sia

An Indra company

Hunting for log4shell compromises

2022 TF-CSIRT Meeting & FIRST Regional Symposium Europe

Table of contents

1. Introductions
2. A christmas tale
3. Some background
4. A sea and a bucket
5. A Tale of two organizations
6. A gift that keeps giving
7. Acknowledgments and indictments
8. Resources

Introductions

Who I am

I'm José Ángel García, Senior Incident Handler in SIA CERT. Involved with cyber security since my final year in university and work in different SOCs during my whole professional career since 2013.

Since 2018 I have been working for the SIA CERT, where we assist public and private organizations in the whole incident handling life cycle.

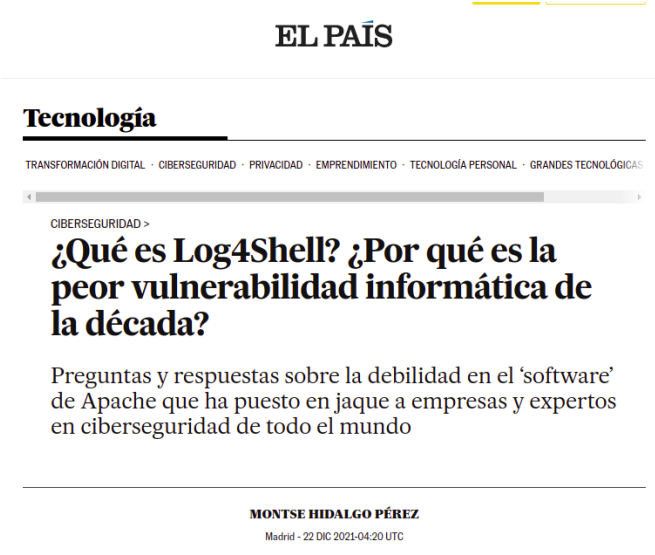
A christmas tale

Log4Shell is the Grinch

CVE-2021-44228 "log4shell" affecting java library log4j was revealed on December 10th 2021. Allowing an unskilled attacker to embed arbitrary code in the user agent field of a request

Notes:

- Revealed on Thursday and not trivial to patch
- Understaffed teams due to the impending holydays and COVID
- In most cases, real mitigation efforts weren't taken until Monday 13th
- So easy to exploit widespread vulnerability left alone for the weekend. What can go wrong?



EL PAÍS

Tecnología

TRANSFORMACIÓN DIGITAL · CIBERSEGURIDAD · PRIVACIDAD · EMPRENDIMIENTO · TECNOLOGÍA PERSONAL · GRANDES TECNOLÓGICAS

CIBERSEGURIDAD >

¿Qué es Log4Shell? ¿Por qué es la peor vulnerabilidad informática de la década?

Preguntas y respuestas sobre la debilidad en el 'software' de Apache que ha puesto en jaque a empresas y expertos en ciberseguridad de todo el mundo

MONTSE HIDALGO PÉREZ

Madrid - 22 DIC 2021-04:20 UTC

Why IRT was activated

- Public exploit available
- Rumors of log4shell used as 0 day on targeted attacks
- Strained detection teams couldn’t take point
- The IRT was activated to supplement local teams
- This is the story of how our team approached this task

