

VULNERABILITY ASSESSMENT AND PENETRATION TESTING

Major project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering/Information Technology

By

KHUSHI SHAH (191516)

UNDER THE SUPERVISION OF

Dr. Pankaj Dhiman

to



Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Wagnaghat, Solan-173234, Himachal Pradesh

CERTIFICATE

This is to certify that the work which is being presented in the internship report titled “**Vulnerability Assessment & Penetration Testing**” in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science & Engineering** and submitted to the Department of Computer Science & Engineering, Jaypee University of Information Technology, Wagnaghat is an authentic record of work carried out by “Khushi Shah(191516) during the period; February 2023 – till date, ensuring proper care towards the rules and regulations as specified by the Non-Disclosure Agreement signed between Khushi Shah and AppSecure.

Khushi Shah (191516)

The above statement made is correct to the best of our knowledge.

Sandeep Hodkasia

CEO & Co-Founder

AppSecure

Dr. Pankaj Dhiman

Assistant Professor

(Senior Grade)

Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Wagnaghat

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT
PLAGIARISM VERIFICATION REPORT

Date:

Type of Document (Tick): PhD Thesis M.Tech Dissertation/ Report B.Tech Project Report Paper

Name: _____ Department: _____ Enrolment No _____

Contact No. _____ E-mail. _____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor)

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	<ul style="list-style-type: none"> • All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String 		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature

Librarian

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

ACKNOWLEDGEMENT

First, I express my gratitude to God who provided me with the courage and fortitude to complete this internship. I would also like to extend my heartfelt gratitude to my college, JUIT, for giving this internship opportunity. My sincere gratitude to our Training and Placement officer, Mr. Pankaj Kumar, and our faculty Coordinator, Dr. Nafis U Khan, for this opportunity. I also wish to express my gratitude to my internship supervisors for their valuable guidance and advice towards my internship.

I am grateful and wish my profound indebtedness to my trainer, and my colleagues in AppSecure without whose kind assistance. My internship would not have been a smooth ride for me.

I would also like to express my gratitude towards my friends and family members, whose immense support has always guided me towards the right direction.

Khushi Shah

191516

TABLE OF CONTENTS

Content	Page no.
Certificate	i
Plagiarism Certificate	ii
Acknowledgment	iv
Table of Figures	v
Abstract	vi
Chapter-1 INTRODUCTION	1-11
Chapter-2 TOOLS AND TECHNOLOGIES	12-30
Chapter-3 SYSTEM DEVELOPMENT	31-42
Chapter-4 PERFORMANCE ANALYSIS	43-47
Chapter-5 CONCLUSION	48
REFERENCES	49

LIST OF FIGURES

Figure No.	Description
1	Process of VAPT steps
2	Steps of VAPT
3	Steps in Assessment
4	Vulnerability scanning
5	Attack Surface
6	Proxy
7	Spider
8	Scanner
9	Intruder
10	Repeater
11	Sequencer
12	Decoder
13	Comparer
14	Extender
15	Nikto
16	SQLmap-Dev
17	Directory Buster
18	Directory Buster
19-20	Waterfall Model
21	Authentication bypass via NoSQL Injection
22	Admin Credentials
23-27	Source Code Disclosure
27-32	Source Code Disclosure

LIST OF ABBREVIATIONS

VAPT	Vulnerable Assessment & Penetration Testing
CVSS	Common Vulnerability Scoring System
XSS	Cross-Site Scripting
OWASP	Open Web Application Services Project
XXE	XML External Entities
ISO	International Organization for Standardization
ACL	Access Control List
CSRF	Cross-site Request Forgery
URL	Uniform Resource Locator
APT	Advanced Persistent Threat

ABSTRACT

Vulnerability Assessment and Penetration Testing (VAPT) is a crucial process to identify and address potential risks in computer systems and networks for any organization and to help maintain its security posture. Evaluating the degree of risk and potential effects of security vulnerabilities in a system or network is part of this systematic and thorough approach. Finding vulnerabilities before they can be used maliciously to undermine the confidentiality, integrity, and availability of data and systems is the most crucial and vital goal of VAPT.

The vulnerability assessment and penetration testing phases make up the VAPT process. Security experts employ a range of instruments and methods during the first stage, known as vulnerability assessment, to find weaknesses in the system or network. This entails running scans, going over logs, and examining the settings of the system. Once vulnerabilities are identified, they are prioritized based on the level of risk they pose to the organization.

The second stage after Vulnerability assessment is penetration testing, security professionals attempt to exploit the vulnerabilities identified in the previous stage to determine the level of risk and potential impact. The objective of this step is to simulate a real-world attack scenario and identify any loopholes or weaknesses in the system's defense mechanisms that may help the attacker enter the organization. Penetration testing provides valuable insights into the effectiveness of the organization's security controls and helps to identify areas where improvements can be made.

VAPT has a lot of benefits when it comes to using it on a larger scale. In the first place, it helps organizations identify vulnerabilities in their network or applications before they can be exploited by malicious actors, resulting in a reduction of the risk of data breaches, theft, and other security incidents. Secondly, it helps businesses ensure compliance with working industry standards and the regulations, such as the Payment Card Industry Data Security Standard (PCI DSS) and the General Data Protection Regulation (GDPR). Apart from the above mentioned, VAPT also helps organizations improve their overall security posture and gain a better understanding of their security risks and vulnerabilities.

Chapter 01: INTRODUCTION

1.1 INTRODUCTION

Vulnerability Assessment and Penetration Testing (VAPT) are critical processes for security engineers to identify and prioritize security issues in a system or network. After conducting VAPT, security engineers can list the identified security issues based on the Common Vulnerability Scoring System (CVSS) score [1] for further risk analysis and deployment of fixes. The security engineer's primary role in VAPT is to handle security issues discovered within the organization and externally by security researchers with caution.

In recent years, security has become increasingly important for companies, and most undergo security testing of their domains and applications. The responsibility for carrying out VAPT falls on security engineers. In this process, they must pay special attention to the Open Web Application Security Project's (OWASP) top 10 security vulnerabilities, which are commonly targeted by attackers.

By using VAPT, security engineers can identify vulnerabilities and address them before they can be exploited by malicious actors. This helps to ensure the confidentiality, integrity, and availability of sensitive data and systems. Additionally, VAPT can help organizations meet regulatory compliance requirements and maintain their reputation by avoiding security breaches and data theft.

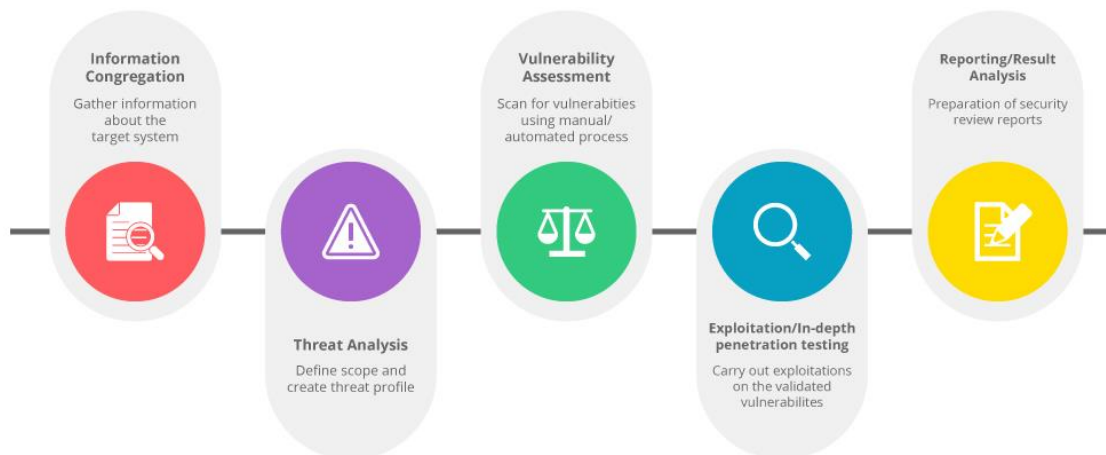


Fig 1 – Process of VAPT [2]

1.2 PROBLEM STATEMENT

The project revolves around identifying potential security weaknesses and vulnerabilities within a system, application, or network that could be exploited by attackers. The goal of the project is to assess the security posture of the target system and determine the level of that it poses to the organization.

A VAPT project is typically initiated when an organization wants to ensure that its systems are secure and free from vulnerabilities.

1.3 OBJECTIVES

The primary objective of a VAPT (Vulnerability Assessment and Penetration Testing) project is to identify and address potential vulnerabilities in a system, application, or network, and assess its security posture. The project aims to proactively identify security weaknesses and mitigate them before they can be exploited by attackers.

Then the company fixes these issues and asks for the revalidation of the issues whether the issue is fixed or not. Some important objectives also behind the VAPT project are:

- **Identifying security weaknesses:** The project aims to identify any vulnerabilities that could be exploited by attackers, including known vulnerabilities, configuration errors, and coding mistakes.
- **Assessing security posture:** The project assesses the overall security posture of the system, application, or network, which includes evaluating its ability to withstand attacks and mitigate potential threats.
- **Mitigating risks:** Once vulnerabilities are identified, the project helps to prioritize them based on their severity and provides recommendations for mitigation to reduce risks.
- **Meeting compliance requirements:** Many organizations are required to comply with industry regulations or government mandates that require periodic security assessments. A VAPT project can help organizations meet these compliance requirements.

1.4 METHODOLOGY

The project's methodology involves understanding how to identify security vulnerabilities in computer systems, and to evaluate them using different tools and techniques. Some of the main steps involved while working on this are:

1. Planning: understanding the scope and objectives of VAPT, which will also include a complete understanding of systems and the applications that are to be tested.
2. Reconnaissance: This involves gathering information about the target, that includes the network architecture, OS, and any possible vulnerability[3].
3. Scanning and Analysis: Automated tools to scan the software and analyze the vulnerabilities and determine their severity.
4. Penetration testing: Exploitation of the identified vulnerabilities using techniques like password cracking, social engineering, etc., to gain unauthorized access to the system/application.
5. Reporting: Documentation of the identified vulnerabilities, their level of severity, their impact on the system, and what remediation can be suggested for the future.
6. Verification: The last step involves the verification that the found vulnerabilities are remediated or not and all the important security measures have been taken.

1.5 ORGANIZATION

This project report is divided into five chapters which are as follows: -

Chapter 1: - This chapter gives a brief introduction of the project. Gives a brief overview of how VAPT works. The chapter also talks about the problem statement of the whole project and the objectives of the project. The chapter also provides a brief introduction to the methodology used for the project and also provides information about the steps involved in a VAPT process.

Chapter 2: - This chapter gives knowledge about the previous work related to VAPT and security. The literature survey chapter includes a broad and complete analysis of existing research that has been done, studies, and publications that are related to vulnerability assessment and penetration testing. The literature survey chapter helps provide a complete understanding of the current state of VAPT, how important it is in the context of cybersecurity, and the different methods, and tools used in the process.

Chapter 3: - This chapter consists of information about the steps and the methodologies that are I have followed to work on the whole project. This chapter also talks about system development and a complete description of the project. Chapter 3 will also provide a detailed analysis of the different tools and technologies of the Assessment. The system development also includes vulnerabilities that were found, their severity level, and recommendations for remediation.

Chapter 4: - The performance analysis in the following chapter provides a thorough assessment of how well the project met its goals.. This chapter will include a descriptive analysis of the vulnerability assessment. The performance analysis aims to calculate how much has been achieved in identifying and addressing vulnerabilities and the overall security posture of the system, application, or network being tested. With respect to VAPT, it is important to keep track of valuable insights into the effectiveness of the testing process and helps organizations make better security decisions on how to improve their cybersecurity measures.

Chapter 5: - The last chapter is all about the conclusion and future work presented in this project report. It provides information about how each step works and generates appropriate results. This chapter also summarize the found vulnerabilities, the end discussion of how the vulnerabilities effect a system.

It also includes the future scope, which discusses about the current situation of VAPT in industry, and its need in the coming future.

Chapter 02: LITERATURE SURVEY

2.1 LITERATURE SURVEY

For any security engineer, Vulnerability Assessment and Penetration testing is a way to find all the security issues in a particular application or system and list them according to the CVSS score for the risk analysis. In this section, I have provided some literature works from cybersecurity researchers and professionals, whom I took help with while working on the project.

1. **Jai Narayan Goel, BM Mehtre[4],**

As we know the complexity of systems keeps on increasing rapidly with time, this also raises the possibility of an increase in the potential for vulnerabilities as well, which can be easily exploited by attackers to hack the system. This can be avoided by identifying and addressing these vulnerabilities before attackers can exploit them. In this case, Vulnerability Assessment and Penetration Testing (VAPT) counts as a powerful cyber defense technology that offers a proactive cyber defense. The research examines how VAPT can be used for proactive cyber defense and details the entire lifecycle of doing VAPT on networks or systems, including taking the initiative to address vulnerabilities that have been found and avert possible data breaches. The paper also examines some prevalent VAPT techniques and gives an overview of some popular and useful open-source VAPT tools.

This paper also discusses the importance of VAPT as a Cyber Defence Technology. By properly removing system vulnerabilities, VAPT can be used as a cyber defense technology, lowering the chance of cyberattacks. The paper explains several Vulnerability & Penetration testing approaches, and provide a comprehensive VAPT life cycle for active defense.

The detailed tutorial in this paper helps in reducing risks and cyber-attacks. The paper explains various methods of conducting Vulnerability Assessment and Penetration Testing and provides a comprehensive life cycle of VAPT for proactive defense.

2. Shaimaa Khalifa Mahmoud, Marco Alfonse, Mohamed Ismail Roushdy, Abdel-Badeeh M. Salem[5]

For we know how important web applications are to us in day-to-day life, they are also powerful in e-commerce, the medical industry, and so much more. These web applications can contain vulnerabilities that can be exploited by a third-party attacker. Among all these critical vulnerabilities, Cross Site Scripting (XSS) attacks.

