# VPN环境搭建总结

## 1 搭建准备

购买的腾讯云（Tencent Cloud）服务器作为服务端搭建隧道服务，操作系统Ubuntu 16.04，VPS类型为KVM，云服务器节点位于香港。客户端使用Win10/Win7操作系统配置完成VPN连接。

下面的VPN搭建方式分手动配置和脚本搭建两种。手动配置方式为实验过程中的配置方式，因为多个VPN在同一服务器上搭建时，防止冲突需要自行手动配置，比较麻烦。在服务器上如果只搭建一种VPN服务，推荐使用脚本搭建方式。

## 2 PPTP

### 2.1 服务端配置

#### 2.1.1 手动配置方式

安装pptpd

$ sudo apt-get install pptpd

编辑配置文件：

$ sudo vi /etc/pptpd.conf

找到最下面，修改ip：

localip 192.168.0.1

remoteip 192.168.0.234-248,192.168.0.245

设置dns

$ sudo vi /etc/ppp/pptpd-options

修改以下部分为google的dns：

ms-dns 8.8.8.8

ms-dns 8.8.4.4

设置账号：

$ sudo vi /etc/ppp/chap-secrets

添加一行，依次为：用户名，服务，密码，限制ip：

"user" pptpd "user" \*

重启服务：

$ sudo /etc/init.d/pptpd restart

设置IP转发

打开这个文件

$ sudo vi /etc/sysctl.conf

去掉文件中这一行的注释：

net.ipv4.ip\_forward=1

使它立刻生效：

sudo sysctl -p

安装iptables，如果你还没有安装的话：

sudo apt-get install iptables

建立一个 NAT：

sudo iptables -t nat -A POSTROUTING -s 192.168.0.0/24-o eth0 -j MASQUERADE

# eth0 为外网网卡地址

Ubuntu系统下保存iptables操作

Iptables-save > /etc/iptables-rules

然后修改脚本/etc/network/interfaces，使系统能重启时自动应用这些规则

auto eth0

iface eth0 inet dhcp

pre-up iptables-restore < /etc/iptables-rules

最后，我们需要重启服务，让配置生效 .

sudo service pptpd restart

#### 2.1.2 脚本搭建方式

获取安装脚本

wget <https://raw.github.com/viljoviitanen/setup-simple-pptp-vpn/master/setup.sh>

执行脚本

Sudo sh setup.sh

修改用户名和密码

Vim /etc/ppp/chap-secrets

和手动配置方式相同，添加一行，依次为：用户名，服务，密码，限制ip：

"user" pptpd "user" \*

重启服务

service pptpd restart

### 2.2 客户端配置

Win10下打开“网络和Internet”设置，打开网络和共享中心，

设置新的连接或网络，选择连接到工作区，创建新连接。

在新连接中选择使用我的Internet连接。Internet地址填写VPS公网IP地址。点击创建。

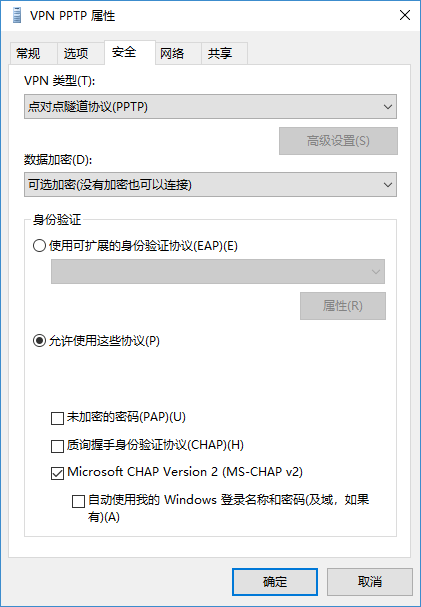
完成创建后在网络和共享中心中选择更改适配器设置，右键点击新建的VPN连接，选择属性。

在安全选项卡中选择VPN类型为点对点隧道协议（PPTP）

数据加密选择可选加密

并最后选择允许Microsoft CHAP Version 2(MS-CHAP v2)

连接VPN，输入之前PPTP服务设置好的用户名和密码即可完成登录。



## 3 IPSec/L2TP

### 3.1 服务端配置

#### 3.1.1 手动配置方式

安装IPSec

apt-get install openswan

修改IPSec配置文件/etc/ipsec.conf

config setup

nat\_traversal=yes

virtual\_private=%v4:10.0.0.0/8,%v4:192.168.0.0/16,%v4:172.16.0.0/12

oe=off

protostack=netkey

conn %default

forceencaps=yes

conn L2TP-PSK-NAT

rightsubnet=vhost:%no,%priv

also=L2TP-PSK-noNAT

conn L2TP-PSK-noNAT

authby=secret

pfs=no

auto=add

keyingtries=3

rekey=no

ikelifetime=8h

keylife=1h

type=transport

left= 填入自己linode的IP

leftprotoport=17/1701

right=%any

rightprotoport=17/%any

设置 PSK 预共享密钥

编辑/etc/ipsec.secrets，插入下面一行内容：

%any %any: PSK “test”

“”中内容为共享密钥，可以修改

对系统的网络策略进行一些调整

for each in /proc/sys/net/ipv4/conf/\*

do

echo 0 > $each/accept\_redirects

echo 0 > $each/send\_redirects

done

将上面这段代码完整地复制一次，加入到 /etc/rc.local 中，使其在每次系统启动时都生效。

重启一次 IPSec 服务

service ipsec restart

运行ipsec verify，结果如下：

ipsec verify

Checking your system to see if IPsec got installed and started correctly:  
Version check and ipsec on-path                                                [OK]  
Linux Openswan U2.6.37/K3.13.7-x86\_64-linode38 (netkey)  
Checking for IPsec support in kernel                                           [OK]  
SAref kernel support                                                                    [N/A]  
NETKEY: Testing XFRM related proc values                               [OK]  
         [OK]  
         [OK]  
Checking that pluto is running                                                       [OK]  
Pluto listening for IKE on udp 500                                                 [OK]  
Pluto listening for NAT-T on udp 4500                                           [OK]  
Two or more interfaces found, checking IP forwarding                  [OK]  
Checking NAT and MASQUERADEing                                          [OK]  
Checking for ‘ip’ command                                                             [OK]  
Checking /bin/sh is not /bin/dash                                                   [OK]  
Checking for ‘iptables’ command                                                   [OK]  
Opportunistic Encryption Support                                                  [DISABLED]

L2TP的部署

安装L2TP

apt-get install xl2tpd

编辑 L2TP 配置文件 /etc/xl2tpd/xl2tpd.conf

文件内容替换为下面的内容：

[global]

