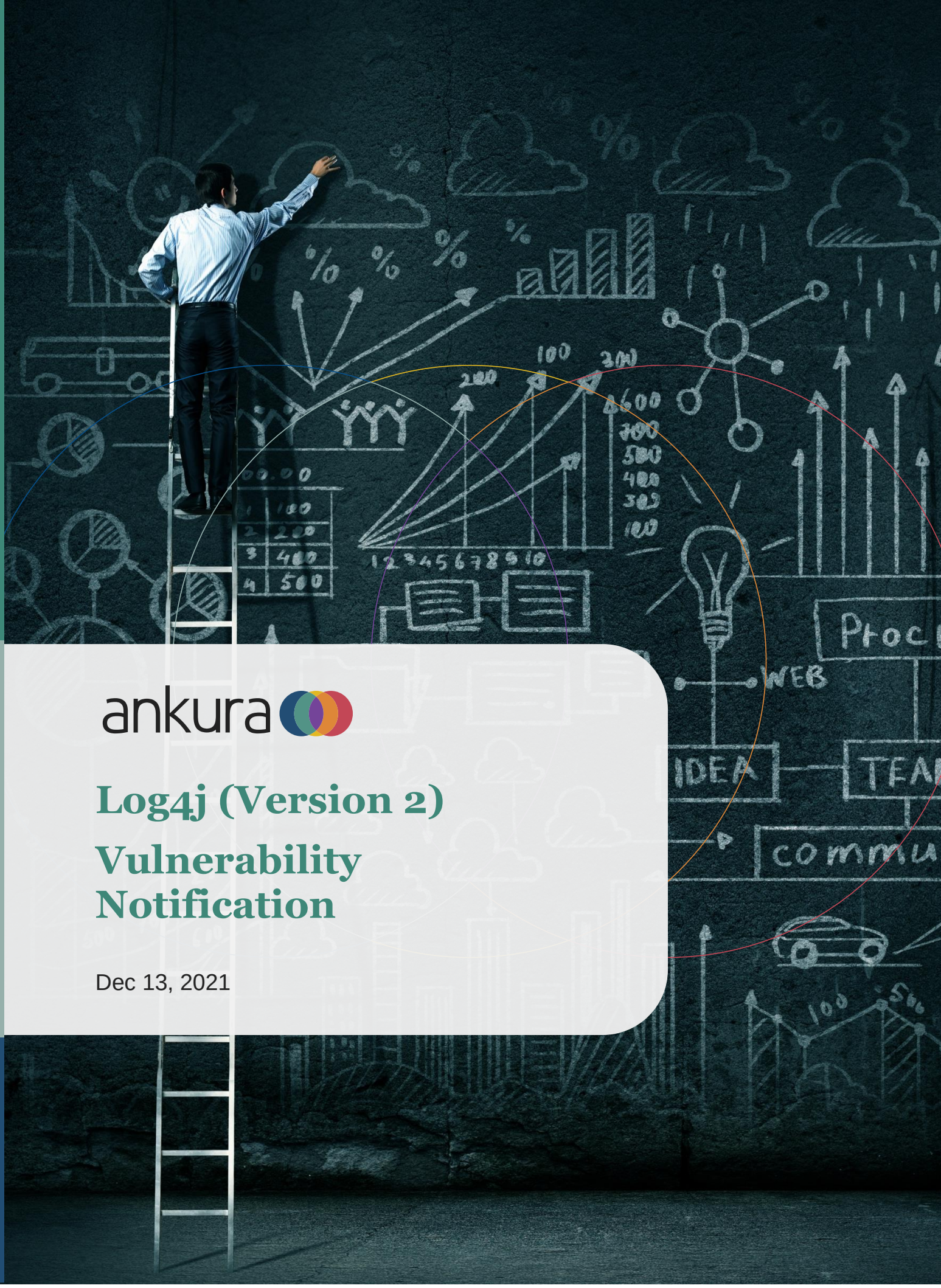

ankura

Log4j (Version 2) Vulnerability Notification

Dec 13, 2021

On the **9th Dec 2021 at 2:25 am,** an exploit that impacts Apache Log4j version 2 was published by "@P0rZ9" on Twitter, along with proof of concept (PoC) code on GitHub. Since this initial publication, a CVE (CVE-2021-44228) was logged for the vulnerability within "Log4j" version 2.

