

OpenVPN 服务器搭建详解

环境简介:

服务器: CentOS 5.2 客户端: XP sp2

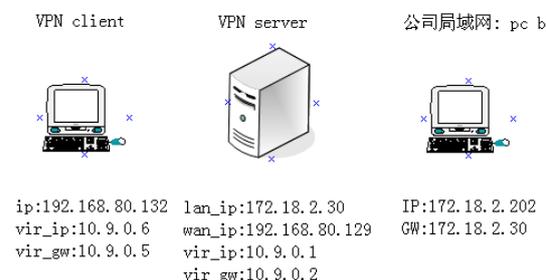
其他软件: `openvpn-2.0.9.tar.gz`
`openvpn-2.0.9-gui-1.0.3-install.exe`
`lzo-2.03.tar.gz`
openssl 为 CentOS 5.2 自带
NTRadPing.exe radius 测试软件
`pam_mysql-0.7RC1.tar.gz`
`radiusplugin_v2.0c.tar.gz`
`libcrypt-1.2.4.tar.gz`
`libpgp-error-1.5.tar.bz2`

所有测试都是在 VMWare Workstation 5.5.1 上完成

(一) 搭建 OpenVPN Server 路由模式

目的: 搭建一台 OpenVPN Server 使出差的员工也可以方便的访问到公司局域网中的共享资料。

网络环境:



OpenVPN Server 基本设定: 连接方式采用路由方式, 认证方式采用证书认证, 虚拟设备使用 tun (比 tap 更高效)

1. 安装 CentOS

这一步我就不详写了

注意: 关闭 SELinux , iptables

2. 安装 OpenVPN

a) 检测 openssl 是否已安装。(一般系统已自带)

[root@localhost ~]# Whereis openssl

如果你的系统没有 OpenSSL 库, 你需要 [下载和安装它](#)。

b) 安装 lzo

如果你想使用 VPN 连接的压缩特性, 或者你想将 OpenVPN 安装为一个 RPM 包, 安装 [LZO Library](#)。

下载: <http://www.oberhumer.com/opensource/lzo/download/lzo-2.03.tar.gz>

解压到 /root/Scripts 目录中, 后面所有的软件都存放到这个目录

```
gzip -cd lzo-2.03.tar.gz | tar -xvf -
make
make install
```

如果你使用Linux 2.2 或更早版本，下载 [TUN/TAP driver](#)。对于Linux 2.4.7 及以上版本的用户TUN/TAP 驱动已经捆绑到内核中。Linux 2.4.0 -> 2.4.6 的用户需要注意[INSTALL](#)文件末尾的注意信息。

c) tarball 安装 OpenVPN

现在下载 OpenVPN 的最新发布版: <http://openvpn.net/release/openvpn-2.0.9.tar.gz>

```
gzip -dc openvpn-2.0.9.tar.gz | tar xvf -
```

```
cd openvpn-2.0.9
```

```
./configure
```

```
make
```

```
make install
```

如果你未下载 LZO Library ，将 `--disable-lzo` 加入到 `configure` 命令中。也可以启用其他的选型，比如 `pthread` (`./configure --enable-pthread`) 用来提高 SSL/TLS 动态密钥交换的响应速度。命令

```
./configure --help
```

将显示所有的配置选型。

d) 配置 TUN/TAP 驱动

仅需一次的配置

如果你使用 Linux 2.4.7 或更高版本，十分幸运 TUN/TAP 驱动已经捆绑到内核中。你可以通过如下命令确认：

```
locate if_tun.h
```

此命令产生类似这样的信息 `/usr/include/linux/if_tun.h` 。

对于 Linux 2.4.7 或更高版本，如果你通过 tarball 安装，输入如下命令配置 TUN/TAP 设备节点（如果你通过 RPM 安装可以忽略这一步，因为 RPM 为你自动创建该节点）：

```
mknod /dev/net/tun c 10 200
```

如果你使用 Linux 2.2，你需要获得 [版本 1.1](#) 的TUN/TAP kernel module 并按照安装说明进行操作。

每次系统启动后需要执行一次的配置

在 Linux 上使用 OpenVPN 或任何用到 TUN/TAP 设备的程序前需要载入 TUN/TAP kernel module:

```
modprobe tun
```

并且启用 IP 转发:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

3. 配置 OpenVPN

a) 生成证书 Key

设置环境变量

```
[root@openvpn ~]# vi /root/.bash_profile
```

 追加如下内容(依据情况改变相应值)

```
D=/root/Scripts/openvpn-2.0.9/easy-rsa
```

```
KEY_CONFIG=$D/openssl.cnf
```

```
KEY_DIR=$D/keys
KEY_SIZE=1024
KEY_COUNTRY=CN
KEY_PROVINCE=GD
KEY_CITY=DG
KEY_ORG="ld"
KEY_EMAIL="colin_xia@luckydragongroup.com"
export KEY_CONFIG KEY_DIR KEY_SIZE KEY_COUNTRY KEY_PROVINCE
KEY_CITY KEY_ORG KEY_EMAIL D
```

同时把以上内容直接粘贴到控制台。

```
[root@openvpn ~]# echo $D          可以看到变量已生效
```

```
[root@localhost local]# cd /root/Scripts/openvpn-2.0.9/easy-rsa/
```

初始化 PKI

Build:

代码:

```
./clean-all
```

```
./build-ca
```

Generating a 1024 bit RSA private key

```
.....++++++
```

```
...++++++
```

writing new private key to 'ca.key'

```
----
```

You are about to be asked to enter information that will be incorporated into your certificate request.

What you are about to enter is what is called a Distinguished 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) [DG]:

Organization Name (eg, company) [ld]:

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

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

Email Address [colin_xia@luckydragongroup.com]:

建立 server key 代码: 代码:

```
./build-key-server server
```

Generating a 1024 bit RSA private key

.....+++++

.....+++++

writing new private key to 'server.key'

You are about to be asked to enter information that will be incorporated into your certificate request.

What you are about to enter is what is called a Distinguished 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) [DG]:

Organization Name (eg, company) [ld]:

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

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

Email Address [colin_xia@luckydragongroup.com]:

Please enter the following 'extra' attributes to be sent with your certificate request

A challenge password []:

An optional company name []:

Using configuration from /root/Scripts/openvpn-2.0.9/easy-rsa/openssl.cnf

Check that the request matches the signature

Signature ok

The Subject's Distinguished Name is as follows

countryName :PRINTABLE:'CN'

stateOrProvinceName :PRINTABLE:'GD'

localityName :PRINTABLE:'DG'

organizationName :PRINTABLE:'ld'

organizationalUnitName:PRINTABLE:'it'

commonName :PRINTABLE:'server'

emailAddress :IA5STRING:'colin_xia@luckydragongroup.com'

Certificate is to be certified until Nov 6 18:18:13 2018 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

#生成客户端 key

代码:

Generating a 1024 bit RSA private key

.....++++++

.....++++++

writing new private key to 'client1.key'

You are about to be asked to enter information that will be incorporated into your certificate request.

