acunetix Vulnerability Assessment Engine: It's an entire web application security testing solution that will be used both standalone and as a part of complex environments. It offers built-in vulnerability assessment and vulnerability management, also as many options for integration with market-leading software development tools. It is not an open-source tool. It is the most expensive tool available.
- Burp Suite Web Vulnerability Scanner: Burp Scanner uses PortSwigger's world-leading research to assist its users to hunt out an honest range of vulnerabilities in web applications, automatically.
- Qualys Web Application Scanner: WAS' dynamic deep scanning covers all apps on your perimeter, in your internal environment and under active development, and even APIs that support your mobile devices. It also covers public cloud instances and provides you instant visibility of vulnerabilities like SQLi and XSS.
- Nessus Vulnerability Scanner: Nessus is the vulnerability assessment solution for security practitioners. The latest intelligence, rapid updates, an easy-to-use interface. It is also the costlier one.

B. Analysis of Vulnerability Scanning

Machine Learning for Web Vulnerability Detection The Case of Cross-Site Request Forgery published within the year 2020 by Stefano Calzavara, Mauro Conti, Riccardo Focardi, Alvise Rabitti, Gabriele Tolomei. It has the key advantage of offering a language-agnostic vulnerability

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 8, Issue 8, August 2021

detection approach, which abstracts from the complexity of scripting languages and offers a consistent interface to the widest possible range of web applications. [1]

An efficient algorithm and tool for detecting dangerous website vulnerabilities in the year 2020 and written by Hoang Viet Long, Tong Anh Tuan, David Taniar, Nguyen Van Can, Hoang Minh Hue. The proposed new technique has the advantage of detecting attacks in nested SQL queries and giving a good performance. [2]

An Automated Composite Scanning Tool with Multiple Vulnerabilities within the year 2019 published by Xun Zhang, Jinxiong Zhao, Fan Yang, Qin Zhang, Zhiru Li, Bo Gong, Yong Zhi, Xuejun Zhang. It enables the automatic detection tool to implement automatic vulnerability scanning. [8]

A Distributed Vulnerability Scanning on Machine Learning in the year 2019 by Xiaopeng TIAN, Di TANG, establishing standardized and quantified data sets for different industries and different businesses is of great help to improve the quality of testing. [7]

Commix: automating evaluation and exploitation of command injection vulnerabilities in Web applications published within the year 2019 by Anastasios Stasinopoulos, Christoforos Ntantogian and Christos Xenakis. It supports a plethora of functionalities that attempt to cover various exploitation scenarios such as different authentication mechanisms, custom headers, tornet working, attack vectors produced by programming languages, system user enumeration. [6]

Dimitris E. Simos, Jovan Zivanovic, Manuel Leithner proposed Automated Combinatorial Testing for Detecting SQL Vulnerabilities in Web Applications in the year 2019. It demonstrates that our approach can successfully evade faulty filtering mechanisms. [5]

III. WEB SECURITY VULNERABILITIES

A. SQL Injection

SQL injection attacks are one among the topmost threats in database-centric web applications and SQL injection vulnerabilities are the foremost serious Vulnerability types. SQL Injection allows the attacker to gain control over the database of an application. [7]

Every other website needs input from the user for a variety of reasons and if they are not validated properly, they might lead to some critical issues. Consider a login function where the user has to provide a username and password. These credentials are then validated at the backend through SQL query statements and if they are correct, then the user is successfully logged in.

Now let's consider a situation where it can be abused. If the user provides some malformed inputs and the application accepts it as it is, then the attacker can leverage this to perform a database attack. From Fig. 2 we can see that the attacker Alice is providing username as 'admin" ;–' and some arbitrary password.

