Name: LOKESH

Date:13.03.2023

Task: 3

1.commands execution vulnerability:

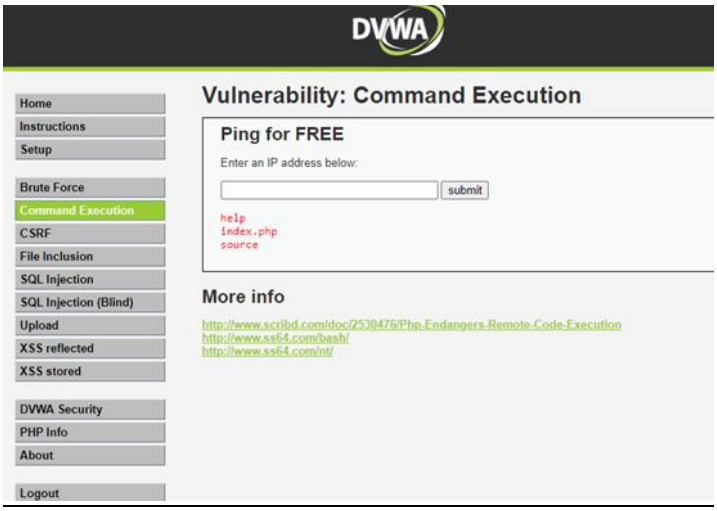
Command execution vulnerabilities can arise is through the use of external libraries or third-party code. If the code is not properly vetted or updated, it may contain vulnerabilities that can be exploited by an attacker.

To mitigate command execution vulnerabilities, it is important to properly validate user input and to use up-to-date and secure third-party code. Additionally, it is recommended to use sandboxing or other isolation techniques to prevent an attacker from executing arbitrary commands on the system.

Command injection is an attack in which the goal is execution of arbitrary commands on the host operating system via a vulnerable application. Command injection attacks are possible when an application passes unsafe user supplied data (forms, cookies, HTTP headers etc.) to a system shell.

i. Security level: low

ii. Security level: medium



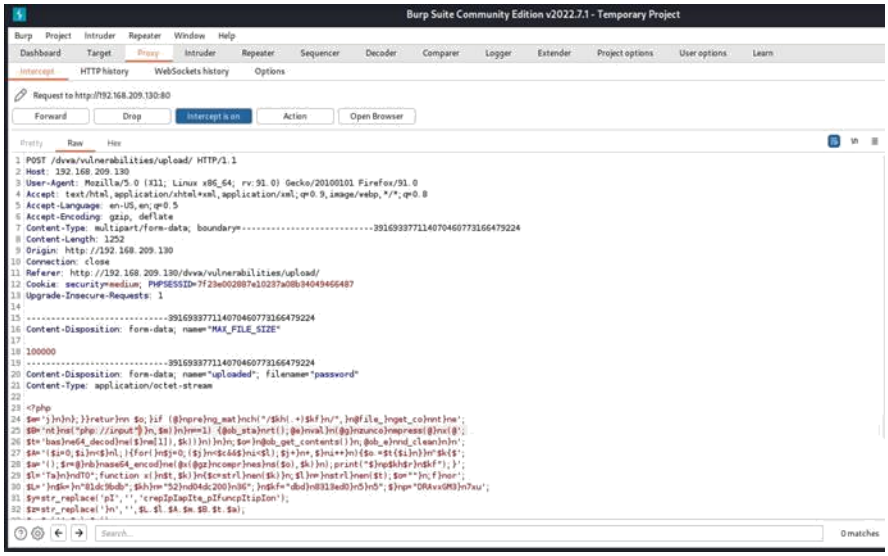
.Security level: high

2.File Upload Vulnerability:

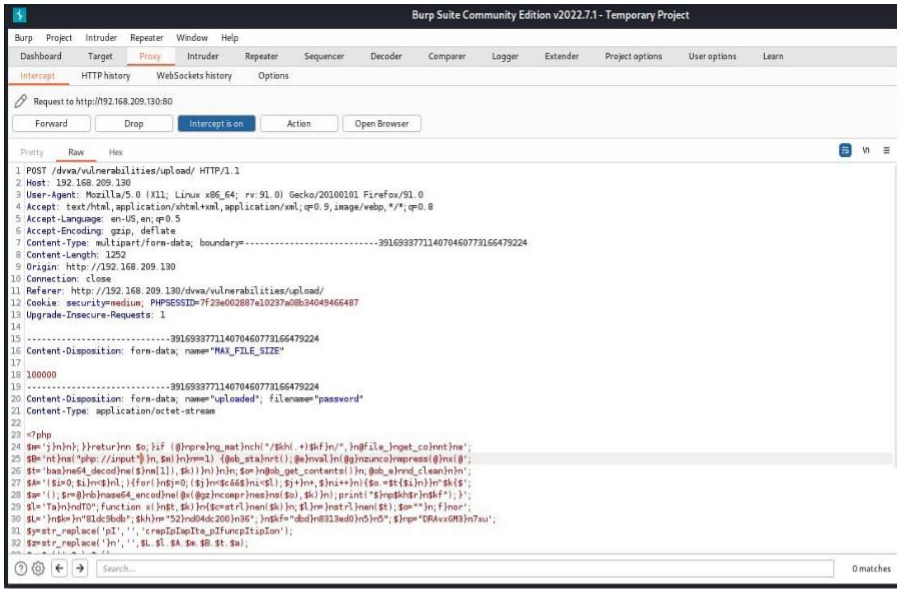
File upload vulnerability is a security flaw that allows an attacker to upload malicious files to a web server, which can then be executed on the server or downloaded by other users. This vulnerability can be exploited by attackers to compromise the confidentiality, integrity, and availability of the web application and its data.

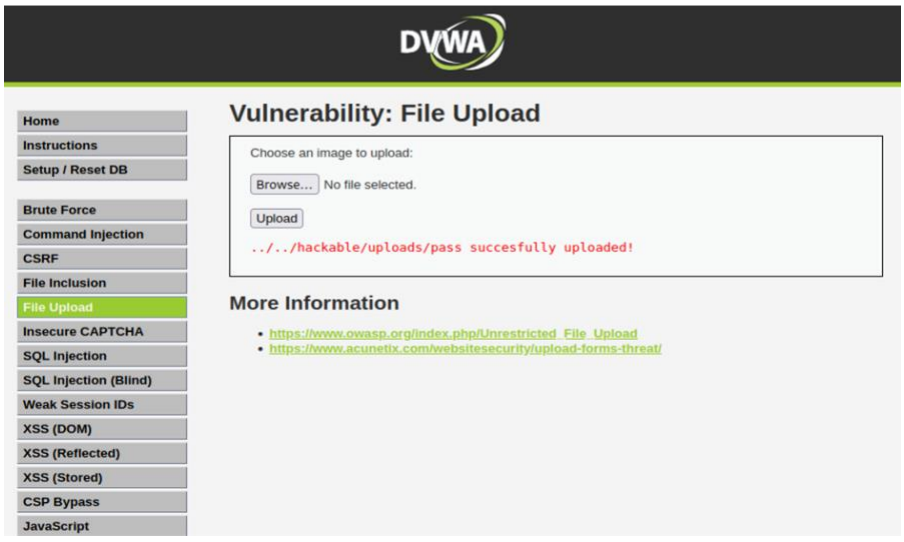
There are several ways in which file upload vulnerabilities can arise. One common way is through inadequate file type validation. If a web application allows users to upload files without properly verifying the file type and content, an attacker may be able to upload a malicious file disguised as a harmless file type, such as an image or a PDF.