Some background

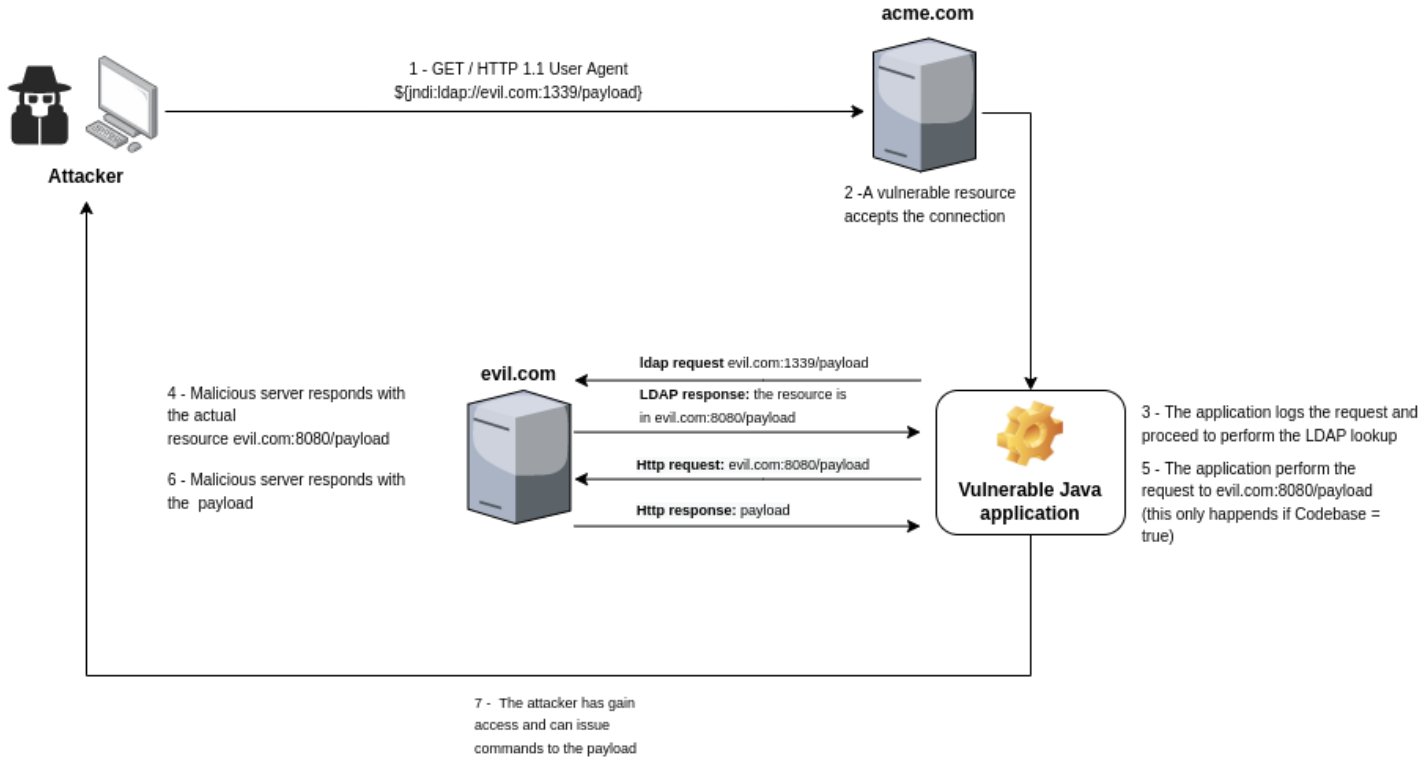
Apache Log4j is everywhere

- Apache log4j is one of the main logging frameworks for Java
- Not only used by webservers, also by other products such as:
  - **CISCO:** firepower, Webex, CloudCenter Suite Admin, Data Center Network Manager, IoT Control Center, Network Services Orchestrator, WAN Automation Engine, …
  - **AWS:** Amazon Web Services
  - **IBM:** IBM Java Runtime (Qradar: User Behavioral Analytics)
  - **Fortinet:** FortiCASB, FortiConverter Portal, and FortiCWP, FortiEDR
  - **VMware:** 40 VMware products are vulnerable to RCE

Timeline

- **2013 The lookup feature is introduced in log4j**
- **2014 Issues of compatibility arise an the %m{nolookups} is introduced**
- **2016 A Black Hat talk analyzes the risk of jndi lookups of untrusted resources**
  - LDAP: JNDI reference, Serialized Object, Remote location
  - RMI: JNDI reference, Remote Object
  - CORBA: IOR
  - Ref: https://www.blackhat.com/docs/us-16/materials/us-16-Munoz-A-Journey-From-JNDI-LDAP-Manipulation-To-RCE.pdf
- **December 1st 2021 first attempts of exploiting log4 as reported by CloudFlare**
- **December 10th 2021 Apache releases CVE-2021-44228 notification**

