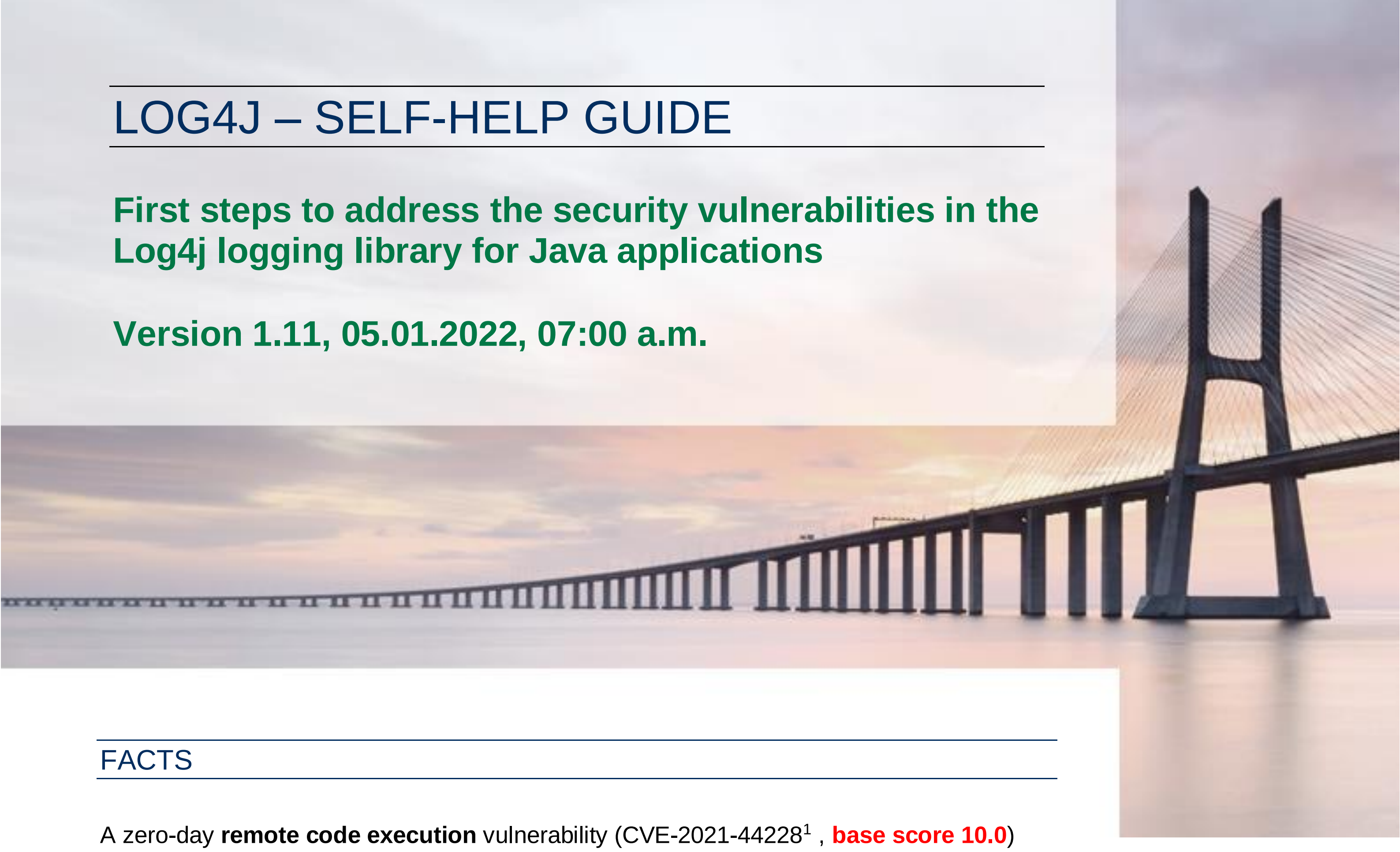
LOG4J – SELF-HELP GUIDE

First steps to address the security vulnerabilities in the Log4j logging library for Java applications

Version 1.11, 05.01.2022, 07:00 a.m.