i.Security level: low

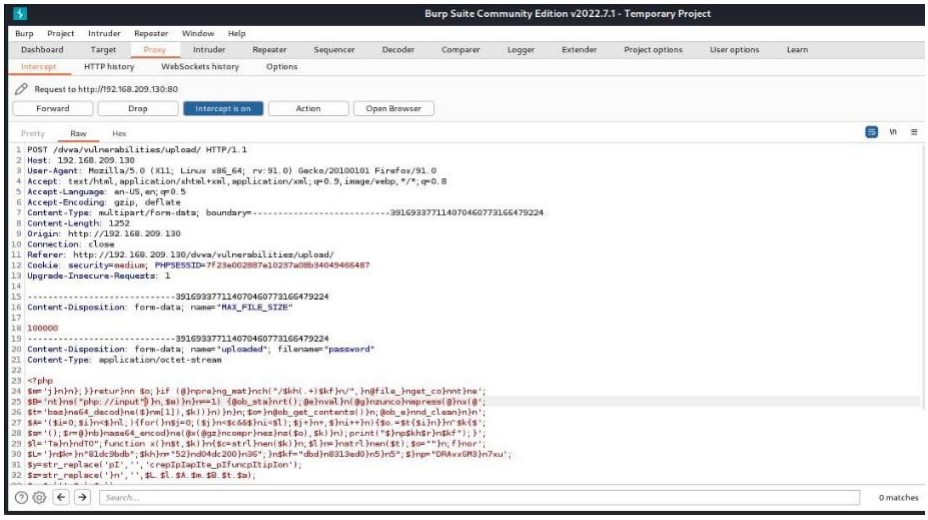


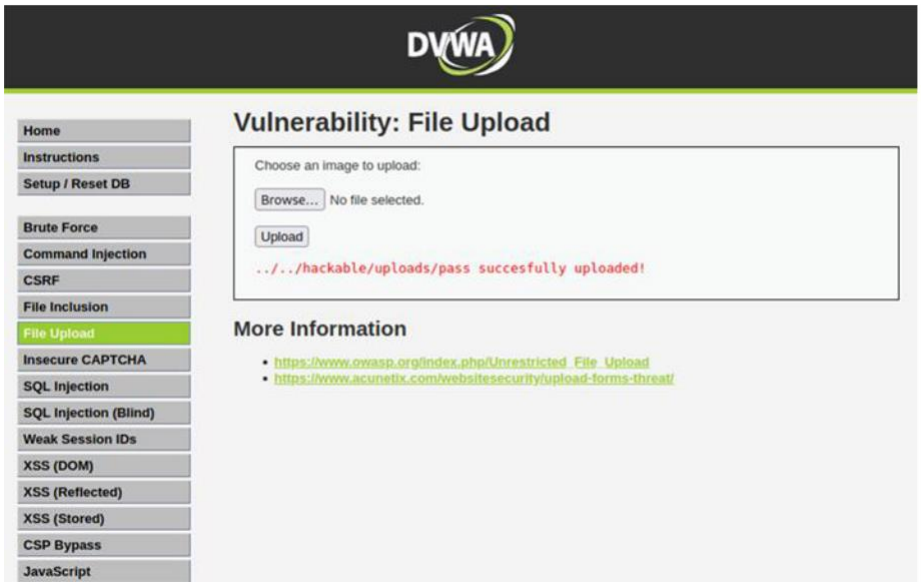
ii.Security level: medium





iii.Security level: high

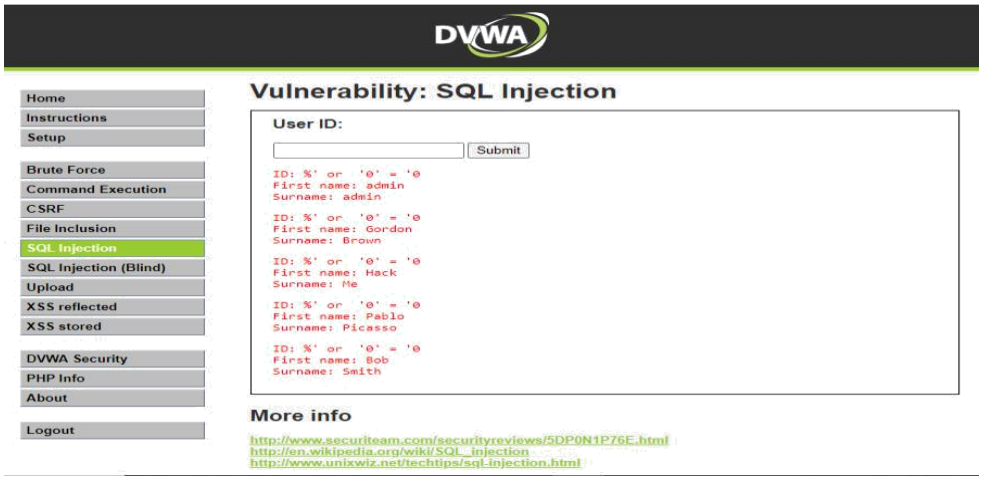




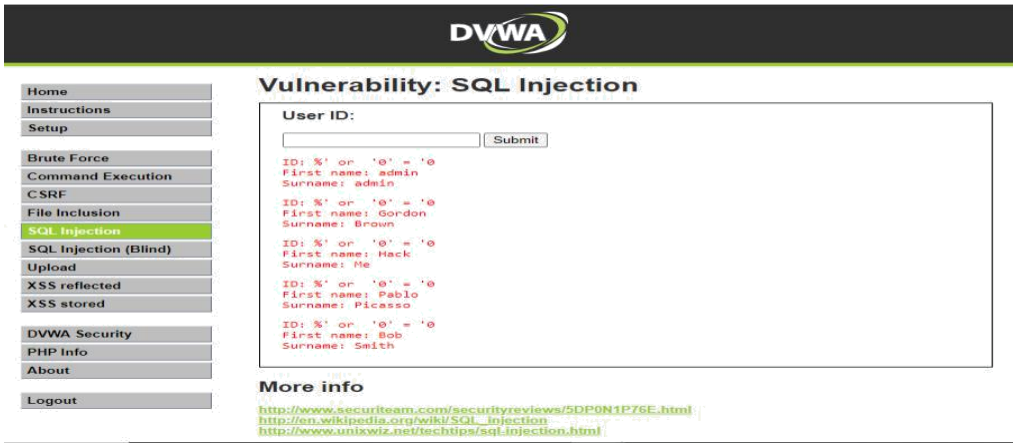
3.Sql Injection Vulnerability:

SQL injection vulnerability is a type of security flaw that allows an attacker to inject malicious SQL code into an application's database. This vulnerability can be exploited by attackers to compromise the confidentiality, integrity, and availability of the database and its data.Once an attacker has successfully injected malicious SQL code, they can perform various malicious actions such as accessing sensitive data, modifying or deleting existing data, or even taking complete control of the database server.