What you are about to enter is what is called a Distinguished 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) [DG]:

Organization Name (eg, company) [ld]:

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

Common Name (eg, your name or your server's hostname) []:client1 #重要:

每个不同的 client 生成的证书, 名字必须不同.

Email Address [colin_xia@luckydragongroup.com]:

Please enter the following 'extra' attributes to be sent with your certificate request

A challenge password []:

An optional company name []:

Using configuration from /root/Scripts/openvpn-2.0.9/easy-rsa/openssl.cnf

Check that the request matches the signature

Signature ok

The Subject's Distinguished Name is as follows

countryName :PRINTABLE:'CN'

stateOrProvinceName :PRINTABLE:'GD'

localityName :PRINTABLE:'DG'

organizationName :PRINTABLE:'ld'

organizationalUnitName:PRINTABLE:'it'

commonName :PRINTABLE:'client1'

emailAddress :IA5STRING:'colin_xia@luckydragongroup.com'

Certificate is to be certified until Nov 6 18:18:36 2018 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

依次类推生成其他客户端证书/key

代码:

```
./build-key client2
```

```
./build-key client3
```

注意在进入 Common Name (eg, your name or your server's hostname) []: 的输入时, 每个证书输入的名字必须不同.

. build: 代码:

创建 Diffie Hellman 参数。Diffie Hellman 用于增强安全性, 在 OpenVPN 是必须的:

```
./build-dh
```

生成 ta.key

```
openvpn --genkey --secret ta.key
```

将 keys 下的所有文件打包下载到本地 (除 ca 的 key, 这个文件要单独保存)

b) 创建 OpenVPN 服务器配置文件

```
vi /usr/local/etc/server.conf
```

```
port 2194
proto udp
dev tun
server 10.9.0.0 255.255.255.0
push "route 172.18.2.0 255.255.255.0"
push "dhcp-option DNS 172.18.2.23"
push "dhcp-option DNS 202.96.128.86"
ifconfig-pool-persist /usr/local/etc/ipp.txt
ca /usr/local/etc/keys/ca.crt
cert /usr/local/etc/keys/server.crt
key /usr/local/etc/keys/server.key
dh /usr/local/etc/keys/dh1024.pem
tls-auth /usr/local/etc/keys/ta.key 0
keepalive 10 120
comp-lzo
status /var/log/openvpn-status.log
verb 4
persist-key
persist-tun
```

按照配置文件所设置的, copy 相应的.key .pem .crt 文件至 /usr/local/etc/keys

c)启动 OpenVPN

```
/usr/local/sbin/openvpn --config /usr/local/etc/server.conf
```

检查服务是否启动

```
lsof -i :2194
```

等调试结束后以后台进程的方式启动 openvpn

```
/usr/local/sbin/openvpn --daemon --config /usr/local/etc/server.conf
```

并把这一句加入到 /etc/rc.local 中

4. 配置 OpenVPN Server 防火墙

配置的关键是允许 tun tap 连入,对从 OpenVPN 客户端来到公司局域网的流量做 NAT

如下(参考配置,实际配置要对应实际情况更改 如测试时可先对 tun tap 全部允许,成功后再做限制。还有注意 NAT 的配置)

Vi /etc/sysconfig/iptables

```
# Generated by iptables-save v1.3.5 on Tue Sep 30 21:34:16 2008
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [42:4060]
:RH-Firewall-1-INPUT - [0:0]
-A INPUT -j RH-Firewall-1-INPUT
-A FORWARD -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
#设备 tap (交换) 和 tun(路由), tap 是二层设备,支持链路层协议。
#tun 是 ip 层的点对点协议,限制稍微多一些,一般是使用 tun
从 tun+虚拟网卡入,目标 ip 为 172.18.2.32 协议 tcp 端口 80 或 83 动作允许
-A RH-Firewall-1-INPUT -d 172.18.2.32 -i tun+ -p tcp -m tcp --dport 80:83 -j ACCEPT
-A RH-Firewall-1-INPUT -d 172.18.2.32 -i tap+ -p tcp -m tcp --dport 80:83 -j ACCEPT
-A RH-Firewall-1-INPUT -d 172.18.2.23 -i tun+ -p udp -m udp --dport 53 -j ACCEPT
-A RH-Firewall-1-INPUT -d 172.18.2.23 -i tap+ -p udp -m udp --dport 53 -j ACCEPT
-A RH-Firewall-1-INPUT -d 172.18.2.23 -i tun+ -p tcp -m tcp --dport 53 -j ACCEPT
-A RH-Firewall-1-INPUT -d 172.18.2.23 -i tap+ -p tcp -m tcp --dport 53 -j ACCEPT
-A RH-Firewall-1-INPUT -s 10.9.0.132 -d 172.18.2.24 -j ACCEPT
-A RH-Firewall-1-INPUT -s 10.9.0.131 -j REJECT --reject-with icmp-port-unreachable
源 ip 为 10.0.0.131 拒绝
#-A RH-Firewall-1-INPUT -s 10.9.0.130 -d 172.18.2.0/255.255.0.0 -j ACCEPT
-A RH-Firewall-1-INPUT -s 10.9.0.130 -d 172.18.2.41 -j ACCEPT
```

```

-A RH-Firewall-1-INPUT -s 10.9.0.130 -d 172.18.2.40 -j ACCEPT
-A RH-Firewall-1-INPUT -i eth1 -p tcp -m state --state NEW -m tcp --dport 22 -j
ACCEPT
允许从 eth1 网卡访问 ssh 服务
-A RH-Firewall-1-INPUT -p udp -m udp --dport 2194 -j ACCEPT
允许 vpn 客户端连接服务器的 vpn 服务端口
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
# Completed on Tue Sep 30 21:34:16 2008
# Generated by iptables-save v1.3.5 on Tue Sep 30 21:34:16 2008
*nat
:PREROUTING ACCEPT [3:518]
:POSTROUTING ACCEPT [1:92]
:OUTPUT ACCEPT [1:92]
-A POSTROUTING -s 10.9.0.0/255.255.255.0 -o eth1 -j SNAT --to-source 172.18.2.30
源地址为 10.9.0.0/255.255.255.0 从 eth1 出口的数据报都 snat 为 172.18.2.30
COMMIT
# Completed on Tue Sep 30 21:34:16 2008

```

配置 iptables 配置文件后，启用配置：
iptables-restore < /etc/sysconfig/iptables

ok 至此 OpenVPN 服务器配置完毕。

5. 配置 OpenVPN windows 客户端
 - a) 安装 openvpn-2.0.9-gui-1.0.3-install.exe
编辑安装目录 config 文件中的 client.ovpn 文件
内容如下：
client

dev tun
proto udp

remote 192.168.80.129 2194

persist-key
persist-tun
ca ca.crt
cert client1.crt
key client1.key
ns-cert-type server

```
comp-lzo
verb 3
;redirect-gateway def1
tls-auth ta.key 1
```

按照配置文件 cp 相应的 .key .crt .pem 文件到 config 文件夹

运行 openvpn 客户端，并从 windows 客户机 ping 公司局域网 pc 172.18.2.202
再 tracert 和 pathping 一下，确认数据流向
OK 实验成功

