Web Application Penetration Testing Report For

[Client_Name]

Ritesh Kumar

riteshraj1405@gmail.com

August 15,2023

Document Details

Title	Details
Website Url:	DVWA
COMPLETED ON:	13 january 2024
REPORT TYPE:	Website penetration testing
Reported By :	Ritesh Kumar
Contact Information:	Riteshraj1405@gmail.com

Executive Summary

This document contains the initial security assessment report for : DVWA

The purpose of this penetration testing effort was to assess the security posture of the website and identify potential weaknesses that could be exploited by malicious actors. The tests were carried out assuming the identity of an attacker or a malicious user but no harm was made to the functionality or working of the application/network.

Assessment Overview and Recommendations

During the course of testing, I uncovered different critical Vulnerabilities material risk to **DVWA** information systems. It includes:

SQL Injection (High Severity): An SQL injection vulnerability was discovered in the website's login page. An attacker could exploit this vulnerability to gain unauthorized access to the database, potentially compromising user data.

Cross-Site Scripting (XSS) (High Severity): Multiple instances of reflected and stored XSS vulnerabilities were found in the user comments section of the website. These vulnerabilities could allow an attacker to inject malicious scripts into the web pages, affecting other users.

Sensitive Data Exposure: Some sensitive data was found to be inadequately protected, increasing the risk of data leaks.

Insecure Authentication: Weak password policies and inadequate session management practices were identified, which could lead to unauthorized access and account compromise.

Outdated Software Components: Several third-party libraries and software components were outdated, making the website susceptible to known vulnerabilities.

Table of Contents

7.conclusion

8. Acknowledgement

1. Introduction

The aim of this web penetration test is to help the technical personnel of the company to make the website more secure. Although this report contains technical terms, a non-technical explanation of the content – which can be found in the appendices – is given along with the test report, for the technical personnel and security consultant to review. This also makes it possible for them to reproduce the tests

DVWA stands for "Damn Vulnerable Web Application." It is a web application intentionally created to be vulnerable to various security issues. DVWA is used for educational and training purposes, allowing security professionals, developers, and students to practice and improve their skills in identifying and mitigating web application vulnerabilities.

DVWA is designed with a range of security vulnerabilities, including but not limited to SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF), file inclusion vulnerabilities, and more. Users can explore and exploit these vulnerabilities in a controlled environment to better understand common web application security issues and learn how to secure applications against such threats.

It's important to note that DVWA should be used responsibly and only in environments where ethical hacking and security training are authorized. Unauthorized use or deploying DVWA on public-facing servers without proper authorization can lead to security risks and legal consequences.

Purpose

Bug Identification: The primary purpose of Penetration testing is to identify and document issues or bugs within a software application or system. These issues can range from functional glitches and usability problems to security vulnerabilities.

Quality Improvement: It helps improve the overall quality of a software product or website by identifying and resolving issues that impact user experience, reliability, and performance.

Security Enhancement: It especially in the context of security testing, aims to uncover vulnerabilities that could potentially be exploited by malicious actors. Identifying and fixing these vulnerabilities strengthens the system's security.

Cost Reduction: By identifying and fixing bugs early in the development lifecycle or before they are exploited in the wild, organizations can reduce the cost and effort associated with later-stage bug fixes, customer support, and potential legal or financial repercussions.

Enhanced User Experience: Ensuring that a software product is free from bugs and issues contributes to a positive user

experience, which can lead to higher user satisfaction and increased user retention.

2. Scope & Approach

A security review was conducted (web application penetration test) of the dvwa web application.

- **gain unauthorized access to the application, systems, individual functions or critical platform components,**
- **read or falsify data without authorization**
- **impair the availability of the application, systems or data**
- **bring the application or systems into an undefined state**

To do so, we have tested the application for the following issues (OWASP top 10 aligned):

I. **SQL Injection**
II. **Broken Authentication**
III. **Sensitive Data Exposure**
IV. **XML External Entities (XXL)**
V. **Broken Access Control**
VI. **Security Misconfiguration**
VII. **Cross-Site Scripting (XSS)**
VIII. **Insecure Deserialization**
IX. **Using Components with Known Vulnerabilities**
X. **Insufficient Logging and Monitoring**

The web penetration test was conducted as a "Grey-Box", implying that the security tested was given prior information about the target applications sample

This was done to simulate as closely as possible the viewpoint of a completely external hacker. We tried to penetrate the website, specifically focusing on transactions and events happening in the application's front-end and back-end. We conducted the tests against industry best practices like the Open Web Application Security Project (OWASP) and the tests were generated and executed based on our vast experience in the field of Web Application Security.

This approach can be summarized as follows:

- Perform broad scans on source gained during “man in the middle” captures to identify discouraged coding practices and points that would increase compilation and program execution overhead.
- Perform targeted code injection at identified break points in code to simulate an attack.
- Identify hard-coded cryptographic hashes, usernames and passwords that would be an easy give-away to an attacker to obtain internal information.
- Obtain database connection configurations, informative data such as hard-coded IP addresses and port numbers.
- Obtain developer comments that could give out too much information concerning the source code and algorithms to an attacker.
- Provide threat ranking of the identified risks based on their level of criticality (i.e. low, medium and high).
- Supply recommendations to enhance security.

3. **Methodology**

Penetration testers often use different methods and tools to comprehensively assess the security of a target system. It's important to note that penetration testing should only be conducted on systems with proper authorization to avoid legal and ethical issues.
Penetration testing, also known as ethical hacking, involves simulating cyberattacks on a system, network, or application to identify vulnerabilities and weaknesses before malicious actors can exploit them. Here are some common methods and tools used in penetration testing:

Methods:

1. Black-Box Testing: In this method, the tester has no prior knowledge of the system's architecture, code, or infrastructure. It simulates an external hacker's perspective.

2. White-Box Testing: Testers have full knowledge of the system, including source code, architecture, and infrastructure. This method helps identify vulnerabilities from an insider's perspective.