How Loj4shell works



A sea and a bucket

Compromise assessment vs threat hunting

Threat hunting

- It is a planned activity
- Uses methods of detection currently available
- Reports:
  - At the end of the activity
  - When malicious activity is found

Compromise assessment

- Not planned
- Supplementary methods are used
- Reports:
  - Daily reports
  - Notify any malicious activity

How to start

- Involve the local team as much as possible
- Check an entire organization is very different than to check a couple systems
  - There is a time limit
  - A plan is needed
- Several issues arise:
  - Web server logs are not indexed or centralized
  - Several appliances are also affected

Iterative approach

1 Identify useful log sources

2 Start with a set of basic detection rules

- In this case something like **jndi:ldap**

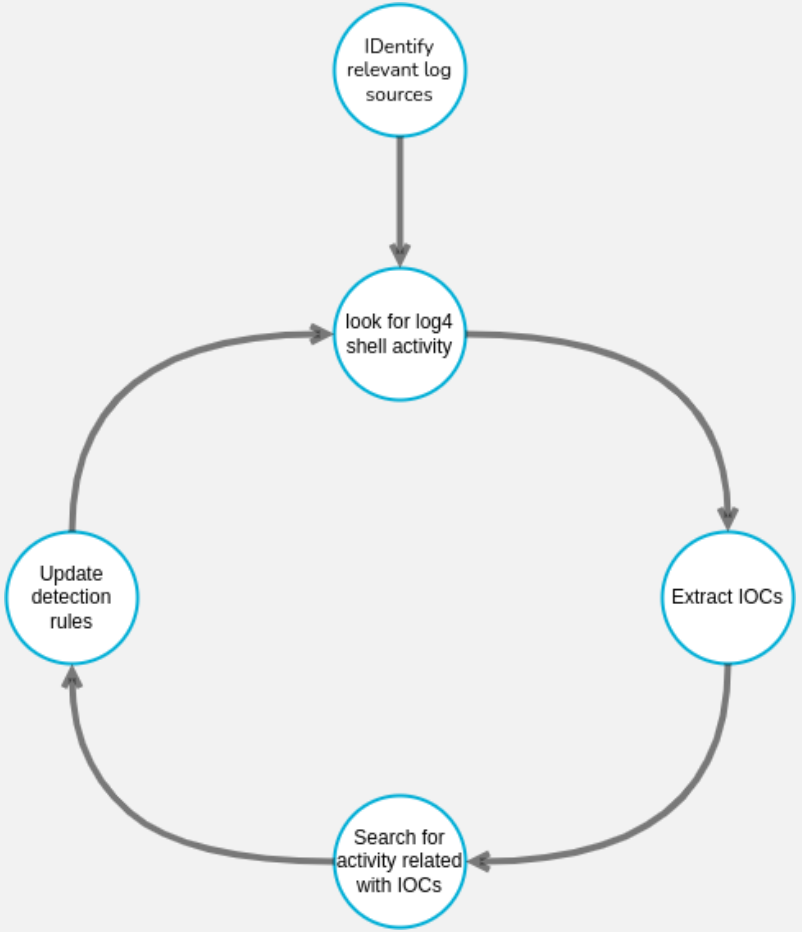
3 Identify the IOCs

4 Improve rule set to cover all log4shell activity ie:

- **${::-j}ndi:dns:**
- **$%7B$%7B::-j%7Dnd$%7B::-i%7D:ldap:**

5 Run the new rules

6 Back to step 2



Final rule

In the end a comprehensive detection rule will emerge

- This is just an example for a regulrar expression
  - Yara rules
  - SIEM queries
  - Splunk queries
  - Elastic queries