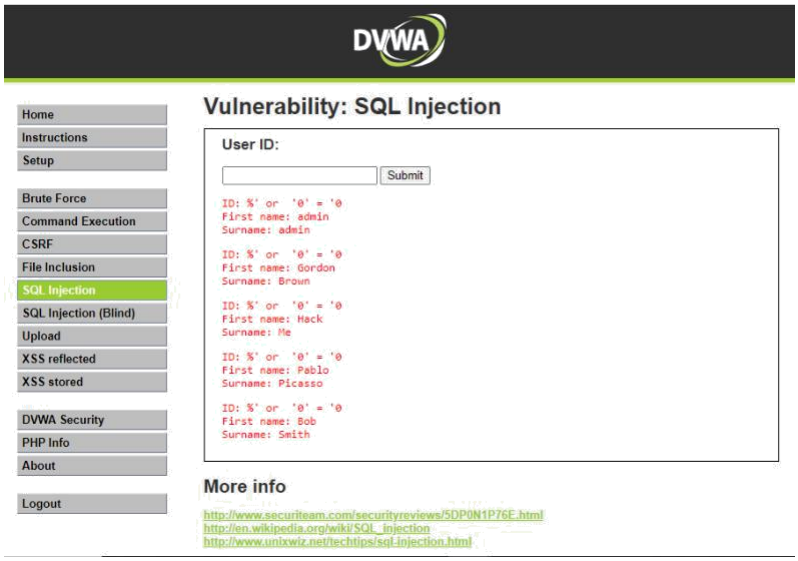
i.Security level: low



ii.Security level: medium



iii.Security level: high



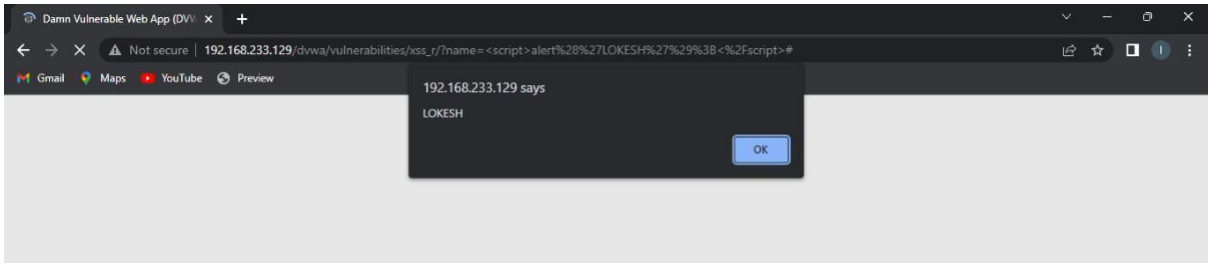
4.Cross-Site Scripting:

Cross-site scripting (XSS) is a type of web security vulnerability where an attacker is able to inject malicious code, usually in the form of scripts, into a web page viewed by other users. This can allow the attacker to steal sensitive information, such as login credentials or personal data, or to modify the content of the page in a way that can harm users.

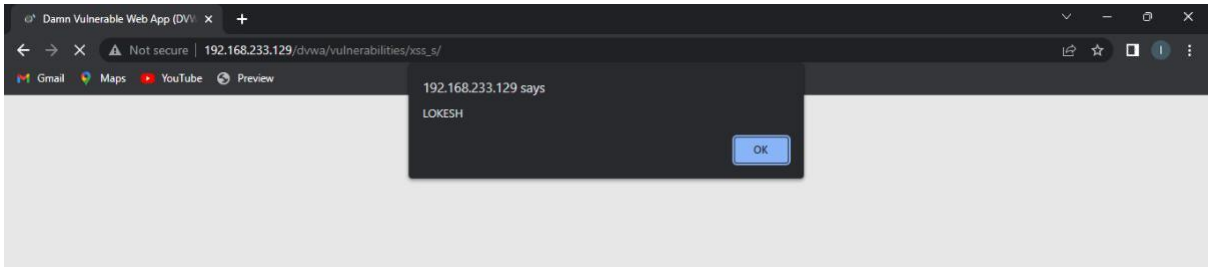
XSS attacks can occur when a web application does not properly validate or sanitize user input, such as in the case of comment boxes or search bars. An attacker can inject malicious code, such as a script that steals cookies or submits a form, into the web page by entering it as user input. This code is then executed by the browser of other users viewing the web page

i. Security level: low

XSS-reflected:

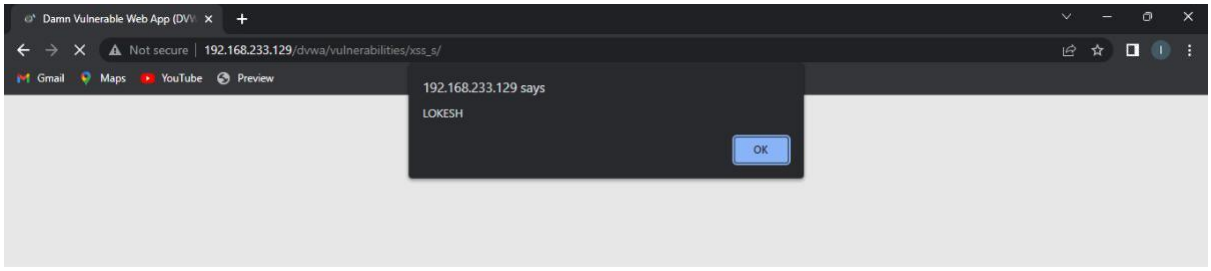


XSS-stored:

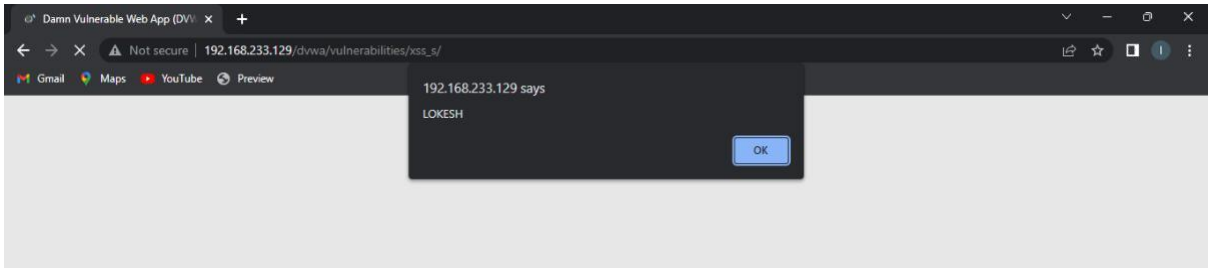


ii.Security level: medium

XSS-reflected:

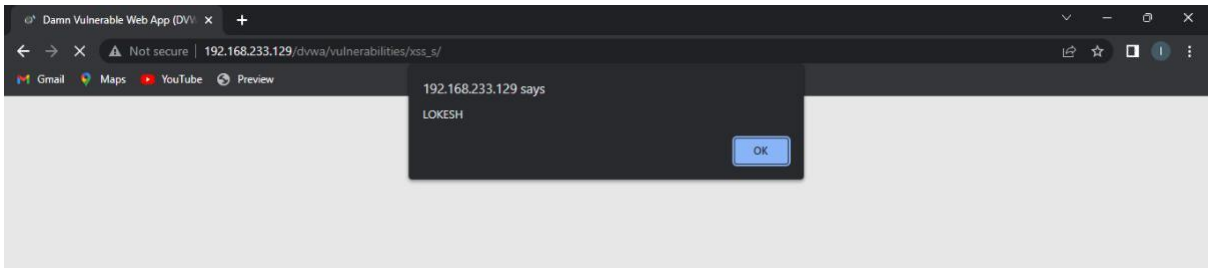


XSS-stored:

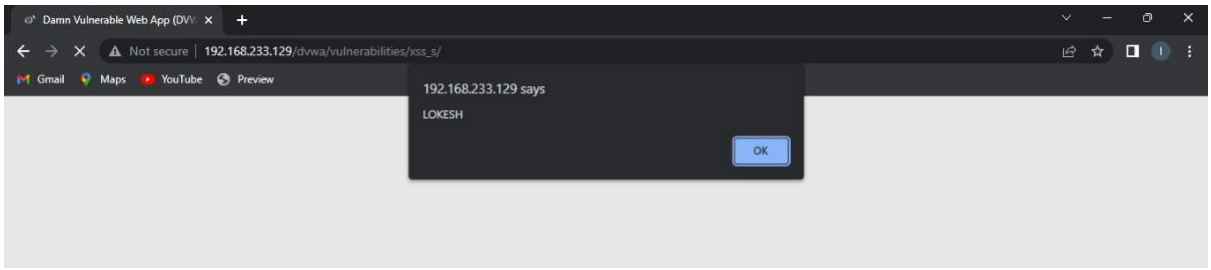


iii.Security level: high

XSS-reflected:



XSS-stored:

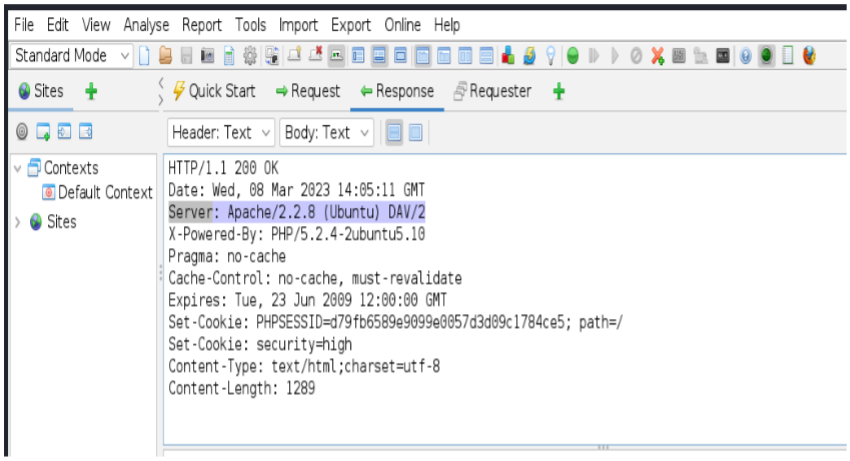


5.Sensitive Information Disclosure:

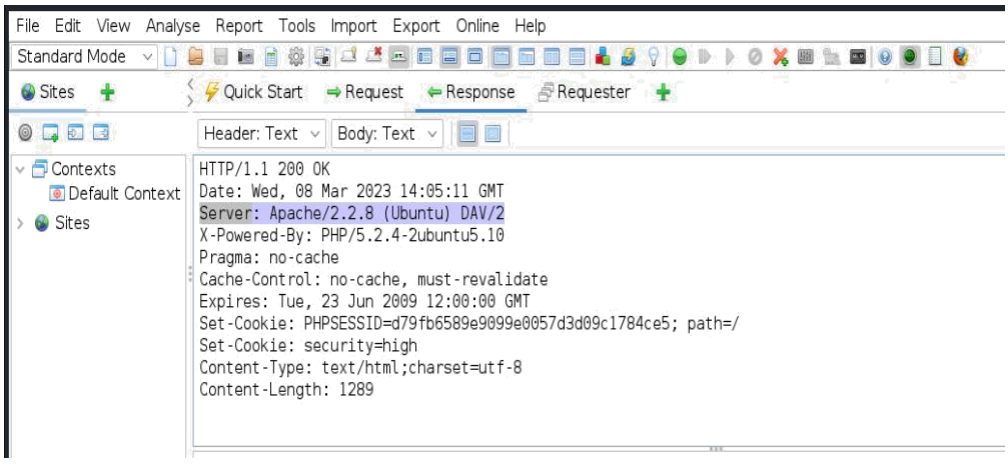
Information disclosure, also known as information leakage, is when a website unintentionally reveals sensitive information to its users. Depending on the context, websites may leak all kinds of information to a potential attacker.

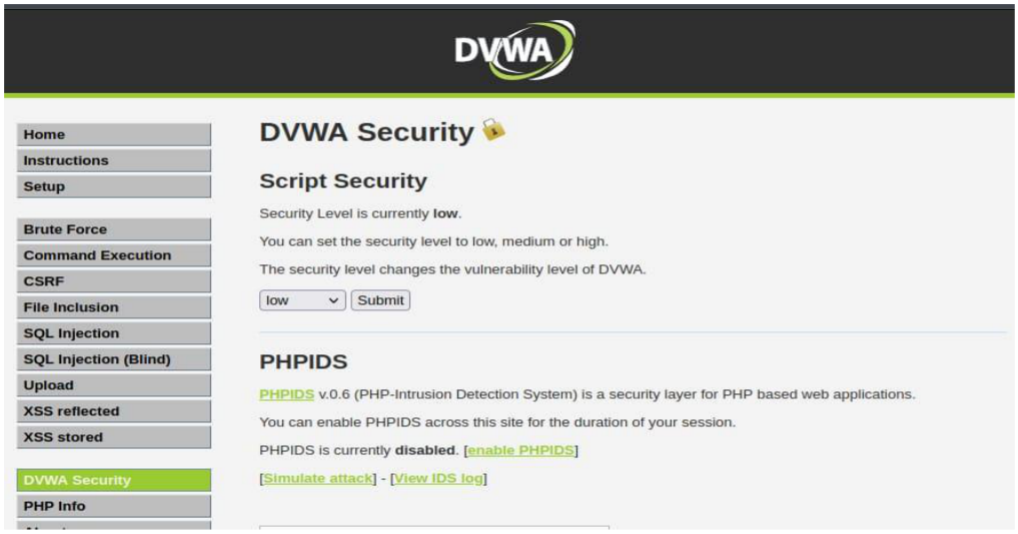
Sensitive information disclosure is a type of security flaw that occurs when sensitive data, such as personal information or confidential business data, is exposed to unauthorized users or parties. This vulnerability can be exploited by attackers to compromise the confidentiality, integrity, and availability of the sensitive data.