In an XSS attack, the hacker will inject a corrupted, malicious code into the web application, on any of the two sides, client, within the browser, or server within the database. XSS attack starts with the injection of malicious code in the web page. When any user opens the website, the malicious code will get exploited and the credentials of the user can be extracted such as credit card numbers or passwords.

The other XSS is the Stored XSS which is much more damaging than the other one. It is implemented every time the user will open the affected page. Unlike the reflected XSS, which gets executed when the user gets involved with any input field, this type affects all those who even view the website. The author has covered all the statistics of how XSS vulnerabilities take place, making it the third in the OWASP top 10.

When it comes to detection and prevention the authors have also referred to Shahriar and Zulkernine model S2XS2, which is capable of detecting an XSS attack automatically from the server side. Next, they talked about the model proposed by Haujie Xu that was developed on the basis of the browsing data of a user and website logic structure and then load to an XML file to parse logical structure. The next step is recording the navigation. The model will display a warning message when navigation is deemed illegitimate.

The paper includes a total of 9 studies done to explore web applications and XSS tracks happening on them. It concluded with the fact that as we know XSS-Stored is more dangerous and needs more study and research for enhancement.

3. Gitanjali Simran T, Saskala D [6].

There has been a noteworthy increase in cyberattacks and exploitation of data and applications. This has eventually led to organizations taking indispensable steps to secure themselves. In this paper the authors aim to work on the overview of VAPT and introduce some of the most efficient and important open-source tools that are most commonly used for testing, and also a complete testing methodology of the VAPT process. Vulnerability Assessment & Penetration Testing has two different forms of

testing which helps in the evaluation of the security posture of any organization or business. While the main motive behind both of them are almost similar, i.e. identification of loopholes and vulnerabilities, they still work on different methods and follow different steps to achieve this goal and also differ in the types of results.

In the first place Vulnerability Assessment involves the part where we need to scan the systems, applications, and networks to identify the vulnerabilities, any misconfigurations, and other fragility. The main aim of this testing is to identify as many vulnerabilities as possible, nonetheless of their severity, and to provide a justified prioritized list of issues that need remediation.

In the second place, we have Penetration Testing. This one involves the simulation of an actual attack on the organization's system, using methods and techniques that any real black hat hacker might use to gain access. In this, the goal is to exploit vulnerabilities that can be used to gain unauthorized access to any system, steal the information, or maybe cause any other type of damage. The outcome of this provides a detailed description of the organization's security posture and the needful actions they need to take, in specific areas.

While both of these have their pros, they are often used and combined together for a more comprehensive result. In the end, the author ultimately defines that a well-designed VAPT program can provide a more complete understanding of an organization's security posture and help identify and mitigate risks to its systems and data.

4. Ahmad Almaarif, Muharman Lubis[7]

This research paper By Ahmad and Muharman makes use of proper engagement to conduct a Vulnerability Assessment and Penetration testing for a government website. The study begins with a complete description of what VAPT is and how important it is for ensuring the security of any organization to protect themselves from facing a security breach. The paper also describes how Vulnerability Assessment and Penetration testing takes place, how each step proceeds and how every step is connected to the previous step. The initial phase determines the scope of the process, the testing phase, and the finalizing of the report phase. The assessment cannot take place without the use of proper and appropriate tools. The paper also consists of detailed study of the tools used. The first being Nmap. It is used to find a web server. Fiddler for the web debugging

proxy. Firefox extensions for inline editing, making changes in the JavaScript and much more.

So basically there can be two ways to perform a penetration test: internally and externally.

There is a function in penetration test for external networks that shows the known vulnerabilities and when exploited by the hackers may appear outside of the boundaries of the network.

SQL is a very important part of the RDBMS and useful to operate the website because of the simplicity it carries. But the SQL Injection attack can occur when the attacker injects any new parameters in the place of the input values that should have been entered. This helps the attacker enter the system and access the sensitive data.

The author also discusses how XSS works. In Cross-Site Scripting (XSS) attacks the attackers embed malicious links into the JavaScript code, VBScript, Flash or even HTML, to access the sensitive information and gain root access.

While this study was being conducted, the authors also covered Full Path Disclosure. This allows the hacker to gain information about the web path. In this scenario, Protecting the information of users is of utmost importance for the companies. Therefore its security should be the top priority. The developed framework in this paper helps in reducing the cost incurred, while also providing the proper security measures for the application.

Chapter 03: IMPLEMENTATION

3.1 OVERVIEW

3.1.1 What is OWASP top 10?

The OWASP top 10 is basically a list of the top 10 most critical vulnerabilities found in web applications, identified by the Open Web Application Security Project(OWASP), which is a nonprofit organization that helps in the improvisation of the security of software.

This list by OWASP is updated every few years to reflect the changing nature of web application security. The list is also a way to provide a framework for developers, security professionals, and organizations to understand the most common and most dangerous web application security risks that they need to protect against and make their organization more secure.

Understanding and addressing the vulnerabilities listed in the OWASP Top 10, it helps developers and organizations to build more secure web applications and reduce the risk of data breaches, unauthorized access, and other security incidents. The top 10 list acts as a standard for implementing web application security best practices and guarantees that security is always given priority throughout the development and deployment process. It is a valuable resource for everyone who works on building or security of online applications and is most frequently used by security experts and organisations as a tool for evaluating the security of web applications.

3.1.2 Description of Top 10 Vulnerabilities

The Top 10 list of vulnerabilities include:

1. Injection
2. Broken Authentication
3. Sensitive data exposure
4. XML External Entities (XXE)
5. Broken Access Control
6. Security Misconfigurations
7. Cross-Site Scripting (XSS)

8. Insecure Deserialization
9. Using components with Known vulnerabilities
10. Insufficient Logging and Monitoring

Hash: Injection vulnerability is a type of web application security flaw that allows the attackers to insert and execute malicious code into an application, or gain access to sensitive data. Injection attacks occur when an attacker can input data into an application, such as through a form or search bar, or any text field, which asks the user to enter some text, and that input is not properly validated or sanitized by the application.

One of the most common examples of an injection attack is SQL injection, which involves the insertion of malicious SQL commands into an application's input fields so that when the user accesses the application, the injection exploits help the attacker to manipulate or retrieve sensitive data stored in a database. It will be a successful attack if the attacker is able to gain access to the username, passwords, and other sensitive data. Apart from SQL injection we also have LDAP injection, XPath injection, and OS command injection. These include inserting malicious scripts into other parts of the application where it has input fields and text boxes so as to gain unauthorized access or execute arbitrary code on the target system. To prevent data loss and maintain a proper security posture, the development team should keep track of the user's input properly in the input field. Their input should be sanitized and validated well before it is processed by the application. This step will involve filtering of input and the encoding, parameterized queries for database access.

Additionally, organizations should implement secure coding practices and regularly test their web applications for vulnerabilities to ensure that they are not susceptible to injection attacks.

You can see one of OWASP's examples below:

```
String query = "SELECT * FROM accounts WHEREcustID = " + request.getParameter("id") + "";
```

This query can be exploited by calling up the web page executing it with the following URL: <http://example.com/app/accountView?id=' or '1'='1>, causing the return of all the rows stored on the database table. The core of a code injection vulnerability is the lack of validation and sanitization of the data consumed by the web application, which means that this vulnerability can be present on almost any type of technology.

Broken Authentication: Broken Authentication is a security vulnerability that takes place when an application's authentication mechanism is poorly design or implemented, that allows the attackers to bypass authentication controls and gain unauthorized access to the sensitive information within an application.

Authentication is the process of verifying the identity of the user or the system, mostly using the credentials of the user such as username and password. Broken Authentication vulnerabilities emerge when the application fails to adequately protect user authentication credentials or lacks the appropriate session management controls, such as the use of session timeouts or secure cookie settings.

Attackers can exploit broken authentication vulnerabilities in a number of ways, including:

- Brute force attacks: Attackers can try to guess usernames and passwords or use automated tools to repeatedly guess credentials until they gain access.
- Credential stuffing: Attackers can use stolen credentials from other sources to try to gain access to an application.
- Session hijacking: Attackers can intercept or steal session tokens to gain unauthorized access to an authenticated session.
- Password cracking: Attackers can easily gain access to an application using particular tools to crack passwords and gain access to an application.

To avoid such situations, developers should use secure authentication techniques, such as strong password limitations multi-factor authentication, and secure session management controls, to prevent broken authentication vulnerabilities. In addition, organizations should frequently inspect their applications for flaws, including checking for weak or default credentials, checking session management safeguards, and keeping an eye out for unusual user behaviour. Organizations can help make sure that their apps are safeguarded from the dangers posed by compromised authentication vulnerabilities by adopting the following actions.

To avoid broken authentication, don't leave the login page for admins publicly accessible to all visitors of the website:

/administrator on Joomla!,

/wp-admin/ on WordPress,

**/index.php/admin on Magento,
/user/login on Drupal.**

Sensitive Data Exposure: Sensitive data exposure is a security flaw that happens when a system or application is not sufficiently secure to prevent the access or disclosure of sensitive data, such as personal or financial information. This flaw might provide hackers access to private information that they could then readily use for malicious purposes.

Sensitive Data Exposure can take place in a number of ways, that includes:

- **Insecure Data:** When Attackers can take advantage of sensitive data that is stored insecurely or that is improperly encrypted.
- **Insufficient data masking:** When an Attacker can access the data when it is displayed or transmitted in plain text or when it is improperly masked.
- **Lack of access controls:** When there aren't enough access controls in place, hackers can access private data by taking advantage of system flaws or weak spots.
- **SQL injection attacks:** when an attacker can access private information kept in a database by inserting SQL code into an application.
- **Misconfigured systems:** When Systems or programs that are improperly configured run the risk of exposing private information to unauthorised users.

Developers and businesses should use safe methods of coding including encryption of data, access controls, and appropriate data masking to prevent vulnerabilities that reveal sensitive data. Additionally, it's important to regularly inspect applications and systems for flaws, including checking for SQL injection attacks and other typical attack routes. To make sure that systems and applications are in accordance with the relevant security standards and regulations, organizations should routinely examine and audit them.

Protecting Data in Transit

Both kinds of data ought to be shielded. When discussing data in transit, having an SSL certificate is one technique to safeguard information on a website. The TLS protocol, which creates an encrypted link between a web server and a browser, is sometimes referred to as SSL but is now deprecated.

XML External Entities (XXE)

XXE (XML External Entity) is a kind of vulnerability that act on applications that parse XML

input. This vulnerability usually takes place when an attacker is able to inject malicious XML code into an application's processing.

XML input can be modified to incorporate external entities, which are files or network resources that are mentioned within the XML document, in XXE attacks. An attacker may be able to access personal data, run remote code, and carry out other nefarious deeds by taking advantage of this vulnerability.

To steal sensitive data, such as user credentials, credit card details, or other private information, an attacker may, for instance, execute an XXE attack. On the targeted system, they might also employ XXE to launch denial-of-service attacks or run arbitrary code.

Developers should employ safe coding practices and input validation procedures to make sure that all user-supplied data is appropriately sanitized before processing in order to prevent XXE attacks.

Additionally, using a secure XML parser library that is designed to mitigate XXE attacks can provide an additional layer of protection against this type of vulnerability.

What Are the Attack Vectors?

According to OWASP, the XML external entities (XXE) main attack vectors are:

- Exploitation of vulnerable XML processors if malicious actors can upload XML or include hostile content in an XML document;
- Exploitation of vulnerable code;
- Exploitation of vulnerable dependencies;
- Exploitation of vulnerable integrations

Example of an XML External Entity Attack

According to OWASP, the easiest way to exploit an XXE is to upload a malicious XML file.

Scenario #1: The attacker attempts to extract data from the server

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
<!DOCTYPE foo
```

```
[<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<foo>&xxe;</foo>
```

Scenario #2: An attacker probes the server's private network by changing the above ENTITY line to:

```
<!ENTITY xxe SYSTEM "https://192.168.1.1/private" >]>
```

Scenario #3: An attacker attempts a denial-of-service attack by including a potentially endless file:

```
<!ENTITY xxe SYSTEM "file:///dev/random" >]>
```

Broken Access Control: Broken Access Control is a type of security vulnerability that occurs when an application does not properly carry out the restrictions on what actions a user is authorized to perform. In other words, it's when an attacker gains access to manipulate sensitive data or to perform unauthorized actions due to weak access controls within an application.

Access controls are created to make sure that users can only access the data and functionality that they are authorized to access. When access controls are broken, it's possible for an attacker to easily gain access to sensitive information, modify or delete it, or perform other unauthorized actions.

For example, let's say a web application allows users to view their own profile information by accessing a URL like "www.example.com/profiles/user123". However, if an attacker changes the URL to "www.example.com/profiles/user456", they may be able to access the profile information of another user. This is a classic example of a Broken Access Control vulnerability.

To prevent Broken Access Control vulnerabilities, developers should implement proper access controls within their applications, including authentication and authorization mechanisms. This means ensuring that users are properly authenticated before allowing them to access sensitive data or functionality, and that they are authorized to perform the specific actions they are attempting to perform. It's also important to test applications for access control vulnerabilities during development and regularly throughout the application's lifecycle.

Examples of Access

- Access to a hosting control / administrative panel.
- Access to a server via FTP / SFTP / SSH
- Access to a website's administrative panel
- Access to other applications on your server
- Access to a database

These are ways attackers can exploit authorization flaws:

- Access unauthorized functionality and/or data;
- View sensitive files;
- Modify other users' data;
- Change access rights, etc.

Security Misconfiguration

When a system or application is not configured properly, Security Misconfiguration can take place, which can leave it open for the attackers to exploit. It is one of the most common security issues found in systems and applications and can result in very serious consequences, such as unauthorized access, data breaches, and system compromise.

A security misconfiguration can occur due to many reasons, such as using default passwords, unpatched software or operating systems, open ports, unnecessary services running, or lack of proper access controls. These misconfigurations can be exploited by attackers to gain unauthorized access to sensitive data, inject malware, or take control of the system.

To reduce security misconfiguration vulnerabilities, it is essential to follow best practices for system and application configuration, this will include regularly updating software and operating systems, disabling unnecessary services, using strong passwords, implementing access controls, and limiting user privileges. Apart from this, conducting regular vulnerability assessments and penetration testing are a great help in identifying and remediating any misconfigurations that may occur in a system or application.

Where can Security Misconfiguration Happen?

Any level of the application stack can experience a security configuration error which can also

include:

- Network services
- Platform,
- Web server,
- Application server,
- Database,
- Frameworks,
- Custom code,
- Pre-installed virtual machines,
- Containers,
- Storage.

Cross-Site Scripting (XSS)

XSS is a kind of vulnerability, that enables an attacker to insert malicious code into web pages that other users can view. By doing this, they can steal sensitive data, like login credentials, cookies, or maybe even the installation of malware on the target's system.

XSS attacks occur in various forms, but the reflected XSS attack is the most prevalent. In this type of attack, the victim receives a unique link, that can contain dangerous software. Once the victim clicks on the corrupted link, the code gets executed by the target's browser and can proceed to steal sensitive information or perform unauthorized actions, for example sending phishing mails.

Another type of XSS attack is the "stored" XSS attack, in which the malicious code is stored on the server and executed every time the page is viewed by any user. This can lead to a widespread compromise of the system and its users.

To prevent XSS attacks, it is essential to sanitize user input and encode any special characters to prevent them from being interpreted as code. Web developers can use security libraries and frameworks to automatically sanitize inputs, and web application firewalls can be used to detect and block XSS attacks. Additionally, users can protect themselves by using browser extensions that block XSS attacks and avoid clicking on suspicious links or downloading unknown files.

Scenario #1:

The application server comes with sample applications that are not removed from the production server. These sample applications have known security flaws attackers use to compromise the server. If one of these applications is the admin console and default accounts weren't changed, the attacker logs in with default passwords and takes over.

Scenario #2:

Directory listing is not disabled on the server. An attacker discovers they can simply list directories. They find and download the compiled Java classes, which they decompile and reverse engineer to view the code. The attacker then finds a serious access control flaw in the application.

Scenario #3:

The application server's configuration allows detailed error messages, e.g. stack traces, to be returned to users. This potentially exposes sensitive information or underlying flaws, such as component versions. They are known to be vulnerable.

Scenario #4:

A cloud service provider has default sharing permissions open to the Internet by other CSP users. This allows stored sensitive data to be accessed within cloud storage.

Types of XSS

According to OWASP, there are three types of XSS:

XSS Type	Server	Client
Stored	Stored Server	Stored Client
Reflected	Reflected Server	Reflected Client
DOM - Based		ClientSubset of Client

Insecure Deserialization

Insecure Deserialization is a process that involves transforming an object's state into a sequence of bytes that can be transmitted or stored over a network. Deserialization is the reverse process of reconstructing an object from its serialized form.

Insecure deserialization can allow attackers to manipulate the data being deserialized and execute arbitrary code on the server or client side. This vulnerability can result in a range of attacks, including remote code execution, privilege escalation, and denial of service.

Attackers can exploit insecure deserialization by tampering with serialized data in transit or by sending maliciously crafted serialized data to a vulnerable application. This can result in the execution of unintended code or the creation of unintended objects, leading to a variety of security issues.

To prevent the system from insecure deserialization vulnerability, it is important to validate and sanitize all the insecure and untrusted data before we deserialize it. Also, developers should only be focusing on deserializing data from trusted sources and ensure that they are using secure serialization frameworks that provide safeguards against common deserialization attacks.

Some examples of Attack Scenarios:

According to OWASP, here are some examples of attack scenarios:

Scenario #1: A React application calls a set of Spring Boot microservices. Being functional programmers, they tried to ensure that their code is immutable. The solution they came up with is serializing the user state and passing it back and forth with each request. An attacker notices the "R00" Java object signature and uses the Java Serial Killer tool to gain remote code execution on the application server.

Scenario #2: A PHP forum uses PHP object serialization to save a "super" cookie, containing the user's user ID, role, password hash, and other states:

```
a:4:{i:0;i:132;i:1;s:7:"Mallory";i:2;s:4:"user";  
i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960"};
```

An attacker changes the serialized object to give themselves admin privileges:

```
a:4:{i:0;i:1;i:1;s:5:"Alice";i:2;s:5:"admin";  
I:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960"};
```

One of the attack vectors presented by OWASP regarding this security risk was a super cookie containing serialized information about the logged-in user. The role of the user was specified in this cookie.

If an attacker is able to deserialize an object successfully, then modify the object to give himself an admin role, they can serialize it again. This set of actions could compromise the whole web application.

- Using Components with unknown vulnerabilities

Using components with known vulnerabilities is an OWASP Top 10 security risk that refers to the practice of using third-party libraries, frameworks, or other software components that have known security vulnerabilities. These vulnerabilities can allow an attacker to exploit the component and gain unauthorized access to the system, steal sensitive data, or execute malicious code.

The use of third-party components is common in software development, as it helps developers save time and effort by not having to create every component from scratch. However, using these components can also introduce security risks into the application, as the vulnerabilities in the component can be exploited by attackers.

To reduce the risk of using elements with known vulnerabilities, developers should always keep their software components updated to date by applying security patches and updates as soon as they become available. It's also important to monitor the security of the components used in an application, especially if they are open-source, and to have a process in place for quickly identifying and addressing any vulnerabilities that may arise. Additionally, developers should carefully vet and review any third-party components before integrating them into their applications to ensure that they are secure and reliable.

Vulnerable Applications

According to OWASP Vulnerable applications are usually if:

- You do not know the versions of all components you use (both client-side and server-side). This includes all the nested dependencies as well as components you directly use.
- The software is vulnerable, unsupported, or maybe out of date. This will include the OS, web/application server, database management system (DBMS), applications, APIs and all components, runtime environments, and libraries.
- There is no fixing or upgrading of the underlying platform, frameworks, and dependencies in a risk-based, timely fashion. This commonly takes place in environments where patching is a monthly or quarterly task under change control, which leaves organizations open to many days or months of unnecessary exposure to fixed vulnerabilities.
- The software developers do not test the compatibility of the libraries that are updated, upgraded, or patched.
- You do not secure the components' configurations.

Insufficient Logging and Monitoring

Insufficient logging and monitoring is the type of vulnerability that is related to the lack of proper event logging and monitoring mechanisms in a system. When any attacker gains access to any system, they may try to cover their tracks by deleting logs or modifying system events to evade detection. Inadequate logging and monitoring can make it tough for the system administrators to detect these malicious activities, and respond appropriately.

It is quite a challenging task to analyze user activity and determine the underlying cause of a security incident without effective monitoring methods. Organizations should develop appropriate logging procedures including recording all important events and keeping them in a secure location, to reduce the risks related to poor logging and monitoring. Additionally, they should make sure that the logs are checked at timely intervals and that safety measures are in place to alert the system administrators of any unusual activity.

Insufficient logging and monitoring can also lead to a delay in incident response and make it difficult to conduct forensic analysis in case of a security breach.

In addition, conducting regular security audits and penetration testing can help identify vulnerabilities and ensure that proper logging and monitoring mechanisms are in place.

Examples of Attack Scenarios

According to OWASP, here are some examples of attack scenarios due to insufficient logging and monitoring:

Scenario #1: An open-source project forum software run by a small team was hacked using a flaw in its software. The attackers managed to wipe out the internal source code repository containing the next version and all of the forum contents. Although the source could be recovered, the lack of monitoring, logging, or alerting led to a far worse breach. The forum software project is no longer active as a result of this issue.

Scenario #2: An attacker scans for users with a common password. They can take over all accounts with this password. For all other users, this scan leaves only one false login behind. After some days, this may be repeated with a different password.

Scenario #3: A major U.S. retailer reportedly had an internal malware analysis sandbox analyzing attachments. The sandbox software had detected potentially unwanted software, but no one responded to this detection. The sandbox had been producing warnings for some time before detecting the breach due to fraudulent card transactions by an external bank.

3.2 ANALYSIS

With the increasing adaptation to technology, especially in the form of IoT devices, networks are becoming more vulnerable to security threats. Hence, Vulnerability Assessment and Penetration Testing (VAPT) plays a very important role in ensuring the security of networks. By conducting VAPT, organizations can easily identify and address various types of vulnerabilities in their applications and networks. There are multiple sectors today that invest heavily in improving their security systems, and VAPT services are a significantly dependable measure to protect networks from hackers and cybercriminals.



Fig 2: Steps involved in VAPT. [8]

<https://www.guru99.com/vulnerability-assessment-testing-analysis.html>

We will go through the detailed analysis of the project in this section:

3.2.1 Goals and Objectives

The purpose of VAPT (Vulnerability Assessment and Penetration Testing) is to identify weak points in the organization's systems and networks and to test the effectiveness of the organization's information security control. The ultimate goal of VAPT is to improve an organization's overall security by identifying and fixing vulnerabilities that attackers can exploit to gain unauthorized access, steal sensitive data, or disrupt operations.

Specifically, the goals of a VAPT typically include:

- Identifying vulnerabilities: The first and primary goal of a VAPT is to detect vulnerabilities in an organization's systems and networks. This includes the vulnerabilities in operating systems, software applications, network devices, and other components that can be exploited by attackers and they can gain access to the organization's network or data.
- Testing security controls: The vulnerability assessment and penetration testing can be applied to examine the effectiveness of an organization's security control, including intrusion detection systems, firewalls, and antivirus software. In order to determine whether the measures taken while performing VAPT are correct and sufficient enough to protect against potential threats.

- Evaluation of incident response procedures: A VAPT can also be used for the evaluation of an organization's incident response procedures. This involves testing on the steps that how well the organization is able to detect, contain, and respond to security incidents.
- Measuring compliance with regulations: A VAPT can also help organizations measure their compliance with relevant regulations, such as HIPAA or PCI-DSS. This involves identifying areas of non-compliance and developing recommendations for addressing them.
- Prioritizing remediation efforts: Finally, prioritizing the remediation efforts of any organization is the main goal of a VAPT. This happens by identifying the most critical vulnerabilities that need to be addressed first. This involves developing a remediation plan that focuses on the vulnerabilities that pose the greatest risk to the organization.

Overall, the main aim of a VAPT is to provide the organization with a better understanding of its security posture and help them improve it by finding out and addressing the vulnerabilities and the bugs in its systems and the network, and also by conducting a proper test to measure the effectiveness of its security controls and the incident response strategy.

3.2.2 Scope

The range of a Vulnerability Assessment and Penetration Testing (VAPT) is dependable on the certain needs and purpose of any organization, as well as the systems and networks that are being tested. However, in general, the scope of a VAPT and this project also includes the following:

1. Systems and applications: The process of vulnerability assessment and penetration testing covers all of the organization's systems and applications, including their servers, databases, and web applications.
2. Network infrastructure: The organization's network infrastructure, which includes routers, switches, firewalls, and other network devices is also included in the VAPT.
3. Wireless networks: If the organization has wireless networks, these should be included in the scope of the VAPT.
4. Web applications: The VAPT also includes testing of all web applications that are

made available all over the internet or internal network.

5. Mobile applications: Any mobile application that the company might have, should also be included in the APT.
6. Social engineering: To evaluate the organization's human security controls, the VAPT should also involve social engineering tests, for example, phishing emails or phone calls.
7. Physical security: If the VAPT includes physical security testing, it should cover all physical access points to the organization's facilities, including doors, windows, and other entry points.

The scope of a VAPT is carefully defined and is targeted to ensure that all relevant systems and networks are tested, and that the testing is conducted in a manner that is completely safe, and compliant with any relevant regulations or standards in particular. Additionally, the scope of the VAPT is reviewed periodically to ensure that it remains relevant and up-to-date in the face of changing threats and technologies.

Apart from these, there are mainly 3 possible scopes as well:

Black box testing: Testing from an external network with no prior knowledge of the internal network and systems.

Grey Box Testing: Testing from either external or internal networks with the knowledge of the internal network and the system. It's the combination of both black and white box testing.

White Box testing: Testing within the internal network with the knowledge of the internal network and system. Also known as Internal Testing

3.2.3 Information Gathering

While performing a VAPT it is very important to have the complete information stored with you to perform the assessment. Information gathering helps the tester to know about the target system and about the application we are going to test. There are a few key activities and point that are important to know when gathering information :

- **Network Scanning:** The very first step is to have a complete scanning of the network to identify the target system and the applications that are being tested. This can be done using tools such as Nmap, Netdiscover, and Angry IP Scanner.

- **Port Scanning:** After the target systems have been scanned and identified, the next step is to scan the ports on those systems to determine which services are running and which ports are open. Port scanning can be achieved using tools such as Nmap, Masscan, and Hping.
- **Service Identification:** Once the port scanning is completed and open ports have been detected, the tester attempt to identify the services that are running on those ports. This provides important information about the victim’s systems. This can be achieved using tools such as BannerGrab, Netcat, and Telnet.
- **OS Fingerprinting:** To learn more about potential vulnerabilities, the tester will also attempt to determine the operating system being used by the targeted systems. Nmap, POf, and Xprobe2 are a few tools that can be used for this..
- **Web application fingerprinting:** If the testing will involve web applications, the tester will also make an effort to identify the kind of web server and the framework being used, since this might provide useful information about potential vulnerabilities. Tools like WhatWeb, Wappalyzer, and NetCraft can be used for this.
- **Social engineering:** In order to learn more about the target organization's employees and security measures, the information collection step may also use social engineering methods like phishing emails or phone calls.

3.2.4 Vulnerability Detection

Out of all the most important and crucial steps of the whole assessment, Detection stands first.. Here is the diagram of how this step takes place.



Fig 3: Steps of assessment [9]

1. Setup

- Begin Documentation
- Secure Permissions
- Update the tools
- Configure the tools

2. Test Execution

Vulnerability Scanning: For this procedure I have used the tool Burp Suite tool:

- Configure Burp Suite: Setup Burp Suite to configure the burp suite to intercept the traffic after configuring the proxy settings in the browser.
- Identifying the target: Determining the target using the browser to navigate to the pages for scanning after entering the URL of a web application..

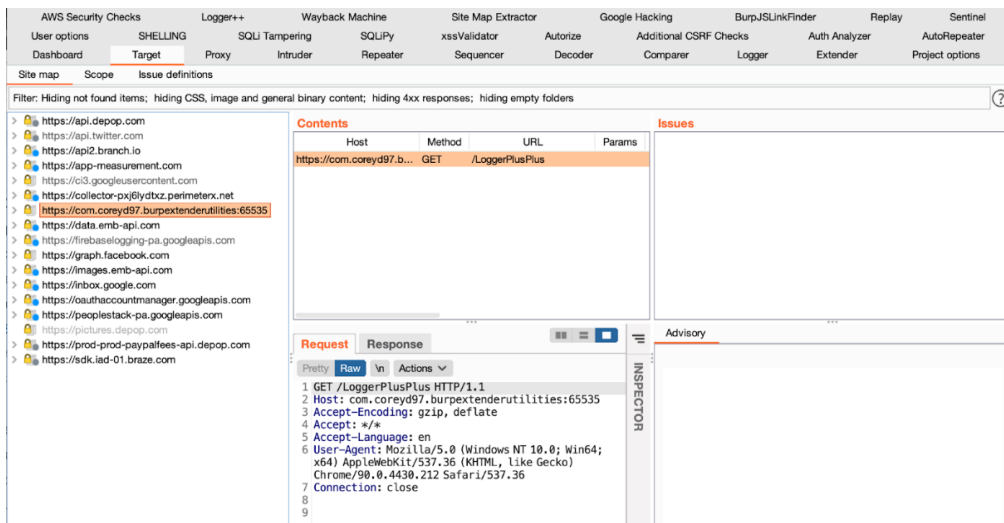


Fig 4: Vulnerability Scanning

- Spider the web application: once the target has been identified, the next step is to spider the web application. This step will involve using Burp Suite's spider tool to crawl the web application and identify all the pages and links on the website.
- Conduct the vulnerability scan: After the spidering is complete, the next step is to conduct the complete vulnerability scan. Burp Suite provides a scanner tool that can be used to automatically scan the web application for vulnerabilities based on the pages and links identified during spidering.

Penetration testing: this step involves attempting to exploit the vulnerabilities that were identified in the previous step to gain unauthorized access, escalate privileges, or perform other malicious activities.

- Analyze the vulnerability: This involves reviewing the report that the Burp suite generated and prioritizing based on the severity.
- ▲ • Choose the attack method: Once the vulnerabilities have been analyzed and prioritized, the next step is to choose the attack method. Burp Suite provides several tools and techniques for conducting attacks, including manual testing and automated attacks.
- Conduct the attack: After selecting the attack method, the next step is to conduct the attack. Burp Suite can be used to send requests to the web application with malicious payloads to exploit vulnerabilities and gain unauthorized access.

3. Vulnerability Analysis

Using burp suite for this step:

- Vulnerability Details Pane: This pane helps in providing detailed information about the vulnerability, which includes its severity level, the affected URL, and also the specific parameters that are involved in the vulnerability. It helps in recommending remediation actions to fix the vulnerability.
- Scanner Issues: This feature provides a list of all the vulnerabilities that are identified during the scan, along with their intensity levels and other details. I have sorted and filtered vulnerabilities using the provided filters based on various criteria, such as severity level, type of vulnerability, and affected URL.

Request	Payload	Status	Error	Timeout	Length	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
1	root	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
2	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
3	test	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
4	guest	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
5	info	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
6	adm	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
7	mysql	200	<input type="checkbox"/>	<input type="checkbox"/>	3186	
8	user	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
9	administrator	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
10	oracle	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
11	ftp	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
12	pi	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	

Request	Response
1	POST /login HTTP/1.1
2	Host: acce1f1c1f1e258680c1144100a000bc.web-security-academy.net
3	Connection: close
4	Content-Length: 72
5	Cache-Control: max-age=0

Fig 5: Attack Surface Analysis.

- Attack Surface Analysis: This feature has helped me analyze the application to identify areas that are vulnerable to attacks, such as input fields or text fields that are not properly validated, some authentication mechanisms that are weak, or access controls that are not properly configured.
- Vulnerability Analysis Reports: Burp Suite provides various reporting options that can be used to analyze and document the vulnerabilities that are identified during the VAPT. These reports provide detailed information about the vulnerabilities, including their severity level, potential impact, and recommended remediation actions.

4. Reporting

Burp Suite provides many integral features and tools that help in generating vulnerability assessment reports. Here are some of the key steps that are involved in generating a report in Burp Suite:

- The first step is to navigate to the “Issues” tab in Burp Suite. The issues tab has a complete list of all the issues (vulnerabilities) that are identified during the scanning process.
- We need to select the issues that are going to be included in the report. There is an option of selecting individual issues or selecting all issues by clicking the checkbox at the top of the list.
- Next is to click on the “Report” button in the top right corner of the “Issues” tab, which opens the “Report Generator” window.
- In the “Report Generator” window, select the report templates needed. Burp Suite provides several built-in report templates, including HTML, XML, and PDF.
- Next I configured the report settings, such as the output file location, the report title, and the report description.
- The last step is to click the “Generate Report” button to generate the report. Burp Suite will generate a report that includes a summary of the issues, detailed information about each issue, and recommended remediation actions.

5. Remediation

Remediation is a very important step in the VAPT (Vulnerability Assessment and Penetration Testing). In this we have to address the vulnerabilities that were identified during the assessment. Remediation is the last step in the VAPT process, and it involves taking steps to fix the vulnerabilities found and to work on reducing the risk of exploitation in future.

3.2.5 Information Analysis and Planning

Includes testing as well

Test Cases:

1. Track data transmitted across wire
2. Track data stored in file

3. Check for secret password saved by programmer in a secret file
4. Check if error page and condition expose any data which might help hacker
5. Check if binary file consist of any sensitive information
6. Check URL for sensitive data
7. Check if internal server contain sensitive information
8. Check if the application returns more data than it is needed
9. Check for multi stage elevation
10. Check for weak discretionary ACL
11. Check for buffer overflow
12. Attempt to modify execution flow for instance serial key validation
13. Try to identify insecure function call for insecure methods
14. Make an attempt to overflow protocol, server name, file name, query string and file extension
15. Check for canonicalization attacks like using /, \ to access roots or may be like using environment variable to denote path
16. Check for DOS attack strategies like changing expected data types, repeat same action over and again, attempt to connect to server concurrently
17. Check for XML injection attack like crashing XML parser, Xquery injection and XML external entity attack
18. Check for format string attack
19. Check for spoofing attack like changing MAC address and IP address
20. Check for HTML script injection attacks
21. Check for COM and ActiveX attacks
22. Check for code disassembling like performing algorithm reversing, analysis of security updates and patching binaries.

3.3 TOOLS USED

- **Burp Suite**

Burp Suite (used most of the times for dynamic analysis of the website and mobile application)

Burp or Burp Suite is a set of tools used for penetration testing of web applications. It is

developed by the company named Portswigger, which is also the alias of its founder Dafydd

Stuttard. BurpSuite aims to be an all-in-one set of tools and its capabilities can be enhanced by installing add-ons that are called BApps.

Burp Suite Modules:

- Target
- Proxy
- Spider
- Scanner
- Intruder
- Repeater
- Sequencer
- Decoder
- Comparer
- Extender.

Target:

The Target tool gives you an overview of your target application’s content and functionality, and lets you drive key parts of your testing workflow. The key steps that are typically involved in using the Target tab are described below.

Proxy:

The Proxy tool lies at the heart of Burp’s user-driven workflow, and gives you a direct view into how your target application works “under the hood”. It operates as a web proxy server, and sits as a man-in-the-middle between your browser and destination web servers. This lets you intercept, inspect and modify the raw traffic passing in both directions.

#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title	Comment	TL
185	https://api.depop.com	POST	/oauth2/access_token/	✓		401	1372	JSON				▼
186	https://api.depop.com	POST	/oauth2/access_token/	✓		200	1910	JSON				▼
187	https://api.depop.com	GET	/api/v1/users/me/			200	2785	JSON				▼
188	https://api2.branch.io	POST	/v1/profile	✓		200	2185	JSON				▼
189	https://api.depop.com	POST	/api/v1/devices/	✓		201	1204	JSON				▼
190	https://api.depop.com	GET	/api/v1/feature/switch/			200	3859	JSON				▼
191	https://graph.facebook.com	GET	/v9.0/me?fields=id%2Cname%2Cfri...	✓		400	947	JSON	0/me			▼
192	https://api.twitter.com	POST	/oauth2/invalidate_token			403	1170	JSON				▼
193	https://api.depop.com	GET	/api/v1/onboarding/profile/			200	1049	JSON				▼
194	https://api.depop.com	GET	/api/v1/users/27892955/			200	2785	JSON				▼
195	https://api.depop.com	GET	/api/v1/users/27892955/counters/			200	918	JSON				▼
196	https://api.depop.com	GET	/api/v1/seller-info			200	1856	JSON				▼
197	https://api.depop.com	GET	/api/v1/users/27892955/following/limit	✓		200	887	JSON				▼

Fig 6: Proxy

Spider

Burp Spider is a tool for automatically crawling web applications. You can use this in conjunction with manual mapping techniques to speed up the process of mapping an application's content and functionality.

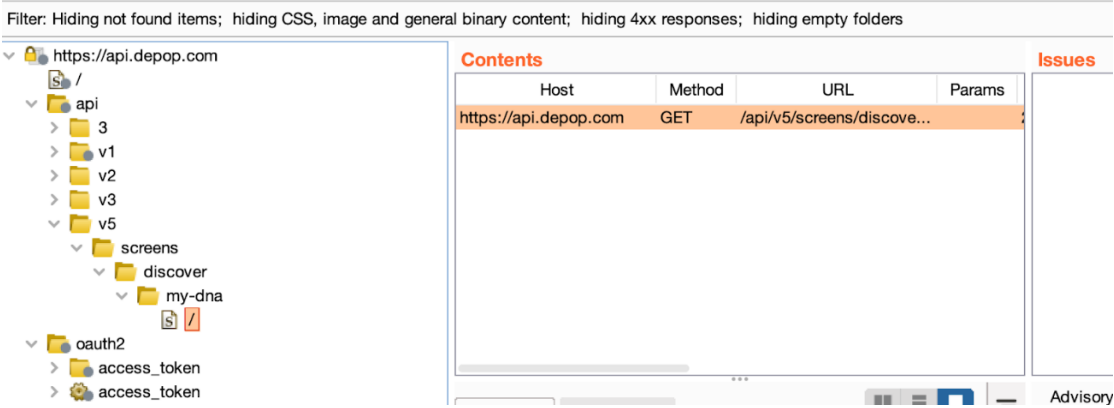


Fig 7: Spider

Scanner

Burp Scanner is a tool for automatically finding security vulnerabilities in web applications. It is designed to be used by security testers, and to fit in closely with your existing techniques and methodologies for performing manual and semi-automated penetration tests of web applications.

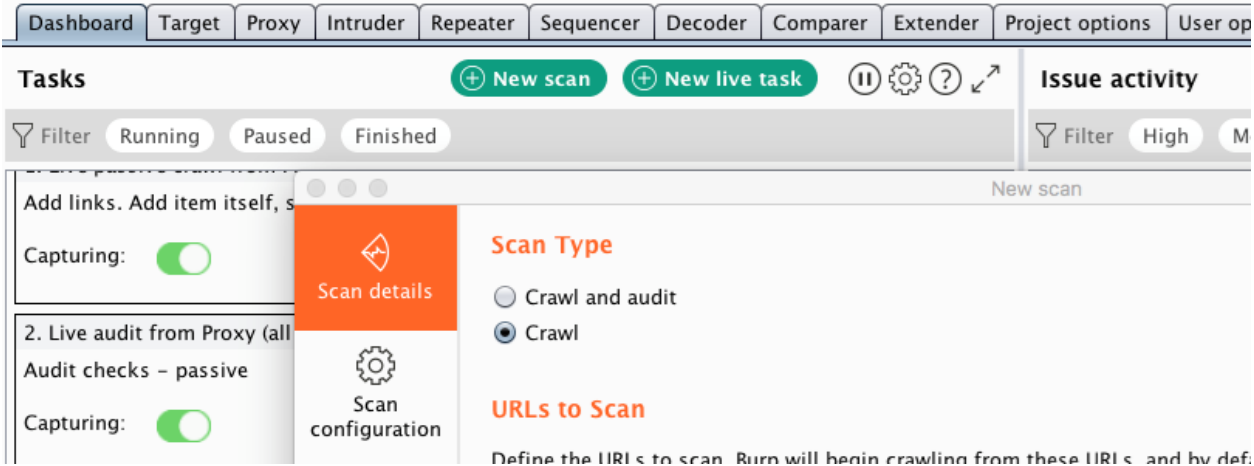


Fig 8: Scanner

Intruder

Burp Intruder is a tool for automating customized attacks against web applications. It is extremely powerful and configurable and can be used to perform a huge range of tasks, from simple brute-force guessing of web directories to active exploitation of complex blind SQL injection vulnerabilities.

Request	Payload	Status	Error	Timeout	Length	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
1	root	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
2	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
3	test	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
4	guest	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
5	info	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
6	adm	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
7	mysql	200	<input type="checkbox"/>	<input type="checkbox"/>	3186	
8	user	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
9	administrator	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
10	oracle	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
11	ftp	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	
12	pi	200	<input type="checkbox"/>	<input type="checkbox"/>	3184	

Request	Response
1 POST /login HTTP/1.1	2 Host: accelfclfle258680c1144100a000bc.web-security-academy.net
3 Connection: close	4 Content-Length: 72
5 Cache-Control: max-age=0	

Fig 9: Intruder

Repeater

Burp Repeater is a simple tool that helps in manual manipulating and reissuing of individual HTTP requests, and analyze the application’s responses. You can use Repeater for all kinds of purposes, such as changing parameter values to test for input-based vulnerabilities, issuing requests in a specific sequence to test for logic flaws, and reissuing requests from Burp Scanner issues to manually verify reported issues.

Request	Response
1 GET / HTTP/1.1	1 HTTP/1.1 301 Moved Permanently
2 Host: technipages.com	2 Date: Mon, 09 Nov 2020 17:38:37 GMT
3 Connection: close	3 Content-Type: text/html; charset=UTF-8
4 Upgrade-Insecure-Requests: 1	4 Connection: close
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.121 Safari/537.36	5 Set-Cookie: __cfduid=dea472cb9bf8769b154fc26c33ced4e0b1604943516; expires
6 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9	6 X-Powered-By: PHP/7.2.31
7 Sec-Fetch-Site: none	7 Referrer-Policy: unsafe-url
8 Sec-Fetch-Mode: navigate	8 X-Frame-Options: SAMEORIGIN
9 Sec-Fetch-User: ?1	9 X-XSS-Protection: 1; mode=block
10 Sec-Fetch-Dest: document	10 X-Content-Type-Options: nosniff
11 Accept-Encoding: gzip, deflate	11 X-Redirect-By: WordPress
12 Accept-Language: en-US,en;q=0.9,en;q=0.8	12 Location: https://www.technipages.com/
	13 Cache-Control: max-age=7776000
	14 Expires: Sun, 07 Feb 2021 17:38:37 GMT
	15 Vary: User-Agent
	16 CF-Cache-Status: DYNAMIC
	17 cf-request-id: 064fafac8e0000f9de5994c000000001
	18 Expect-CT: max-age=604800, report-uri="https://report-uri.cloudflare.com/
	19 Set-Cookie: __cf_bm=2e542cc537a7b2ed135fcbde524070a4b97954b-1604943517-1
	20 Report-To: [{"endpoints":[{"url":"https://a.nel.cloudflare.com/report?8
	21 NEL: {"report_to":"cf-nel","max_age":604800}
	22 Server: cloudFlare
	23 CF-RAY: Sef94ef41caef9de-PRG
	24 Content-Length: 0
	25

Fig 10: Repeater

Sequencer

Burp Sequencer is a tool for analyzing the quality of randomness in a sample of data items. The data items can either be application’s session ID’s, CSRF tokens, password reset or forget password tokens or any specific unpredictable ID generated by the application.

The Burp Sequencer is one of the most amazing tools that try to capture the randomness or the variances in the session ID's by employing some standard statistical tests which are based on the principle of testing a hypothesis against a sample of evidence, and calculating the probability of the observed data occurring.

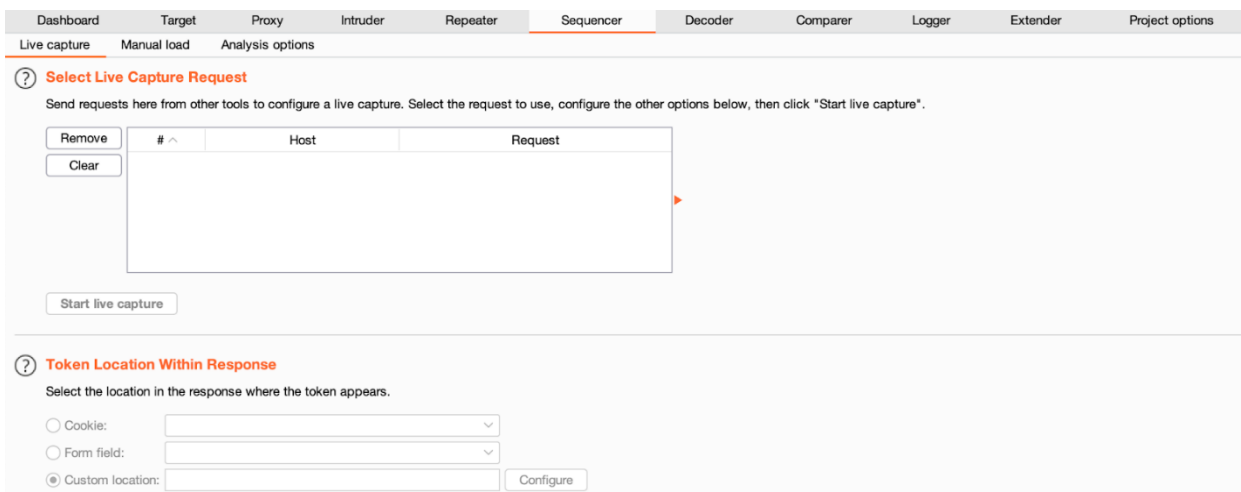


Fig 11: Sequencer

Decoder

Burpsuite Decoder can be said as a tool which is used for transforming encoded data into its real form, or for transformation of raw data into various encoded and hashed forms. This tool helps in recognition of several encoding formats using defined techniques. Encoding is the process of putting a sequence of character's (letters, numbers, punctuation, and symbols) into a specialized format which is used for efficient transmission or storage. Decoding is the opposite process of encoding the conversion of an encoded format back into the original format. Encoding and decoding can be used in data communications, networking, and storage.

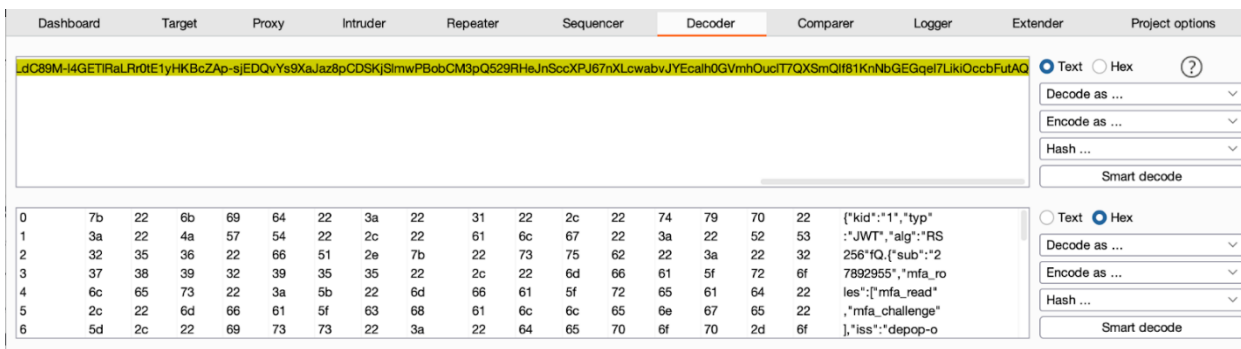


Fig 12: Decoder

Comparer

- **Nikto**

Nikto is an open-source, free vulnerability assessment and penetration testing (VAPT) web server scanner. The tool aids in detecting potential security flaws and vulnerabilities in web servers and software applications.

The Nikto tool runs a wide range of tests, involving looking for out-of-date web servers and web applications, looking via all of the default files and directories for any that might hold private data, checking for incorrectly configured web servers and web applications, and looking for widespread vulnerabilities like SQL injection and cross-site scripting (XSS).

Furthermore, Nikto has the ability to scan numerous web servers and applications at once and may produce thorough reports on the vulnerabilities it finds. Additionally, the program supports a number of authentication techniques, such as basic and NTLM authentication, and can configure to work with various plugins and custom tests.