- Thanks to Florian Roth that provide us with the base
  - https://gist.github.com/Neo23x0/e4c8b03ff8cdf1fa63b7d15db6e3860b

```
(?:\$|%(?:25)*24|\\(?:0024|0{0,2}44))(?:{|%(?:25)*7[Bb]|\\(?:007[Bb]|0{0,2}173
)).{0,30}?((?:[Jj]|%(?:25)*[46][Aa]|\\(?:00[46][Aa]|0{0,2}1[15]2)).{0,30}?(?:[
Nn]|%(?:25)*[46][Ee]|\\(?:00[46][Ee]|0{0,2}1[15]6)).{0,30}?(?:[Dd]|%(?:25)*[46
]4|\\(?:00[46]4|0{0,2}1[04]4)).{0,30}?(?:[Ii]|%(?:25)*[46]9|\\(?:00[46]9|0{0,2
}1[15]1)|1).{0,30}?(?::|%(?:25)*3[Aa]|\\(?:003[Aa]|0{0,2}72)).{0,30}?((?:[L1]|
%(?:25)*[46][Cc]|\\(?:00[46][Cc]|0{0,2}1[15]4)).{0,30}?(?:[Dd]|%(?:25)*[46]4|\
\(?:00[46]4|0{0,2}1[04]4)).{0,30}?(?:[Aa]|%(?:25)*[46]1|\\(?:00[46]1|0{0,2}1[0
4]1)).{0,30}?(?:[Pp]|%(?:25)*[57]0|\\(?:00[57]0|0{0,2}1[26]0))(?:.{0,30}?(?:[S
s]|%(?:25)*[57]3|\\(?:00[57]3|0{0,2}1[26]3)))?|(?:[Rr]|%(?:25)*[57]2|\\(?:00[5
7]2|0{0,2}1[26]2)).{0,30}?(?:[Mm]|%(?:25)*[46][Dd]|\\(?:00[46][Dd]|0{0,2}1[15]
5)).{0,30}?(?:[Ii]|%(?:25)*[46]9|\\(?:00[46]9|0{0,2}1[15]1)|1)|(?:[Dd]|%(?:25)
*[46]4|\\(?:00[46]4|0{0,2}1[04]4)).{0,30}?(?:[Nn]|%(?:25)*[46][Ee]|\\(?:00[46]
[Ee]|0{0,2}1[15]6)).{0,30}?(?:[Ss]|%(?:25)*[57]3|\\(?:00[57]3|0{0,2}1[26]3))|(
?:[Nn]|%(?:25)*[46][Ee]|\\(?:00[46][Ee]|0{0,2}1[15]6)).{0,30}?(?:[Ii]|%(?:25)*
[46]9|\\(?:00[46]9|0{0,2}1[15]1)|1).{0,30}?(?:[Ss]|%(?:25)*[57]3|\\(?:00[57]3|
0{0,2}1[26]3))|(?:.{0,30}?(?:[Ii]|%(?:25)*[46]9|\\(?:00[46]9|0{0,2}1[15]1)|1))
{2}.{0,30}?(?:[Oo]|%(?:25)*[46][Ff]|\\(?:00[46][Ff]|0{0,2}1[15]7)).{0,30}?(?:[
Pp]|%(?:25)*[57]0|\\(?:00[57]0|0{0,2}1[26]0))|(?:[Cc]|%(?:25)*[46]3|\\(?:00[46
]3|0{0,2}1[04]3)).{0,30}?(?:[Oo]|%(?:25)*[46][Ff]|\\(?:00[46][Ff]|0{0,2}1[15]7
)).{0,30}?(?:[Rr]|%(?:25)*[57]2|\\(?:00[57]2|0{0,2}1[26]2)).{0,30}?(?:[Bb]|%(?
:25)*[46]2|\\(?:00[46]2|0{0,2}1[04]2)).{0,30}?(?:[Aa]|%(?:25)*[46]1|\\(?:00[46
]1|0{0,2}1[04]1))|(?:[Nn]|%(?:25)*[46][Ee]|\\(?:00[46][Ee]|0{0,2}1[15]6)).{0,3
0}?(?:[Dd]|%(?:25)*[46]4|\\(?:00[46]4|0{0,2}1[04]4)).{0,30}?(?:[Ss]|%(?:25)*[5
7]3|\\(?:00[57]3|0{0,2}1[26]3))|(?:[Hh]|%(?:25)*[46]8|\\(?:00[46]8|0{0,2}1[15]
0))(?:.{0,30}?(?:[Tt]|%(?:25)*[57]4|\\(?:00[57]4|0{0,2}1[26]4))){2}.{0,30}?(?:
[Pp]|%(?:25)*[57]0|\\(?:00[57]0|0{0,2}1[26]0))(?:.{0,30}?(?:[Ss]|%(?:25)*[57]3
|\\(?:00[57]3|0{0,2}1[26]3)))?).{0,30}?(?::|%(?:25)*3[Aa]|\\(?:003[Aa]|0{0,2}7
2)).{0,30}?(?:\/|%(?:25)*2[Ff]|\\(?:002[Ff]|0{0,2}57)|\${)|(?:[Bb]|%(?:25)*[46
]2|\\(?:00[46]2|0{0,2}1[04]2)).{0,30}?(?:[Aa]|%(?:25)*[46]1|\\(?:00[46]1|0{0,2
}1[04]1)).{0,30}?(?:[Ss]|%(?:25)*[57]3|\\(?:00[57]3|0{0,2}1[26]3)).{0,30}?(?:[
Ee]|%(?:25)*[46]5|\\(?:00[46]5|0{0,2}1[04]5)).{2,60}?(?::|%(?:25)*3[Aa]|\\(?:0
03[Aa]|0{0,2}72))(JH[s-v]|[\x2b\x2f-9A-Za-z][CSiy]R7|[\x2b\x2f-9A-Za-
z]{2}[048AEIMQUYcgkosw]ke[\x2b\x2f-9w-z]))
```