3. Gray-Box Testing: Testers have partial knowledge of the system, often limited to high-level details. This approach

balances the realism of black-box testing with the depth of white-box testing.

4. Internal Testing: Testers work from within the organization's network, simulating attacks that could occur from a compromised internal user.

5. External Testing: Testers operate from outside the organization's network, imitating external threats like cybercriminals.

6. Social Engineering: This method involves manipulating people, often through deceptive means, to gain access to systems or information. Common techniques include phishing, pretexting, and baiting.

Tools:

1. Nmap, Zenmap : A powerful network scanner used to discover open ports, services, and vulnerabilities on networked systems.

2. Metasploit: An exploitation framework that assists testers in finding, exploiting, and validating vulnerabilities.

3. Wireshark: A network protocol analyser for capturing and analysing network traffic, which can be used to identify

security issues.

4. Burp Suite: A web vulnerability scanner and proxy tool for identifying and testing web application vulnerabilities, such as SQL injection and cross-site scripting (XSS).

5. Nessus: A widely-used vulnerability scanner that helps identify vulnerabilities in network infrastructure, servers, and applications.

6. OWASP ZAP (Zed Attack Proxy): An open-source web application security scanner used for finding vulnerabilities in web applications and APIs.

7. Netcat (nc): A versatile networking utility often used for banner grabbing, port scanning, and creating reverse shells.

8. Robotex DNS Lookup :Shows comprehensive information about the target website.

9. Hydra: A fast and flexible password-cracking tool that supports numerous protocols, including HTTP, SSH, FTP, and more.

10. SQLMap: A tool for detecting and exploiting SQL injection vulnerabilities in web applications.

11. DirBuster: A tool for brute-forcing directories and files on

web servers, useful for finding hidden resources.

12. BeEF (Browser Exploitation Framework): A tool for testing and exploiting web browser vulnerabilities

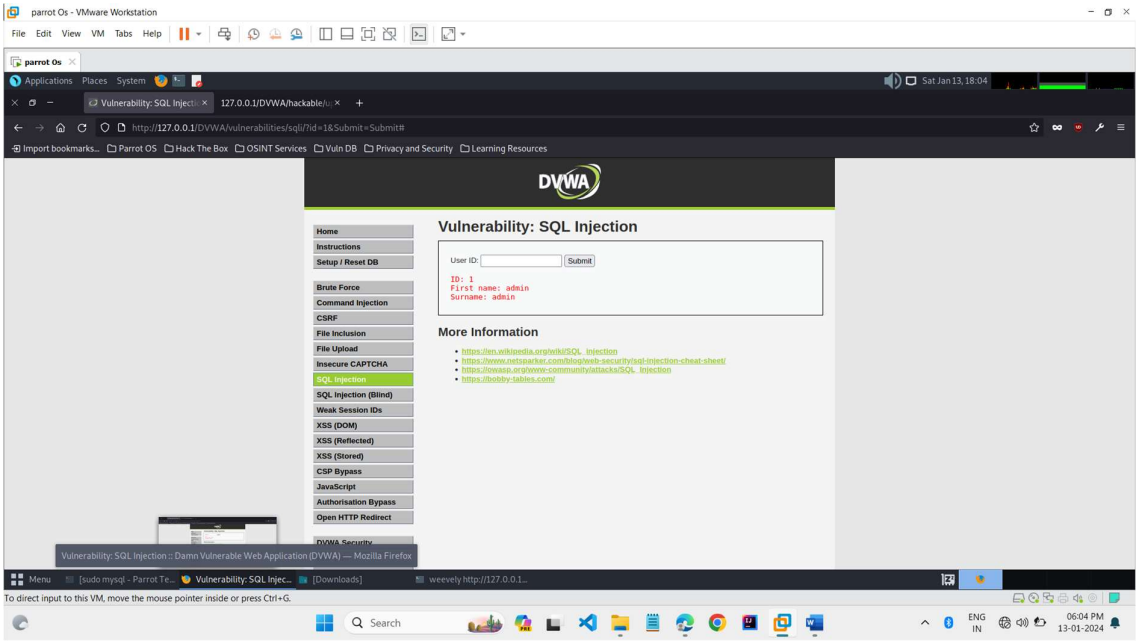
13. Maltego()