6. 再做一个小测试 吊销客户端证书

```
root@localhost ~]# cd Scripts/openvpn-2.0.9/easy-rsa/
[root@localhost easy-rsa]# ./revoke-full client1          吊销 client1 的证书
```

这条命令执行完成之后，会在 keys 目录下面，生成一个 crl.pem 文件，这个文件中包含了吊销证书的名单。

成功注销某个证书之后，可以打开 keys/index.txt 文件，可以看到被注销的证书前面，已标记为 R。

- a. 复制 easy-rsa 目录下 keys 目录中的 crl.pem 到 /usr/local/etc/keys 目录中
- b. 确保 server.conf 中包含这样一句

```
crl-verify /usr/local/etc/keys/crl.pem
```

- c. 重新启动 openvpn 。并测试

Client 连接失败 测试成功

7. 使用 tap 设备的路由模式

把服务器和客户端配置文件中 dev tun 改为 dev tap 即可

关于使用 tap 和 tun 设备的区别，见附录 tun 和 tap 区别。

(二) 搭建 OpenVPN Server 桥接模式

还是 (一) 的网络环境

目的: 客户端还是连接到公司局域网中, 但出于一些目的必须使用桥接模式。

如客户端要获得公司局域网 ip, 某些应用只能使用桥接模式, 节省 ip。

不管什么原因, 反正就是必须使用桥接模式

OpenVPN Server 基本设定: 连接方式采用桥接, 认证方式采用证书认证, 虚拟设备使用 tap

1. 将 OpenVPN 服务器的 LAN 网卡和虚拟网卡桥接

关闭 OpenVPN 服务 `killall openvpn`

a) 确认你已经安装了 **bridge-utils** 软件包。

可以 `vi /root/install.log` 或 `rpm -qa`, 没有安装可以去安装光盘找或者直接 `yum` 安装

b) 编辑下面的 `bridge-start` 脚本。依照你要桥接的物理以太网接口设置 `br`, `tap`, `eth`, `eth_ip`, `eth_netmask` 和 `eth_broadcast` 参数。确保使用的接口是私有的且它连接的局域网由防火墙保护其到 `internet` 的连接。可以使用 `ifconfig` 命令获得要填入 `bridge-start` 参数的网络接口的信息

c) 这个脚本可以从

d) <http://openvpn.net/index.php/documentation/miscellaneous/ethernet-bridging.html#linuxscript> 获得, 或者 `openvpn` 的模版文件目录中也有

e) 运行 `bridge-start` 脚本。它将创建一个永久 (服务器重起前) 的 `tap0` 接口并将它和活跃的以太网接口桥接。

f) 把这个脚本也加入到 `/etc/rc.local`

2. 编辑服务器配置文件

a) 在 “(一)” 步骤地服务器配置文件基础上

i. 更改 `dev tun` 为 `dev tap0`

ii. 注释 `server` 开头的行

iii. 添加如下行

iv. `server-bridge` 网桥 ip 分给 vpn 客户端的 ip 范围

```
server-bridge 172.18.2.30 255.255.255.0 172.18.2.128 172.18.2.254
```

v. `push "route 172.18.2.0 255.255.255.0"` 这句也可以注释掉, 当然只要知道这句是 `push` 一条路由给 `vpn client` 就可以灵活掌握。

vi.

3. 设置防火墙使数据包在新建的 `tap0` 和 `br0` 借口上自由传送

`vi /etc/sysconfig/iptables`

添加如下几条

```
-A RH-Firewall-1-INPUT -i tap0 -j ACCEPT
```

```
-A RH-Firewall-1-INPUT -i br0 -j ACCEPT
```

那条 `NAT` 的规则也可以去掉。

开启 `openvpn`

```
/usr/local/sbin/openvpn --daemon --config /usr/local/etc/server.conf
```

从 winxp client 连接到服务器

```
测试 ok ipconfig /all tracert pathping
```

OpenVPN 服务器启动和停止

```
运行 bridge-start
```

```
运行 openvpn
```

```
停止 openvpn
```

```
运行 bridge-stop
```

Ok OpenVPN 桥接设置完毕

(三) 搭建 OpenVPN Server 路由模式 + 口令认证+ MYSQL

还是 (一) 的网络环境

目的: 客户端还是连接到公司局域网中, 但为了用户管理方便, 认证数据从 mysql 数据库中提取。

OpenVPN Server 基本设定: 连接方式采用路由, 认证方式采用 mysql 认证, 虚拟设备使用 tun

1. 首先检查 pam-devel 包是否安装, 否则从系统盘安装改软件包

a) rpm -qa | grep pam

2. 检查 Mysql 是否安装, 确认 mysql-devel 包已经安装, 否则从系统盘安装改软件包

```
rpm -qa | grep msq
```

```
mysql-5.0.45-7.el5
```

```
mysql-devel-5.0.45-7.el5
```

```
mysql-server-5.0.45-7.el5
```

注: 从系统盘安装改软件包时有一系列的依赖问题, 我晕

可以把系统盘设置为 yum 源, 使用 yum 安装这样依赖问题就会自动解决

3. 我们安装 pam_mysql 使用 MySQL 数据库存储用户数据, 其它数据库可以找相应的 PAM 验证模块 (我是在 cu 下载的 pam_mysql-0.7RC1.tar.gz)

```
[root@localhost ~]# gzip -cd pam_mysql-0.7RC1.tar.gz | tar xvf -
```

```
[root@localhost ~]# cd pam_mysql-0.7RC1
```

```
[root@localhost pam_mysql-0.7RC1]# ./configure
```

```
[root@localhost pam_mysql-0.7RC1]# make
```

```
[root@localhost pam_mysql-0.7RC1]# cp .libs/pam_mysql.so /lib/security/
```

4. 配置 mysql 数据库 和 pam

Server mysqld start

使用 chkconfig 命令让 mysql 数据库, 开机自启动

配置数据库

以管理员身份登录数据库:

```
mysql> create database vpn;
```

```
mysql> GRANT ALL ON vpn.* TO vpn@localhost IDENTIFIED BY 'vpn123';
```

```
mysql> flush privileges;
```

```
mysql> use vpn;
```

```
mysql> CREATE TABLE vpnuser (
```

```
-> name char(20) NOT NULL,
```

```
-> password char(128) default NULL,
```

```
-> active int(10) NOT NULL DEFAULT 1,
```

```
-> PRIMARY KEY (name)
```

```
-> );
```

```
mysql> insert into vpnuser (name,password) values('elm', 'elm' );
```

#创建 vpn 用户, 对 vpn 这个 database 有所有操作权限, 密码为 vpn123

#active 不为 1, 无权使用 VPN

#增加用户 用户名: elm 密码: elm

配置 pam_mysql 模块

创建/etc/pam.d/openvpn 文件，文件内容如下：

```
=====CUT Here=====
auth sufficient pam_mysql.so user=vpn passwd=vpn123 host=localhost db=vpn \
table=vpnuser usercolumn=name passwdcolumn=password \
where=active=1 sqllog=0 crypt=0
account required pam_mysql.so user=vpn passwd=vpn123 host=localhost db=vpn \
table=vpnuser usercolumn=name passwdcolumn=password \
where=active=1 sqllog=0 crypt=0
=====Cut Here=====
```

crypt(0) -- Used to decide to use MySQL's PASSWORD() function or crypt()

0 = No encryption. Passwords in database in plaintext. NOT recommended!

