https://oracle-base.com/blog/2021/12/19/log4j-vulnerabilities-my-random-thoughts/

For technical information keep checking the [Apache Log4j Security Vulnerabilities](https://logging.apache.org/log4j/2.x/security.html) page for updates.

Someone on Twitter asked me to write something about the Log4j issue and my response was I’m not really qualified to do that. After some consideration I thought maybe my uneducated opinion would be useful to others, so here goes…

**Basic Context**

This is a variation of something I wrote on an internal P1 incident, to give people some context. Remember, there are a range of people reading this P1, so it was written to be understandable to a wide audience.

* Log4j is an open source library used for logging in many Java applications. If you are not using Java apps, you are not using Log4j, so you are safe. If you are using Java apps, the vendor may not have used Log4j to do logging. This is why it is important to scan servers and check with the vendor to see if their software is vulnerable or not.
* This is not an issue with Apache HTTPS Server. Apache is a software foundation, which manages many commonly used open source software products, including the Apache HTTP Server. When you see “Apache Log4j”, the word “Apache” is a reference to the software foundation, not the HTTP server. As a result, it’s not safe to assume that if you don’t use Apache HTTP Server you are safe.
* Client applications running on your PC are low risk compared to server applications. Most of the attacks are based around sending requests containing dodgy payloads to application servers. Your local PC applications don’t handle such requests, so are extremely unlikely to be affected. They should still be patched as soon as patches are available, but you don’t need to obsess about them.

Here’s a quick summary:

* Not a Java application. Don’t worry.
* Java application that doesn’t use Log4j. Don’t worry.
* Java application that uses Log4j 1.x. Don’t worry about these vulnerabilities. Of course, older code may be susceptible to other vulnerabilities.
* Java application that uses Log4j 2.x. Java 8 (or later), upgrade to Log4j release 2.17.0\*.
* Java application that uses Log4j 2.x. Java 7, upgrade to Log4j release 2.12.2\*.
* If upgrading Log4j is not an immediate option, maybe you are waiting for a vendor to release a patch, consider mitigations until upgrades are possible.

\* These versions were correct at the time of writing. Keep checking the [Apache Log4j Security Vulnerabilities](https://logging.apache.org/log4j/2.x/security.html) page for updates.

**Mitigations are not Solutions**

Upgrading Log4j is the only way to be sure.

* Java 8 (or later) users should upgrade to Log4j release 2.17.0.
* Java 7 users should upgrade to Log4j release 2.12.2.

In the early days of the vulnerabilities, most people focused on mitigations. Probably the most common was to add this JVM parameter.

-Dlog4j2.formatMsgNoLookups=true

Or to set this environment variable.

LOG4J\_FORMAT\_MSG\_NO\_LOOKUPS=true

These worked for the initial vulnerability, but don’t stop all attacks. They are listed as “discredited” on the [Apache Log4j Security Vulnerabilities](https://logging.apache.org/log4j/2.x/security.html) page. It’s still worth doing this while you wait for patches from vendors, but this only limits your exposure. It’s not a complete fix. Do not do this and assume it’s game over!

Another option was to remove the JndiLookup.class completely. This is still listed as a valid mitigation if you are not able to upgrade. It may seem scary, but if a vendor patch is not forthcoming, you need to weigh up the risks.

zip -q -d log4j-core-\*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class

In addition to the direct mitigations, you also need to consider the bigger picture. Applications that are available to the outside world are clearly at enormous risk. If applications are only available inside your company network, then the risk is reduced. I’m not saying ignore internal applications, but prioritise the higher risk systems first maybe?

**Which of my systems are at risk?**

If you work in a mixed shop with lots of 3rd party products, that is not necessarily an easy question to answer. You can’t just search the file system for \*log4j\* and think that’s good enough. The log4j libraries are often deployed inside other JAR files or zip files.

I’ve been using the [log4j-detector](https://github.com/mergebase/log4j-detector) tool by [MergeBase](https://mergebase.com/) to scan servers. They’ve been releasing new versions regularly since this issue started. It seems to work well, but I’m sure other tools are available.

I would suggest you scan systems, even if your Vendor says they are safe!

**Log4j Developers**

I’ve read a number of comments where people have criticised the Log4j developers, and for the most part I think they are totally out of order. The vast majority of companies have no engagement with open source software. They don’t commit code, and they don’t donate money to the projects they rely on. If you are just taking without giving anything back, I feel like you are not in a position to complain.

I understand some of the criticisms from a technical perspective, but hindsight is a wonderful thing. You could have spent time looking through the source code and highlighted stuff you didn’t like, but you didn’t. You could have got involved, but you didn’t.

I suspect there are developers of other common libraries checking their code for “exotic” features that need to be turned off by default…

**Open Source**

I’ve seen some people using the recent Log4j issues as a way to attack open source on a more general level. If people couldn’t see the source, they wouldn’t find the exploits right? I don’t buy this. Security through obscurity is not security. I wonder how much longer these vulnerabilities would have existed if the source code was not freely available?

**Vendor Reactions**

The reaction of vendors has been really interesting. Some have been really quick to react. Some, not so much. At times like this it’s really important that vendors release a statement as soon as possible, even if that is a message that says they are aware of the issue and are investigating. Your typical “watch this space” message. If your vendors were slow to react, or didn’t react at all, then I think you need to question whether you should be working with their products.

This is true for vendors of products that don’t even use Java. In addition to scanning, we have been compiling statements from vendors regardless of their technology stack. For a vulnerability this high profile, I think it’s important all vendors release a statement. It may sound ridiculous to you, but not every person involved in the process has a grasp of what technology stack is used by each product. If a vendor provides a clear statement, then it makes life a lot easier.

**Oracle Advisory**

The Oracle advisory came out pretty quick, and has been updated frequently over the last week as more patches have been released. Keep an eye on it over the next few days, as I expect some existing patches will be reissued with Log4j 2.17.

<https://www.oracle.com/security-alerts/alert-cve-2021-44228.html>

You still need to use your brain when determining the risk. The Oracle database is marked as not vulnerable, but there are some items shipped with the database that use vulnerable log4j versions. For example SQL Developer is shipped with the database, but this is a client tool. It is not receiving HTTPS requests from users, so it’s not a threat. There are patched versions of this available, but do you care? In a similar manner Enterprise Manager is vulnerable, and you should patch it, but it shouldn’t be accessible publicly, so the threat is lower. The chances are only your DBAs have firewall access to this server, so it represents a smaller threat than a public facing application server.

**Conclusion**

It has been a shit show, and there are little signs of it calming down much before Christmas, but you have to use this as a learning experience. Please apply patches as soon as they are available. If your vendor is slow off the mark, apply the mitigations while you wait.

As I’ve said, I’m not an expert, just someone trying to cope with these issues. If you see anything you think is factually incorrect, please tell me so I can correct it.