; listen-addr = 192.168.1.98

[lns default]

ip range = 10.1.1.2-10.1.1.255

local ip = 10.1.1.1

require chap = yes

refuse pap = yes

require authentication = yes

name = LinuxVPNserver

ppp debug = yes

pppoptfile = /etc/ppp/options.xl2tpd

length bit = yes

配置PPP

安装PPP

apt-get install pop

配置/etc/ppp/options.xl2tpd

先从 xl2tpd 文档中复制一个配置文件样例到我们的配置文件目录：

cp /usr/share/doc/xl2tpd/examples/ppp-options.xl2tpd /etc/ppp/options.xl2tpd

编辑/etc/ppp/options.xl2tpd，内容如下：

ipcp-accept-local

ipcp-accept-remote

ms-dns 8.8.8.8

ms-dns 8.8.4.4

noccp

auth

crtscts

idle 1800

mtu 1410

mru 1410

nodefaultroute

debug

name l2tpd

lock

proxyarp

connect-delay 5000

注意下这个name很重要，需要记住，后面需要用到。

添加用户账户，“账户”都在 /etc/ppp/chap-secrets 中

# Secrets for authentication using CHAP

# client server secret IP addresses

test l2tpd test \*

这里的server与上一步配置中的name必须保持一致。

重启xl2tpd

service xl2tpd restart

转发设置

启用转发设置：

编辑/etc/sysctl.conf，找到net.ipv4.ip\_forward=1这一行，将前面的#去掉，然后运行sysctl -p令其生效。

永久生效

编辑/etc/rc.local，添加下面一行：

iptables -t nat -A POSTROUTING -s 10.1.1.0/24 -o eth0 -j MASQUERADE

在终端运行一次上述 iptables 命令，即可令转发立即生效。

#### 3.1.2 脚本搭建方式

获取安装脚本

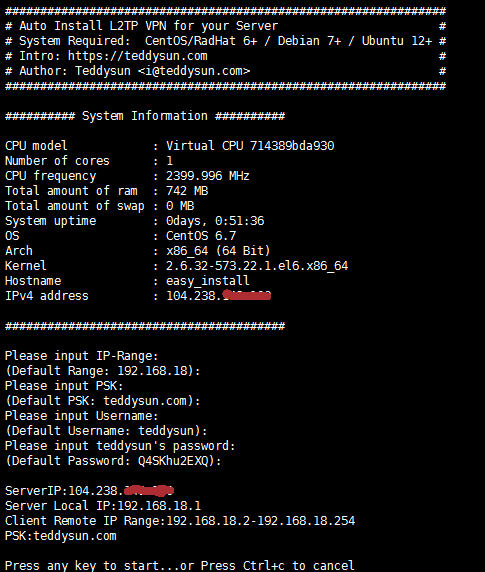
wget --no-check-certificate <https://raw.githubusercontent.com/teddysun/across/master/l2tp.sh>

执行脚本

chmod +x l2tp.sh

./l2tp.sh

执行后，会有如下交互界面



Please input IP-Range:

(Default Range: 192.168.18):

输入本地IP段范围（本地电脑连接到VPS后给分配的一个本地IP地址），直接回车意味着输入默认值192.168.18

Please input PSK:

(Default PSK: teddysun.com):

PSK意为预共享密钥，即指定一个密钥将来在连接时需要用到，直接回车意味着输入默认值teddysun.com

Please input Username:

(Default Username: teddysun):

Username意为用户名，即第一个默认用户。直接回车意味着输入默认值teddysun

Please input teddysun’s password:

(Default Password: Q4SKhu2EXQ):

输入用户的密码，默认会随机生成一个10位包含大小写字母和数字的密码，当然你也可以指定密码。

ServerIP:your\_server\_main\_IP

显示你的 VPS 的主 IP（如果是多 IP 的 VPS 也只显示一个）

Server Local IP:192.168.18.1

显示你的 VPS 的本地 IP（默认即可）

Client Remote IP Range:192.168.18.2-192.168.18.254

显示 IP 段范围

PSK:teddysun.com

显示 PSK

Press any key to start…or Press Ctrl+c to cancel

按下任意按键继续，如果想取消安装，请按Ctrl+c键

### 3.2 客户端配置

Win10下打开“网络和Internet”设置，打开网络和共享中心，

设置新的连接或网络，选择连接到工作区，创建新连接。

在新连接中选择使用我的Internet连接。Internet地址填写VPS公网IP地址。点击创建。

完成创建后在网络和共享中心中选择更改适配器设置，右键点击新建的VPN连接，选择属性。

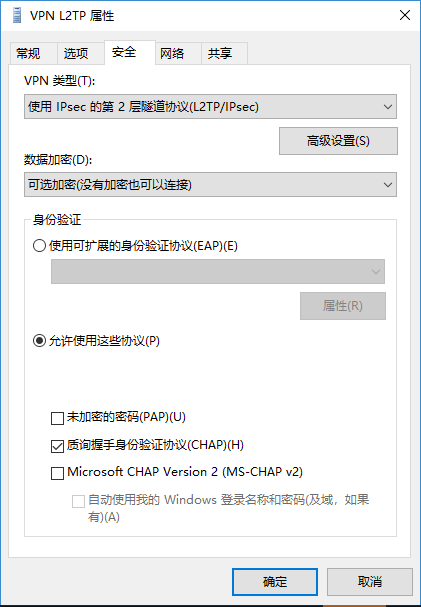
在安全选项卡中选择VPN类型为使用IPsec的第2层隧道协议（L2TP/IPsec）

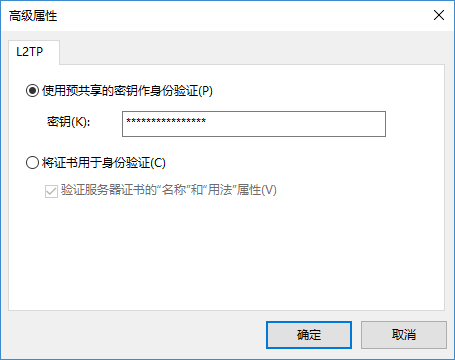
数据加密选择可选加密

并最后选择允许质询握手身份验证协议（CHAP）（H）

在高级设置中选择使用预共享的密钥作身份验证，输入之前设置的PSK

连接VPN，输入之前来l2tpd服务设置好的用户名和密码即可完成登录。





## 4 IPSec IKE

### 4.1 服务端配置

#### 4.1.1 手动配置方式

编译安装Strongswan

安装必须的库

apt-get update

apt-get install libpam0g-dev libssl-dev make gcc

下载strongswan并解压(\*代表当前Strongswan版本号)

wget http://download.strongswan.org/strongswan.tar.gz

tar xzf strongswan.tar.gz

cd strongswan-\*

编译Strongswan:

Xen、KVM使用以下参数:

./configure --enable-eap-identity --enable-eap-md5 \

--enable-eap-mschapv2 --enable-eap-tls --enable-eap-ttls --enable-eap-peap \

--enable-eap-tnc --enable-eap-dynamic --enable-eap-radius --enable-xauth-eap \

--enable-xauth-pam --enable-dhcp --enable-openssl --enable-addrblock --enable-unity \

--enable-certexpire --enable-radattr --enable-tools --enable-openssl --disable-gmp

编译并安装:

make; make install

编译完成后如果没有报错,且使用ipsec version指令能出现版本信息,则表示安装成功.