An Indra company

Filtering hits

- Everybody was trying to exploit the vulnerability
- Impossible to do a full DFIR on every impacted resource
- Better idea to look for the second stage:
  - Outgoing connections can be checked on the SIEM or EDR
  - However, direct access to the firewall logs is better
- A dedicated workstation or server for its analysis is advisable

A Tale of Two Organizations

Different folks, (mostly) same approach

- **All the previous is fine but no plan survives reality**
- **The following factors had to be accommodated:**
  - Different levels of maturity
  - Different sectors
  - Single country vs multinational
  - Organizational culture

Organization 1: old school but still cool

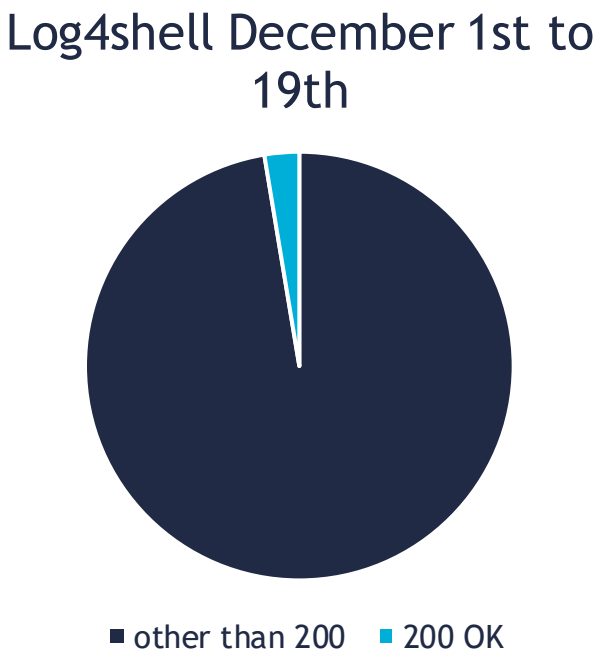
- **Healthcare sector**
- **Simple well though network architecture:**
  - Only one real egress point for the organization
  - 2 years of logs on everything
  - Everything in house: no cloud, no nonsense
- **Not all the old-school things are good:**
  - No remote access, the team was deployed in person
  - No EDR
  - No real SIEM to speak about (grep everything)

Organization 1: investigation results

Thousands of log4shell requests. No second stage activity was found, the request seem scan like.