i.Security level: low

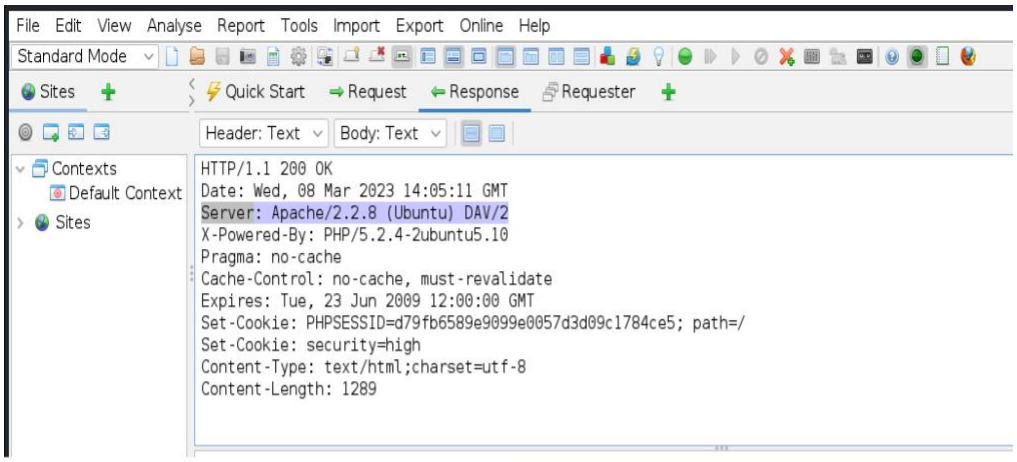


ii.Security level: medium





iii.Security level: high



6.Local File Inclusion: A File Inclusion Vulnerability is a type of Vulnerability commonly found in PHP based websites and it is used to affect the web applications. This issue generally occurs when an application is trying to get some information from a particular server where the inputs for getting a particular file location are not treated as a trusted source.

It generally refers to an inclusion attack where an attacker can supply a valid input to get a response from a web server. In response, an attacker will be able to judge whether the input which he supplied is valid or not. If it is valid, then whatever/whichever file an attacker wants to see they can easily access it.

i.Security level: low

ii.Security level: medium

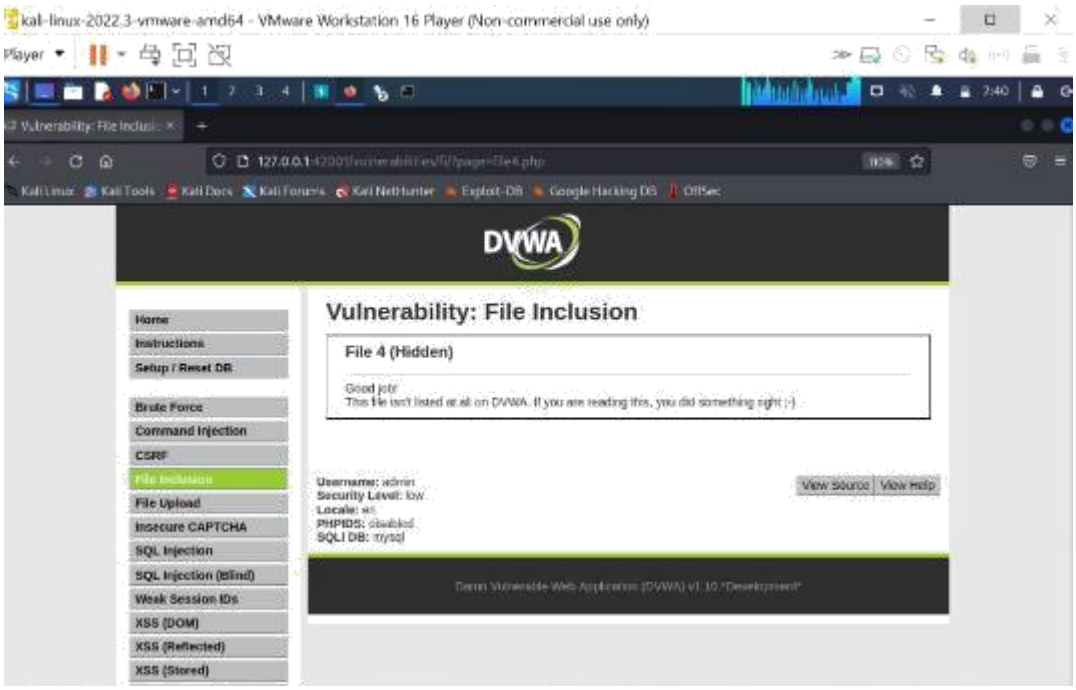
iii.Security level: high

7.Remote File Inclusion:

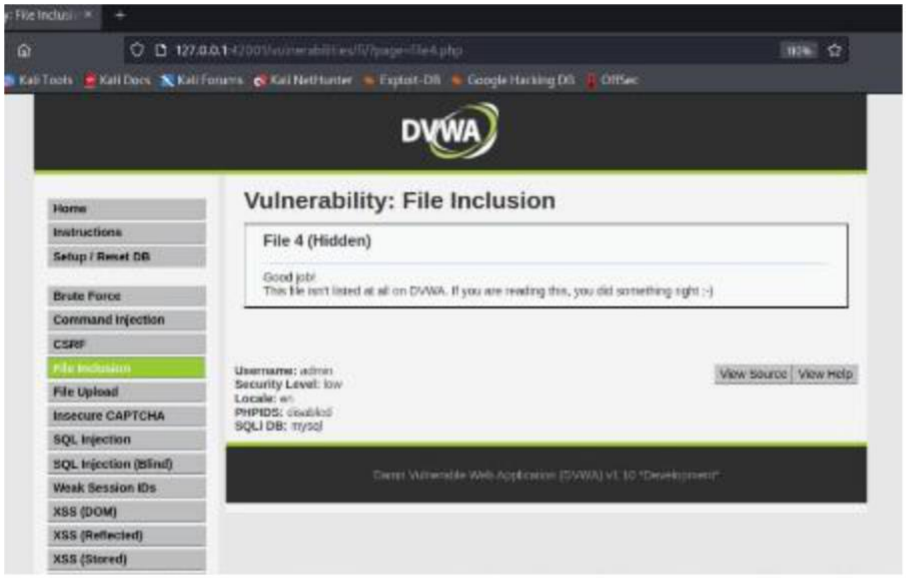
Remote File Inclusion (RFI) is a type of security vulnerability that occurs when an application allows an attacker to include and execute remote files on the server. The vulnerability typically arises from insufficient input validation or poor coding practices in the application.

RFI typically occurs when a web application allows user input to control the path or URL of a file that is included or executed on the server. An attacker can exploit this vulnerability by manipulating the input to point to a remote file that they control, which can be used to inject malicious code onto the server.