FACTS

A zero-day **remote code execution** vulnerability (CVE-2021-44228[1] , **base score 10.0**) was discovered in Apache **Log4j,** a widely used Java logging library, allowing attackers to take full control of systems without authentication.

The vulnerability was publicly disclosed via GitHub on 2021/12/09[2]. In addition, further vulnerabilities were found after publication.[3] At least versions 2.0 to 2.17.0 of Apache Log4j are affected. Versions 1.x, which are no longer maintained, have a similar vulnerability with CVE-2021-4104. According to the current status, the vulnerability has been exploited since 2021/12/01,[4] but mass attacks only became known with the release of the Proof of Concept on 2021/12/09.

So as of today **every version below Log4j version 2.17.1[5]** is vulnerable under certain conditions. This also includes the patches already provided by the product manufacturers, as they often only install version 2.15 or 2.16 and might therefore still be vulnerable.

Log4j log output allows for the inclusion of variables. However, this functionality also allows attackers to call external Java libraries via **${jndi:ldap://** and **${jndi:ldaps://**, which opens the possibility to perform shell dropping without much additional effort. In addition, threat actors can use **${jndi:rmi** to execute commands within the current environment. In cloud services, Log4j log output could be used to read credentials - such as access keys - possibly allowing wide-ranging access to cloud services.[6]

[1] https://nvd.nist.gov/vuln/detail/CVE-2021-44228
[2] https://github.com/apache/logging-log4j2/pull/608
[3] last updated 12/18/2021, CVE-2021-45105
[4] https://twitter.com/eastdakota/status/1469800951351427073
[5] Log4j version 2.12.2 is not affected by the vulnerability because it was made for compatibility with Java version 7 only.
[6] https://twitter.com/Laughing_Mantis/status/1469804737566425089

HiSolutions AG
Castle road 1
12163 Berlin

info@hisolutions.com
www.hisolutions.com

+49 30 533 289-0
+49 30 533 289-900

Note: In the following, the information currently available is used as a basis to define a possible course of action. The document is continuously updated and adapted to current conditions and new findings.

This document is intended for IT experts and administrators who can check and secure systems on their own. The German Federal Office for Information Security has published an explanation for consumers.[7]

Make sure you always use the latest version:

https://research.hisolutions.com

Please notify us of any technical additions, which we will be happy to include.

[7] https://www.bsi.bund.de/DE/Themen/Verbraucherinnen-und-Verbraucher/Cyber-Sicherheitslage/Schwachstelle-log4Shell-Java-Bibliothek/log4j_node.html

AFFECTED SYSTEMS AND VULNERABILITY

AFFECTED SYSTEMS

There is currently no complete list of affected systems, as the component is used in many software applications. Currently maintained lists of vulnerable software include the following

- https://www.techsolvency.com/story-so-far/cve-2021-44228-log4j-log4shell/#affected-products
- https://gist.github.com/SwitHak/b66db3a06c2955a9cb71a8718970c592
- https://github.com/NCSC-NL/log4shell/tree/main/software
- https://github.com/cisagov/log4j-affected-db

Even if you do not use any of the products listed above, every system should be checked for the vulnerability if possible. In addition to Windows and Linux servers, this also includes appliances such as firewalls, logging servers, hypervisors, routers, managed switches and other systems on which Java can potentially be used. In these infrastructure systems, it is mostly the management components that are affected.

POTENTIAL THREAT EVALUATION – SITUATIONAL AWARENESS

According to the Federal Office for Information Security in Germany, exposed IT systems on the Internet and lateral movement in the local network are the main problems.

Web applications often pass on the received log strings, or these are monitored on the way by security systems such as a web application firewall or a monitoring tool. This can lead to affected systems automatically attempting to load malicious code. There is no limitation to certain protocols. Cases have already been reported in which LDAP and DNS have been used.

However, even if no malicious code is executed, sensitive environment variables such as access keys or passwords could still exfiltrated on, which could then be used for further attacks.

In addition, "lateral movement" can also make internal systems such as virtualization systems or database systems the target of the attack, even though these might not be accessible from the Internet.

INDUSTRIAL CONTROL SYSTEMS

Additionally, the software can also be used in machines in industrial and production environments. Therefore, these systems and components (ICS, SCADA, DCS, PLC, IIoT and IoT) should also be checked for this vulnerability and the manufacturers should be contacted.

