A terminal window with a dark background and a menu bar at the top containing 'File', 'Actions', 'Edit', 'View', and 'Help'. The terminal shows a user 'labuser' at a prompt '(labuser@labuser)-[~]'. They enter '\$ sudo su', which prompts '[sudo] password for labuser:'. The prompt changes to '(root@labuser)-[/home/labuser]'. The user enters '# service mysql start'. The prompt remains '(root@labuser)-[/home/labuser]'. The user enters '# service apache2 start'. The prompt remains '(root@labuser)-[/home/labuser]'. The user enters '#', followed by a white cursor block.

```
File Actions Edit View Help

(labuser@labuser)-[~]
$ sudo su
[sudo] password for labuser:
(root@labuser)-[/home/labuser]
# service mysql start

(root@labuser)-[/home/labuser]
# service apache2 start

(root@labuser)-[/home/labuser]
#
```

Login into DVWA application with username as “admin” and password as “password”

Welcome :: Damn Vulnerable Web Application :: Kali Linux on localhost

127.0.0.1/DVWA/index.php

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to **practice some of the most common web vulnerabilities**, with **various levels of difficulty**, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possibly could by using that particular vulnerability.

Please note, there are **both documented and undocumented vulnerability** with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate how adding another layer of security may block certain malicious actions. Note, there are also various public methods at bypassing these protections (so this can be seen as an extension for more advanced users!)

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

- 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

Set the DVWA Security level to LOW as shown below and submit

DVWA Security

Security Level

Security level is currently: **low**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is as an example of how web application vulnerabilities manifest through bad coding practices and to as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerability source code to the secure source code.
Prior to DVWA v1.9, this level was known as 'high'.

Low Submit

PHPIDS

PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and

- 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
- DVWA Security**

File Actions Edit View Help

(labuser@labuser)-[~]

\$ nano rogue file.php

[sudo] password for labuser:

labuser@labuser: ~\$ sudo su - /home/labuser

labuser@labuser: ~\$ service mysql start

File Actions Edit View Help

GNU nano 7.2

<?php

[sudo] password for labuser:

echo "sample vulnerable code";

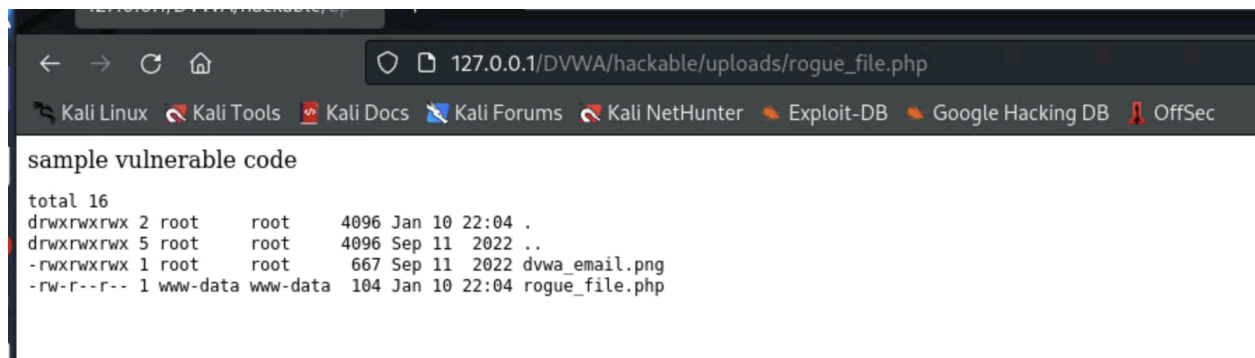
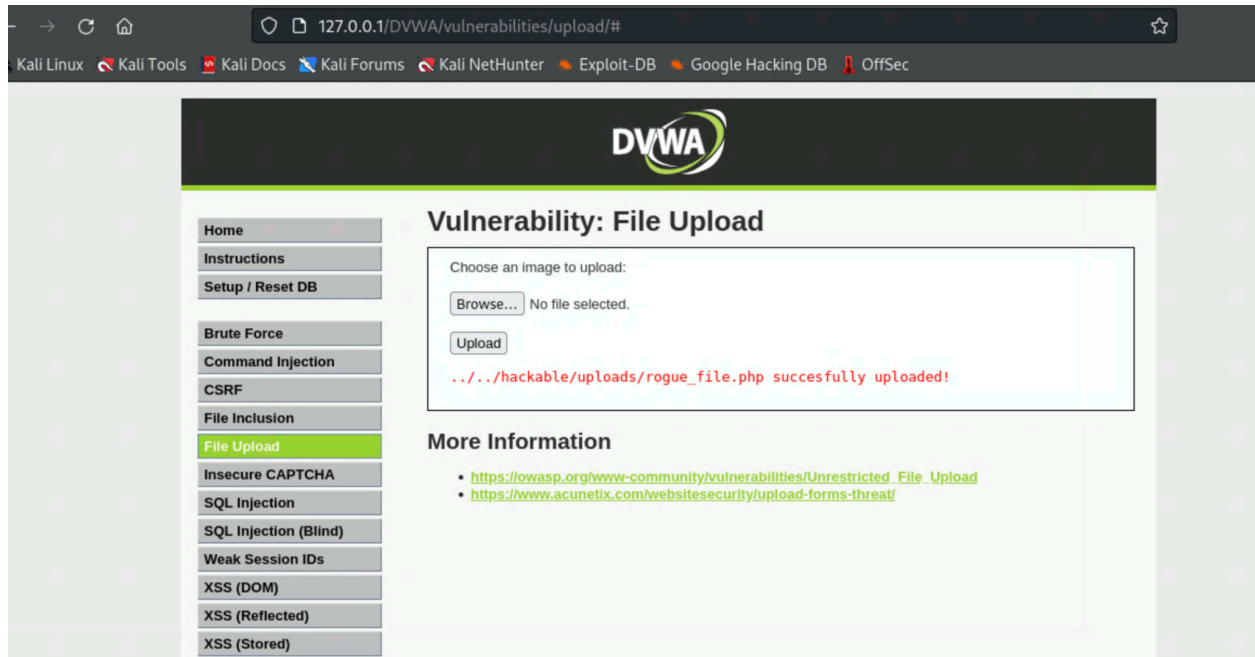
\$output = shell_exec('ls -la');

echo "<pre>\$output</pre>";

labuser@labuser: ~\$ sudo su - /home/labuser

?> service apache2 start

labuser@labuser: ~\$ sudo su - /home/labuser



Mitigation:

1. Strict validation of file types with only allowed list of extensions
2. Strict validation of file content or content inspection
3. Restrict the application to validation if the file is malware or not (Integrate the Antivirus tool during the file upload inspection)
4. Validate the Size of files as well
5. Restrict the permissions to process the file types in OS level (Don't run the application as root user)