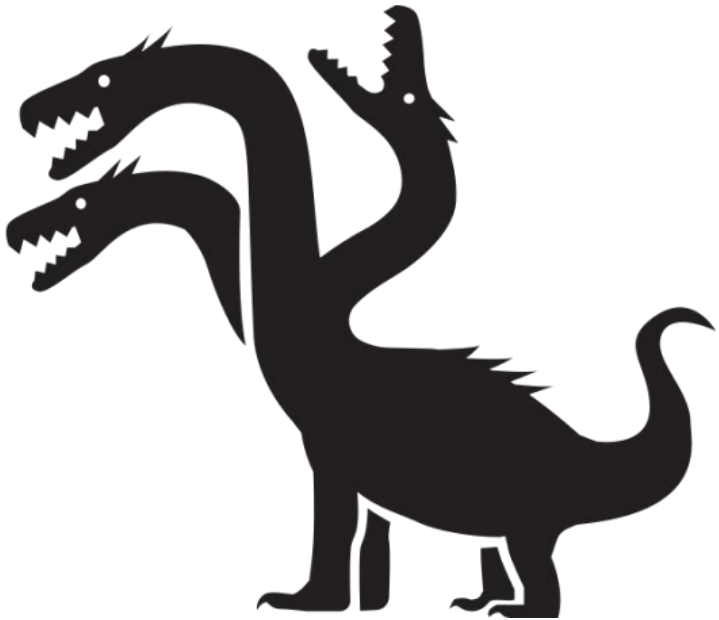
Three main sources of data:

- **Web server logs**
  - Regex and Yara search all of them
- **DMZ firewalls**
  - Look for the incoming IOCs
  - Look for the payload distribution IOCs
- **Proxy logs**
  - Outgoing connections to payload distribution IOCs



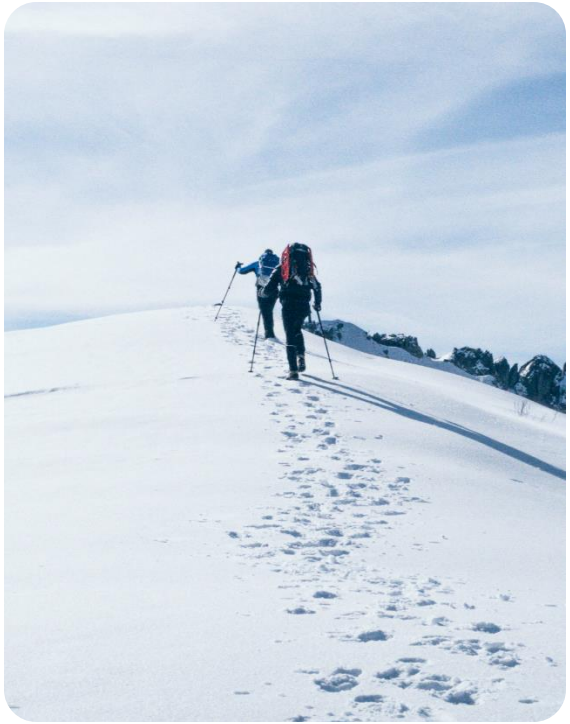
Organization 2: one body a hundred heads

- **Energy sector**
- **Complex network architecture**
  - Multiple egress points
  - Difficult to trace one request end to end
  - Shadow IT
  - Complex chain of command
- **Good usage of security technologies**
  - Good coverage of EDR
  - Real SIEM
  - Threat hunting processes already in place
  - Already established relation with the local team