1 = Use crypt

2 = Use MySQL PASSWORD() function

下面可以测试 pam_mysql 是否工作正常，先检查 saslauthd 是否安装：

```
[root@localhost pam_mysql-0.7RC1]# rpm -qa | grep sasl
```

```
cyrus-sasl-plain-2.1.22-4
```

```
cyrus-sasl-devel-2.1.22-4
```

```
cyrus-sasl-2.1.22-4
```

```
cyrus-sasl-lib-2.1.22-4
```

有 cyrus-sasl-2.1.22-4 应该就可以了，如果没有请安装相应的软件包，不安装也行，可以通过其它方法测试

```
[root@vpn ~]# saslauthd -a pam
```

把这句加入到 /etc/rc.local 记得写全路径 用 whereis 或 locate 找

```
[root@vpn ~]# testsaslauthd -u elm -p elm -s openvpn
```

```
0: OK "Success."
```

恭喜，pam_mysql 工作正常了，下面可以开始配置 OpenVPN 服务器了。

这里我第一次测试时失败了

编辑 /etc/pam.d/openvpn

两句最后都加上

verbose=1 **详细日志**

```
[root@localhost pam_mysql-0.7RC1]# ls -ltr /var/log
```

看哪个日志文件被更新了

再开两个窗口监视这两个文件

```
[root@localhost pam_mysql-0.7RC1]# tail -f /var/log/messages
```

```
[root@localhost pam_mysql-0.7RC1]# tail -f /var/log/secure
```

重新 做一次 saslauthd 的启动 和 testsaslauthd

日志也没有看到明显错误

最后才记起来

Mysql 的 password 函数因为 mysql 的加密函数和 pam_mysql-0.7RC1 的加密函数不相同。

crypt (plain)

The method to encrypt the user's password:

0 (or "plain") = No encryption. Passwords stored in plaintext.

HIGHLY DISCOURAGED.

1 (or "Y") = Use crypt(3) function.

2 (or "mysql") = Use MySQL PASSWORD() function. It is possible that the encryption function used by PAM-MySQL is different from that of the MySQL server, as PAM-MySQL uses the function defined in MySQL's C-client API instead of using PASSWORD() SQL

function

in the query.

3 (or "md5") = Use plain hex MD5.

4 (or "sha1") = Use plain hex SHA1.

于是把 crypt 设置为 0，新建一个密码不加密的帐户。再测试就通过了

5. 生成 openvpn-auth-pam.so

```
[root@localhost etc]# cd /root/Scripts/openvpn-2.0.9/plugin/auth-pam/  
[root@localhost auth-pam]# make  
[root@localhost auth-pam]# cp openvpn-auth-pam.so /usr/local/etc/lib/
```

6.配置服务器配置文件, 在（一）的基础上

添加如下几行:

```
#说明使用的插件, openvpn为插件的参数, 使用pam的servicesname  
plugin /usr/local/etc/lib/openvpn-auth-pam.so openvpn  
#不请求客户的CA证书, 使用User/Pass验证  
client-cert-not-required  
#使用客户提供的UserName作为Common Name  
username-as-common-name
```

7.配置客户端配置文件

注释掉

```
;cert client1.crt  
;key client1.key
```

增加

```
#询问用户名和密码  
auth-user-pass
```

使用客户端连接服务器 测试 ok

(四) 搭建 OpenVPN Server 路由模式 + 口令认证+TEXT/POP3

还是(一)的网络环境

目的: 客户端还是连接到公司局域网中, 但为了用户管理方便, 认证数据从 TEXT 数据库中提取。

OpenVPN Server 基本设定: 连接方式采用路由, 认证方式采用 TEXT/POP3 认证, 虚拟设备使用 tun

1. 下载 TEXT 认证脚本 checkpsw.sh, 并且复制到 /usr/local/etc/ 目录中同时 chmod u+x 。

<http://openvpn.se/files/other/>

注意: 1.脚本保存到 windows 中再上传到 linux 有问题, 最好在那个目录新建一个同名文件再从控制台粘贴进去。

2. 脚本的开始不是 #! 要改正。

2. 配置服务器配置文件, 在(一)的基础上

添加如下几行:

```
# auth-user-pass-verify cmd method: Query client for username/password and
# run script cmd to verify.  If method='via-env', pass
# user/pass via environment, if method='via-file', pass
# user/pass via temporary file.
auth-user-pass-verify /usr/local/etc/checkpsw.sh via-env
#不请求客户的CA证书, 使用User/Pass验证
client-cert-not-required
#使用客户提供的UserName作为Common Name
username-as-common-name
```

3. 配置客户端配置文件

注释掉

```
;cert client1.crt
;key client1.key
```

增加

```
#询问用户名和密码
auth-user-pass
```

4. 更改 checkpsw.sh 中的 PASSFILE 变量为。

```
PASSFILE="/usr/local/etc/psw-file"
```

5. 创建 /usr/local/etc/psw-file 内容如下:

格式: 用户名 Tab 密码

```
User1 pass
User2 pass
```

注: 实验成功后我自己仿照 checkpsw.sh 用 perl 重写了一遍(目录中的 checkpsw.pl), 也可以使用。

6. 同样的原理我们还可以使用 POP3 认证(^_^ 其他的当然也可以)