14. Whois Lookup (Find information about owner of the target)

15. Netcraft site Report (Show technology used on the target)

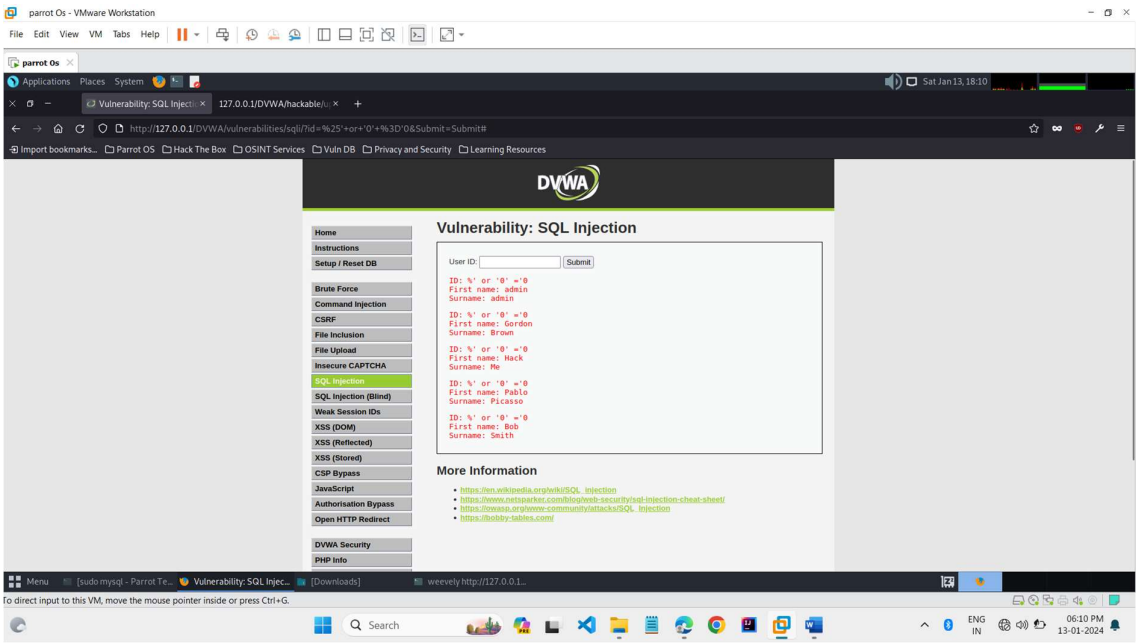
I. SQL Injection

SQL Injection is a technique used by attackers to fetch the database related information by inserting sql query in the input field. There are various set of queries that can be utilized to retrieve the system data but somehow, in this case burp suite will be used to intercept the simple query and its output will be used in the terminal by using the sqlmap to fetch the database information as much as possible.

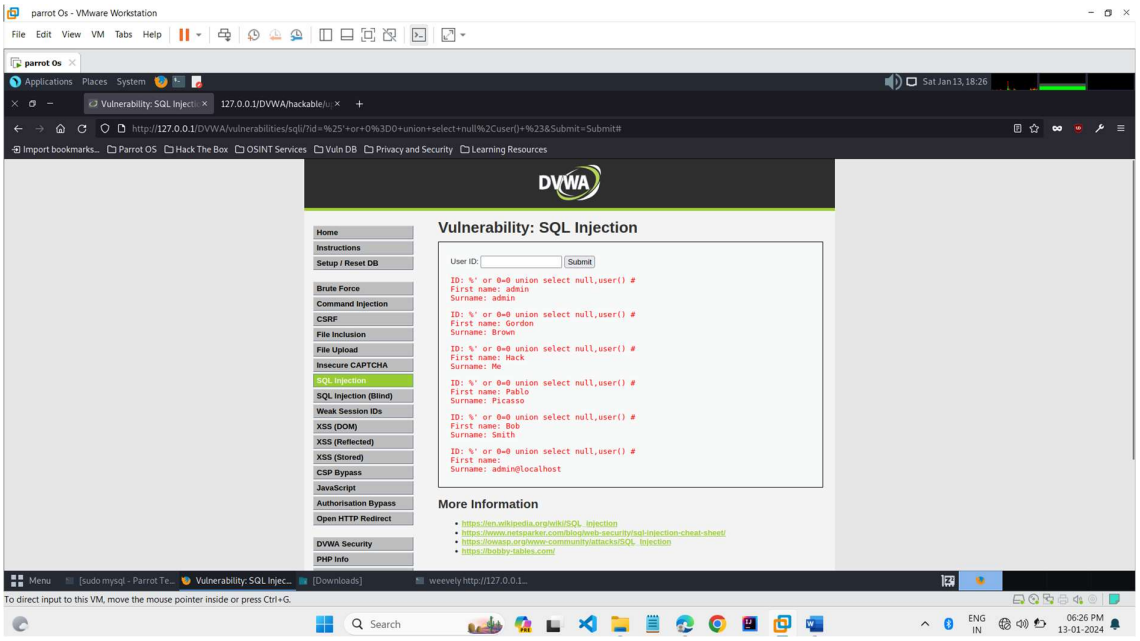


select First name ,Surname from users where ID='%' or '0' ='0';