Organization 2: adapt and overcome

- 2 challenges:
  - There is no full access to all the requests logs
  - Difficulty to trace back second stage activity
- Use EDR to look for exploitation related activity:
  - Commands related with Log4Shell exploitation
  - Connections or commands launched from Java parent process
  - Commands launched from Java parent process and contains an IP in command line
- PaloAlto Panorama and SIEM to trace back connections
  - Queries looking for activity associated with second stage IOCs were created and run

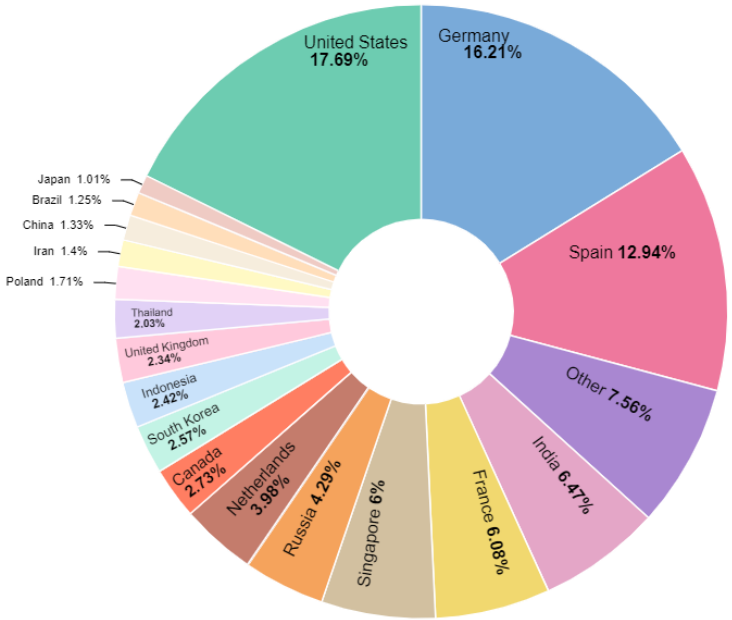


Organization 2: investigation results

No activity related to successful exploitation was found.

Main data sources:

- **Web server logs**
  - Application logs were indexed in ES
  - Regex all of them
- **EDR**
  - Look suspicious process activity
  - Outgoing connections to
- **PaloAlto Panorama**
  - Outgoing connections to payload distribution iocs



A gift that keeps giving

A vulnerability for years to come

We are going to see this vulnerability to compromise internal systems:

Mainly patched in the perimeter

Legacy environments may not allow the application of patches or mitigations

Shadow IT and incomplete CMDBs guarantee that this vulnerability will persist in the future

Use of illegal copies or out of support software that can't be patched

The vulnerability can be used for lateral movement by attackers

Acknowledgments

Acknowledgments and indictments

- To Alvaro Muñoz & Oleksandr Mirosh for they work on lookups an RCE in Java
- Florian Roth for the wealth of detection rules provided
- To my co-workers involved in this investigation
  - Jose Alberto López
  - Sergio Sanz
  - Javier Garcia
- To Abel Gonzalez to put me up for this talk

Resources

EDR Queries (CrowdStrike)