脚本见目录中的 popauth.pl

(五) 搭建 OpenVPN Server 路由模式 + 口令认证+RADIUS

网络环境: 在 (一) 的基础上添加一台 win2003 服务器 ip 为 192.168.80.130

^^ 不好意思, 我把服务器给放到了公网上了。不过还好是测试。

目的: 客户端还是连接到公司局域网中, 但为了用户管理方便, 认证数据从 radius 数据库中提取。

OpenVPN Server 基本设定: 连接方式采用路由, 认证方式采用 radius 认证, 虚拟设备使用 tun

1. 搭建 Radius 服务器

见附录 搭建 win2003 下的 IAS 服务

2. 配置 radiusplugin

1.radiusplugin_v2.0.tar.gz: 可以编译得到 radiusplugin.so

到 <http://www.nongnu.org/radiusplugin/> 下载

2.libcrypt 支持库: 可以编译得到/usr/lib/libcrypt.so.11

到 <ftp://ftp.gnupg.org/gcrypt/libcrypt/libcrypt-1.2.4.tar.gz> 下载

3.libgpg-error 支持库: 可以编译得到/usr/local/lib/libgpg-error.so.0

到 <ftp://ftp.gnupg.org/gcrypt/libgpg-error/libgpg-error-1.5.tar.gz> 下载

简单的编译以上 3 个支持库, configure;make;make install。

我们要用到 radiusplugin.so, 其他是 radiusplugin.so 的支持库。

好了如果能够得到 radiusplugin.so, 已经成功了 80%, 其他的就是配置了。

把 radiusplugin.so 拷贝到/usr/local/etc/lib 下, 并配置其配置文件 radiusplugin.conf 内容如下:

```
# The NAS identifier which is sent to the RADIUS server
```

```
NAS-Identifier=OpenVpn
```

```
# The service type which is sent to the RADIUS server
```

```
Service-Type=5
```

```
# The framed protocol which is sent to the RADIUS server
```

```
Framed-Protocol=1
```

```
# The NAS port type which is sent to the RADIUS server
```

```
NAS-Port-Type=5
```

```
# 这是运行 openvpn 服务器的 ip, 作为 radius 客户端
```

```
NAS-IP-Address=192.168.80.129
```

```
#这里指明 openvpn 的配置位置
```

```
OpenVPNConfig=/usr/local/etc/server.conf
```

```
# 这里定义 radius server 参数可以超过 1 个作为备份 server
```

```
{
```

```
# The UDP port for radius accounting.
```

```
acctport=1813
```

```
# The UDP port for radius authentication.
```

```
authport=1812
```

```
# 这是我 radius 服务器的 ip, 并添加了用户。
```

```
name=192.168.80.130
```

```
# How many times should the plugin send the if there is no response?
retry=1
# How long should the plugin wait for a response?
wait=1
# The shared secret.共享密钥，在 winradius 里配置，设置-系统-NAS 密钥
sharedsecret=123456
}
```

3.配置服务器配置文件, 在（一）的基础上

添加如下几行:

```
#说明使用的插件
plugin /usr/local/etc/lib/radiusplugin.so /usr/local/etc/radius.conf
#不请求客户的CA证书，使用User/Pass验证
client-cert-not-required
#使用客户提供的UserName作为Common Name
username-as-common-name
```

4.配置客户端配置文件

注释掉

```
;cert client1.crt
;key client1.key
```

增加

```
#询问用户名和密码
auth-user-pass
```

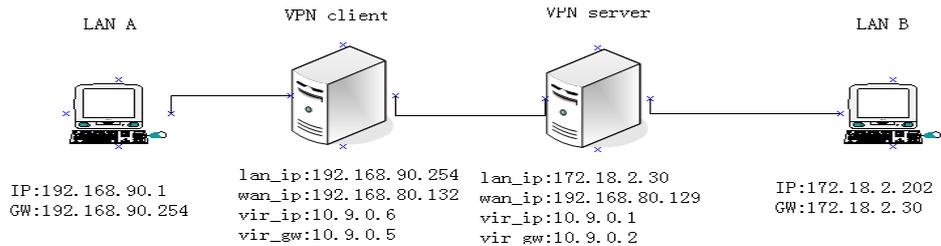
使用客户端连接服务器 测试 ok

(六) 搭建 OpenVPN Site to Site

目的: 公司局域网 LAN B 和在其它城市公司分部网络 LAN A 要互相连通, 共享彼此的资源。

OpenVPN Server 基本设定: 连接方式采用路由, 认证方式采用证书认证, 虚拟设备使用 tun

网络环境:



1. vpn client 和 vpn server 两台设备都将作为各自网络的 GW, 系统为 CentOS 5.2 上面安装 OpenVPN 软件。并开启 ip 转发, 防火墙已设置妥当。

2. 配置 openvpn server 在 (一) 的基础上:

添加:

```
route 192.168.90.0 255.255.255.0
```

```
client-config-dir /usr/local/etc/ccd
```

配置要点:

使用 tun 点对点的虚拟连接 (路由)

dev tun

在本机添加一条到 192.168.90.0 255.255.255.0 via 就是对端 (10.9.0.2)

```
route 192.168.90.0 255.255.255.0
```

push 一条路由到客户端, 告诉客户机 172.18.2.0 255.255.255.0 在它的对端 (10.9.0.5)

```
push "route 172.18.2.0 255.255.255.0"
```

为 vpn clien 设置单独配置文件, 内容是: iroute 192.168.90.0 255.255.255.0

```
client-config-dir /usr/local/etc/ccd
```

3. 配置 VPN Client

同 (一)

4. 使用 ping pathping tracert 等命令测试, LAN A 和 LAN B 两个网络连通 ok

至此 OpenVPN Site to Site 配置完成。

(七) Open VPN 其它配置选项

1. 服务器选项

- a) #定义最大连接数
;max-clients 100
- b) #记录日志，每次重新启动 openvpn 后删除原有的 log 信息
log /var/log/openvpn.log
#和 log 一致，每次重新启动 openvpn 后保留原有的 log 信息，新信息追加到文件最后
;log-append openvpn.log
- c) #如果可以让 VPN Client 之间相互访问直接通过 openvpn 程序转发，
#不用发送到 tun 或者 tap 设备后重新转发，优化 Client to Client 的访问效率
client-to-client
- d) #使 Client 的默认网关指向 VPN，让 Client 的所有 Traffic 都通过 VPN 走
;push "redirect-gateway"
- e) 在 vpn client 连接或退出时，执行脚本。可以用于生成用户访问日志。
client-connect /usr/local/etc/connect
client-disconnect /usr/local/etc/disconnect
脚本 见附录
- f) #定义运行 openvpn 的用户
user nobody
group nobody

2. 客户端选项

- a) 可以有多个 vpn 服务器做负载均衡，然后客户端配置这样写
remote 61.1.1.2 1194
remote 22.1.1.2 1194
随机选择一个 Server 连接，否则按照顺序从上到下依次连接
;remote-random
- b) 如果 remote 后接的是域名
始终重新解析 Server 的 IP 地址（如果 remote 后面跟的是域名），
保证 Server IP 地址是动态的使用 DDNS 动态更新 DNS 后，Client 在自动重新连接时重新解析 Server 的 IP 地址
这样无需人为重新启动，即可重新接入 VPN
resolv-retry infinite
- C) # 如果你使用 HTTP 代理连接 VPN Server，把 Proxy 的 IP 地址和端口写到下面
如果代理需要验证，使用 http-proxy server port [authfile] [auth-method]
其中 authfile 是一个 2 行的文本文件，用户名和密码各占一行，auth-method 可以省略，详细信息查看 Manual
;http-proxy-retry # retry on connection failures
;http-proxy [proxy server] [proxy port #]