This vulnerability occurs when Apache Log4j parses user input that contains malicious code, which causes Apache Log4j to execute the malicious code. This vulnerability is not a problem with the Apache webserver software; Log4j is not part of the Apache webserver. Any string logged via Log4j could potentially be used to exploit this vulnerability.

To date, these strings have predominantly contained URL's that cause Apache Log4j to fetch content on an attacker's hosted website, which is then retrieved by the vulnerable Apache Log4j model and executed on the system running Apache Log4j. The result of this vulnerability would give an attacker control of the system the exploit was executed on, and this would result in complete system compromise. It is also likely that threat actors could use this compromised system to move laterally.

Reporting to date has shown that this vulnerability is already being scanned and exploited in the wild. Victims have seen a combination of security researchers scanning for the vulnerability and threat actors using it to conduct resource hijacking, such as installing Coin Miners and installing Cobalt Strike on systems[1].

It is essential to understand that Apache Log4j could be used in software developed by your organisation and software provided by many software vendors. If you cannot directly patch or mitigate this vulnerability, you should actively detect exploitation attempts.

WHAT SOFTWARE IS IMPACTED?

- Systems running Apache Log4j version 2.0-beta to version 2.14.1 are vulnerable to this exploit.

Apache Log4j may be used in many Apache-based systems that are public-facing to the internet. These systems are used widely by various cloud servers and vendors. To date, the systems that are likely impacted include; Struts 2, Solr, Druid, Flink, and Swift.

WHAT IS THE OUTLOOK FOR THIS VULNERABILITY?

Given we've already seen threat actors scanning for this vulnerability and exploitation already occurring in the wild, more threat actors will likely start to leverage this exploit. While there have not been any reports of ransomware threat actors leveraging this vulnerability, they will likely use it to gain initial access to victims in the coming days/weeks.

[1] https://www.microsoft.com/security/blog/2021/12/11/guidance-for-preventing-detecting-and-hunting-for-cve-2021-44228-log4j-2-exploitation/

HOW DO I DETECT IF MY SYSTEMS ARE VULNERABLE?

Several techniques can be used to determine if your systems are currently vulnerable, and you should use whichever of the below techniques you are most comfortable with.

- PowerShell for Windows Systems
  - `gci 'C:\' -rec -force -include *.jar -ea 0 | foreach {select-string "JndiLookup.class" $_} | select -exp Path`
- Bash for Linux Systems
  - `find / 2>/dev/null -regex ".*.jar" -type f | xargs -I{} grep JndiLookup.class "{}"`
- Go-based vulnerability scanner (cross platform)
  - https://github.com/hillu/local-log4j-vuln-scanner
- YARA signature for scanning with EDR tools:
  - https://github.com/darkarnium/CVE-2021-44228/blob/main/rules/vulnerability/log4j/CVE-2021-44228.yar

HOW DO I FIX VULNERABLE SYSTEMS/SOFTWARE?

- You should update to Apache Log4j version 2.15.0
  - An update has been provided on Maven Central for “log4j-core.jar.”
- If you cannot update to Apache Log4j version 2.15.0, possible mitigations have been provided below.
- If you cannot update or apply the mitigations, you should detect exploitation attempts.

To mitigate exploitations against vulnerable versions of Apache Log4j version 2.10 to 2.14.1, you can set the system property “log4j2.formatMsgNoLookups” or the environment variable “LOG4J_FORMAT_MSG_NO_LOOKUPS” to “true”. You can configure this in the start-up script of the Java Virtual Machine with “-Dlog4j2.formatMsgNoLookups=true”. Please note that these mitigations may not always be effective; it has already been reported that it is not sufficient for systems using Logstash 6.8.21 or 7.16.1, which may also be the case for other vendor software.

To mitigate this vulnerability in Kubernetes, administrators should use “kubectl set env” to set the LOG4J_FORMAT_MSG_NO_LOOKUPS=”true” environment variable to apply the mitigation across Kubernetes clusters.