配置证书

生成CA证书的私钥

ipsec pki --gen --outform pem > ca.pem

使用私钥，签名CA证书

ipsec pki --self --in ca.pem --dn "C=com, O=myvpn, CN=VPN CA" --ca --outform pem >ca.cert.pem

提示:多个vps使用同个CA根证书:

如果需要多个vps使用同一个CA根证书,则以上两步只执行一次,之后所有vps都使用上面生成的这两个ca.pem和ca.cert.pem文件进行后续的操作.

然后把多台vps解析到同一个域名的不同二级域名下.

这样客户端连接各个服务器时,只需要客户端安装一次根证书ca.cert.pem即可.

生成服务器证书所需的私钥:

ipsec pki --gen --outform pem > server.pem

用CA证书签发服务器证书

请先确认你的服务器的IP地址或域名,以后客户端连接时只能使用证书中的地址连接(多服务器使用相同根证书CA的,请先做好服务器的域名解析),

然后将下面命令中的123.123.123.123替换为自己服务器的IP地址或域名,一共需要替换两处:

ipsec pki --pub --in server.pem | ipsec pki --issue --cacert ca.cert.pem \

--cakey ca.pem --dn "C=com, O=myvpn, CN=123.123.123.123" \

--san="123.123.123.123" --flag serverAuth --flag ikeIntermediate \

--outform pem > server.cert.pem

注意以上命令中的”C=”和”O=”的值要与第2步CA中的C,O的值保持一致.

生成客户端证书所需的私钥:

ipsec pki --gen --outform pem > client.pem

用CA签名客户端证书(C,O的值要与上面第2步CA的值一致,CN的值随意):

ipsec pki --pub --in client.pem | ipsec pki --issue --cacert ca.cert.pem --cakey ca.pem --dn "C=com, O=myvpn, CN=VPN Client" --outform pem > client.cert.pem

生成pkcs12证书:

openssl pkcs12 -export -inkey client.pem -in client.cert.pem -name "client" -certfile ca.cert.pem -caname "VPN CA" -out client.cert.p12

注意以上命令中的”-caname”后面的引号里的值必须要与第2步CA中的”CN=”的值保持一致.

安装证书:

cp -r ca.cert.pem /usr/local/etc/ipsec.d/cacerts/

cp -r server.cert.pem /usr/local/etc/ipsec.d/certs/

cp -r server.pem /usr/local/etc/ipsec.d/private/

cp -r client.cert.pem /usr/local/etc/ipsec.d/certs/

cp -r client.pem /usr/local/etc/ipsec.d/private/

配置Strongswan

编辑/usr/local/etc/ipsec.conf文件:

vim /usr/local/etc/ipsec.conf

修改为以下内容(点击展开):

config setup

uniqueids=never

conn iOS\_cert

keyexchange=ikev1

# strongswan version >= 5.0.2, compatible with iOS 6.0,6.0.1

fragmentation=yes

left=%defaultroute

leftauth=pubkey

leftsubnet=0.0.0.0/0

leftcert=server.cert.pem

right=%any

rightauth=pubkey

rightauth2=xauth

rightsourceip=10.31.2.0/24

rightcert=client.cert.pem

auto=add

conn android\_xauth\_psk

keyexchange=ikev1

left=%defaultroute

leftauth=psk

leftsubnet=0.0.0.0/0

right=%any

rightauth=psk

rightauth2=xauth

rightsourceip=10.31.2.0/24

auto=add

conn networkmanager-strongswan

keyexchange=ikev2

left=%defaultroute

leftauth=pubkey

leftsubnet=0.0.0.0/0

leftcert=server.cert.pem

right=%any

rightauth=pubkey

rightsourceip=10.31.2.0/24

rightcert=client.cert.pem

auto=add

conn windows7

keyexchange=ikev2

ike=aes256-sha1-modp1024!

rekey=no

left=%defaultroute

leftauth=pubkey

leftsubnet=0.0.0.0/0

leftcert=server.cert.pem

right=%any

rightauth=eap-mschapv2

rightsourceip=10.31.2.0/24

rightsendcert=never

eap\_identity=%any

auto=add

使用vim编辑/usr/local/etc/strongswan.conf文件:

charon {

load\_modular = yes

duplicheck.enable = no

compress = yes

plugins {

include strongswan.d/charon/\*.conf

}

dns1 = 8.8.8.8

dns2 = 8.8.4.4

nbns1 = 8.8.8.8

nbns2 = 8.8.4.4

}

include strongswan.d/\*.conf

使用vim编辑/usr/local/etc/ipsec.secrets文件:

: RSA server.pem

: PSK "myPSKkey"

: XAUTH "myXAUTHPass"

[用户名] %any : EAP "[密码]"

将上面的myPSKkey单词更改为你需要的PSK认证方式的密钥;

将上面的myXAUTHPass单词更改为你需要的XAUTH认证方式的密码,该认证方式的用户名是随意的;

将上面的[用户名]改为自己想要的登录名,[密码]改为自己想要的密码([]符号去掉),可以添加多行,得到多个用户,这即是使用IKEv2的用户名+密码认证方式的登录凭据.

提示:wp8.1客户端连接的用户名问题

由于wp8.1连接IKEv2的vpn时,默认会加上与手机名称相同的域,于是连接时会显示用户名或密码错误.这里有两种解决方法:

方法1:将上面/usr/local/etc/ipsec.secrets文件的最后一行改为%any %any : EAP “[密码]” ,这样就可以使用任意用户名登录,带上域也不会出错.

方法2:用FreeRADIUS过滤掉登录名的域,参考连接:让FreeRADIUS去掉登陆用户名中的Windows登录域

配置防火墙

编辑/etc/sysctl.conf，将net.ipv4.ip\_forward=1一行前面的#号去掉(否则Ikev2 vpn连接上后将无法访问外网)，保存后执行sysctl -p(如果执行后有报错的,重新打开sysctl.conf将报错的部分#注释掉保存,直到执行sysctl -p不再报错为止)。

此外,如果需要对TCP连接及速度进行进一步优化,可以参见本人的另一篇文章《各平台vps快速搭建shadowsocks及优化总结》中的TCP部分(注意不要把现有vpn配置的ip\_forward给覆盖了).

配置iptables:

Xen、KVM执行:

iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -A FORWARD -s 10.31.0.0/24 -j ACCEPT

iptables -A FORWARD -s 10.31.1.0/24 -j ACCEPT

iptables -A FORWARD -s 10.31.2.0/24 -j ACCEPT

iptables -A INPUT -i eth0 -p esp -j ACCEPT

iptables -A INPUT -i eth0 -p udp --dport 500 -j ACCEPT

iptables -A INPUT -i eth0 -p tcp --dport 500 -j ACCEPT

iptables -A INPUT -i eth0 -p udp --dport 4500 -j ACCEPT

iptables -A INPUT -i eth0 -p udp --dport 1701 -j ACCEPT

iptables -A INPUT -i eth0 -p tcp --dport 1723 -j ACCEPT

iptables -A FORWARD -j REJECT

iptables -t nat -A POSTROUTING -s 10.31.0.0/24 -o eth0 -j MASQUERADE

iptables -t nat -A POSTROUTING -s 10.31.1.0/24 -o eth0 -j MASQUERADE

iptables -t nat -A POSTROUTING -s 10.31.2.0/24 -o eth0 -j MASQUERADE

PS:如果VPS有 static IP ,可将上述防火墙规则的最后3条NAT规则替换为以下3条来提升处理效率:

iptables -t nat -A POSTROUTING -s 10.31.0.0/24 -o eth0 -j SNAT --to-source ELASTIC\_IP

iptables -t nat -A POSTROUTING -s 10.31.1.0/24 -o eth0 -j SNAT --to-source ELASTIC\_IP

iptables -t nat -A POSTROUTING -s 10.31.2.0/24 -o eth0 -j SNAT --to-source ELASTIC\_IP

注意将上述3条规则的ELASTIC\_IP替换为vps的静态ip地址.