（八） OpenVPN 配置参数详解

```
##号和;号开头的都是注释
# 设置监听 IP，默认是监听所有 IP
#local 116.6.45.23
#Openvpn 服务器监听端口
port 2194
# 设置用 TCP 还是 UDP 协议?
;proto tcp
proto udp
# 设置创建 tun 的路由 IP 通道，还是创建 tap 的以太网通道
# 路由 IP 容易控制，所以推荐使用它；但如果如 IPX 等必须
# 使用第二层才能通过的通讯，则可以用 tap 方式，tap 也
# 就是以太网桥接
dev tun
# 配置 VPN 使用的网段，OpenVPN 会自动提供基于该网段的 DHCP
# 服务，但不能和任何一方的局域网段重复，保证唯一
# server 端 ip 默认会设为.1 的地址。
server 10.9.0.0 255.255.255.0
# 为客户端创建对应的路由,以另其通达公司网内部服务器
# 但记住，公司网内部服务器也需要有可用路由返回到客户端
push "route 172.18.2.0 255.255.255.0"
# 维持一个客户端和 virtual IP 的对应表，以方便客户端重新
# 连接可以获得同样的 IP
ifconfig-pool-persist /usr/local/etc/ipp.txt
# 用 OpenVPN 的 DHCP 功能为客户端提供指定的 DNS、WINS 等
push "dhcp-option DNS 172.18.2.23"
push "dhcp-option DNS 202.96.128.86"
# 这里是重点，必须指定 SSL/TLS root certificate (ca),
# certificate(cert), and private key (key)
# ca 文件是服务端和客户端都必须使用的，但不需要 ca.key
# 服务端和客户端指定各自的.crt 和.key
# 请注意路径,可以使用以配置文件开始为根的相对路径,
# 也可以使用绝对路径
# 请小心存放.key 密钥文件
ca /usr/local/etc/keys/ca.crt
cert /usr/local/etc/keys/server.crt
key /usr/local/etc/keys/server.key
# 指定 Diffie hellman parameters.
dh /usr/local/etc/keys/dh1024.pem
```

```
#用于吊销客户证书
crl-verify /usr/local/etc/keys/vpn-crl.pem
#增强安全性
# Generate with:
# openvpn --genkey --secret ta.key
#
# The server and each client must have
# a copy of this key.
# The second parameter should be 0
# on the server and 1 on the clients.
tls-auth /usr/local/etc/keys/ta.key 0

# 设置服务端检测的间隔和超时时间 每 10 秒 ping 一次, 如果 120 秒没有回应则认为对方已经 down
keepalive 10 120
# 使用 lzo 压缩的通讯,服务端和客户端都必须配置
comp-lzo
# 输出短日志,每分钟刷新一次,以显示当前的客户端
status /var/log/openvpn-status.log
#设置日志要记录的级别。
#0 只记录错误信息。
#4 能记录普通的信息。
#5 和 6 在连接出现问题时能帮助调试
#9 是极端的, 所有信息都会显示, 甚至连包头等信息都显示 (像 tcpdump)
verb 4
#相同信息的数量, 如果连续出现 20 条相同的信息, 将不记录到日志中。
mute 20
# 让 OpenVPN 以 nobody 用户和组来运行 (安全)
user nobody
group nobody
# The persist options will try to avoid
# accessing certain resources on restart
# that may no longer be accessible because
# of the privilege downgrade.
# 重启时仍保留一些状态
persist-key
persist-tun

##### 其他参数 #####

# 为特定的客户端指定 IP 或指定路由,该路由通常是客户端后面的
# 内网网段,而不是服务端连接的网段
# ccd 是/etc/openvpn 下的目录, 其中建有希望限制的客户端 Common
# Name 为文件名的文件,并通过下面的命令写入固定 IP 地址
# 例如 Common Name 为 client1,则在/etc/openvpn/ccd/client1 写有:
```

```
# ifconfig-push 10.9.0.1 10.9.0.2
client-config-dir /usr/local/etc/ccd
# 若客户端希望所有的流量都通过 VPN 传输,则可以使用该语句
# 其会自动改变客户端的网关为 VPN 服务器,推荐关闭
# 一旦设置, 请小心服务端的 DHCP 设置问题
;push "redirect-gateway"
# 如果您希望有相同 Common Name 的客户端都可以登陆
# 也可以注释下面的语句,推荐每个客户端都使用不同的 Common Name
# 常用于测试
;duplicate-cn
# 设置最大用户数
#max-clients 3
# 打开管理界面,可以定义监控的 IP 和端口
management localhost 7505
# 缺省日志会记录在系统日志中, 但也可以导向到其他地方
# 建议调试的使用先不要设置,调试完成后再定义
;log /var/log/openvpn/openvpn.log
;log-append /var/log/openvpn/openvpn.log
# 配置为以太网桥模式,但需要使用系统的桥接功能
# 这里不需要使用
;server-bridge 10.8.0.4 255.255.255.0 10.8.0.50 10.8.0.100
#记录日志, 每次重新启动 openvpn 后删除原有的 log 信息
log /var/log/openvpn.log
#和 log 一致, 每次重新启动 openvpn 后保留原有的 log 信息, 新信息追加到文件最后
;log-append openvpn.log
#定义运行 openvpn 的用户
user nobody
group nobody
#Run script or shell command cmd to validate client
#virtual addresses or routes. 具体查看 manual
;learn-address ./script
#其它的一些需要 PUSH 给 Client
#用于记录某个 Client 获得的 IP 地址, 类似于 dhcpd.lease 文件,
#防止 openvpn 重新启动后“忘记”Client 曾经使用过的 IP 地址
ifconfig-pool-persist ipp.txt
#Bridge 状态下类似 DHCPD 的配置, 为客户分配地址, 由于这里工作在路由模式, 所以不使用
;server-bridge 10.8.0.4 255.255.255.0 10.8.0.50 10.8.0.100
# 随机选择一个 Server 连接, 否则按照顺序从上到下依次连接
;remote-random
# 始终重新解析 Server 的 IP 地址 (如果 remote 后面跟的是域名),
# 保证 Server IP 地址是动态的使用 DDNS 动态更新 DNS 后, Client 在自动重新连接时重新解析 Server 的 IP 地址
# 这样无需人为重新启动, 即可重新接入 VPN
resolv-retry infinite
```

```
# 在本机不绑定任何端口监听 incoming 数据，Client 无需此操作，除非一对一的 VPN 有必要
nobind
# 如果你使用 HTTP 代理连接 VPN Server，把 Proxy 的 IP 地址和端口写到下面
# 如果代理需要验证，使用 http-proxy server port [authfile] [auth-method]
# 其中 authfile 是一个 2 行的文本文件，用户名和密码各占一行，auth-method 可以省略，详细信息查看 Manual
;http-proxy-retry # retry on connection failures
;http-proxy [proxy server] [proxy port #]
# Server 使用 build-key-server 脚本生成的，在 x509 v3 扩展中加入了 ns-cert-type 选项
# 防止 VPN client 使用他们的 keys + DNS hack 欺骗 vpn client 连接他们假冒的 VPN Server
# 因为他们的 CA 里没有这个扩展
ns-cert-type server
a.定义 tun 为使用路由方式的 VPN
b.小心处理证书的路径，.key 文件要保存好，特别是 ca.key。
（ca.key 不需要在 OpenVPN 中用到，可以另外保存）
注意，每个虚拟 tun 网卡都是成对的，只有 inet addr 标识的才是用于 VPN 通讯。并且必须在/30 网段
```