%' or '0' ='0



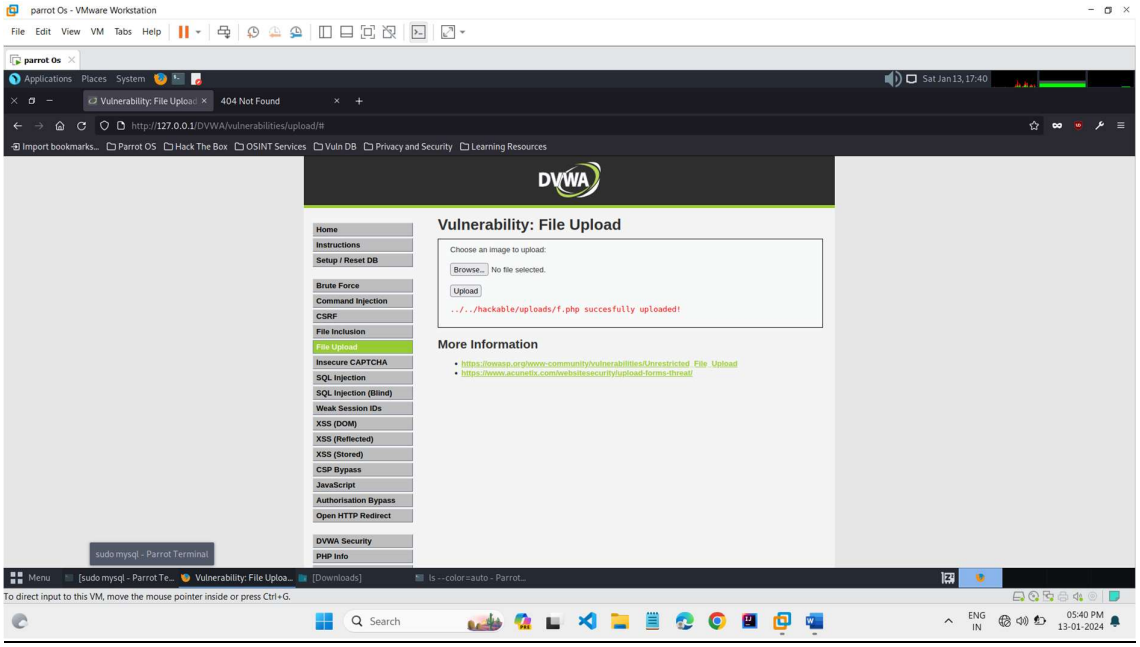
%' or 0=0 union select null,user() #

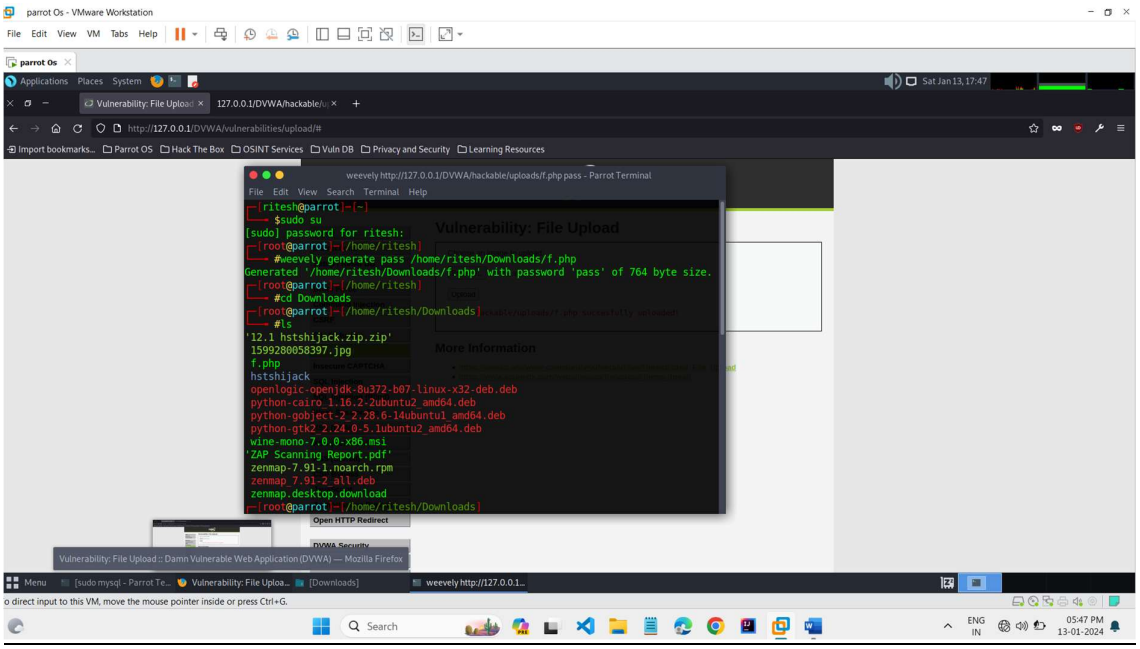


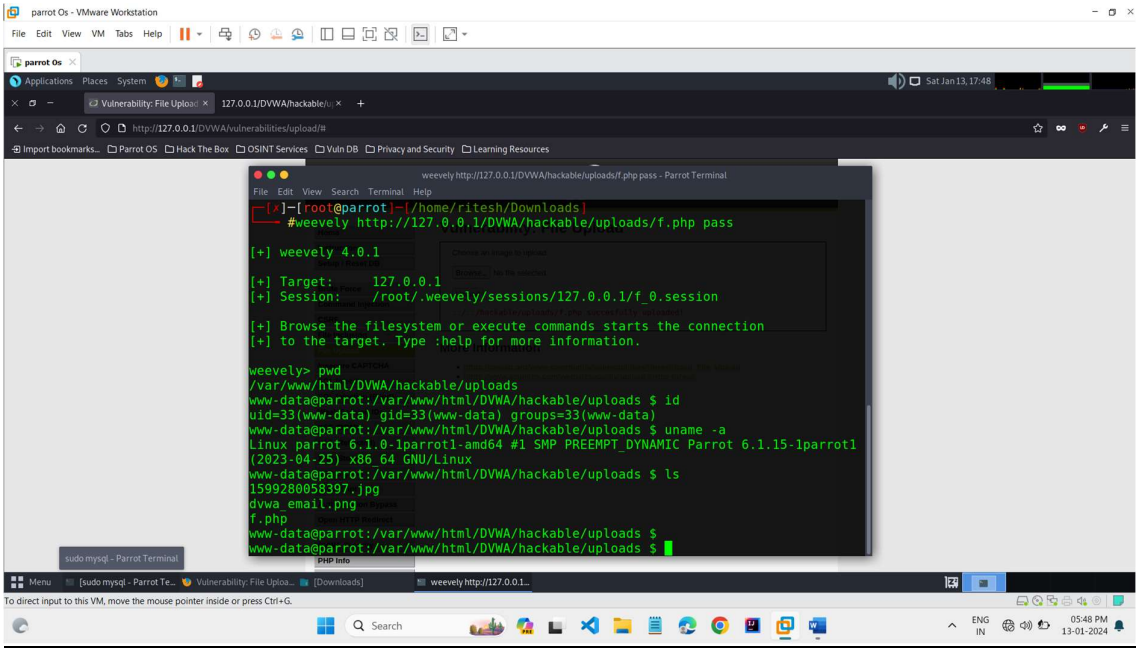
II) File Upload Vulnerabilities

File upload vulnerabilities are security issues that arise when a web application improperly validates or controls the uploading of files. Exploiting these vulnerabilities can lead to serious consequences, including unauthorized access, execution of arbitrary code, and compromise of the application or server

file upload vulnerabilities pose significant security risks, and their exploitation can lead to severe consequences. Web developers should implement strong file upload validation mechanisms to mitigate such vulnerabilities.