```
(xenia)fxbg@localhost:~$ nikto -Cgidirs None -h aworkoutroutine.com
- Nikto v2.1.5
-----
+ Target IP:      104.27.184.222
+ Target Hostname: aworkoutroutine.com
+ Target Port:    80
+ Start Time:    2018-03-07 02:59:50 (GMT-7)
-----
+ Server: cloudflare
+ Uncommon header 'cf-ray' found, with contents: 3f7c362c4658399a-PHX
+ Uncommon header 'x-frame-options' found, with contents: SAMEORIGIN
^G^G+ Server banner has changed from 'cloudflare' to 'cloudflare-nginx' which may suggest a WAF, load balancer or proxy is in place
- STATUS: Completed 3270 tests (~50% complete, 2.7 minutes left: currently in plugin 'Nikto Tests')
+ 6545 items checked: 0 error(s) and 2 item(s) reported on remote host
+ End Time:      2018-03-07 03:05:10 (GMT-7) (320 seconds)
-----
+ 1 host(s) tested
```

The ease of use and adaptability of Nikto are two of its main benefits. It is simple to connect with other tools and scripts because it can be run from the command line. Nikto is also extremely flexible, offering a variety of features and settings that may be modified to meet the individual requirements of the user.

- **Sqlmap-dev**

An open-source command-line penetration testing tool called SQLmap-dev is made to identify and take advantage of SQL injection flaws in any given web application. It is built in Python and can be used to test a variety of databases, including PostgreSQL, Oracle, MySQL, and Microsoft SQL Server. As we all know, SQL injection is a very common attack that involves adding malicious SQL code into any text field in a web application, such as a login form, a search box, or any other text field, in order to gain

In order to figure out whether a given directory or file is present on the server, DirBuster sends HTTP queries to the web server and analyses the return codes. It can also be set up to use different wordlists, such as well-known lists of frequently used directories and web page names, to improve the accuracy of brute force attacks.

```
root@llamasec:~# gobuster -u http://192.168.1.2/ -w /usr/share/wordlists/dirb/co
mmon.txt -fw