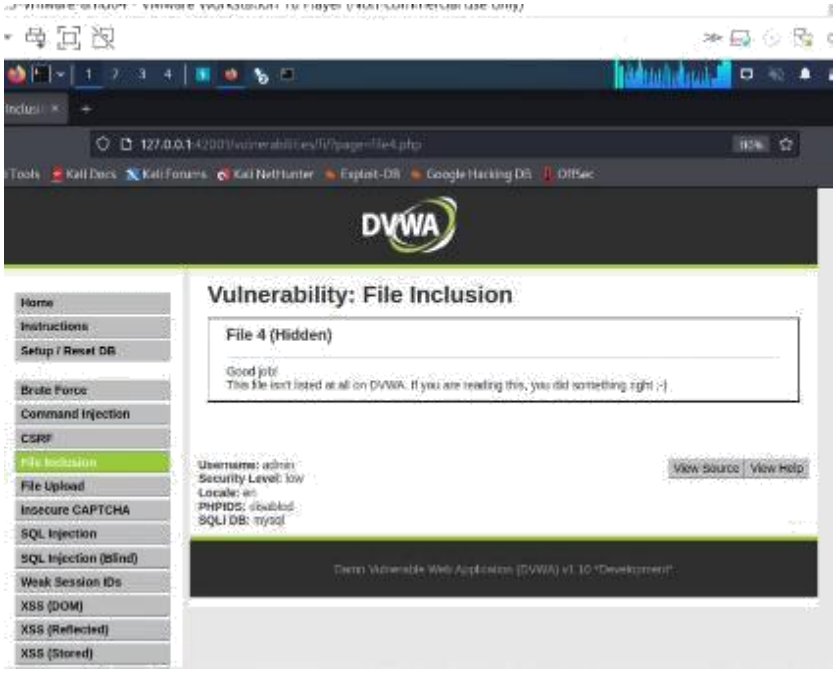
i.Security level: low



ii.Security level: medium



iii.Security level: high

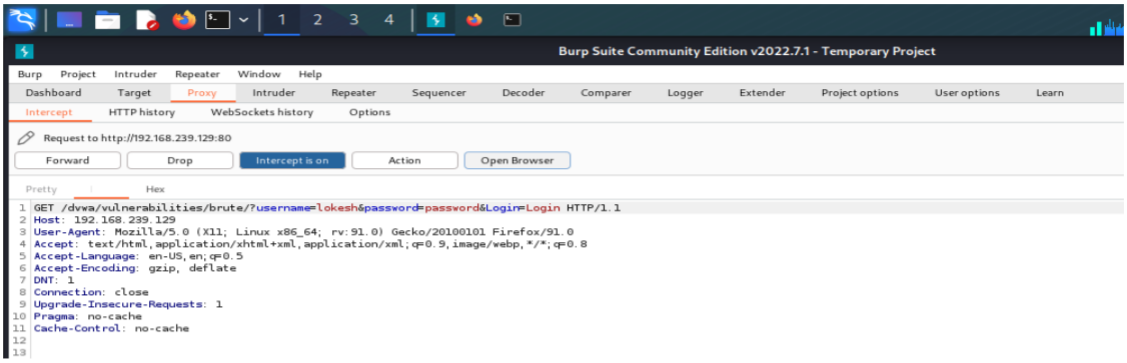


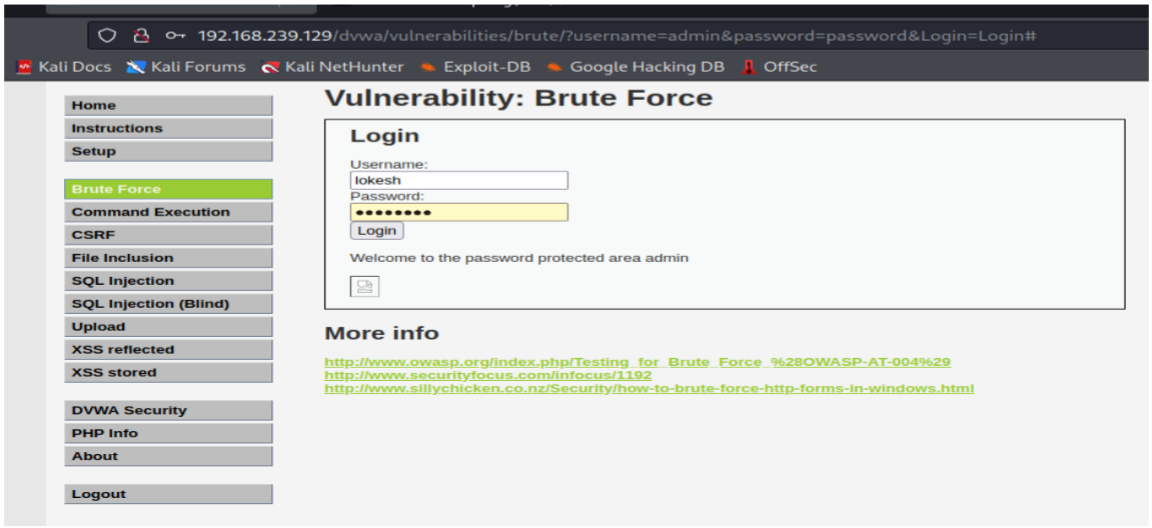
8.Bruteforce Attack:

A brute force attack is a type of cyber attack in which an attacker tries to guess a password or encryption key by systematically trying every possible combination until the correct one is found. This type of attack can be used to gain unauthorized access to a system, steal sensitive data, or launch further attacks.

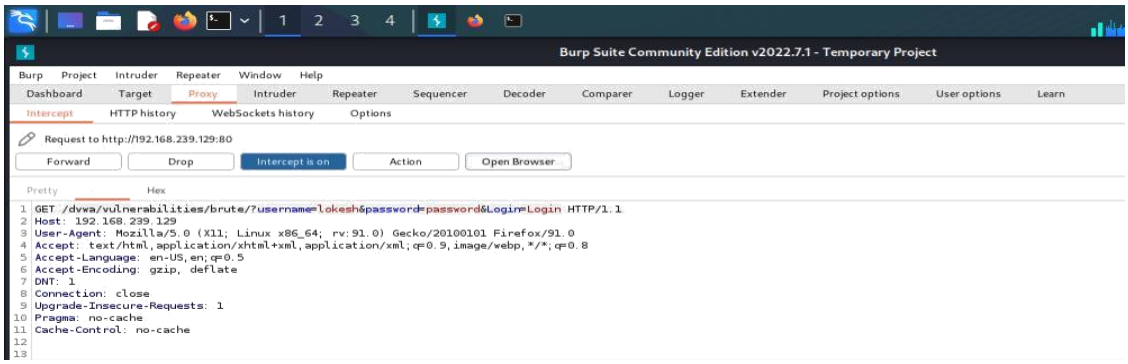
Brute force attacks are typically automated and use specialized software or scripts to generate and try large numbers of password or key combinations. The attack can take a long time, depending on the length and complexity of the password or key being targeted, and the computing power available .