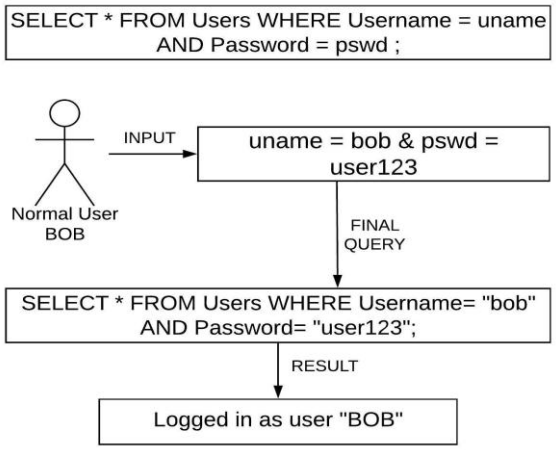

Fig. 1. SQL Injection - User's Perspective

This results in breaking of the structure of the SQL query used at the backend. So the effective query will be 'SELECT * FROM Users WHERE Username="admin";–" AND Password="random";'. This will to lead to a change in the application logic, as the double-quote entered in the username will match the starting double-quote of the query and as the '–' is considered as an identifier for comment in most of the relational databases, it simply comments out the succeeding part of the query. So the new query will be 'SELECT * FROM Users WHERE Username="admin";'. The attacker can now log in to an account without knowing the password.

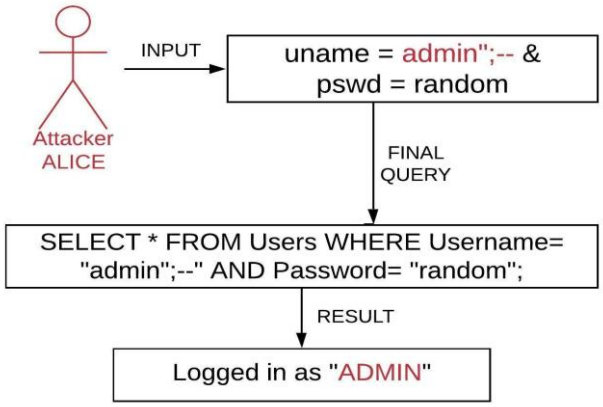

Fig. 2. SQL Injection - Attacker's Perspective

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 8, Issue 8, August 2021

B. Cross Site Scripting

The Cross Site Scripting attack is a critical vulnerability that affects web application's security. XSS attack is an injection of malicious script code into the web application by the attacker in the client-side within user's browser or in the serverside within the database, this malicious script is written in JavaScript code and injected within untrusted input data on the web application [8].

Many applications provide the facility to search for specific content. Whenever the user searches for the required content, the relevant results are displayed on the webpage along with a search keyword entered by the user.

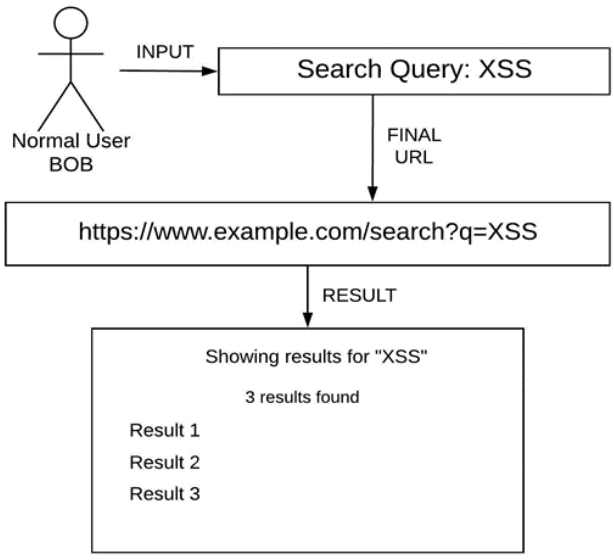

Fig. 3. XSS - User's Perspective

Now let us consider the attacker's perspective. The malicious user makes use of this search functionality as any of the normal user and checks whether the searched keyword gets reflected on the resultant page returned by the application. If it succeeds, then the attacker comes to know that there is a possibility of XSS to take place at that particular location.

If the application does not perform either encoding or filtering on the search query given by the user, then it might be possible for an attacker to break out of the previous HTML tag and insert a new one. From the figure given below, we can see that the attacker is able to insert a new script tag that can be used for malicious purposes.

Being able to insert a new script tag can have several consequences, including but not limited to, stealing session cookies, bypassing CSRF protections, theft of user's personal and sensitive data. In some extreme scenarios, it might be possible to exploit the user's browser by leveraging the XSS.