=====
Gobuster v2.0.1                OJ Reeves (@TheColonial)
=====
[+] Mode           : dir
[+] Url/Domain     : http://192.168.1.2/
[+] Threads       : 10
[+] Wordlist       : /usr/share/wordlists/dirb/common.txt
[+] Status codes  : 200,204,301,302,307,403
[+] Timeout       : 10s
=====
2019/03/25 23:40:34 Starting gobuster
=====
/.hta (Status: 403)
/.htaccess (Status: 403)
/.htpasswd (Status: 403)
/cgi-bin/ (Status: 403)
/index.html (Status: 200)
/passwords (Status: 301)
/robots.txt (Status: 200)
=====
```

Fig 17: Directory buster

With the help of DirBuster, users can select a unique user-agent string, manage the number of threads utilized in the brute force attack, and set the program to automatically follow redirection.

3.4 SYSTEM DEVELOPMENT CYCLE

We must understand the Software Development Life Cycle (SDLC) idea in order to produce software or any other type of product. The SDLC can help in the context of VAPT to make sure that the project is handled successfully and that the eventual VAPT solution is developed in a systematic and consistent manner. A simple way to implement SDLC in VAPT project can be using Waterfall method:



Fig 18: Waterfall Method

- The procedures for adding an SDLC to a VAPT project are as follows:
- Planning: The project scope, objectives, resources, and timelines are all stated during this phase.
- Analysis: In the second stage, the VAPT team collects data on the systems and applications that will be put to the test, such as architecture diagrams, network layouts, system configurations, user roles, and permissions. A testing plan that specifies the precise tests that must be performed is created using the data provided.
- Design: During the design phase, the VAPT team creates the testing environment and architecture, as well as the test cases and scenarios, the testing tools to be used, and the protocols and procedures to be followed.

- **Implementation:** In this phase, the VAPT team executes the testing plan, using the selected tools and methodologies to identify and evaluate vulnerabilities in the target systems and applications.
- **Testing:** During this step, the VAPT team evaluates the testing results, ensuring that all vulnerabilities have been found and resolved and that the overall VAPT solution fulfils the objectives that were established.
- **Deployment:** During this step, the suggested security controls and measures are put into place in order to address the vulnerabilities that discovered during the testing phase. The deployment process is crucial because it entails closing security gaps and addressing vulnerabilities found during testing, which lowers the system or application's overall risk.
- **Maintenance:** This stage mostly comprises VAPT's post-completion work. Because threats and vulnerabilities can vary over time, security controls and procedures must be updated and maintained to stay up with those changes. This is why the maintenance phase is vital.

Chapter 04: PERFORMANCE ANALYSIS & RESULTS

4.1 System Configuration

4.1.1 Software Requirements

These are the software configurations used:

Operating System: Windows 11.

Tool: Burp Suite, SQLmap-dev

4.1.2 Hardware Requirements

These are the Hardware interfaces used Processor: Intel i5 9th Gen 6 cores 12 threads

RAM: 8GB DDR4 RAM

Monitor: 17.3'' full HD color monitor

Mouse: Scroll or optical mouse

Graphics Device: Nvidia GTX 1650 4GB DDR5 VRAM

Keyboard: Standard 110 keys keyboard

4.2 Critical Vulnerabilities Found

4.2.1 Authentication bypass via NoSQL Injection

A web application that uses a NoSQL database may have a security weakness known as a NoSQL injection vulnerability. This kind of vulnerability permits an attacker to bypass authentication, gain access to sensitive information, edit it, or even obtain full control over the web application. The chief reason of NoSQL injection attacks is the lack of proper data sanitization, which allows an attacker to manipulate queries and inject malicious code into the database.

Vulnerable URL: <http://139.59.66.27/api/users/signin>

Affected Parameter: email, password

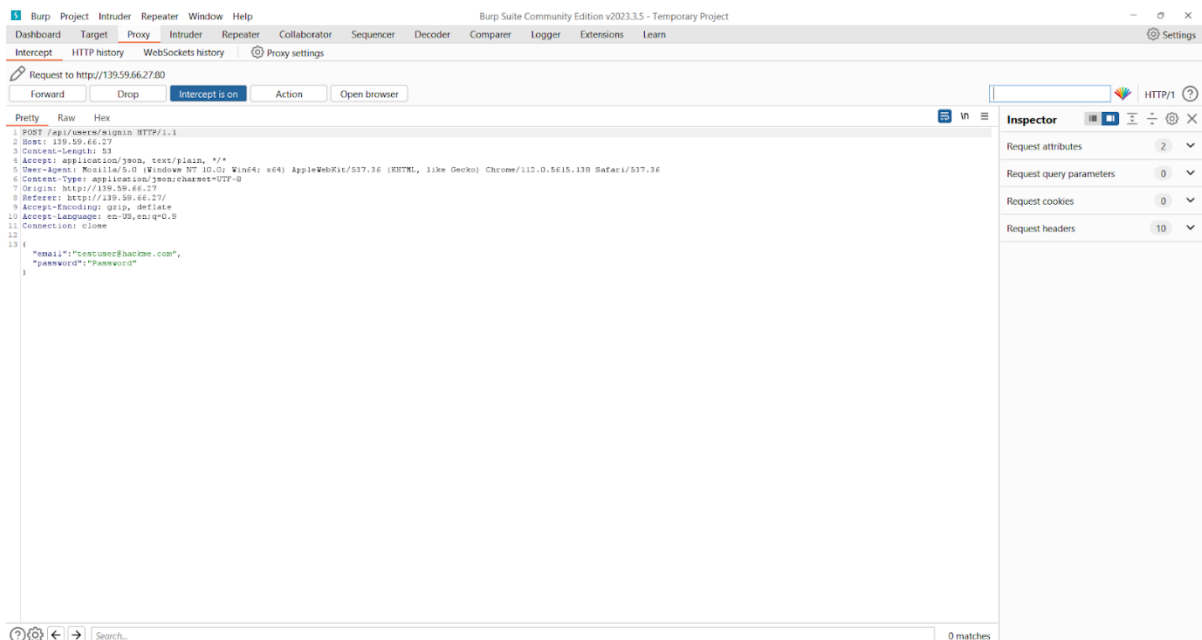
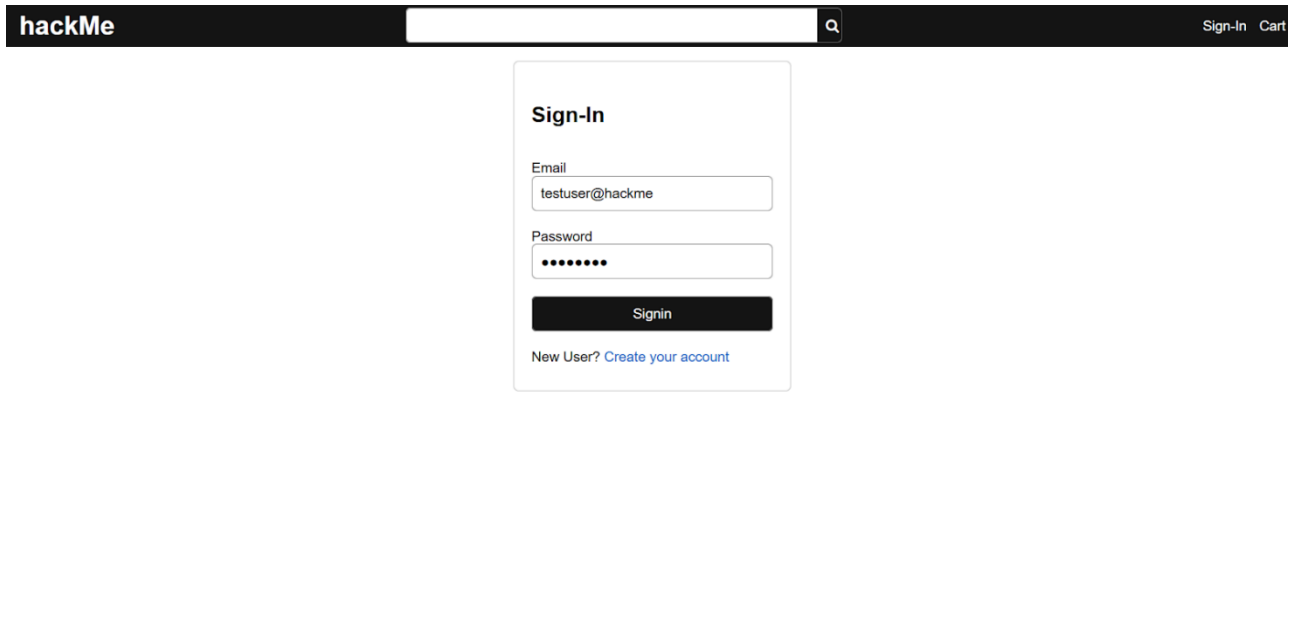


Fig 19-20: Authentication bypass via NoSQL Injection

The **technical impact** of a NoSQL injection vulnerability is important as an attacker can bypass authentication and gain access to the admin user account. This access allows the attacker to manipulate orders, products, modify, and delete data, ultimately leading to full ownership of the website.

To **remediate** this vulnerability, it is crucial to ensure that NoSQL API calls are constructed without unsensitized data using techniques such as existing escaping libraries. Although NoSQL databases

like MongoDB offer built-in features to prevent JavaScript in database queries, input validation is still necessary. Input validation can be achieved using sanitization libraries like mongo-sanitize or mongoose.

4.2.2 Admin Credentials Exposed

Description: In this testing, the admin credentials are exposed and publicly available hackers can get these credentials via fuzzing or even content discovery and can easily get access to the complete admin account.

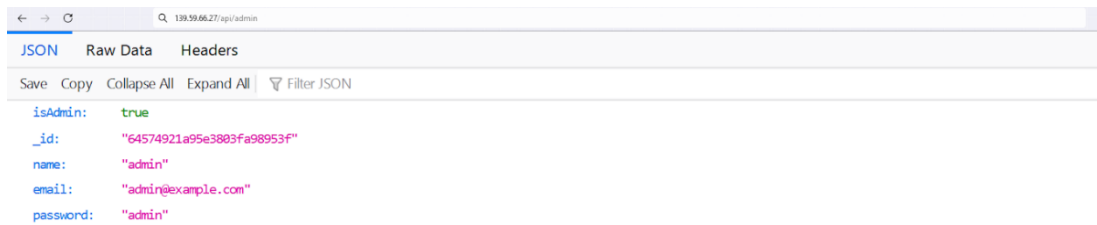


Fig 21: Admin Credentials Exposed

4.2.3 Source Code Disclosure

Description: The source code of the application is exposed publicly and accessible to anyone.

Technical Impact: The Source code of application is accessible publically. Attacker can read, observe and can find vulnerability via disclosed source code.

Remediation: Before deploying the production check for sensitive data and directory.

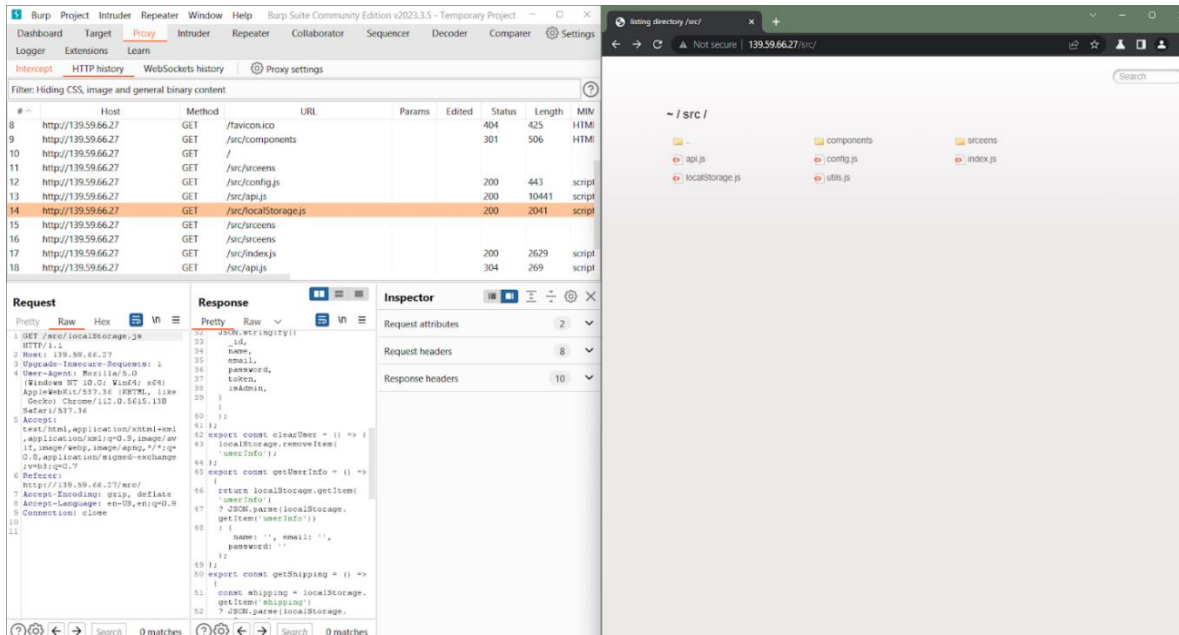


Fig 22: Source code disclosure

4.2.4 Price Manipulation leads to free product purchase

Description: This business logic vulnerability enables hackers to purchase any product for free.

Affected Parameter: totalPrice

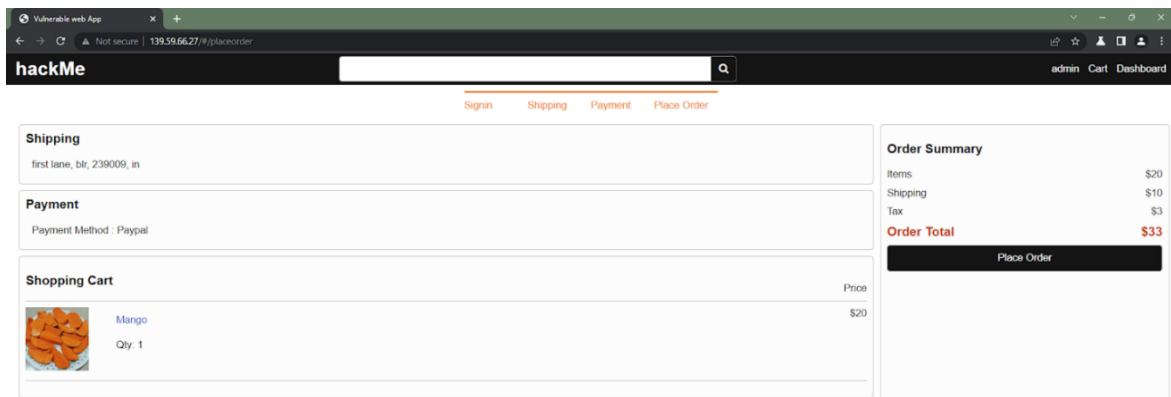
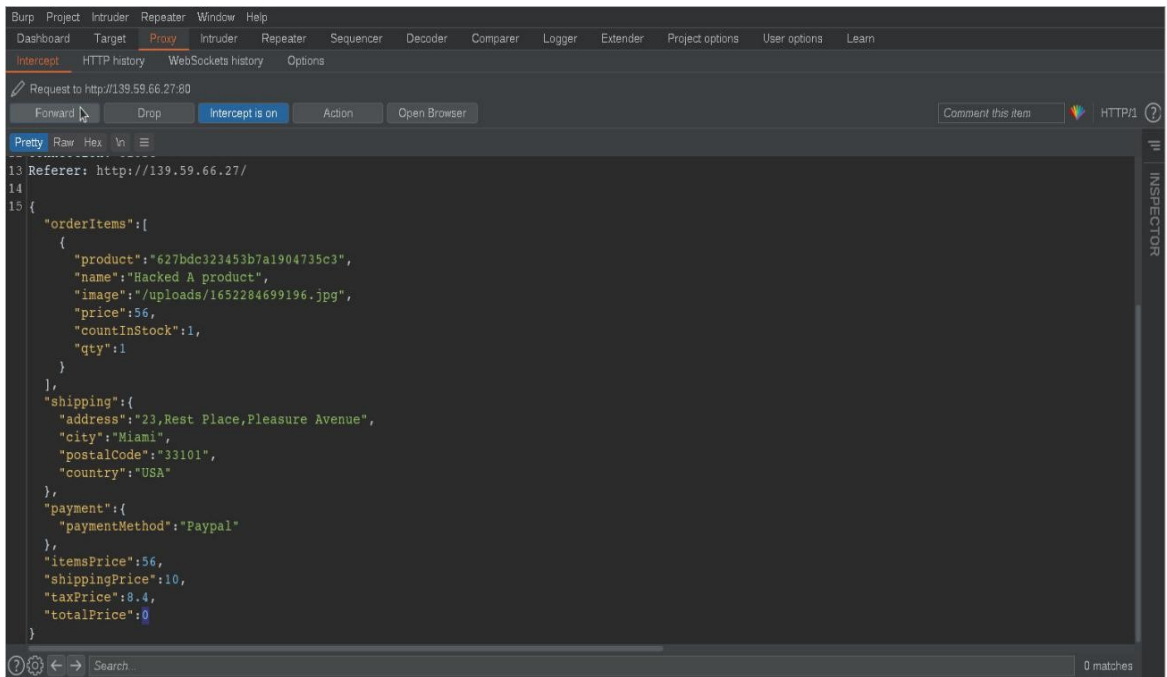
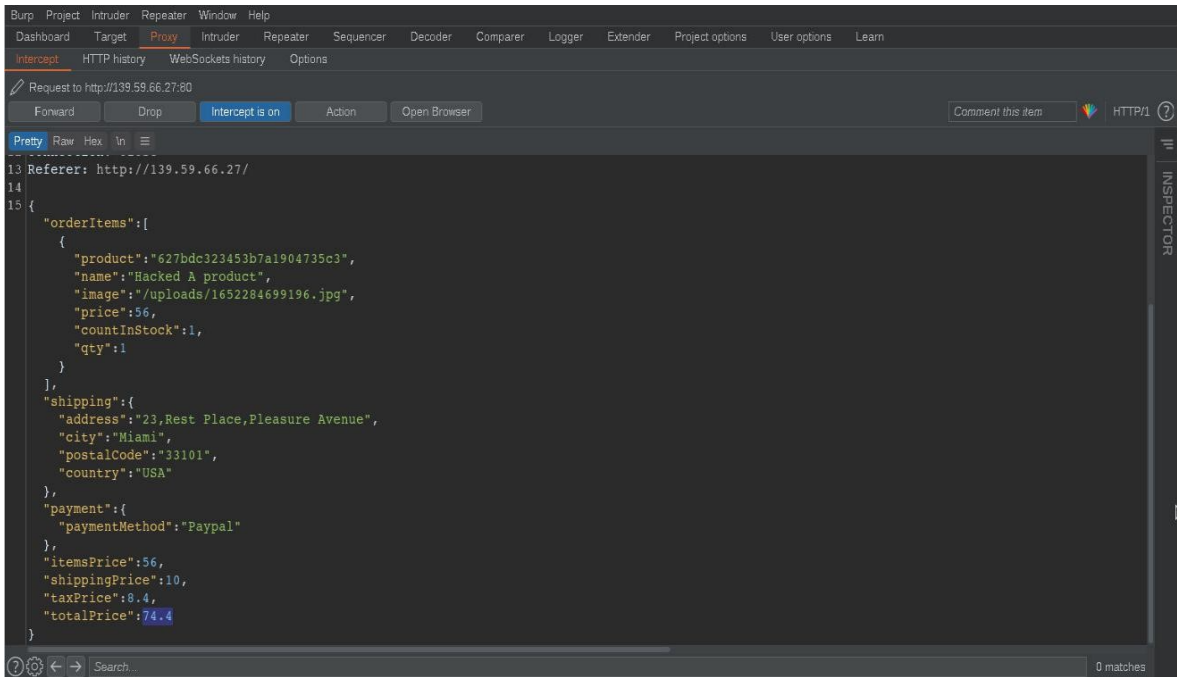
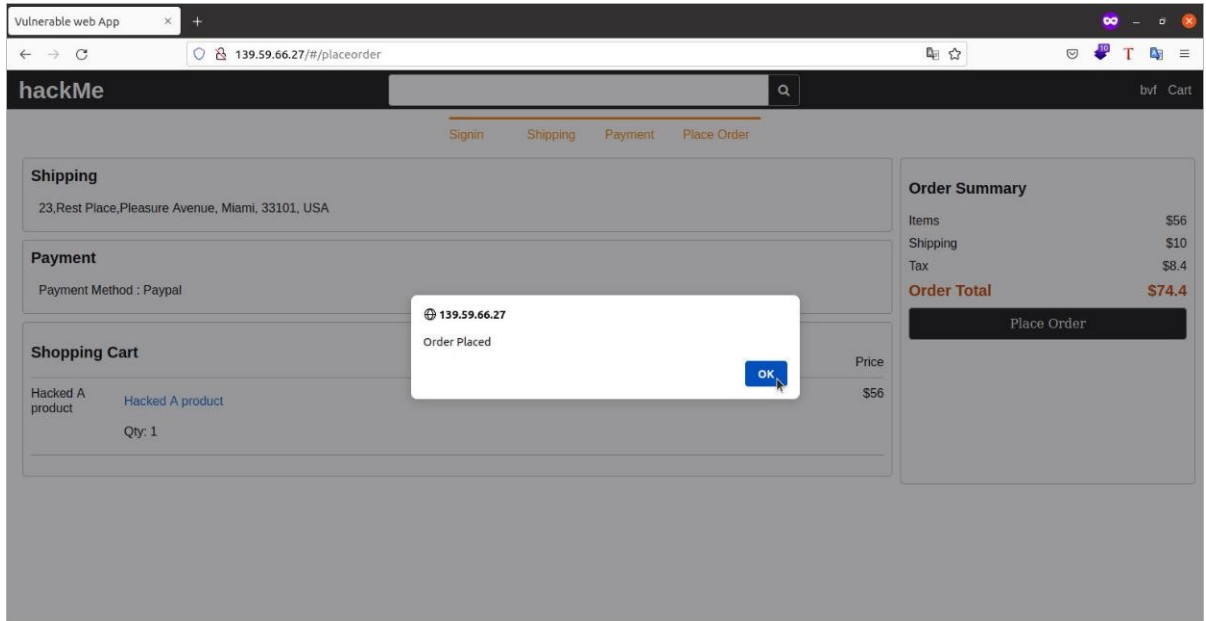


Fig 23: Price maipulation





627f56d53453b7a1904750c1	2022-05-14T07:14:29.316Z	74.4	Yes	No
627f56d53453b7a1904750c4	2022-05-14T07:14:29.430Z	74.4	Yes	No
627f56d53453b7a1904750c7	2022-05-14T07:14:29.545Z	74.4	Yes	No