开机自动载入iptables:

iptables-save > /etc/iptables.rules

cat > /etc/network/if-up.d/iptables<<EOF

#!/bin/sh

iptables-restore < /etc/iptables.rules

EOF

chmod +x /etc/network/if-up.d/iptables

至此,IPSec/IKEv2 VPN便搭建好了!现在启用服务就可以使用了:

启动服务

ipsec start

#### 4.1.2 脚本搭建方式

下载脚本:

Wget --no-check-certificate https://raw.githubusercontent.com/quericy/one-key-ikev2-vpn/master/one-key-ikev2.sh

注:如需使用其他分支的脚本,请将上述url中的master修改为分支名称,各分支区别详见本页的分支说明节点

运行脚本：

chmod +x one-key-ikev2.sh

bash one-key-ikev2.sh

等待自动配置部分内容后，选择vps类型（OpenVZ还是Xen、KVM），选错将无法成功连接，请务必核实服务器的类型。输入服务器ip或者绑定的域名(连接vpn时服务器地址将需要与此保持一致,如果是导入泛域名证书这里需要写\*.域名的形式)；

one-key-install-ikev2-vpn

选择使用使用证书颁发机构签发的SSL证书还是生成自签名证书：

如果选择no,使用自签名证书（客户端如果使用IkeV2方式连接，将需要导入生成的证书并信任）则需要填写证书的相关信息(C,O,CN)，为空将使用默认值(default value)，确认无误后按任意键继续,后续安装过程中会出现输入两次pkcs12证书的密码的提示(可以设置为空)

如果选择yes，使用SSL证书（如果证书是被信任的，后续步骤客户端将无需导入证书）请在继续下一步之前，将以下文件按提示命名并放在脚本相同的目录下（SSL证书详细配置和自动续期方案可见https://quericy.me/blog/860/ ）：

ca.cert.pem 证书颁发机构的CA，比如Let‘s Encrypt的证书,或者其他链证书；

server.cert.pem 签发的域名证书；

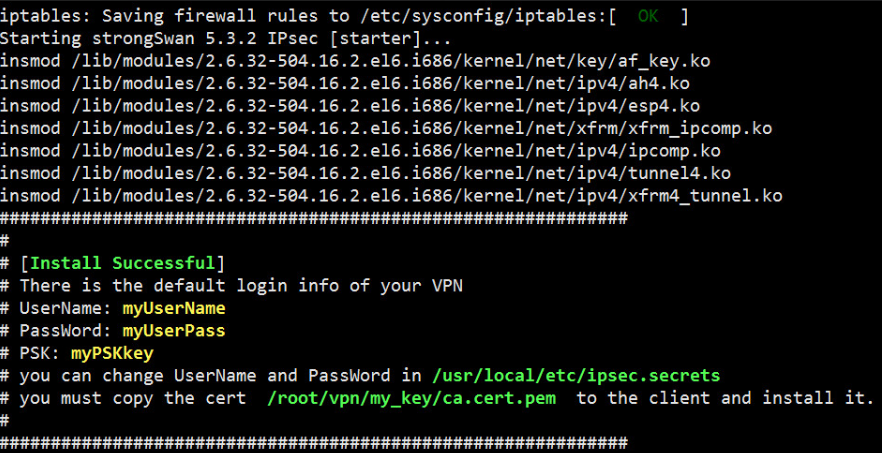
server.pem 签发域名证书时用的私钥；

是否使用SNAT规则(可选).默认为不使用.使用前请确保服务器具有不变的静态公网ip,可提升防火墙对数据包的处理速度.如果服务器网络设置了NAT(如AWS的弹性ip机制),则填写网卡连接接口的ip地址(参见KinonC提供的方案:#36).

防火墙配置.默认配置iptables(如果使用的是firewall(如CentOS7)请选择yes自动配置firewall,将无视SNAT并跳过后续的补充网卡接口步骤).补充网卡接口信息,为空则使用默认值(Xen、KVM默认使用eth0,OpenVZ默认使用venet0).如果服务器使用其他公网接口需要在此指定接口名称,填写错误VPN连接后将无法访问外网)

看到install Complete字样即表示安装完成。默认用户名密码将以黄字显示，可根据提示自行修改配置文件中的用户名密码,多用户则在配置文件中按格式一行一个(多用户时用户名不能使用%any),保存并重启服务生效。

将提示信息中的证书文件ca.cert.pem拷贝到客户端，修改后缀名为.cer后导入。ios设备使用Ikev1无需导入证书，而是需要在连接时输入共享密钥，共享密钥即是提示信息中的黄字PSK.



### 4.2 客户端配置

将服务器上安装目录下的my\_key文件夹中的ca.cert.pem传给win10客户端

将ca.cert.pem后缀改为.cer后，双击安装证书，导入证书需要导入到“本地计算机”的”受信任的根证书颁发机构”，以”当前用户”的导入方式是无效的。推荐运行mmc添加本地计算机的证书管理单元来操作。

Win10下打开“网络和Internet”设置，打开网络和共享中心，

设置新的连接或网络，选择连接到工作区，创建新连接。

在新连接中选择使用我的Internet连接。Internet地址填写VPS公网IP地址。点击创建。

完成创建后在网络和共享中心中选择更改适配器设置，右键点击新建的VPN连接，选择属性。

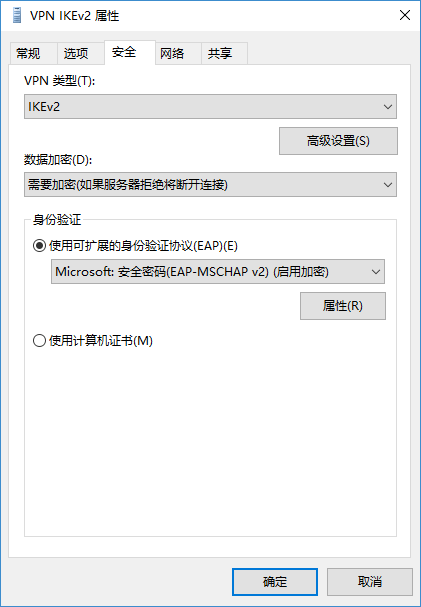
在安全选项卡中选择VPN类型为IKEv2

数据加密选择需要加密

并最后选择身份验证中使用可扩展的身份验证协议（EAP）（E）

在选择栏里选择Microsoft：安全密码（EAP-MSCHAP v2）（启用加密）

连接VPN，输入之前ipsec服务设置好的用户名和密码即可完成登录。



## 5 OpenVPN

### 5.1 服务端配置

#### 5.1.1 手动配置方式

安装OpenVPN

OpenVPN在Ubuntu的默认仓库中是可用的，所以我们可用使用apt来安装。我们还需安装一个easy-rsa包，这个包可以帮助我们建立一个内部CA（certificate authority）用于使用我们VPN。