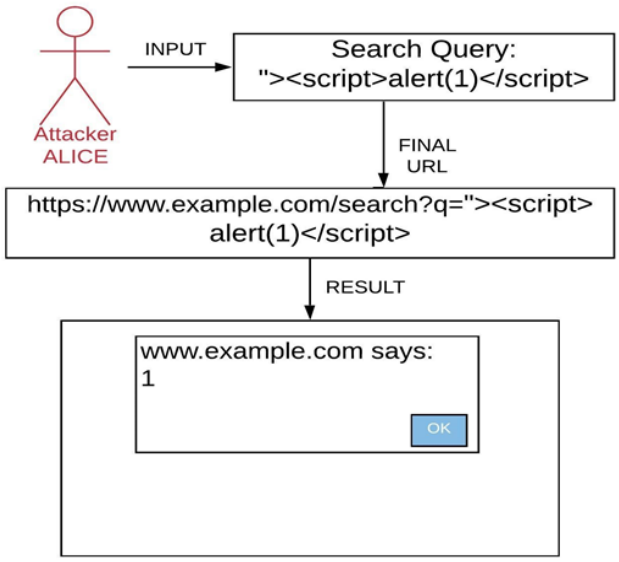

Fig. 4. XSS - Attacker's Perspective

IV. PROPOSED SYSTEM

A. System Architecture

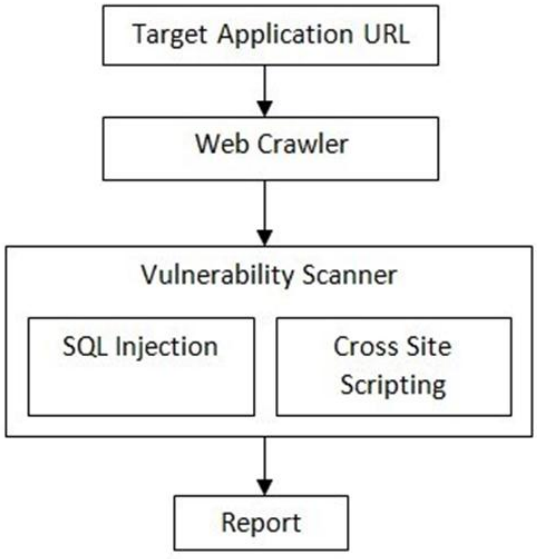

Fig. 5. System Architecture

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 8, Issue 8, August 2021

B. Project Scope

- If the web application is not having the robots.txt file then the user has to explicitly specify the restricted URLs.
- The system will scan the target application and check if the web application is having any of these vulnerabilities:
  - SQL Injection
  - Cross Site Scripting
- The report will be generated consisting of endpoint affected, payload used, and generalized remediation.

C. User classes and characteristics

• WebSpider:

– Robots.txt Parsing

∗ Checks for the presence of the robots.txt file and if present, collect allowed and disallowed URLs.

– URL Parsing:

∗ All URLs specified within the anchor tag from the current page are saved in a List.

∗ Multiple threads will be created to crawl different hyperlinks simultaneously.

∗ Relative URLs (like /admin or #footer) are converted into Absolute URL (like https://example.com/admin or https://example.com#footer)

∗ URLs which are not in the scope of target application are removed from the list (for example twitter.com or instagram.com)

∗ Hyperlinks with ‘mailto:’ or ‘javascript:’ and those pointing to static file types like images, pdfs, fonts, etc. are also removed.

- WebScanner:
  - Search for form elements in crawled URLs. From the listed form elements, find out input fields.
  - Pass the appropriate payloads to the input field and save the response received from the server.
- SQL Injection:
  - For an Error-Based SQL Injection attack, we try to break the syntax of the SQL query being used by the server by passing SQL-special characters (E.g.’,”, etc.) through user input.
  - For Union SQL injections, a dataset consisting of SQL queries of specific types will be created.
  - In the user input, these payloads will be passed to the server to check if SQL query is well formed after inserting the given payload.
  - The system requires the target URL to be entered by the user.
  - If the response from the server for the payload is similar to the usual response, then we can infer that SQL injection is possible.
- Cross Site Scripting:
  - For Cross Site Scripting, a dataset will be created consisting of different XSS payloads.
  - These payloads will then be tested against all input fields present on that web page.
  - The responses are checked for the presence of a particular payload and to check if XSS is successful.
  - If successful, that part of the web page is considered vulnerable and a report will be generated giving a detailed knowledge about the vulnerability detected.