parrot Os - VMware Workstation
File Edit View VM Tabs Help
parrot Os
Applications Places System
Sat Jan 13, 17:40
Vulnerability: File Upload
404 Not Found
http://127.0.0.1/DVWA/vulnerabilities/upload/#
Import bookmarks... Parrot OS Hack The Box OSINT Services Vuln DB Privacy and Security Learning Resources
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Authorisation Bypass
Open HTTP Redirect
DVWA Security
PHP Info
Vulnerability: File Upload
Choose an image to upload:
Browse... No file selected.
Upload
../../hackable/uploads/f.php succesfully uploaded!
More Information
https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload
https://www.acunetix.com/websitesecurity/upload-forms-threat/
sudo mysql - Parrot Terminal
Menu
[sudo mysql - Parrot Te...
Vulnerability: File Uploa...
[Downloads]
ls --color=auto - Parrot...
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
Search
ENG
IN
05:40 PM
13-01-2024


parrot Os - VMware Workstation
File Edit View VM Tabs Help
parrot Os
Applications Places System
Sat Jan 13, 17:47
Vulnerability: File Upload
127.0.0.1/DVWA/hackable/u
http://127.0.0.1/DVWA/vulnerabilities/upload/#
Import bookmarks... Parrot OS Hack The Box OSINT Services Vuln DB Privacy and Security Learning Resources
weevely http://127.0.0.1/DVWA/hackable/uploads/f.php pass - Parrot Terminal
File Edit View Search Terminal Help
[ritesh@parrot]~[~]
$sudo su
[sudo] password for ritesh:
[root@parrot]~[/home/ritesh]
#weevely generate pass /home/ritesh/Downloads/f.php
Generated '/home/ritesh/Downloads/f.php' with password 'pass' of 764 byte size.
[root@parrot]~[/home/ritesh]
#cd Downloads
[root@parrot]~[/home/ritesh/Downloads]
#ls
'12.1 hstshijack.zip.zip'
1599280058397.jpg
f.php
hstshijack
openlogic-openjdk-8u372-b07-linux-x32-deb.deb
python-cairo_1.16.2-2ubuntu2_amd64.deb
python-gobject-2_2.28.6-14ubuntu1_amd64.deb
python-gtk2_2.24.0-5.1ubuntu2_amd64.deb
wine-mono-7.0.0-x86.msi
'ZAP Scanning Report.pdf'
zenmap-7.91-1.noarch.rpm
zenmap_7.91-2_all.deb
zenmap.desktop.download
[root@parrot]~[/home/ritesh/Downloads]
Vulnerability: File Upload
More Information
Open HTTP Redirect
DVWA Security
Vulnerability: File Upload :: Damn Vulnerable Web Application (DVWA) — Mozilla Firefox
Menu
[sudo mysql - Parrot Te...
Vulnerability: File Uploa...
[Downloads]
weevely http://127.0.0.1...
o direct input to this VM, move the mouse pointer inside or press Ctrl+G.
Search
ENG
IN
05:47 PM
13-01-2024



Recommendation

1. Understand the Objective:

- Clearly define your objectives. Are you trying to bypass file type restrictions, manipulate content types, or exploit other aspects of file upload functionality? Having a clear goal will help you focus your testing.

2. Explore Security Levels:

- DVWA offers different security levels (low, medium, high) for file upload. Start with the lowest security level and gradually progress to higher levels. Each level introduces additional security measures.

3. Check File Type Validation:

- Upload files with different extensions and observe how the application handles them. Try changing the file extension or using double extensions (e.g., `malicious.php.jpg`) to bypass filters.

4. Content Type Manipulation:

- Test if the application checks the content type of uploaded files. Attempt to manipulate the content type header to trick the server into interpreting the file differently.

5. Size and Length Limits:

- Check for file size and length limitations. Attempt to upload files that exceed the specified limits and observe the application's response.

6. Malicious Payloads:

- Experiment with uploading files containing malicious payloads. For example, try uploading a file with embedded JavaScript or PHP code to test if the server executes it.

7. Double-Check Server Configuration:

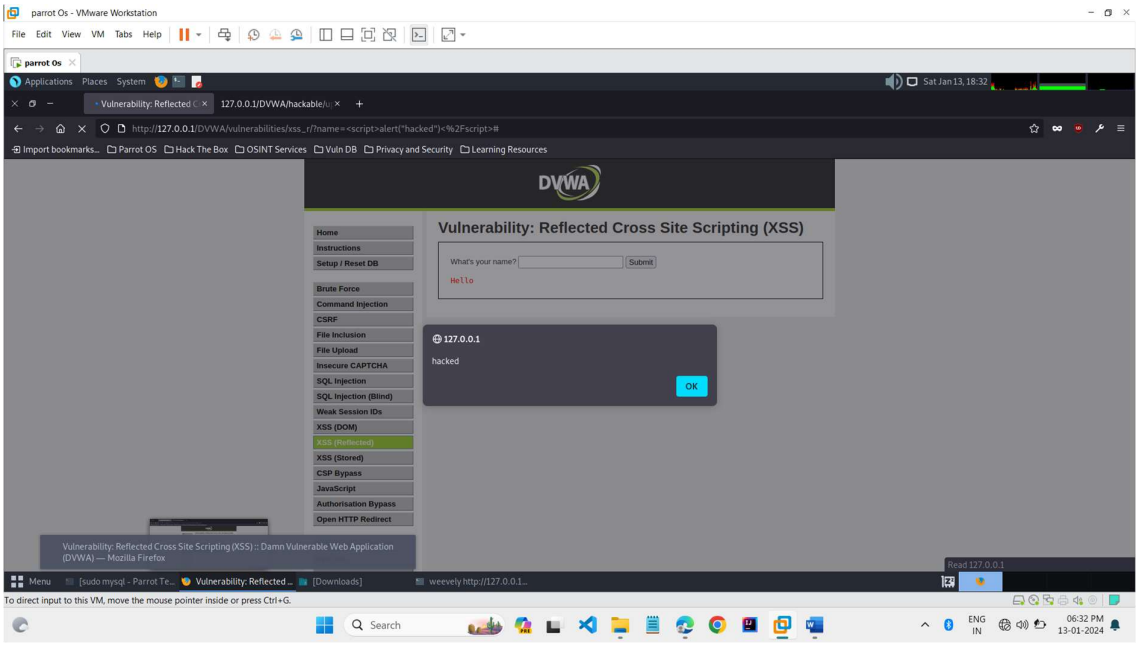
- Ensure that your DVWA environment is properly configured to allow file upload vulnerabilities. Check the server configuration and make sure it matches the intended setup.