更新你的服务器包索引并安装必要的包类型：

$ sudo apt-get update

$ sudo apt-get install openvpn easy-rsa

现在你的服务器上面有了所需的软件，接下来是相关的配置。

建立CA目录

OpenVPN是一个TLS/SSLVPN，这意味着它需要使用证书来在客户端和服务器之间加密数据。为了发布可信的证书，我需要建立我们自己的简单CA（certificate authority）。

开始之前，我们可以使用make-cadir命令，用于复制easy-rsa临时目录到我们的home目录下面：

$ make-cadir ~/openvpn-ca

上面这条命令也用下面的两条命令来做：

$ mkdir ~/openvpn-ca

$ cp -r /usr/share/easy-rsa/\* ~/openvpn-ca

然后进入到我们新创建的目录中来开始配置CA：

$ cd ~/openvpn-ca

配置CA变量

我们需要编辑当前目录下vars文件来配置CA要用到的值，用你的文本编辑器打开vars文件：

$ nano vars

在这个文件里面，你会发现许多变量，这些变量用于决定怎样生成你的证书，你可以修改这些变量的值。在这里我们只需要关注几个变量就行。

在文件的底部，找到如下信息：

~/openvpn-ca/vars

. . .

exportKEY\_COUNTRY="US"

exportKEY\_PROVINCE="CA"

exportKEY\_CITY="SanFrancisco"

exportKEY\_ORG="Fort-Funston"

exportKEY\_EMAIL="me@myhost.mydomain"

exportKEY\_OU="MyOrganizationalUnit"

. . .

将这些红色的值编辑成任何你喜欢的，但不要让它们空着，比如我的：

~/openvpn-ca/vars

. . .

export KEY\_COUNTRY="CN"

export KEY\_PROVINCE="GD"

export KEY\_CITY="ShanTou City"

export KEY\_ORG="STU"

export KEY\_EMAIL="qfuqin@163.com"

export KEY\_OU="University"

. . .

到这步之后，我们再编辑KEY\_NAME的值，简单起见，我们将它命名为server，如下：

~/openvpn-ca/vars

export KEY\_NAME="server"

完成之后，保存并退出。

制作CA

现在我们可以使用我们刚刚设置的变量，用easy-rsa包来制作CA。确保你处在你的CA目录下面，然后source我们刚刚编辑的vars文件。

$ cd ~/openvpn-ca

$ source vars

如果source正确的话，你会看到如下信息：

NOTE: If you run ./clean-all, I will be doing a rm -rf on /home/sammy/openvpn-ca/keys

通过输入如下命令确保我们的操作处于一个clean环境中:

$ ./clean-all

现在我们可以制作我们的root CA：

$ ./build-ca

上面这一步会开始制作根证书颁发机构密钥（rootcertificate authority key ）和证书（certificate），由于我们刚刚填了vars文件，证书制作所需要变量的值都会自动填充，制作过程中你只需要回车来确认就行。消息输出如下：

Generating a 2048 bit RSA private key

..........................................................................................+++

...............................+++

writing new private key to 'ca.key'

-----

You are about to be asked to enter information that willbe incorporated

into your certificate request.

What you are about to enter is what is called aDistinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

-----

Country Name (2 letter code) [CN]:

State or Province Name (full name) [GD]:

Locality Name (eg, city) [ShanTou City]:

Organization Name (eg, company) [STU]:

Organizational Unit Name (eg, section) [University]:

Common Name (eg, your name or your server's hostname) [STUCA]:

Name [server]:

Email Address [qfuqin@163.com]:

现在我们得到了一个CA文件，CA文件可以用于制作我们接下来所需的文件。

制作Server端的 Server Certificate, Key, Encryption Files

接下来，我们将制作服务端所需要的证书，这些证书就像传统的用于加密过程的文件一样。通过键入如下命令来生成服务端所需的证书：

$ ./build-key-server

上面红色字体的server就是我们在vars文件中填入的export KEY\_NAME="server"，如果你不是填入的server这个名称，则./build-key-server后面输入你自己填的那个名称。制作过程中一路回车，中间出现 challenge password ，不要输入任何值回车就行，最后的会有两个问题，输入y就行，如下所示：

. . .

Certificate is to be certified until May 1 17:51:16 2026 GMT (3650 days)

Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y

Write out database with 1 new entries

Data Base Updated

如果以上操作无误的话应该可以看到生成的server.crt、server.key和server.csr三个文件。其中server.crt和server.key两个文件是我们所需要的。现在再为服务器生成加密交换时的Diffie-Hellman文件。输入以下命令：

$ ./build-dh

完成这一步可能需要几分钟时间。

最后，我们可以生成一个HMAC签名来增强服务器的TLS完整性验证能力:

$ openvpn --genkey --secret keys/ta.key

制作Client端的 Client Certificate, Key

尽管客户端的相关证书可以在客户端的机器上面生成，为了简单起见，在这篇教程中我们在服务器上面来生成客户端的相关证书，然后再把服务器上生成的客户端证书下载到本地客户端上面。我们用client1来命名我们的第一个证书/密钥对，用build-key命令来生成没有密码情况下的凭证，并且用于自动连接：

$ cd ~/openvpn-ca

$ source vars

$ ./build-key

如果你想生成一个带密码保护的凭证，可以使用build-key-pass命令：

$ cd ~/openvpn-ca

$ source vars

$ ./build-key-pass

然后一路回车，中间出现challenge password，不要输入任何值回车就行，最后的会有两个问题，输入y就行。

配置OpenVPN服务器

下来，我们利用已经生成的相关文件来配置OpenVPN服务器。

把相关证书复制到OpenVPN的目录下面

开始之前，把我们需要的相关文件复制到/etc/openvpn这个配置目录中去，即把~/openvpn-ca/keys目录下面的ca.crt，ca.key，server.crt，server.key，HMAC签名以及Diffie-Hellman文件复制到/etc/openvpn这个目录下面：

$ cd ~/openvpn-ca/keys

$ sudo cp ca.crt ca.key server.crt server.key ta.key dh2048.pem /etc/openvpn

然后从OpenVPN自带的配置模板中复制配置文件:

$ cp /usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz /etc/openvpn/

$ cd /etc/openvpn/

$ gzip -d server.conf.gz

修改OpenVPN配置文件

$ sudo nano /etc/openvpn/server.conf

基本配置步骤如下：

首先，通过查找tls-auth指令找到HMAC部分，移除";"来解注释tls-auth，并且在tls-auth的下面，增加一个key-direction参数，设置其参数值为0，如下：