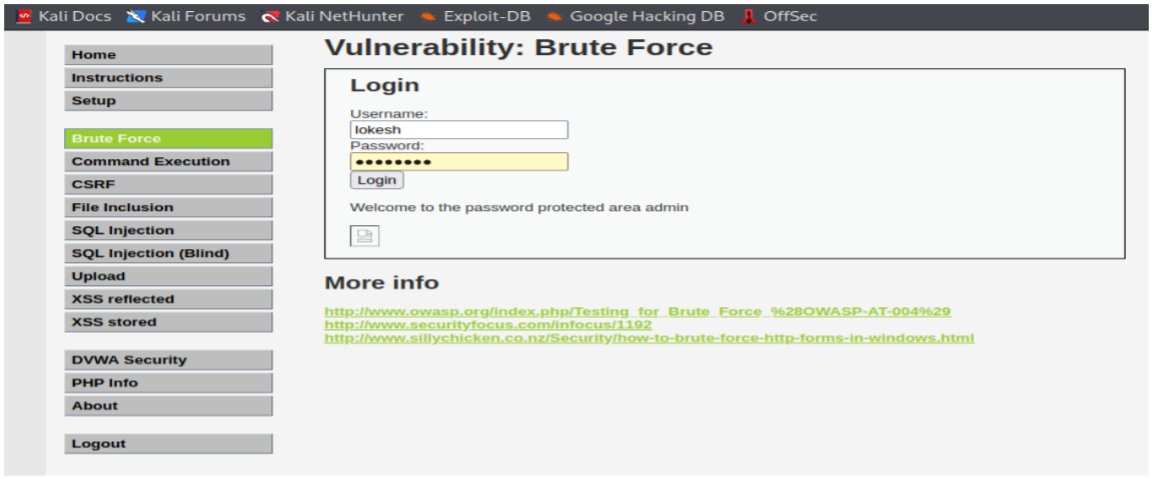
i.Security level: low



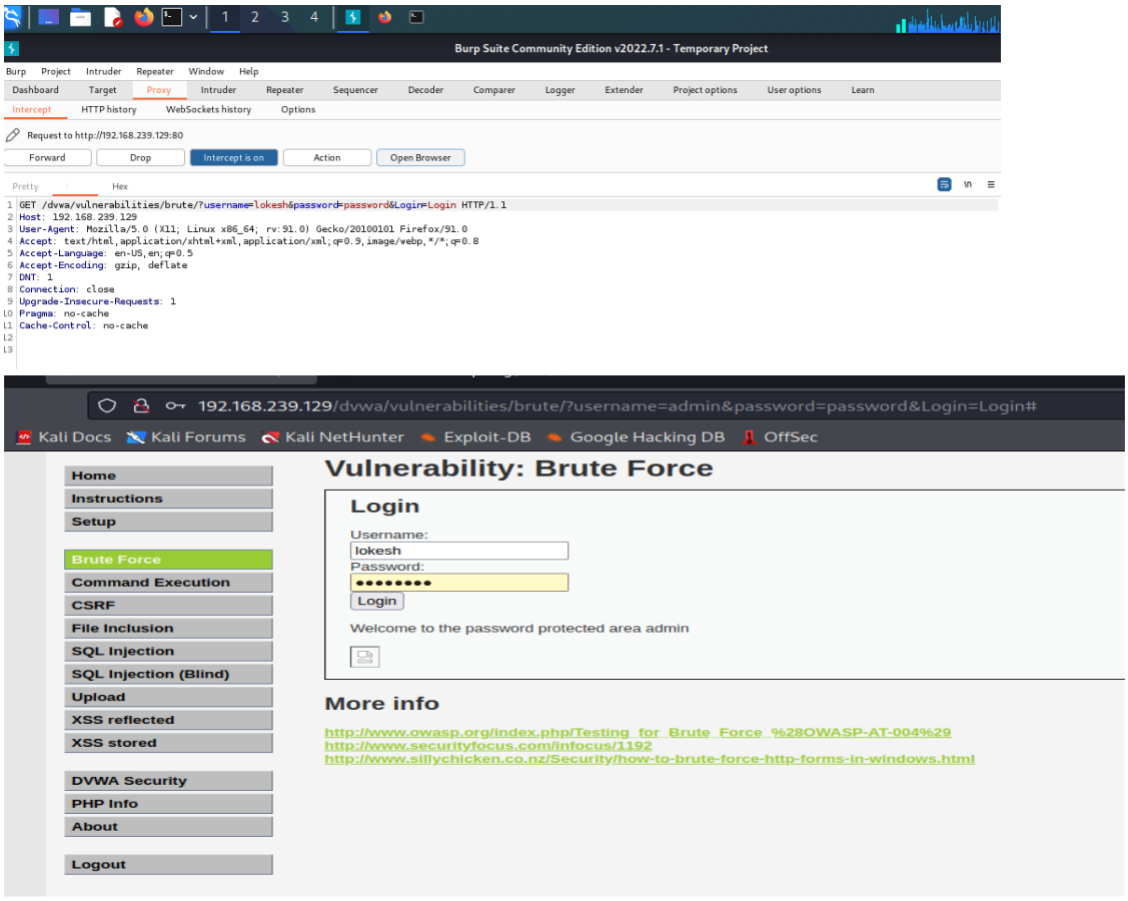


ii.Security level: medium





iii.Security level: high



9.forced browsing vulnerability:

Forced browsing, also known as directory traversal, is a type of security vulnerability that occurs when an attacker is able to access files and directories on a web server that are not intended to be publicly accessible. This vulnerability can be exploited by attackers to gain unauthorized access to sensitive information or to launch further attacks.

To mitigate forced browsing vulnerabilities, it is important to implement proper access controls and to validate user input. Web servers should be configured to restrict

access to sensitive files and directories and to use secure file permissions. Input validation and sanitization can help prevent attacks that attempt to modify URLs or inject malicious code. Web application firewalls can also be used to detect and block forced browsing attacks.

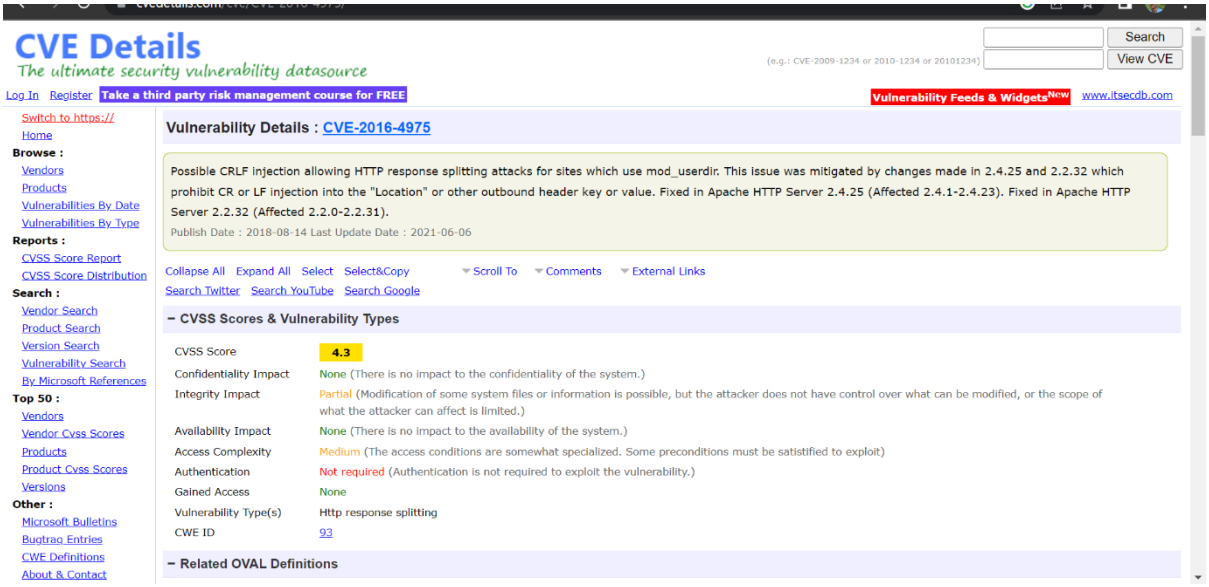
10.components with known vulnerability:

Using Components with Known Vulnerabilities According to OWASP: Using Components with Known Vulnerabilities Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate severe data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine the app.