bridge-start

```
#!/bin/bash

#####
# Set up Ethernet bridge on Linux
# Requires: bridge-utils
#####
更改蓝色字体部分即可，桥接后的ip为内网卡ip
# Define Bridge Interface
br="br0"

# Define list of TAP interfaces to be bridged,
# for example tap="tap0 tap1 tap2".
tap="tap0"

# Define physical ethernet interface to be bridged
# with TAP interface(s) above.
eth="eth1"
eth_ip="172.18.2.30"
eth_netmask="255.255.255.0"
eth_broadcast="172.18.2.255"

for t in $tap; do
    openvpn --mktun --dev $t
done

brctl addbr $br
brctl addif $br $eth

for t in $tap; do
    brctl addif $br $t
done

for t in $tap; do
    ifconfig $t 0.0.0.0 promisc up
done

ifconfig $eth 0.0.0.0 promisc up

ifconfig $br $eth_ip netmask $eth_netmask broadcast $eth_broadcast
```

bridge-stop

Bridge stop 脚本（其中内容也是要根据情况更改）

```
#!/bin/bash

#####
# Tear Down Ethernet bridge on Linux
#####

# Define Bridge Interface
br="br0"

# Define list of TAP interfaces to be bridged together
tap="tap0"

ifconfig $br down
brctl delbr $br

for t in $tap; do
    openvpn --rmtun --dev $t
done
```

checkpsw.sh

```
#!/bin/sh
#####
# checkpsw.sh (C) 2004 Mathias Sundman <mathias@openvpn.se>
#
# This script will authenticate OpenVPN users against
# a plain text file. The passfile should simply contain
# one row per user with the username first followed by
# one or more space(s) or tab(s) and then the password.

PASSFILE="/etc/openvpn/psw-file"
LOG_FILE="/var/log/openvpn-password.log"
TIME_STAMP=`date "+%Y-%m-%d %T"`

#####

if [ ! -r "${PASSFILE}" ]; then
    echo "${TIME_STAMP}: Could not open password file \"${PASSFILE}\" for reading." >> ${LOG_FILE}
    exit 1
fi

# ${TIME_STAMP} 是一个参数替换 相当于 $TIME_STAMP 的值, 也就是 `date "+%Y-%m-%d %T"`

CORRECT_PASSWORD=`awk '!/^/&&!/^#/&&$1=="${username}""{print $2;exit}' ${PASSFILE}`
```

```

# 模式          ##动作          ## 输入文件 #

if [ "${CORRECT_PASSWORD}" = "" ]; then
    echo "${TIME_STAMP}: User does not exist: username=\"${username}\", password=\"${password}\"." >>
    ${LOG_FILE}
    exit 1
fi

if [ "${password}" = "${CORRECT_PASSWORD}" ]; then
    echo "${TIME_STAMP}: Successful authentication: username=\"${username}\"." >> ${LOG_FILE}
    exit 0
fi

echo "${TIME_STAMP}: Incorrect password: username=\"${username}\", password=\"${password}\"." >>
    ${LOG_FILE}
exit 1

```

checkpsw.pl

```
#!/usr/bin/perl -w
```

```
use strict;
```

```
my $PASSFILE = "/usr/local/etc/psw-file";
```

```
my $LOG_FILE = "/var/log/openvpn-password.log";
```

```
my $TIME_STAMP = `date "+%Y-%m-%d %T"`;
```

```
my $username;
```

```
my $password;
```

```
open ("LOG", ">>", "$LOG_FILE") or die "can't open $LOG_FILE: $!";
```

```
open ("PASS", "<", "$PASSFILE") or die "can't open $PASSFILE";
```

```
sub get_vpnpass{
```

```
    while(<PASS>){
```

```
        next if m/^\;/;
```

```
        next if m/^\#/;
```

```
        ($username,$password) = split(/\s+/, $_);
```

```
        return $password if $username eq $ENV{username};
```

```
    }
```

```
}
```

```
#####
```

```
my $CORRECT_PASSWORD = get_vpnpass();
```

```
unless ( -r $PASSFILE){
```

```
    print LOG "$TIME_STAMP: Could not open password file \"$PASSFILE\" for reading.";
```

```
    exit 1;
```

```
}
```

```
if ( $CORRECT_PASSWORD eq ""){
```

```
    print LOG "$TIME_STAMP: USER does not exist:"
```

```
username=\"$ENV{username}\",password=\"$ENV{password}\".";
```

```
    exit 1;
```

```
}
```

```
if ( $CORRECT_PASSWORD eq $ENV{password}){
```

```
    print LOG "$TIME_STAMP: Successful authentication:username=\"$ENV{username}\".";
```

```
    exit 0;
```

```
}
```

```
print LOG "$TIME_STAMP: Incorrect password:"
```

```
username=\"$ENV{username}\",password=\"$ENV{password}\".";
```

```
exit 1;
```

connect

```
#!/bin/bash
```

```
day=`date +%F`
```

```
if [ -f /var/log/openvpn/$day ];then
```

```
echo "`date +%F %H:%M:%S` User $common_name is logged in" >>/var/log/openvpn/$day
```

```
else
```

```
touch /var/log/openvpn/$day
```

```
echo "`date +%F %H:%M:%S` User $common_name is logged in" >>/var/log/openvpn/$day
```

```
fi
```

disconnect

```
#!/bin/bash
```

```
day=`date +%F`
```

```
if [ -f /var/log/openvpn/$day ];then
```

```
echo "`date +%F %H:%M:%S` User $common_name is logged off" >>/var/log/openvpn/$day
```

```
else
```

```
touch /var/log/openvpn/$day
```

```
echo "`date +%F %H:%M:%S` User $common_name is logged off" >>/var/log/openvpn/$day
```