/etc/openvpn/server.conf

tls-auth ta.key 0 # This file is secret

key-direction 0

然后，通过查找被注释的clipher这一行找到加密密码的部分，aes128-cbc密码提供了很好的加密级别，并且得到了很好的支持。移除";"来解注释cipher AES-128-CBC，如下：

/etc/openvpn/server.conf

cipher AES-128-CBC

在这一行的下面，添加一个auth行（身份验证行）来选择HMAC消息摘要算法，这里推荐SHA256消息摘要算法：

etc/openvpn/server.conf

auth SHA256

最后，找到user和group参数，去除它们之前的";"，如下：

/etc/openvpn/server.conf

user nobody

group nogroup

（可选配置）推动DNS更改让VPN重定向所有流量

上面的配置可以在客户端和服务器端上创建VPN连接，但是没有强迫连接去使用tunnel。如果你希望用VPN来路由你的所有流量，你需要更改你的客户端机器的DNS设置。

你可以按照以下步骤设置，解注释一些指令使得你的客户端机器把所有的web流量重定向到VPN上面，找到redirect-gateway部分然后移除它之前的";"，如下：

/etc/openvpn/server.conf

push "redirect-gateway def1 bypass-dhcp"

在这条指令的下面，找到dhcp-option部分，移除这两行之前的";"，如下：

push "dhcp-option DNS 223.5.5.5"

push "dhcp-option DNS 114.114.114.114"

这就可以协助客户版重新配置DNS，以便使用VPN tunnel来作为默认网关。

（可选配置）修改OpenVPN服务器的端口和协议

OpenVPN服务器默认使用1194端口和UDP协议来接收客户端的连接，也许由于客户端那边的网络环境限制，你需要使用一个不同的端口，那么你可以改变port选项，如果你的Ubuntu16.04服务器没有托管web服务，那么443端口是一个可替换1194的不错选择，因为防火墙往往对443端口不受限。

/etc/openvpn/server.conf

# Optional!

port 443

我们同样可以将协议从UDP换成TCP：

/etc/openvpn/server.conf

# Optional!

proto tcp

如果你没有更换端口的需求，最好将上述的两项保持默认设置。

（可选配置）指定非默认的凭证（Point to Non-Default Credentials）

如果你在之前的./build-key-server命令用了不同的名字（我们之前用的server这个名字），那么修改cert和key这两行，将这两行的值设为你之前用的那个名字.crt和你之前那个名字.key，如果你默认使用的server这个名字，那么这里你已经正确的设置好了，如下：

/etc/openvpn/server.conf

cert server.crt

key server.key

以上设置完毕之后，保存并退出server.conf这个文件。

调整Ubuntu16.04服务的网络配置

接下来，我们需要调整Ubuntu16.04服务器上面网络的一些配置，以便于OpenVPN服务器可以正确的路由流量。

允许IP转发

首先，我们需要让我们的服务器来转发流量，这是我们需要VPN服务器来提供的最基本的功能。我们可以通过修改/etc/sysctl.conf文件来调整网络设置：

$ sudo nano /etc/sysctl.conf

在这个文件里面，找到net.ipv4.ip\_forward，去除这一行之前的"#"来解注释这个参数：

etc/sysctl.conf

net.ipv4.ip\_forward=1

完成后保存并退出该文件。

为了读取sysctl.conf文件并且让调整后设置对当前系统的session生效，键入如下命令：

$ sudo sysctl -p

调整防火墙（UFW）规则来伪装客户端的连接

在这篇教程中我们需要配置防火墙规则来引导进入服务器的一些流量，我们需要修改防火墙规则文件来建立伪装规则，iptables的概念用于提供动态的NAT，从而正确地路由客户端连接。在打开防火墙配置文件以添加伪装规则之前，我们需要找到我们Ubuntu服务器的公共网络接口，输入如下命令：

$ ip route | grep default

你的公共网络接口应当紧跟在单词"dev"后面，例如，我的接口名字为eth0：

default via 172.18.31.253 dev eth0

当你有一个与你的默认路由相关联的接口的时候，打开/etc/ufw/before.rules这个文件并添加相应的规则：

$ sudo nano /etc/ufw/before.rules

这个文件处理在加载常规UFW规则之前应该被放置的文件，在这个文件的最开始处加入下面红色部分的内容。这样可以为nat表设置POSTROUTING默认规则，并且为来自VPN的任何流量设置伪装连接。

/etc/ufw/before.rules

#

# rules.before

#

# Rules that should be run before the ufw command lineadded rules. Custom

# rules should be added to one of these chains:

# ufw-before-input

# ufw-before-output

# ufw-before-forward

#

# START OPENVPN RULES

# NAT table rules

\*nat

:POSTROUTING ACCEPT [0:0]

# Allow traffic from OpenVPN client to eth0(changeto the interface you discovered!)

-A POSTROUTING -s 10.8.0.0/8 -o eth0 -jMASQUERADE

COMMIT

# END OPENVPN RULES

# Don't delete these required lines, otherwise there willbe errors

\*filter

. . .

完成后保存并退出。

然后告诉防火墙默认允许转发包：

$ sudo nano /etc/default/ufw

在这个文件里面，找到DEFAULT\_FORWARD\_POLICY指令，将它的值从DROP改成ACCEPT：

/etc/default/ufw

DEFAULT\_FORWARD\_POLICY="ACCEPT"

完成后保存并退出。

打开OpenVPN端口并且使变化生效

　　接下来，调整防火墙本身，以允许流量到OpenVPN。如果你在/etc/openvpn/server.conf文件中没有修改OpenVPN的端口号和协议类型，那么直接配置防火墙允许UDP流量到1194端口，如果你改变了端口和协议类型，那么根据你自己设置的端口和协议类型进行配置。

$ sudo ufw allow 1193/udp

$ sudo ufw allow OpenSSH

现在，我们可以从所有修改过的文件中装载配置来关闭和重启防火墙：

$ sudo ufw disable

$ sudo ufw enable

到这里我们的服务器可以正确地处理OpenVPN流量了。

开启OpenVPN服务

在systemd单元文件的后面，我们通过指定特定的配置文件名来作为一个实例变量来开启OpenVPN服务，我们的配置文件名称为/etc/openvpn/server.conf，所以我们在systemd单元文件的后面添加@server来开启OpenVPN服务:

$ sudo systemctl start openvpn@server

通过如下命令再次确认OpenVPN服务已经成功地开启了：

$ sudo systemctl status openvpn@server

如果一切正常的话，你的输出应当跟如下类似:

● openvpn@server.service - OpenVPN connection to server

Loaded: loaded(/lib/systemd/system/openvpn@.service; disabled; vendor preset: enabled)