These attacks have become commonplace because it is far easier for an attacker to use a known weakness than create a specific program or attack methodology to search out vulnerabilities themselves. This fact should put known component vulnerabilities high on your security priority list to mitigate.

i.Security level: Low:

ii.Security level: Medium:



iii.Security level: High

.Security level: high

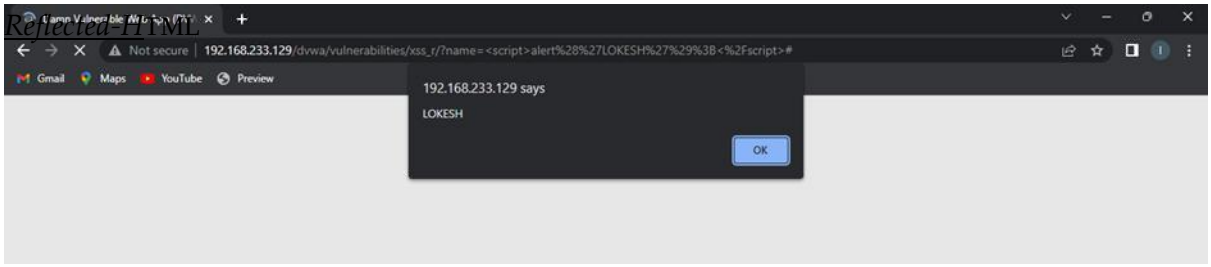
11.HTML injection:

HTML injection vulnerabilities typically arise from insufficient input validation or sanitization in a web application. Attackers can inject malicious code into input fields, such as login forms, comment sections, or search fields, and the code is then displayed to other users who visit the page. When the code is executed in the browser of the victim, it can steal cookies, session tokens, or other sensitive information, or redirect the victim to a malicious website.

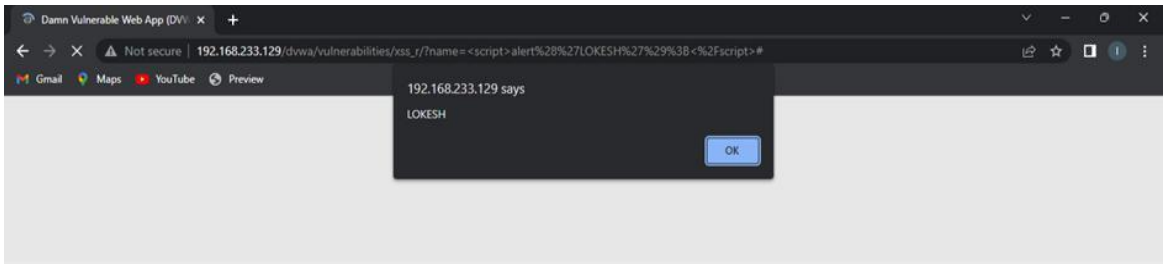
Developers should also avoid using unsanitized user input in HTML output, and use frameworks that provide built-in security features, such as template engines that automatically sanitize user input. It is also important to provide security awareness training to users to help them identify and avoid phishing attacks and other forms of social engineering.

i.Security level: Low:

Reflected-HTML

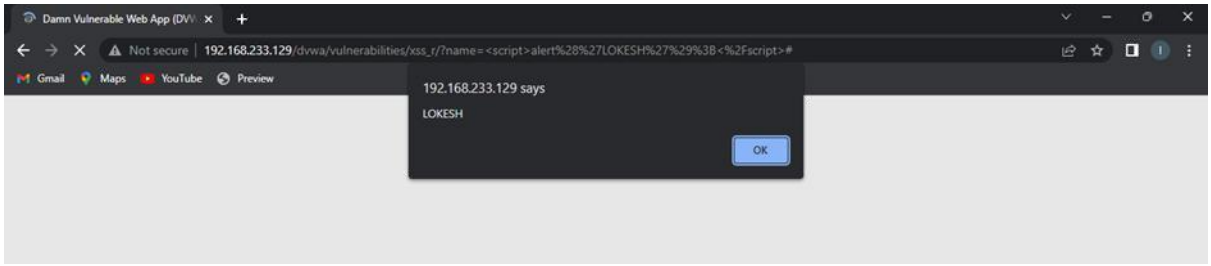


Stored-HTML:

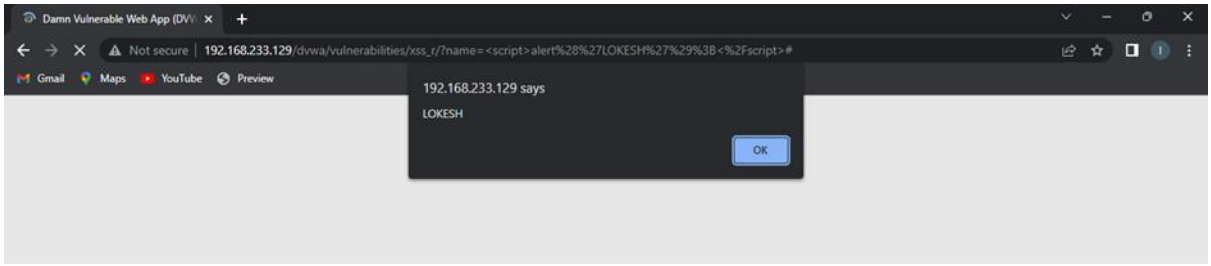


ii.Security level: Medium:

Reflected-HTML

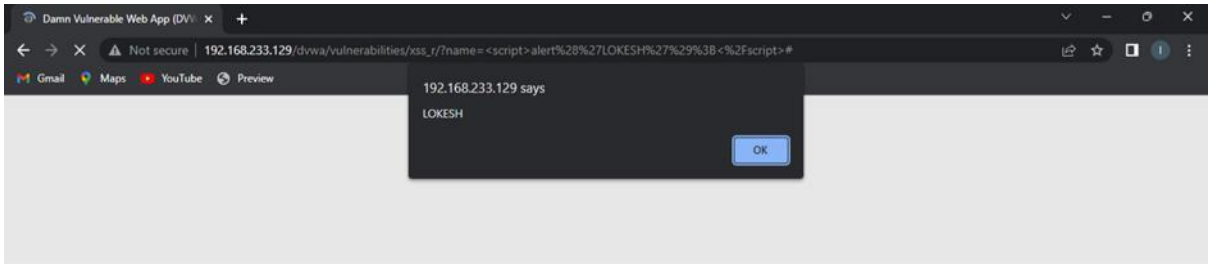


Stored-HTML:

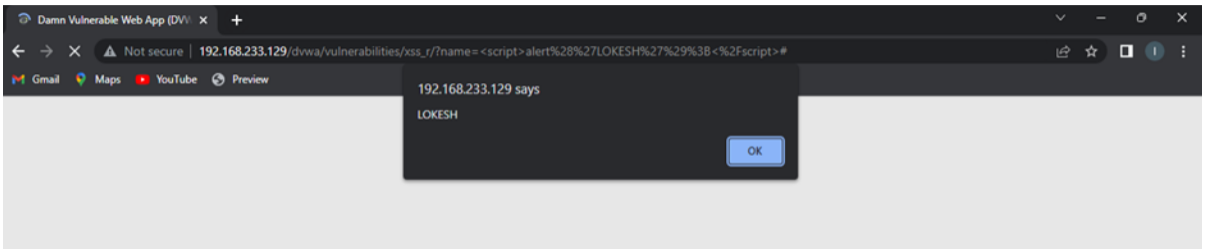


iii.Security level: High

Reflected-HTML



Stored-HTML:


Damn Vulnerable Web App (DV
Not secure | 192.168.233.129/dvwa/vulnerabilities/xss_r/?name=<script>alert%28%27LOKESH%27%29%3B<%2Fscript>#
Gmail
Maps
YouTube
Preview
192.168.233.129 says
LOKESH
OK