To mitigate this vulnerability in Log4j version 2.0-beta9 to 2.10.0, remove the “JndiLookup” class from the “classpath” using “zip -q -d log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class”

HOW DO I CHECK IF MY SYSTEMS ARE COMPROMISED?

Based on the PoC that has been published, threat actors will have to leverage strings injected into user input fields starting with the following examples:

```
${jndi:ldap
${jndi:dns
${jndi:rmi
${jndi:nis
${jndi:nds
${jndi:corba
${jndi:iiop
```

An example of what these attacks may look like in your logs can be found here.

However, threat actors are already finding ways to obfuscate their malicious strings to bypass common detection techniques. It would be prudent to use several detection techniques while threat actors are still evolving their attacks towards this vulnerability.

- Bash for Linux Systems:
  - `sudo egrep -I -i -r '\$(\{|%7B)jndi:(ldap[s]?|rmi|dns|nis|iiop|corba|nds|http):/[^\n]+' /var/log`
  - `sudo find /var/log -name \*.gz -print0 | xargs -0 zgrep -E -i '\$(\{|%7B)jndi:(ldap[s]?|rmi|dns|nis|iiop|corba|nds|http):/[^\n]+'`
  - `sudo find /var/log/ -type f -exec sh -c "cat {} | sed -e 's/\${lower://'g | tr -d '}' | egrep -I -i 'jndi:(ldap[s]?|rmi|dns|nis|iiop|corba|nds|http):'" \;`
  - `sudo find /var/log/ -name '*.gz' -type f -exec sh -c "zcat {} | sed -e 's/\${lower://'g | tr -d '}' | egrep -i 'jndi:(ldap[s]?|rmi|dns|nis|iiop|corba|nds|http):'" \;`
- Python-based scanner
  - https://github.com/Neo23x0/log4shell-detector
- Suricata and Snort IDS signature from Proofpoint
  - https://rules.emergingthreatspro.com/open/
- YARA signature for scanning with EDR tools:
  - https://github.com/Neo23x0/signature-base/blob/master/yara/expl_log4j_cve_2021_44228.yar
- Velociraptor EDR scanning module:
  - https://docs.velociraptor.app/exchange/artifacts/pages/log4jrce/

In addition to the above detection techniques, there are also indicators of compromise (IoC's) being collected by several vendors and threat researchers. These can be useful as part of an investigation if you detect a compromise.

I'VE DETECTED A COMPROMISED SYSTEM, WHAT NOW?

You should conduct your Incident Response procedures immediately to determine if you have detected someone scanning for this exploit on your systems or if a threat actor has managed to exploit the vulnerability on your systems successfully. If you cannot determine this yourself, reach out to your Incident Response team or your third-party consultant to assist.

Also, be aware that if you have detected a threat actor successfully exploiting this vulnerability, it's likely they may have already laterally moved. So you need to identify any lateral movement that may have occurred from your compromised systems.

Lastly, if you are running vulnerable third-party software, you should continue to check for updates from your software vendor while also conducting your own investigation if your system(s) becomes compromised.

REFERENCES

- https://logging.apache.org/log4j/2.x/security.html
- http://cve.mitre.org/cgi-bin/cvename.cgi?name=2021-44228
- https://www.microsoft.com/security/blog/2021/12/11/guidance-for-preventing-detecting-and-hunting-for-cve-2021-44228-log4j-2-exploitation/
- https://www.reddit.com/r/blueteamsec/comments/rd38z9/log4j_0day_being_exploited/
- https://www.lunasec.io/docs/blog/log4j-zero-day/
- https://msrc-blog.microsoft.com/2021/12/11/microsofts-response-to-cve-2021-44228-apache-log4j2/
- https://isc.sans.edu/forums/diary/Log4j+Log4Shell+Followup+What+we+see+and+how+to+defend+and+how+to+access+our+data/28122/
- https://github.com/curated-intel/Log4Shell-IOCs
- https://gist.github.com/Neo23x0/e4c8b03ff8cdf1fa63b7d15db6e3860b
- https://github.com/eromang/researches/tree/main/CVE-2021-44228
- https://blog.cloudflare.com/how-cloudflare-security-responded-to-log4j2-vulnerability/

Address

Level 8, 333 George Street
Sydney NSW 2000
Australia

Phone

T +61.2.9036.3560