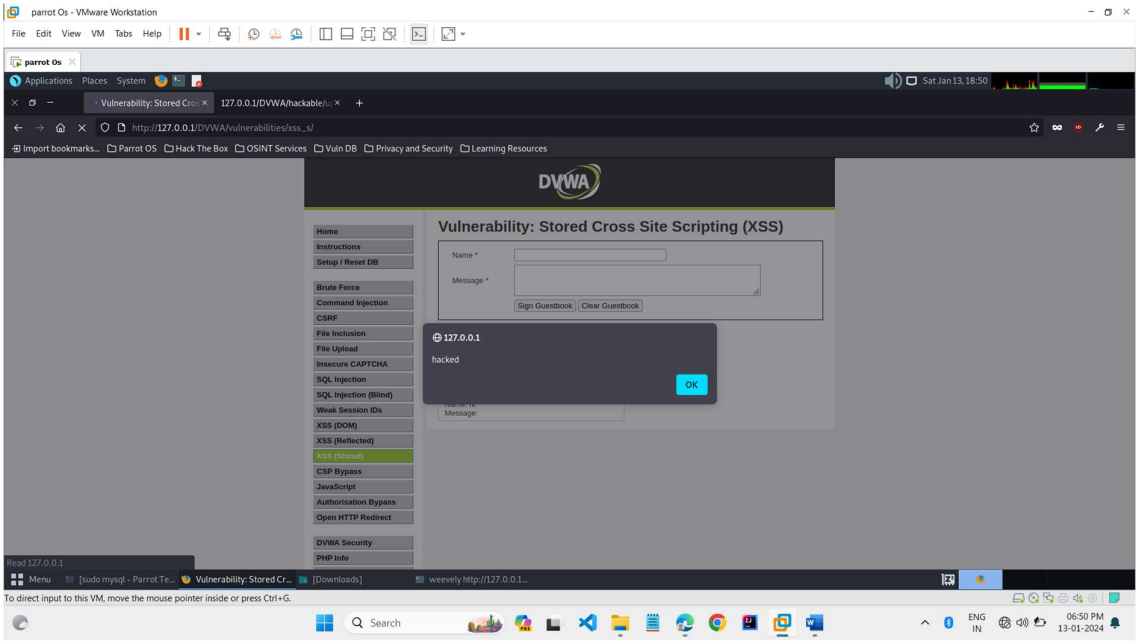
Cross-Site Scripting (XSS)

Problem:

Cross-Site Scripting (XSS) vulnerabilities are a critical focus in web penetration testing, demanding rigorous examination and mitigation strategies. These vulnerabilities allow attackers to inject malicious scripts into web applications, potentially compromising user data, session tokens, or redirecting users to malicious sites. In a penetration testing context, it is imperative to identify and remediate XSS vulnerabilities by implementing secure coding practices. Input validation, output encoding, and the use of security mechanisms like Content Security Policy (CSP) play crucial roles in preventing XSS attacks.

<script>alert("hacked")</script>





Recommendation:

Firstly, stringent input validation must be enforced to ensure that user inputs are thoroughly sanitized and validated before being rendered on web pages. Employing secure coding practices, such as output encoding, is crucial to mitigate the impact of potential XSS attacks. Implementation of a robust Content Security Policy (CSP) further fortifies the defense against malicious script injections. Regular security assessments, including manual testing and the use of automated scanning tools, help identify and remediate XSS vulnerabilities promptly. Moreover, continuous education and awareness programs for developers and stakeholders are vital for fostering a proactive security culture, empowering teams to understand, prevent, and mitigate XSS risks effectively.

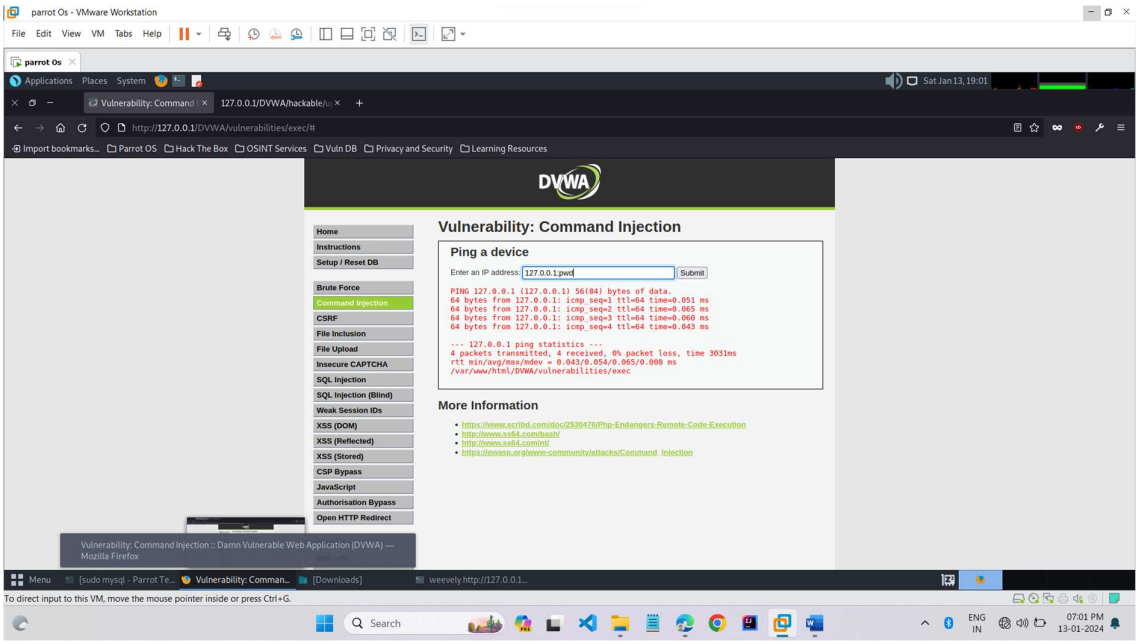
Command Injection

Problem:

Command execution vulnerabilities in web applications pose a significant security risk, allowing attackers to execute arbitrary

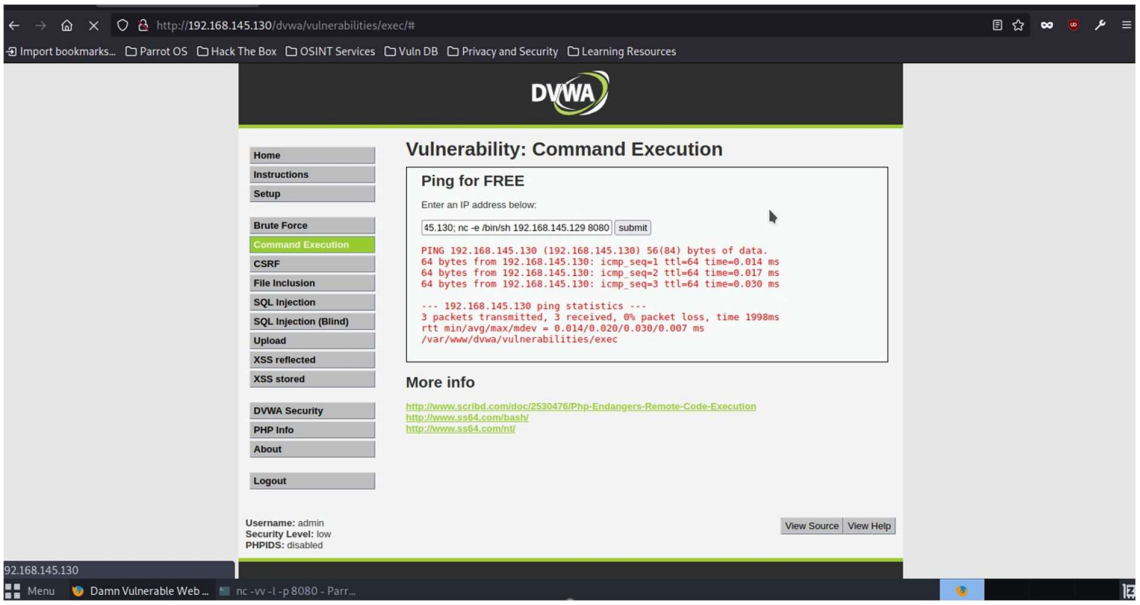
commands on the underlying server. During web penetration testing, it is imperative to identify and remediate such vulnerabilities to prevent unauthorized access, data breaches, and potential server compromise

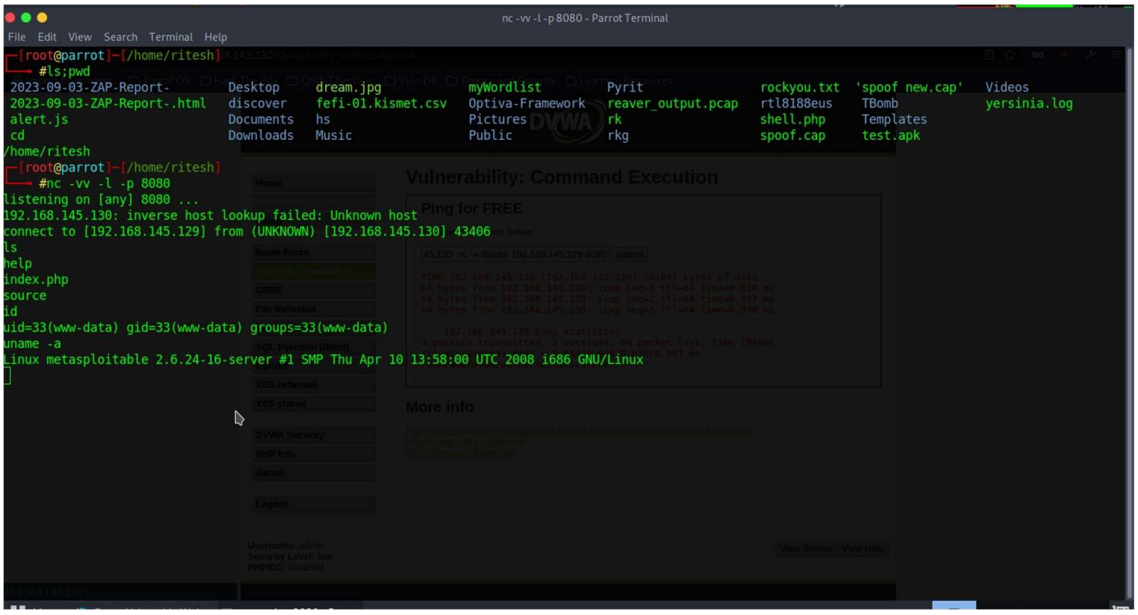
.



Recommendation

It includes adopting strict input validation and sanitization practices is critical to ensure that user input is treated as data and not as executable code, preventing injection attacks. Employing parameterized queries and prepared statements helps segregate user input from command structures, reducing the likelihood of command injection. Additionally, implementing proper access controls and least privilege principles ensures that even if a command execution vulnerability is exploited, the impact is limited. Routine security assessments, penetration testing, and continuous monitoring of web applications are essential for detecting and remedying command execution vulnerabilities promptly.