We recommend contacting the relevant departments and checking whether all devices and software components have been accounted for in asset management systems. Systems for automated data acquisition (scanners, cameras, RFID, BLE,...) are often operated in separate networks and thus "escape" the view of a network monitoring system. In addition, these are often managed by manufacturers or integrators, so that responsibility is not always clearly defined here.

PRIORITIZATION

When defining the next steps, we recommend prioritizing the patching and update process according to the needs for availability and protection. The relevant security systems (such as Security Information and Event Management (SIEM), intrusion detection systems and web application firewalls) should be patched first. Then all systems that are directly connected to the Internet (front-end systems). Then all other systems.

FEATURES OF POTENTIALLY VULNERABLE SOFTWARE

All systems are potentially affected that show the following characteristics:

- Java (or JVM, so also jRuby, jPython, Scala, Kotlin, ...) as the technological basis + Log4j (vulnerable version).
  Note: In version 2.16.0 the property log4j2.enableJndi was introduced. This property is disabled by default.[8]

- The attack path currently works in **any** Java version (up to version 2.15.x, from 2.16.0 only if the default value has been changed to "true") as long as the classes used in the serialized payload are in the application's classpath.

- User inputs are processed and logged at least in part via Log4j (even if the inputs are forwarded within the network). This may also include messages about failed login attempts!

- The system has access to the Internet or DNS resolution which also works for addresses on the Internet.

[8] "When true, Log4j components that use JNDI are enabled . When false, the default, they are disabled. " (cf. https://logging.apache.org/log4j/2.x/manual/configuration.html)

Note: It was discovered that the fix for CVE-2021-44228 in Apache Log4j 2.15.0 was **incomplete** in certain non-standard configurations. CVE-2021-45046[9] was assigned for this vulnerability. In addition, **another bug** (CVE-2021-45105) was found that was fixed in version Apache Log4j 2.17.0. This version also comes with a vulnerability which was later assigned CVE-2021-44832. It can only be exploited if an attacker already has write access to the log configuration files. Apache Log4j 2.17.1 fixes this vulnerability.

[9] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-45046

STEPS TO ADDRESS THE VULNERABILITY

Step 1: Check for possible vulnerability and take initial action

LOG4J DETECTOR

Run Log4j Detector from https://github.com/mergebase/log4j-detector on all systems where possible. This requires a Java environment to be installed on the system. Since many applications bring their own Java environment, a system without a globally installed Java environment may also be affected.

The Detector is started under Linux/Unix via

```
    java -jar log4j-detector-2021.12.15.jar / > hits.txt
```

or under Windows by means of

```
    java -jar log4j-detector-2021.12.15.jar C:\ > hits.txt
```

(This should also be run for other drives if necessary).

If entries are found in `hits.txt` after a successful run, the system is vulnerable via the Log4Shell vulnerability and should be disconnected from the network, analyzed and then secured as soon as possible.

CHECKING LINUX SYSTEMS

You can use the following console commands to scan Linux systems for the presence of log4j:

```
    ps aux | grep -i 'log4j'

    find / -iname "*log4j*"

    lsof | grep -i "log4j"
```

If files are found using this command, they must be checked individually with the following command to see if they are vulnerable:

```
    unzip -p found-log4j-file.jar META-INF/MANIFEST.MF | grep bundle
version
```

Through this console command you can find out where your applications store log data:

```
    lsof | grep '\.log'
```

Note: This search will only find currently active Log4j instances and installations directly in the file system, but not Log4j files present in (nested) archives.

LOG4J-SCANNERS BY DIFFERENT PROVIDERS

By now many established companies provided antivirus or endpoint-protection software, vulnerability scanners or network security functionalities for their software to scan for vulnerable Log4j-versions. These can be deployed if you have their products in active use. Please pay attention to the fact, that detection over the network without any credentials only includes specific protocols and might not detect all vulnerable versions sufficiently. Local scanning tools need to be run with read rights on all directories that need to be accessed and scanned.