627f56d53453b7a1904750ca	2022-05-14T07:14:29.680Z	74.4	Yes	No
627f56d53453b7a1904750cd	2022-05-14T07:14:29.830Z	74.4	Yes	No
627f56d53453b7a1904750d0	2022-05-14T07:14:29.989Z	74.4	Yes	No
627f56d63453b7a1904750d3	2022-05-14T07:14:30.115Z	74.4	Yes	No
627f56d63453b7a1904750d6	2022-05-14T07:14:30.278Z	74.4	Yes	No
627f56d93453b7a19047511e	2022-05-14T07:14:33.564Z	74.4	Yes	No
627f86f53453b7a190476be0	2022-05-14T10:39:49.310Z	74.4	Yes	No
627f8d123453b7a190476d45	2022-05-14T11:05:54.495Z	0	Yes	No

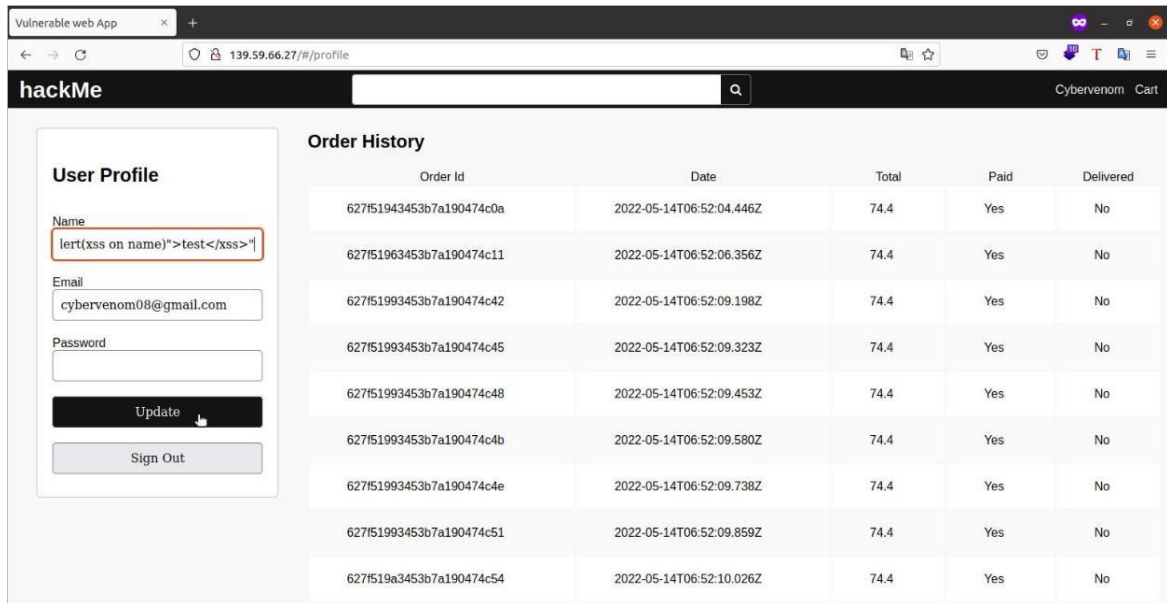
Fig 24-27: Price Manipulation

4.2.5 Stored Cross-site scripting on name

Description: This application fails to sanitize user input which is then stored in the database and executed as a javascript.

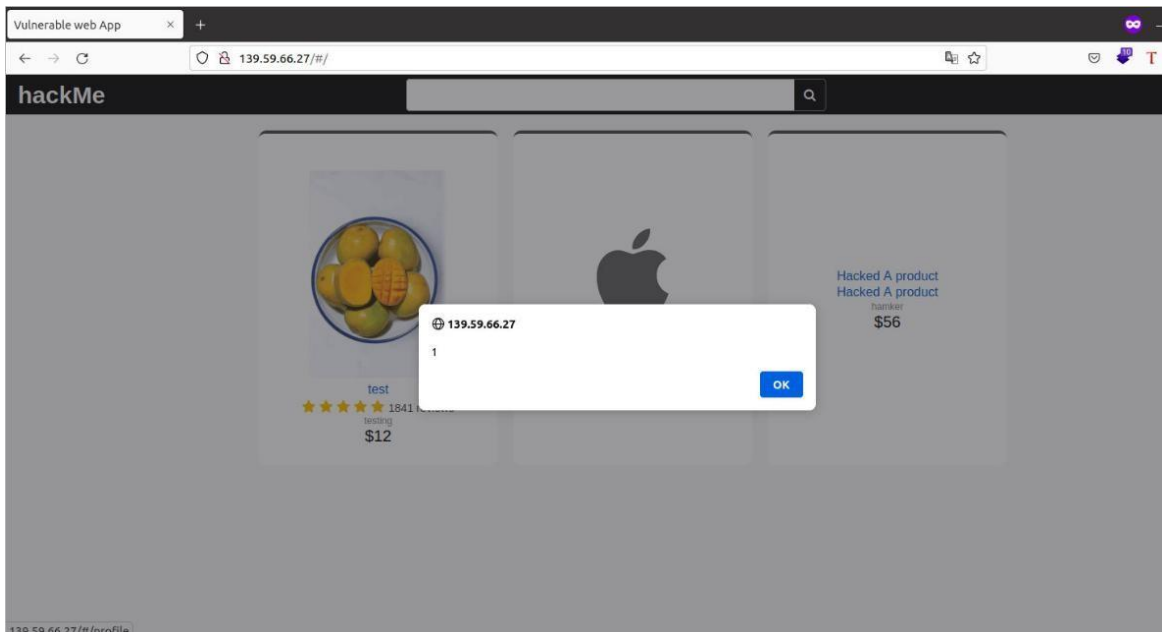
Affected parameters: name

Payload: `<xss onmouseover="alert(1)" style=display:block>test</xss>`

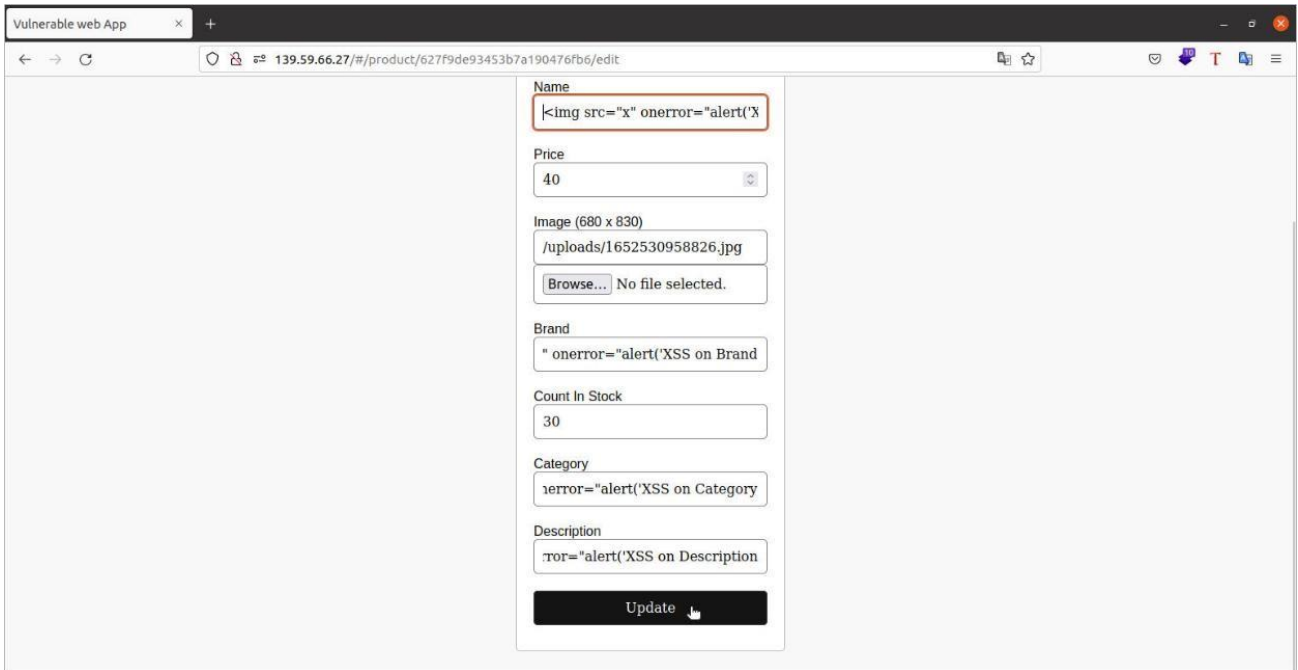
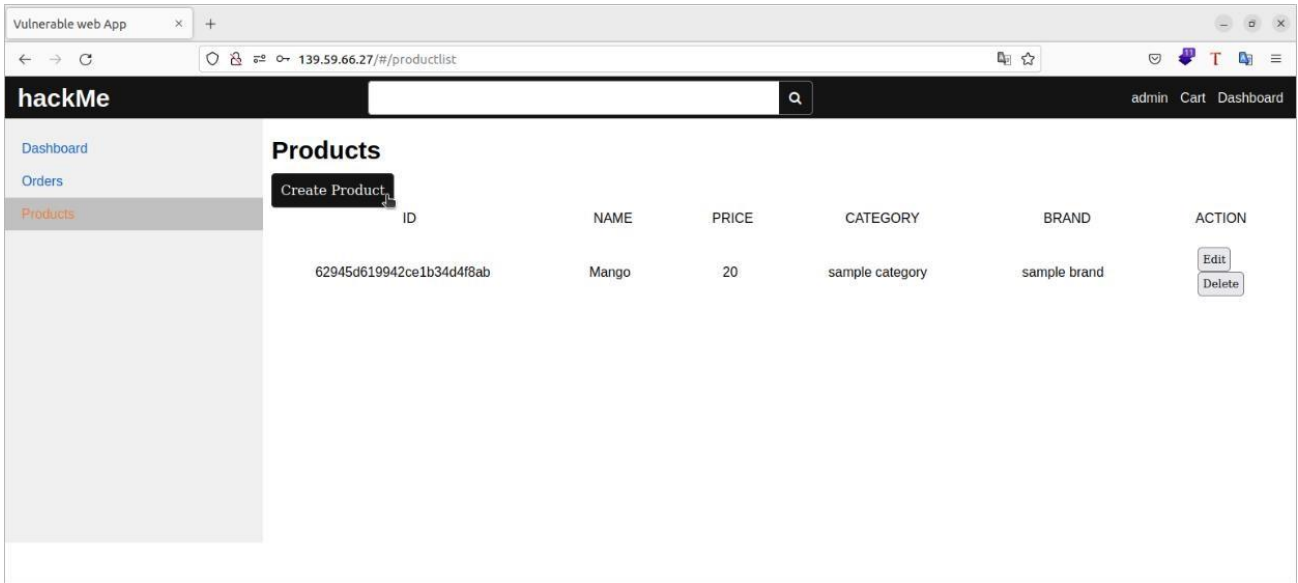


The screenshot shows the 'hackMe' user profile page. The 'User Profile' section has a 'Name' field containing the payload: `!ert(xss on name)">test</xss>¶`. The 'Order History' table is visible on the right.

Order Id	Date	Total	Paid	Delivered
627f51943453b7a190474c0a	2022-05-14T06:52:04.446Z	74.4	Yes	No
627f51963453b7a190474c11	2022-05-14T06:52:06.356Z	74.4	Yes	No
627f51993453b7a190474c42	2022-05-14T06:52:09.198Z	74.4	Yes	No
627f51993453b7a190474c45	2022-05-14T06:52:09.323Z	74.4	Yes	No
627f51993453b7a190474c48	2022-05-14T06:52:09.453Z	74.4	Yes	No
627f51993453b7a190474c4b	2022-05-14T06:52:09.580Z	74.4	Yes	No
627f51993453b7a190474c4e	2022-05-14T06:52:09.738Z	74.4	Yes	No
627f51993453b7a190474c51	2022-05-14T06:52:09.859Z	74.4	Yes	No
627f519a3453b7a190474c54	2022-05-14T06:52:10.026Z	74.4	Yes	No



The screenshot shows the 'hackMe' product page. An alert box is displayed in the center of the screen, containing the IP address `139.59.66.27` and the number `1`. The background shows a product listing for 'test' with a price of \$12 and a 5-star rating.



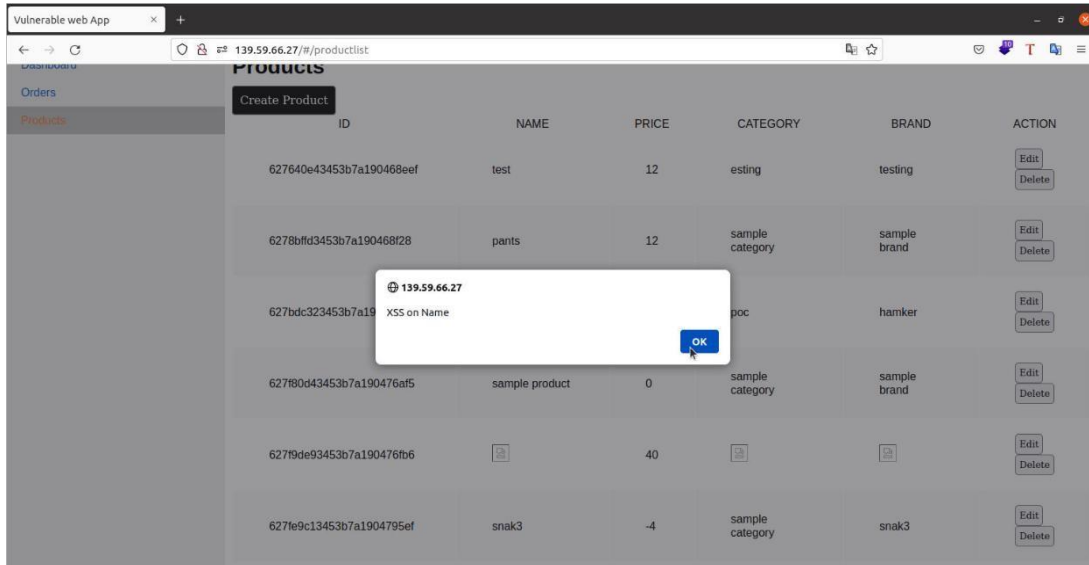


Fig 28-32: Stored Cross-site Scripting

Technical impact: The Cross Site Scripting (XSS) attack is a severe security vulnerability that poses a significant threat to web applications. The attack involves the injection of malicious script code into a web application by an attacker, either on the client-side within a user's browser or on the server-side within the database. This malicious script is usually written in JavaScript code and injected into untrusted input data on the web application. The ultimate goal of an XSS attack is to compromise the confidentiality, integrity, or availability of the web application and its users' data.[6]

Chapter 05: CONCLUSION

5.1 CONCLUSION

In conclusion, a Vulnerability Assessment and Penetration Testing (VAPT) project is a critical component of any organization's overall security strategy. It helps identify security vulnerabilities and weaknesses in the IT infrastructure and web applications, which can be exploited by attackers. Through the VAPT process, organizations can implement some robust actions to proactively detect and remediate vulnerabilities before they are exploited.

The VAPT project involves a comprehensive and structured approach that includes information gathering, vulnerability scanning, penetration testing, vulnerability analysis, and reporting. The VAPT process includes some of the useful tools for the enhancement without which it cannot happen, for example Burp Suite, Nikto, SQLmap-dev, and Directory Buster.

This internship provided me with a great knowledge on various security vulnerabilities that affect web applications. The hands-on experience I gained, helped me develop and enhance my penetration testing skills, helping me to better identify and exploit weaknesses in web applications. Through vulnerability assessments of websites, I was able to gain insights into the main cause of vulnerabilities and how to effectively manage them.

Apart from this, I have also gained a deeper understanding of reconnaissance techniques and testing methodologies, which has made a lot of my concepts clear in methods that I follow for penetration testing with a more comprehensive perspective. The knowledge and skills I acquired during this internship have not only helped me broaden my understanding of cybersecurity but have also encouraged me to work harder toward this concept and tackle real-world security challenges with confidence.

5.2 FUTURE WORK

There is a growing and never-ending discussion among cybersecurity professionals about how important and useful penetration testing is, both presently and in the coming future. While some argue that the current form of penetration testing is a useless activity, others argue that it is still an essential aspect of ensuring robust cybersecurity measures.

Both arguments have some merit.

Correctly performed penetration testing can identify vulnerable loopholes and functions that can be exploited by attackers who pierce into an organization's systems. However, if the aim and scope of testing are not defined correctly, it may not be of much use.

Still, there are a lot of companies that do not use penetration testing to evaluate the actual security of their systems, but only with the intention to meet compliance requirements. In such cases, the test results are often of little use.

While vulnerability assessments are definitely very useful in identifying potential attack surfaces, they are still not a replacement for proper penetration testing, which requires security engineers to use their skills and experience to simulate real-world attacks. The one problem that needs to be worked on is to create a proper communication channel, because the explanation of testing results to non-technical decision makers, becomes difficult.

There seems only one problem that needs to be worked on is to create a proper communication channel, because explanation of testing results to non-technical decision makers, becomes difficult.

REFERENCES

- [1] B. Zhou, B. Sun, T. Zang, Y. Cai, J. Wu, and H. Luo, "Security Risk Assessment Approach for Distribution Network Cyber Physical Systems Considering Cyber Attack Vulnerabilities," in *IEEE Access*, vol. 8, pp. 234443-234454, 2020, doi: 10.1109/ACCESS.2020.3041243.