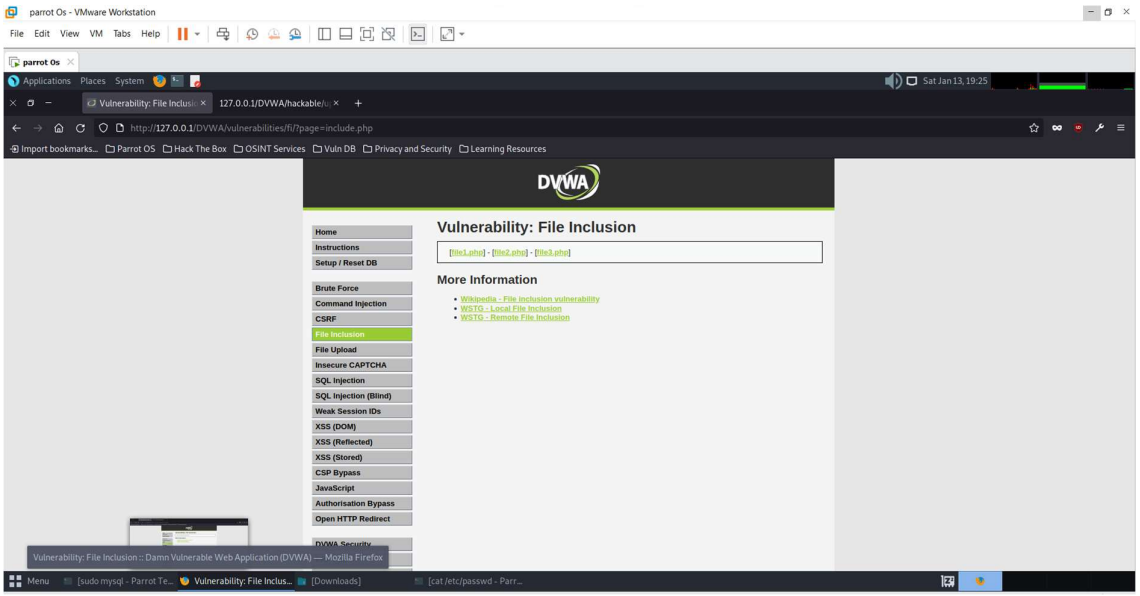


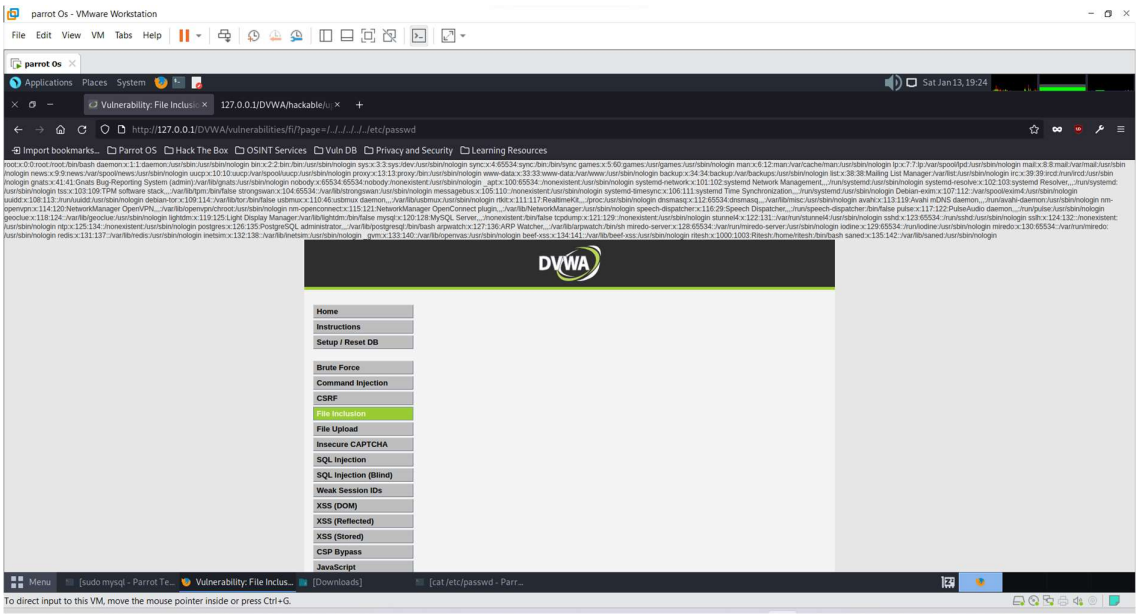


File Inclusion

Problem

File inclusion is a common vulnerability targeted in penetration testing, referring to the ability of an attacker to manipulate a web application's file inclusion mechanisms to include unauthorized files or exploit directory traversal vulnerabilities. This security flaw often occurs when user input is improperly handled in file paths used by the application. By manipulating input parameters that dictate file paths, attackers can traverse directories, access sensitive files, and even execute malicious code on the server





Recommendation

implementing strict input validation and sanitization mechanisms to ensure that user input cannot manipulate file paths. Additionally, utilizing secure file inclusion methods, such as avoiding the use of user-controlled input directly in file paths and employing whitelisting approaches for allowed file inclusions, enhances the application's resilience against malicious manipulation. Regular security audits and code reviews are essential to identify and promptly address any emerging file inclusion vulnerabilities, ensuring robust protection against unauthorized access and potential code execution on the server.

7.conclusion

In conclusion, the penetration testing conducted on Damn Vulnerable Web Application (DVWA) revealed several vulnerabilities that pose potential risks to the security of the application. The assessment identified and exploited common security issues such as SQL injection, file inclusion, and command execution. Recommendations for mitigation include implementing parameterized queries, input validation, and secure coding practices to fortify against SQL injection. For file inclusion vulnerabilities, adopting strict input validation, utilizing secure file inclusion methods, and conducting regular security audits are recommended. To address command execution risks, the implementation of

comprehensive input validation, access controls, and routine security assessments is crucial. Overall, enhancing the security posture of DVWA requires a multi-layered approach that includes secure coding, regular testing, and proactive measures to minimize potential threats and vulnerabilities.