V. OTHER SPECIFICATIONS

A. Advantages

- Supports automated and reliable crawling.
- Optimized use of the number of threads to control the load on the target application.
- Detailed vulnerability analysis.
- User-friendly GUI.

B. Limitations

- Model can currently handle non-CAPTCHA registrations and logins.
- Possible to detect first-order SQL Injection and XSS vulnerabilities through the way of automated scanning.
- Current focus is on small to medium-sized web applications.

C. Applications

- Identifying and reporting vulnerabilities present in a web application.

VI. CONCLUSION AND FUTURE SCOPE

We have tried to find some of the common vulnerabilities on the web, such as SQL Injection and Cross Site Scripting. We have proposed an algorithm and further enhancements in the system to improve the efficiency of the vulnerability detection in the web application. We proposed a system that will crawl the entire web application, scan different types of vulnerabilities, and generate a report specifying an overview of the detected vulnerabilities.

There is a scope of improvement in various aspects of the developed system. We can incorporate more vulnerabilities to further increase the scope of scanning. The time required for both crawling and scanning can be improved so that more complex applications can also be tested for vulnerabilities. Furthermore, options for the users to access their past scans and their results could be

provided.

Acknowledgment

We take this to express our deep sense of gratitude towards our esteemed guide Prof. Dr. H. K. Khanuja for giving us this splendid opportunity to select and present this preliminary report for the final year project and also providing facilities for successful completion. We thank Dr. H. K. Khanuja, Head, Department of Computer Engineering, for opening the doors of the department towards the realization of the final year project, all the staff members, for their indispensable support, priceless suggestion, and for most valuable time lent as and when required. With respect and gratitude, we would like to thank all the people, who have helped us directly or indirectly.

REFERENCES

[1] Stefano Calzavara, Mauro Conti, Riccardo Focardi, Alvise Rabitti, and Gabriele Tolomei. Machine learning for web vulnerability detection: The case of cross-site request forgery. IEEE Security & Privacy, 18(3):8–16, 2020.

[2] Hoang Viet Long, Tong Anh Tuan, David Taniar, Nguyen Van Can, Hoang Minh Hue, and Nguyen Thi Kim Son. An efficient algorithm and tool for detecting dangerous website vulnerabilities. International Journal of Web and Grid Services, 16(1):81–104, 2020.

[3] S. K. Mahmoud, M. Alfonse, M. I. Roushdy, and A. M. Salem. A comparative analysis of cross site scripting (xss) detecting and defensive techniques. In 2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS), pages 36–42, 2017.

[4] C. Sharma and S. C. Jain. Analysis and classification of sql injection vulnerabilities and attacks on web applications. In 2014 International Conference on Advances in Engineering Technology Research (ICAETR - 2014), pages 1–6, 2014.

[5] Dimitris E Simos, Jovan Zivanovic, and Manuel Leithner. Automated combinatorial testing for detecting sql vulnerabilities in web applications. In 2019 IEEE/ACM 14th International Workshop on Automation of Software Test (AST), pages 55–61. IEEE, 2019.

[6] Anastasios Stasinopoulos, Christoforos Ntantogian, and Christos Xenakis. Commix: automating evaluation and exploitation of command injection vulnerabilities in web applications. International Journal of Information Security, 18(1):49–72, 2019.

[7] TIAN Xiaopeng and TANG Di. A distributed vulnerability scanning on machine learning. In 2019 6th International Conference on Information Science and Control Engineering (ICISCE), pages 32–35. IEEE, 2019.

[8] Xun Zhang, Jinxiong Zhao, Fan Yang, Qin Zhang, Zhiru Li, Bo Gong, Yong Zhi, and Xuejun Zhang. An automated composite scanning tool with multiple vulnerabilities. In 2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), pages 1060–1064. IEEE, 2019.