Active: active (running) since Tue 2016-05-03 15:30:05 EDT;47s ago

Docs:man:openvpn(8)

https://community.openvpn.net/openvpn/wiki/Openvpn23ManPage

https://community.openvpn.net/openvpn/wiki/HOWTO

Process: 5852ExecStart=/usr/sbin/openvpn --daemon ovpn-%i --status /run/openvpn/%i.status 10--cd /etc/openvpn --script-security 2 --config /etc/openvpn/%i.conf --writepid/run/openvpn/%i.pid (code=exited, sta

Main PID: 5856(openvpn)

Tasks: 1(limit: 512)

CGroup:/system.slice/system-openvpn.slice/openvpn@server.service

└─5856/usr/sbin/openvpn --daemon ovpn-server --status /run/openvpn/server.status 10--cd /etc/openvpn --script-security 2 --config /etc/openvpn/server.conf--writepid /run/openvpn/server.pid

May 03 15:30:05 openvpn2 ovpn-server[5856]: /sbin/ip addradd dev tun0 local 10.8.0.1 peer 10.8.0.2

May 03 15:30:05 openvpn2 ovpn-server[5856]: /sbin/iproute add 10.8.0.0/24 via 10.8.0.2

May 03 15:30:05 openvpn2 ovpn-server[5856]: GID set tonogroup

May 03 15:30:05 openvpn2 ovpn-server[5856]: UID set tonobody

May 03 15:30:05 openvpn2 ovpn-server[5856]: UDPv4 linklocal (bound): [undef]

May 03 15:30:05 openvpn2 ovpn-server[5856]: UDPv4 linkremote: [undef]

May 03 15:30:05 openvpn2 ovpn-server[5856]: MULTI:multi\_init called, r=256 v=256

May 03 15:30:05 openvpn2 ovpn-server[5856]: IFCONFIGPOOL: base=10.8.0.4 size=62, ipv6=0

May 03 15:30:05 openvpn2 ovpn-server[5856]: IFCONFIG POOLLIST

May 03 15:30:05 openvpn2 ovpn-server[5856]:Initialization Sequence Completed

你可以通过如下命令来确认OpenVPN tun0接口是否可用

$ ip addr show tun0

你应该可以看到一个配置接口：

4: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER\_UP>mtu 1500 qdisc noqueue state UNKNOWN group default qlen 100

link/none

inet 10.8.0.1peer 10.8.0.2/32 scope global tun0

valid\_lftforever preferred\_lft forever

如果一切运行正常，将OpenVPN设置为开机自启动：

$ sudo systemctl enable openvpn@server

构造客户端所需的.ovpn文件配置工厂

这一步，我们建立一个更加容易生成客户端所需配置文件的系统。

创建客户端配置目录结构

在你的home目录下面创建一个目录结构来保存客户端的相应配置文件：

$ mkdir -p ~/client-configs/files

由于客户端的配置目录里面将包含客户端的密钥，所以我们应该对这个目录进行权限的锁定：

$ chmod 700 ~/client-configs/files

创建基本的配置文件

这一步，我们将OpenVPN自带的客户端配置模板复制到我们刚刚创建的客户端配置目录中作为一个客户端的基本配置文件：

$ cp /usr/share/doc/openvpn/examples/sample-config-files/client.conf ~/client-configs/base.conf

打开base.conf并做一些修改：

首先，找到remote指令，这条指令用于指定OpenVPN服务器所在的地址，这个地址应当是你的OpenVPN所在的服务器的公网IP地址。如果你改变了OpenVPN监听的端口，那么把1194换成你所设置的端口：

~/client-configs/base.conf

. . .

# The hostname/IP and port of the server.

# You can have multiple remote entries

# to load balance between the servers.

remote server\_IP\_address 1194

. . .

确保你在server配置文件(/etc/openvpn/server.conf文件)里面你所设置的协议类型：

~/client-configs/base.conf

proto udp

然后去除user和group指令前面的";"，如下：

~/client-configs/base.conf

# Downgrade privileges after initialization (non-Windowsonly)

user nobody

group nogroup

然后找到设置ca，cert和key的指令，将这些指令注释掉，因为我们会在这个配置文件里面自己设置certs和keys的值：

~/client-configs/base.conf

# SSL/TLS parms.

# See the server config file for more

# description. It's best to use

# a separate .crt/.key file pair

# for each client. A single ca

# file can be used for all clients.

#ca ca.crt

#cert client.crt

#key client.key

然后镜像（或者说模仿）我们在/etc/openvpn/server.conf文件里面设置的cipher和auth：

~/client-configs/base.conf

cipher AES-128-CBC

auth SHA256

接下来，在文件的某位置处增加key-direction指令，这个指令的值必须设置为1才能与服务器合作：

~/client-configs/base.conf

key-direction 1

最后添加一些注释信息，我们希望配置信息能够用于所有的客户端，但是下面的注释信息只能用于Linux客户端：

~/client-configs/base.conf

# script-security 2

# up /etc/openvpn/update-resolv-conf

# down /etc/openvpn/update-resolv-conf

如果你的客户端运行在Linux上面并且有一个/etc/openvpn/update-resolv-conf文件，你应当将上面的三条指令解注释。

最后保存并退出base.conf文件。

写一个自动生成客户端.ovpn配置文件的脚本

这一步，我们写一个简单的脚本，用于将相关的certificate，key和加密文件编译我们刚刚制作的base.conf文件，编译完成后得到一个.ovpn客户端文件。

在~/client-configs目录下面创建make\_config.sh文件:

$ nano ~/client-configs/make\_config.sh

在将如下内容复制到make\_config.sh文件中：

#!/bin/bash

# First argument: Client identifier

KEY\_DIR=/etc/openvpn/easy-rsa/keys

OUTPUT\_DIR=~/client-configs/files

BASE\_CONFIG=~/client-configs/base.conf

cat ${BASE\_CONFIG} \

<(echo -e '<ca>') \

${KEY\_DIR}/ca.crt \

<(echo -e '</ca>\n<cert>') \

${KEY\_DIR}/<span style="color:#FF0000;">client1</span>.crt \

<(echo -e '</cert>\n<key>') \

${KEY\_DIR}/<span style="color:#FF0000;">client1</span>.key \

<(echo -e '</key>\n<tls-auth>') \

${KEY\_DIR}/ta.key \

<(echo -e '</tls-auth>') \

> ${OUTPUT\_DIR}/${1}.ovpn

完成后保存并关闭该文件，通过以下命令标记该文件为可执行文件：

$ chmod 700 ~/client-configs/make\_config.sh

生成客户端.ovpn配置文件

现在我们可以轻松地生成客户端配置文件，只需要如下简单的命令即可：

$ cd ~/client-configs

$ ./make\_config.sh

如果一切顺利的话，在~/client-configs/files目录下面会生成一个client-for-android.ovpn文件，client-for-android是随意起的名字，你也可以按照你自己的想法命名，最后在~/client-configs/files文件夹内会生成你所命名的.ovpn文件。

接下来把client-for-android.ovpn下载到你的客户端即可，这里可以使用相应的工具下载，我本人是在Ubuntu16.04里面搭建了vsftpd服务，所以传输文件比较方便。