Descripción	Query
processes that run a possible Log4Shell payload	event_simpleName IN (ProcessRollup2, SyntheticProcessRollup2) | fields ProcessStartTime_decimal ComputerName FileName CommandLine | search CommandLine="*jndi:ldap:*" OR CommandLine="*jndi:rmi:*" OR CommandLine="*jndi:ldaps:*" OR CommandLine="*jndi:dns:*" OR CommandLine="*jndi:iiop:*" | rex field=CommandLine ".*(?<stringOfInterest>\$\{jndi\:(ldap|rmi|ldaps|dns|iiop)\:.*\}).*" | table ProcessStartTime_decimal ComputerName FileName stringOfInterest CommandLine
connections or processes launched from Java parent process	(index=main sourcetype=ProcessRollup* event_simpleName=ProcessRollup2 ParentBaseFileName IN (java) FileName IN (sh, bash, dash, ksh, tcsh, zsh, curl, python, ruby, php, wget)) OR (index=main sourcetype=NetworkConnectIP4 event_simplename=NetworkConnectIP4 RemotePort_decimal IN (1389, 389, 1099, 53, 5353, 80, 443)) | eval falconPID=mvappend(TargetProcessId_decimal,ContextProcessId_decimal) | table ComputerName ParentBaseFileName FileName CommandLine RemoteIP RemotePort_decimal
commands and scripts launched from Java process and contains an IP in command line	(index=main sourcetype=ProcessRollup* event_simpleName=ProcessRollup2 ParentBaseFileName IN (java) FileName IN (sh, bash, dash, ksh, tcsh, zsh, curl, python, ruby, php*, wget)) | where match (CommandLine, "([0-9]{1,3}[.]){3}[0-9]{1,3}") | eval falconPID=mvappend(TargetProcessId_decimal,ContextProcessId_decimal) | table ComputerName ParentBaseFileName FileName CommandLine RemoteIP RemotePort_decimal

ElastichSearch Queries

Descripción	Query
suspicious errors in application logs	message:"Error looking up JNDI resource" OR message:"FATAL log4j - Message: BadAttributeValueException: " OR message:"header with value of BadAttributeValueException: " OR message:".log4j.core.net.JndiManager.lookup(JndiManager"

Yara Queries

Descripción	Query
Set of yara rules for detecting log4shell	https://github.com/flypig5211/JNDIExploit

PaloAlto Panorama Query

Descripción	Query
Activity according to threatid	((threatid eq 91991) or (threatid eq 91994) or (threatid eq 91995))

PaloAlto Panorama Query

Descripción	Query
Linux systems	dpkg -l | egrep 'liblog4j|log4' find / -name log4j-core-*.jar Locate log4j | grep -v log4js Java 8 : versión < 2.17.x Java 7: versión < 2.12.2
Windows	$RESULT=(C:\ProgramData\checkmk\agent\bin\log4j2-scan.exe --all-drives) $FILES_VUL=((echo $RESULT | Select-String -Pattern "vulnerable files" | Select-String -Pattern "potentially vulnerable files" -NotMatch | Select-String -Pattern "Fixed" -NotMatch) -split ' ')[1] $FILES_POTVUL=((echo $RESULT | Select-String -Pattern "potentially vulnerable files") -split ' ')[1] $FILES_MIT=((echo $RESULT | Select-String -Pattern "mitigated files") -split ' ')[1] $SCANNED=(echo $RESULT | Select-String -Pattern "Scanned" | Select-String -Pattern "Running scan" -NotMatch) $RUNTIME=((echo $RESULT | Select-String -Pattern "Completed in") -split ' ')[2] $DRIVES_SCANNED=((echo $RESULT | Select-String -Pattern "Scanning drives") -split ' ')[2] $SHORT="Files: $FILES_VUL vulnerable, $FILES_POTVUL potentially vulnerable, $FILES_MIT mitigated, $SCANNED, Runtime: $RUNTIME s, drives: $DRIVES_SCANNED" $PERFDATA="vulnerable=$FILES_VUL;1;1|potential_vulnerable=$FILES_POTVUL;1;1\|mitigated=$FILES_MIT;;|real_time=${RUNTIME};;;1;" $LONG=$RESULT -join "\n" # if someone needs the count metric uncomment the next line (count and vulnerable metric are identical) #$PERFDATA="count=$FILES_VUL;1;1|vulnerable=$FILES_VUL;1;1\|potential_vulnerable=$FILES_POTVUL;1;1|mitigated=$FILES_MIT;;\|real_time=${RUNTIME};;;1;" echo "P CVE-2021-44228_log4j $PERFDATA $SHORT\n$LONG"

An Indra company

Thanks for your time

BEYOND CYBERSECURITY