LIST ALL PRODUCTS USED

It is recommended to create a list of all products that use Log4j. This list should then be checked regularly. It should be recorded whether the version is vulnerable, whether there is already a patch, when this was installed and, in the case of larger organizational structures, who is responsible for this, for example a specific internal department or an IT service provider.

Note: This allows you to strategically determine not only which systems you have patched, but also whether an attacker has already successfully infiltrated and patched the vulnerability itself to deprive further attackers of the option to exploit the vulnerability.

If you have vulnerable products, be sure to follow the recommendations in steps 2 to 4.

COORDINATION WITH PRODUCT MANUFACTURERS AND SERVICE PROVIDERS

Contact your product manufacturers and service providers to obtain and install updates. In particular, make sure that the manufacturer / service provider is also aware that version 1.x is also vulnerable. In general, everything below Log4j version 2.17.1 (as of today) is vulnerable under certain conditions.

Note: Just because no JNDI is used in the software does not mean that Log4j does not respond to JNDI requests.

Note: Statements like "we don't use Java at all" can also be wrong: For example, there can be a jPython/Jython, jRuby, Scala, Kotlin, Groovy, Clojure, or similar installation based on the JVM. Log4j is flexible and is then often used "quickly" as an established logging standard.

Step 2: Prevent further attacks

Note: The following steps contain measures that should be performed by technically experienced personnel. If in doubt, please give this document to your IT service providers, who can work through these steps with you.

DISABLE VULNERABLE SERVICES

If you are using products that are currently vulnerable and cannot be patched, they should - if possible - be switched off or network connections interrupted or segmented. Please note that systems not connected to the Internet could also be vulnerable, as the "attack strings" are passed on internally in the network. For example, this could also be intranet portals such as internal booking systems or HR reporting systems.

Servers should generally only be allowed to establish connections (especially to the Internet) that are absolutely necessary for the intended purpose. Other connections should be prevented by appropriate control instances such as packet filters and application layer gateways. More about this in the section Egress filter (prevent outgoing traffic) .

Note: DNS queries executed via JNDI are sometimes sufficient to compromise a system.

Data backups (using snapshots if possible) should be created for a detailed analysis, that might become necessary later. Virtual systems that cannot be protected immediately with the previously mentioned measures should be put on standby to preserve RAM for possible forensic analysis.

PREFERRED PROTECTIVE MEASURE: IMMEDIATE PATCHING

If already available, please apply the manufacturer's patches immediately!

A restart may be necessary. Please check several times a day whether patches have been made available and, if necessary, actively ask the manufacturers or open a support ticket.

After patching, verify (for example with Log4j Detector) that no vulnerable Log4j libraries are used anymore.

PROTECTIVE MEASURES FOR AFFECTED SYSTEMS WITHOUT PATCH

As soon as an update of the application has been provided by the manufacturer, it must be installed immediately.

If no update is available, all possibilities to exploit the vulnerability should be disabled. To do this, the JNDI lookup function of the Log4J implementation must be disabled.

The most reliable way to do this is to delete the corresponding .class file. In most cases, this should not affect the application.

Note: To be on the safe side, please create a backup copy for each file identified by Log4j Detector in `hits.txt` before you perform the following steps!

If the JndiLookup.class file is located directly in the found file (this can be a zip, jar, war, ear or aar file), it can be deleted directly from it:

- using 7zip or another GUI archive program
(path can be taken from the `hits.txt`)
- using the following call (adjust path according to `hits.txt`):

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

- If nested archives were found (recognizable by the ! between the paths in the `hits.txt`), the individual archives must first be unpacked, then cleaned up and afterwards packed again.

If replacing the .class file is not an option, the vulnerable version used can be replaced with the latest version of the log4j library, after consulting the manufacturer of the software.

This can be found at https://logging.apache.org/log4j/2.x/download.html
and should have at least version 2.17.0. Only the log4j-core.jar file is required from the archive. At this point, however, it cannot be guaranteed that this solution will work smoothly. Therefore, please check this option carefully with your product manufacturers and IT service providers.