在客户端OpenVPN上安装配置文件

这个步骤比较简单，先根据你的操作系统平台选择下载相应的客户端OpenVPN，然后把上一步中的.ovpn配置文件导入到客户端的OpenVPN即可，正常情况下可以直接连接到服务器端的OpenVPN。

#### 5.1.2 脚本搭建方式

获取安装脚本

wget <https://github.com/Nyr/openvpn-install/raw/master/openvpn-install.sh>

执行脚本

bash openvpn-install.sh

运行此条命令bash openvpn-install.sh，最后显示：

…

Generating a 2048 bit RSA private key

…………….+++

………………………..+++

writing new private key to ‘/etc/openvpn/easy-rsa/pki/private/server.key.bMvlQu5Gie’

—–

Using configuration from /etc/openvpn/easy-rsa/openssl-1.0.cnf

Check that the request matches the signature

Signature ok

The Subject’s Distinguished Name is as follows

commonName :PRINTABLE:’server’

Certificate is to be certified until Jul 2 04:22:38 2027 GMT (3650 days)

Write out database with 1 new entries

Data Base Updated

Generating a 2048 bit RSA private key

………..+++

……………………………………+++

writing new private key to ‘/etc/openvpn/easy-rsa/pki/private/client2.key.7nwUyrULSK’

—–

Using configuration from /etc/openvpn/easy-rsa/openssl-1.0.cnf

Check that the request matches the signature

Signature ok

The Subject’s Distinguished Name is as follows

commonName :PRINTABLE:’client2′

Certificate is to be certified until Jul 2 04:22:39 2027 GMT (3650 days)

Write out database with 1 new entries

Data Base Updated

Using configuration from /etc/openvpn/easy-rsa/openssl-1.0.cnf

An updated CRL has been created.

CRL file: /etc/openvpn/easy-rsa/pki/crl.pem

\* Stopping virtual private network daemon(s)… \* No VPN is running.

\* Starting virtual private network daemon(s)… \* Autostarting VPN ‘server’

Finished!

Your client configuration is available at /root/client2.ovpn

If you want to add more clients, you simply need to run this script again!

在脚本执行时会进行交互式的选择OpenVPN底层实现方式是TCP或UDP，并且配置偏好的DNS，我选择的是Google的8.8.8.8.

修改服务端配置（以UDP为例）

Vim /etc/openvpn/server.conf

将其中的ca、cert、key、dh、crl-verify修改为绝对路径

ca /etc/openvpn/ca.crt

cert /etc/openvpn/server.crt

key /etc/openvpn/server.key

dh /etc/openvpn/dh.pem

tls-auth /etc/openvpn/ta.key 0

crl-verify /etc/openvpn/crl.pem

然后进行Iptables配置

echo 1 > /proc/sys/net/ipv4/ip\_forward

（或者：sysctl -w net.ipv4.ip\_forward=1）

iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -o eth0 -j MASQUERADE

Ubuntu系统下保存iptables操作

Iptables-save > /etc/iptables-rules

然后修改脚本/etc/network/interfaces，使系统能重启时自动应用这些规则

auto eth0

iface eth0 inet dhcp

pre-up iptables-restore < /etc/iptables-rules

重启openvpn服务

Service openvpn restart

### 5.2 客户端配置

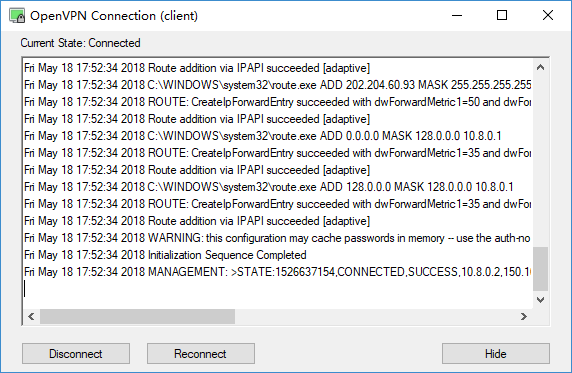
将VPS服务器root目录下的client.ovpn拷贝到win10客户端。

客户端下载openvpn客户端软件，官网下载需要翻墙。

将client.ovpn文件放置在安装好的openvpn软件的安装目录下的config文件夹下

以管理员身份运行OpenVPN GUI

在任务栏右下角会弹出openvpn的图标，右键选择connect，完成连接。



## 6 抓包脚本

抓包脚本使用的是windows下的VB Script完成本地自动抓包工作。

如需要进行鼠标操控则需要安装AutoIt3控件。

安装完成后以管理员身份运行cmd

将当前目录移动到AutoIt3中的AutoItX目录下

进行dll文件注册，64位操作系统注册AutoItX3\_x64.dll，在cmd中输入regsvr32.exe AutoItX3\_x64.dll。

VB script脚本示例：

For i=201 to 230

    set windump = CreateObject("Wscript.Shell")'创建脚本

    windump.run("C:\WINDOWS\system32\cmd.exe")'打开cmd窗口

    wscript.sleep 2000 '等待2秒

    windump.SendKeys "c:\windump.exe -i 1 -s 0 -w C:\Users\lenovo\Desktop\pptp\vimeo\_"&i&".pcap host 43.239.158.85" '命令行下输入windump抓包命令

    wscript.sleep 2000

    windump.SendKeys "{ENTER}" '输入回车开始抓包

    wscript.sleep 2000

    set oAutoIt = CreateObject("AutoItX3.Control") '调用鼠标控件

    oAutoIt.MouseClick "left",1500,879 '鼠标单击指定位置

    wscript.sleep 2000

    oAutoIt.MouseClick "left",1457,647

    wscript.sleep 2000

    oAutoIt.MouseClick "left",1522,678

    wscript.sleep 2000

    oAutoIt.MouseClick "left",720,578

    wscript.sleep 10000

    oAutoIt.MouseClick "left",720,578

    wscript.sleep 10000

    oAutoIt.MouseClick "left",720,578

    wscript.sleep 20000

    oAutoIt.MouseClick "left",720,578

    wscript.sleep 20000

    set chrome = CreateObject("Wscript.Shell") '打开chrome浏览器

    'chrome.run("C:\Users\lenovo\Desktop\chrome.exe")

    oAutoIt.MouseClick "left",281,878

    wscript.sleep 3000

    chrome.SendKeys "https://vimeo.com/channels/staffpicks/264838990" '输入网址

    chrome.SendKeys "{ENTER}" '输入回车，访问网页

    wscript.sleep 20000

    chrome.SendKeys "%{F4}" '关闭chrome浏览器

    wscript.sleep 2000

    oAutoIt.MouseClick "left",1500,879

    wscript.sleep 2000

    oAutoIt.MouseClick "left",1457,629

    wscript.sleep 2000

    oAutoIt.MouseClick "left",1533,663

    wscript.sleep 2000

    oAutoIt.MouseClick "left",1300,600

    wscript.sleep 2000

    windump.SendKeys "^c"

    wscript.sleep 2000

    'windump.SendKeys "%{F4}"

    oAutoIt.MouseClick "left",1419,204

    wscript.sleep 2000

Next

## 7 部分网站中国地区访问不支持