- [2] Image 1: <https://www.netrika.com/services/information-security/security-testing-vapt-network-application-and-web-security>
- [3] P. Kashyap, V. Selvarajah, "Analysis of Different Methods of Reconnaissance", in *ICII 2021*, <https://doi.org/10.2991/ahis.k.210913.064>
- [4] Goel, Jai Narayan and Babu M. Mehtre. "Vulnerability Assessment & Penetration Testing as a Cyber Defence Technology." *Procedia Computer Science* 57 (2015): <https://doi.org/10.1016/J.PROCS.2015.07.458>.
- [5] "Volume (2): Proceeding papers," 2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS), Cairo, Egypt, 2017, pp. 1-6, doi: 10.1109/INTELCIS.2017.8260018.
- [6] G. Simran T, Sasikala D, "Vulnerability Assessment of Web Application using Penetration testing", ISSN: 2277-3878, Volume-8 Issue-4, November 2019
- [7] A. Almaarif and M. Lubis. 2021. "Vulnerability Assessment and Penetration Testing (VAPT) Framework: Case Study of Government's Website". *Int. Journal on Advanced Science Engineering Information Technology* vol. 10 (5), ISSN: 2088-5334.
- [8] Image <https://www.guru99.com/vulnerability-assessment-testing-analysis.html>
- [9] Image: <https://www.guru99.com/vulnerability-assessment-testing-analysis.html>
- [10] Image: <https://www.javatpoint.com/jira-waterfall-model>

KHUSHI SHAH 191516

ORIGINALITY REPORT

8%

SIMILARITY INDEX

%

INTERNET SOURCES

8%

PUBLICATIONS

%

STUDENT PAPERS

PRIMARY SOURCES

1

Goel, Jai Narayan, and B.M. Mehtre. "Vulnerability Assessment & Penetration Testing as a Cyber Defence Technology", *Procedia Computer Science*, 2015.

Publication

1%

2

"Advances in Computational Intelligence and Communication Technology", Springer Science and Business Media LLC, 2021

Publication

1%

3

Julia Sorrentino, Priscila Silva, Gaspard Baye, Gokhan Kul, Lance Fiondella. "Covariate Software Vulnerability Discovery Model to Support Cybersecurity Test & Evaluation (Practical Experience Report)", 2022 IEEE 33rd International Symposium on Software Reliability Engineering (ISSRE), 2022

Publication

1%

4

Shaimaa Khalifa Mahmoud, Marco Alfonse, Mohamed Ismail Roushdy, Abdel-Badeeh M. Salem. "A comparative analysis of Cross Site Scripting (XSS) detecting and defensive

1%

techniques", 2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS), 2017

Publication

5

Howard Poston. "Mapping the OWASP Top Ten to Blockchain", Procedia Computer Science, 2020

Publication

<1%

6

Nguyen Duc Thai, Nguyen Huu Hieu. "A Framework for Website Security Assessment", Proceedings of the 2019 7th International Conference on Computer and Communications Management - ICCCM 2019, 2019

Publication

<1%

7

Maria Teresa Baldassarre, Vita Santa Barletta, Danilo Caivano, Michele Scalera. "Integrating security and privacy in software development", Software Quality Journal, 2020

Publication

<1%

8

Matthew Hickey, Jennifer Arcuri. "Hands on Hacking", Wiley, 2020

Publication

<1%

9

Jeff T Parker, Michael Gregg. "Application Security and Penetration Testing", Wiley, 2019

Publication

<1%

10

Aishwarya Bhalme, Akash Pawar, Aditi Borkar, Pranav Shriram. "Cyber Attack Detection and

<1%

Implementation of Prevention Methods For
Web Application", 2022 IEEE Bombay Section
Signature Conference (IBSSC), 2022

Publication

11 David Basin, Patrick Schaller, Michael
Schlöpfer. "Applied Information Security",
Springer Nature, 2011

Publication

12 Ved Prakash Bhardwaj. "An Algorithmic
Approach to Minimize the Conflicts in an
Optical Multistage Interconnection Network",
Communications in Computer and
Information Science, 2011

Publication

13 "European Privacy by Design", Corvinus
University of Budapest, 2023

Publication

14 Keyur Patel. "A Survey on Vulnerability
Assessment & Penetration Testing for Secure
Communication", 2019 3rd International
Conference on Trends in Electronics and
Informatics (ICOEI), 2019

Publication

15 "General chairs", 2012 2nd IEEE International
Conference on Parallel Distributed and Grid
Computing, 2012.

Publication

16

Ankit Mundra, Nitin Rakesh. "Chapter 81 Online Hybrid Model for Online Fraud Prevention and Detection", Springer Science and Business Media LLC, 2014

Publication

<1%

17

Sagar Rahalkar. "A Complete Guide to Burp Suite", Springer Science and Business Media LLC, 2021

Publication

<1%

18

Basim Mahmood. "Prioritizing CWE/SANS and OWASP Vulnerabilities: A Network-Based Model", International Journal of Computing and Digital Systems, 2021

Publication

<1%

19

Twana Assad Taha, Murat Karabatak. "A proposed approach for preventing cross-site scripting", 2018 6th International Symposium on Digital Forensic and Security (ISDFS), 2018

Publication

<1%

20

"M817 Block 1 week 3 fundamentals 1 cryptography WEB097764", Open University

Publication

<1%

21

Jai Narayan Goel, Mohsen Hallaj Asghar, Vivek Kumar, Sudhir Kumar Pandey. "Ensemble based approach to increase vulnerability assessment and penetration testing accuracy", 2016 International Conference on

<1%

Innovation and Challenges in Cyber Security (ICICCS-INBUSH), 2016

Publication

22

Gabriel Avramescu, Mihai Bucicoiu, Daniel Rosner, Nicolae Tapus. "Guidelines for Discovering and Improving Application Security", 2013 19th International Conference on Control Systems and Computer Science, 2013

Publication

<1 %

23

Pauli, Josh. "Web Application Exploitation with Broken Authentication and Path Traversal", The Basics of Web Hacking, 2013.

Publication

<1 %

24

"Emerging Research in Computing, Information, Communication and Applications", Springer Science and Business Media LLC, 2022

Publication

<1 %

25

Sanjib Sinha. "Bug Bounty Hunting for Web Security", Springer Science and Business Media LLC, 2019

Publication

<1 %

26

Bipin C. Desai, Arlin L. Kipling, Reethu Navale, Jainhu Zhu. "The web", Proceedings of the 24th Symposium on International Database Engineering & Applications, 2020

Publication

<1 %

27

"Attacking iOS Applications", The Mobile Application Hacker s Handbook, 2015.

Publication

<1%

28

Eric Conrad, Seth Misenar, Joshua Feldman. "Domain 3: Security Architecture and Engineering", Elsevier BV, 2023

Publication

<1%

29

Rytis Sileika. "Pro Python System Administration", Springer Science and Business Media LLC, 2010

Publication

<1%

Exclude quotes Off

Exclude bibliography On