Furthermore, the parameter `log4j2.formatMsgNoLookups=True` should be set to globally prevent JNDI lookups. This should be done in addition to the above measures, as none of them can guarantee 100% security.
The parameter should be set in the following places:

- As environment variable of the user running the Java process:
  `JAVA_OPTS=-Dlog4j2.formatMsgNoLookups=True`
- As a global environment variable:
  `LOG4J_FORMAT_MSG_NO_LOOKUPS=True`
- Append as Java transfer parameter in the script used to start the application:
  `-Dlog4j2.formatMsgNoLookups=True`
- As of Java 8u121 set the following parameters at Java startup:
  `-Dcom.sun.jndi.rmi.object.trustURLCodebase=False`
  `-Dcom.sun.jndi.cosnaming.object.trustURLCodebase=False`

Attention: Restart the entire system after implementing these measures!

PROACTIVE RECOGNITION OF REQUESTS (MAYBE)

It can be assumed that current web application firewalls or IPS can detect the attacks only to a certain extent. If you have the technical capabilities, you can also check for the attack strings to better detect the attacks.

However, the critical request parts such as jndi:ldap:// are already partially masked, for example according to the following patterns:

```
${jndi:${lower:l}${lower:d}${lower:a}${lower:p}://${hostName}.}
${${::-j}${::-n}${::-d}${::-i}:${::-l}${::-d}${::-a}${::-
p}://${hostName}.}
```

There are countless ways to obfuscate JNDI calls. Simply filtering the requests does **not** provide **sufficient protection in any case**!

EGRESS FILTER (PREVENT OUTGOING TRAFFIC)

A general procedure, which should actually already be implemented, is to limit data traffic. For example, a database should only be allowed to connect to clearly defined IT systems and should never establish a direct connection to the Internet. This way, possible lateral movement can be contained if necessary. Another focus should be possible DNS and ICMP packets, which often allow data to be easily tunneled to the Internet.

BACKUPS AND CI/CD PIPELINES

The above measures serve to protect the environment short-term. Please also be aware that the environment may become vulnerable again by restoring backups or by libraries that have not yet been adapted in the context of a CI/CD pipeline. You should therefore ensure that the updates or customizations have been integrated in all "golden images", as well as in the development chain.

Step 3: Recommendation of measures (in case of suspected compromise)

If you were vulnerable, the systems should be disconnected from the network and scanned separately. In addition, you canus the Thor-Lite-Scanner[10] to scan for possible webshells or anything else, such as Ransomware. We have published instructions on how to use it on our Research Blog.[11]

Note: Due to the large number of possible attackers, this can only be seen as an indication. Repeat the check regularly with updated signatures.

You should thoroughly scan your systems for possible ransomware. For this, we recommend our guide published in the context of the Hafnium exploit chain[12].

In the event of a potential compromise, it is definitely advisable that the following measures are carried out to detect or even prevent possible follow-up attacks:

- Increase logging
    - Increase the minimum retention period of log files with sufficient free storage on the following IT systems:
        - Security log of the domain controller
        - Netflow data (if available)
        - Web Proxy Log
        - Firewall logs
    - Checking the log files for particularly relevant event messages
        - Here, the SIGMA rules [13] can be used to monitor the log files automatically
        - Monitor externally reachable remote accesses for unauthorized access without multi-factor authentication
- Use the Thor-Lite Scanner. We have published instructions on how to use it in our Research Blog.
- Systematicly change logindata such as passwords, especially for administrative user accounts
- Check the systems for newly created user accounts, services or additionally executed programs or stored files
- Run a full virus scan of your IT systems on a regular basis
- Consider implementing multi-factor authentication for external access to your network
- If not already in place, create and store an offline data backup[14]

[10] https://www.nextron-systems.com/thor-lite/

[11] https://research.hisolutions.com/

[12] https://research.hisolutions.com/wp-content/uploads/2021/03/HiSolutions-Ueberwachen-Ihrer-Systeme.pdf

[13] https://github.com/SigmaHQ/sigma/tree/master/rules/windows/builtin

[14] https://research.hisolutions.com/2021/03/schutz-gegen-ransomware-hisolutions-selbsthilfe-offline-backup/

- Raise awareness of staff and service providers with regard to possible phishing attacks/CEO fraud scenarios. In particular, warn against e-mails that refer to changed bank details or delivery addresses with reference to existing e-mails or contain unusual attachments. The federal office for information security of Germany provides assistance in recognizing suspicious e-mails on its website.[15]

We recommend monitoring systems according to these recommendations for a period of 12 months. In the past, we have often seen 'quiet periods' (after initial compromise until the actual attack) of 2-4 months, in some cases lasting up to 9 months.

In the near future, regularly check the pages listed under Step 4: Observation of the situation for new tipps to increase security.

DATA PRIVACY

A German privacy officer additionally points out that **closing the vulnerability** is **not enough.** Those responsible must **also check whether successful attacks have** already occurred. In this case, further measures must be taken accordingly and it must be checked whether a notification of personal data breaches must be made in accordance with Art. 33 GDPR.[16]

The Bavarian State Office for Data Protection has published a "Tipps for a Log4Shell-Quick analysis"[17] with a checklist of measures to be taken.

[15] https://www.bsi.bund.de/DE/Themen/Verbraucherinnen-und-Verbraucher/Cyber-Sicherheitslage/Methoden-der-Cyber-Kriminalitaet/Spam-Phishing-Co/spam-phishing-co_node.html

[16] https://datenschutz.hessen.de/pressemitteilungen/unmittelbarer-handlungsbedarf-wegen-schwachstelle-in-java-bibliothek-log4j

[17] https://www.lda.bayern.de/media/veroeffentlichungen/handreichung_log4shell_baylda.pdf

OBSERVATION OF THE SITUATION AND KNOWN ATTACKS

Step 4: Observation of the situation

We expect that exploitation of the vulnerability will continue to evolve and that step by step more possible attack paths against various applications and systems will we publicized in the coming days and weeks.

Use the time to find and patch all - or at least as many as possible - of the applications you use with vulnerable versions of Log4j or apply the workaround.

In the near future, check the following pages regularly for new additions, as the documents will be continuously updated and adapted:

- **HiSolutions**: Log4Shell vulnerability in Log4j: overview - self-help guide
  https://research.hisolutions.com/log4shell
- **BSI**: Critical vulnerability in Java library Log4j
  https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Informationen-und-Empfehlungen/Empfehlungen-nach-Angriffszielen/Webanwendungen/log4j/log4j_node.html
- **BSI**: Critical vulnerability in log4j published (CVE-2021-44228)
  https://www.bsi.bund.de/SharedDocs/Cybersicherheitswarnungen/DE/2021/2021-549032-10F2.pdf
- **BSI**: Working Paper Detection and Response
  https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Cyber-Sicherheit/Vorfaelle/log4j-Schwachstelle-2021/log4j_Schwachstelle_Detektion_Reaktion.pdf
- **BSI**: Critical "Log4Shell" vulnerability in widely used logging library Log4j (CVE-2021-44228)
  https://www.bsi.bund.de/SharedDocs/Cybersicherheitswarnungen/DE/2021/2021-549177-1032.pdf

KNOWN ATTACKS

It is not yet known who is attacking servers and for what purpose. The number of groups of attackers has increased significantly since the release. For this reason, it is currently not possible to assess well which measures are sufficient to clean up the systems. Currently publicly disclosed attacks are (excerpt):

Attack Group	Type of Attack	Affected Systems	References
Khonsari	Ransomware	Windows	https://therecord.media/first-log4shell-attacks-spreading-ransomware-have-been-spotted/
MIRAI	Botnetz	IoT devices with ARC-Prozessor	https://blog.netlab.360.com/threat-alert-log4j-vulnerability-has-been-adopted-by-two-linux-botnets/
Muhstik (Tsunami Variant)	Botnet + Backdoor	IoT devices and Linux	https://blog.netlab.360.com/threat-alert-log4j-vulnerability-has-been-adopted-by-two-linux-botnets/
Elknot	DDoS Botnet	Linux and Windows	https://blog.netlab.360.com/ten-families-of-malicious-samples-are-spreading-using-the-log4j2-vulnerability-now/
m8220	Mining Botnet	Linux and Windows	https://blog.netlab.360.com/ten-families-of-malicious-samples-are-spreading-using-the-log4j2-vulnerability-now/
SitesLoader	Botnet	Linux	https://blog.netlab.360.com/ten-families-of-malicious-samples-are-spreading-using-the-log4j2-vulnerability-now/
xmirg.pe	Mining Botnet	Linux and Windows	https://blog.netlab.360.com/ten-families-of-malicious-samples-are-spreading-using-the-log4j2-vulnerability-now/
xmirg.ELF	Mining Botnet	Linux	https://blog.netlab.360.com/ten-families-of-malicious-samples-are-spreading-using-the-log4j2-vulnerability-now/
attack tool 1	Botnet	Linux	https://blog.netlab.360.com/ten-families-of-malicious-samples-are-spreading-using-the-log4j2-vulnerability-now/
attack tool 2	Botnet	Linux	https://blog.netlab.360.com/ten-families-of-malicious-samples-are-spreading-using-the-log4j2-vulnerability-now/
Orcus	Remote Access Trojaner	Windows	https://businessinsights.bitdefender.com/technical-advisory-zero-day-critical-vulnerability-in-log4j2-exploited-in-the-wild
Unknown	Reverse Bash Shell	Linux and Windows	https://businessinsights.bitdefender.com/technical-advisory-zero-day-critical-vulnerability-in-log4j2-exploited-in-the-wild

StealthLoader	Trojaner	Windows	https://blog.checkpoint.com/2021/12/14/a-deep-dive-into-a-real-life-log4j-exploitation/
PHOSPHORUS	Ransomware	Windows	https://www.microsoft.com/security/blog/2021/12/11/guidance-for-preventing-detecting-and-hunting-for-cve-2021-44228-log4j-2-exploitation/
HAFNIUM	Ransomware	Virtualization	https://www.microsoft.com/security/blog/2021/12/11/guidance-for-preventing-detecting-and-hunting-for-cve-2021-44228-log4j-2-exploitation/
Kinsing	Backdoor	Linux	https://medium.com/proferosec-osm/log4shell-massive-kinsing-deployment-9aea3cf1612d
State actor	Espionage		https://twitter.com/kevincollier/status/1471099511179259904
TellYouThePass	Ransomware		https://www.curatedintel.org/2021/12/tellyouthepass-ransomware-via-log4shell.html
Unknown	Mining		https://www.bleepingcomputer.com/news/security/log4j-attackers-switch-to-injecting-monero-miners-via-rmi/
Conti	Ransomware	VMware	https://www.bleepingcomputer.com/news/security/conti-ransomware-uses-log4j-bug-to-hack-vmware-vcenter-servers/
Lapsus	Ransomware		https://therecord.media/lapsus-ransomware-gang-hits-sic-portugals-largest-tv-channel/
Aquatic Panda	Unknown		https://venturebeat.com/2021/12/29/china-based-group-used-log4j-flaw-in-attack-crowdstrike-says/

The situation is likely to remain confusing as the various attackers have very different approaches and objectives. Worm-like spreads can also be expected.

Note: Since the attacks have been running since at least 1.12.2021, current log files should be backed up so that they can be evaluated later. If possible, Netflow logs should be created to monitor the data flow.

INAPPROPRIATE MEASURES

Filter on WAF / ReverseProxy

As shown above, many requests are obfuscated - currently this is the case for about half of the requests. This means, this is not a decent protective measure, since only one successful exploitation attempt by the attackers is sufficient. In addition - depending on the application - the attacks can also be triggered by entries in the UserAgent, cookies, content data (photo-exif data, metadata from S/MIME/SSL certificates, etc.).

Decrease the log level

Changing the log settings within Log4j will not help because the data to be logged will be evaluated and abused by Log4j before the log settings come into play.

Logging/alerting in SIEM

The usual SIEM systems such as Splunk, Graylog, and anything that uses ElasticSearch are affected and, in the worst case, only serve to further propagate a compromise.

(Continued) use of the obsolete Log4j version 1.x

According to current knowledge, this version is also vulnerable via Log4Shell and additionally has unpatched vulnerabilities that are not patched thanks to the EOL status of the code.[18]

One of these vulnerabilities is very similar to this Log4Shell vulnerability.[19]

[18] https://logging.apache.org/log4j/1.2/

[19] https://access.redhat.com/security/cve/CVE-2021-4104