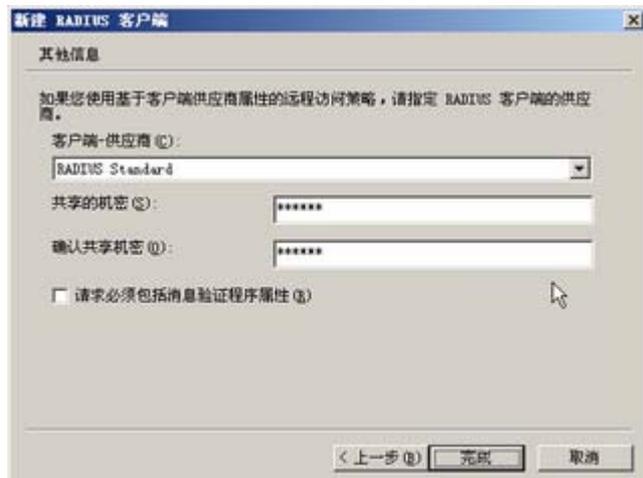
```
fi
```

(一) 搭建 Radius 服务器 这里使用的是 windows 2003 的 IAS

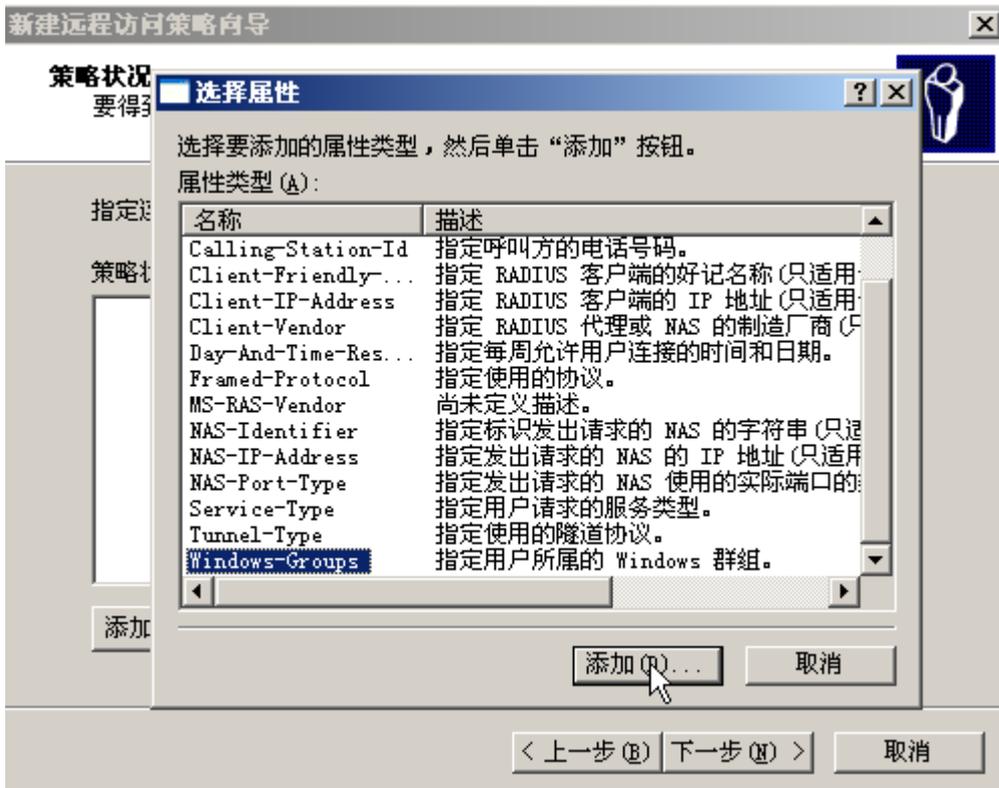
1. 安装 IAS
2. 配置 IAS
 - A. 新建 RADIUS 客户端

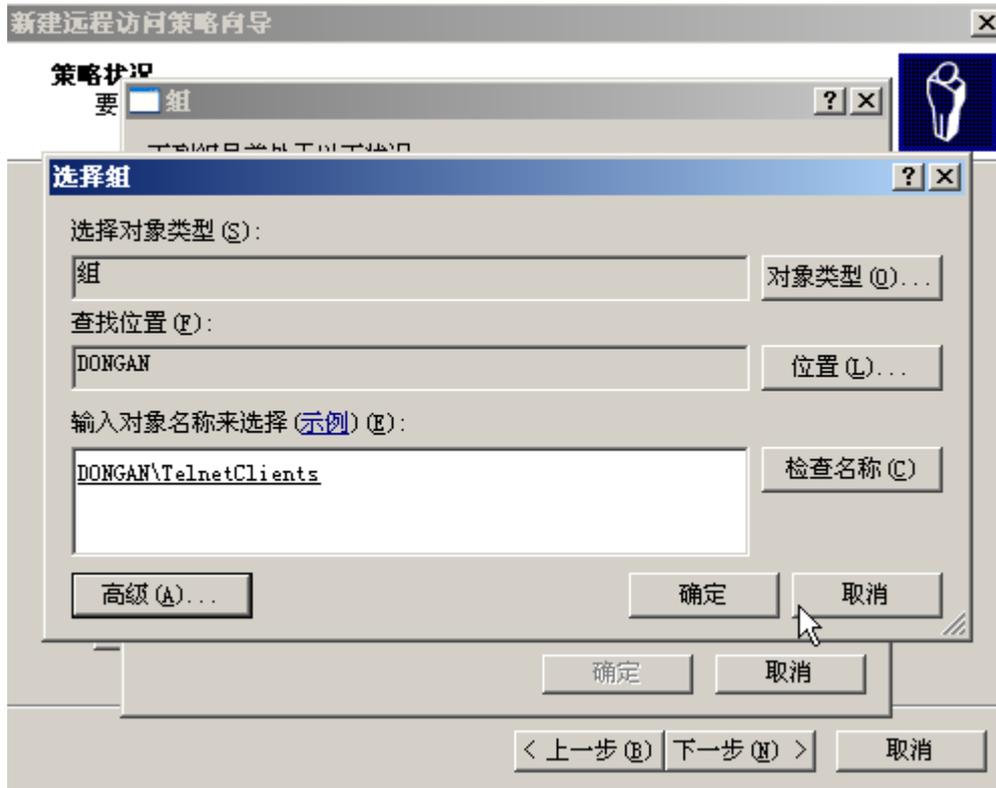


名称随便填，客户端的地址就是 openvpn 服务器地址



B 新建远程访问策略



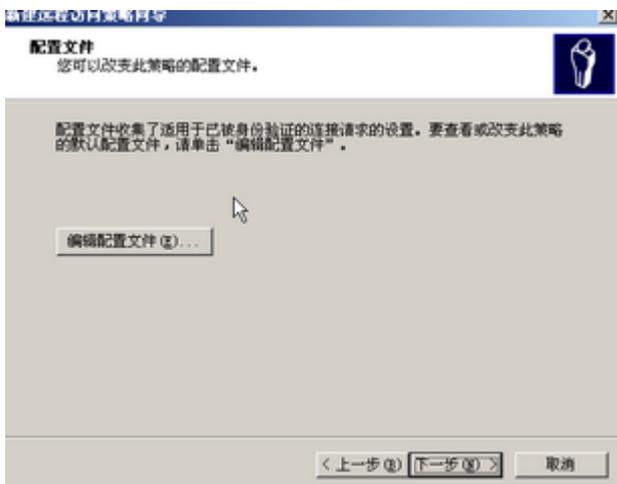


授予 TelnetClients 组远程访问权限

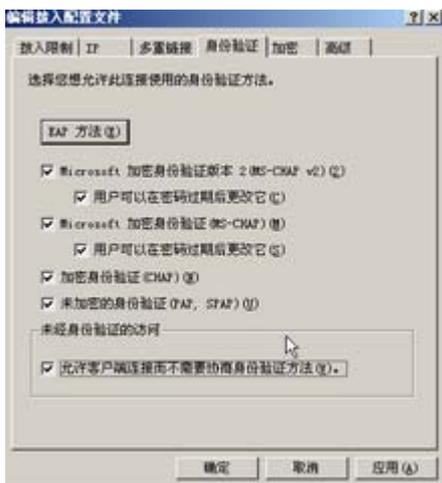




重要：点击-> 编辑配置文件



身份验证选项卡 如下配置



Ok 确定之后 radius server 配置完成

接下来测试 radius 服务器

1. 在服务器上 新建一个用户

用户名: vpn 密码: test 加入 TelnetClients 组 确保它的“属性”中“拨入”选项卡的“远程访问权限”设置为“通过远程访问策略控制访问”

2. 查看服务端口是否开启

```
:\Documents and Settings\Administrator>netstat -na | find "1645"  
UDP    0.0.0.0:1645          *:*  
  
:\Documents and Settings\Administrator>
```

Radius 的常用端口

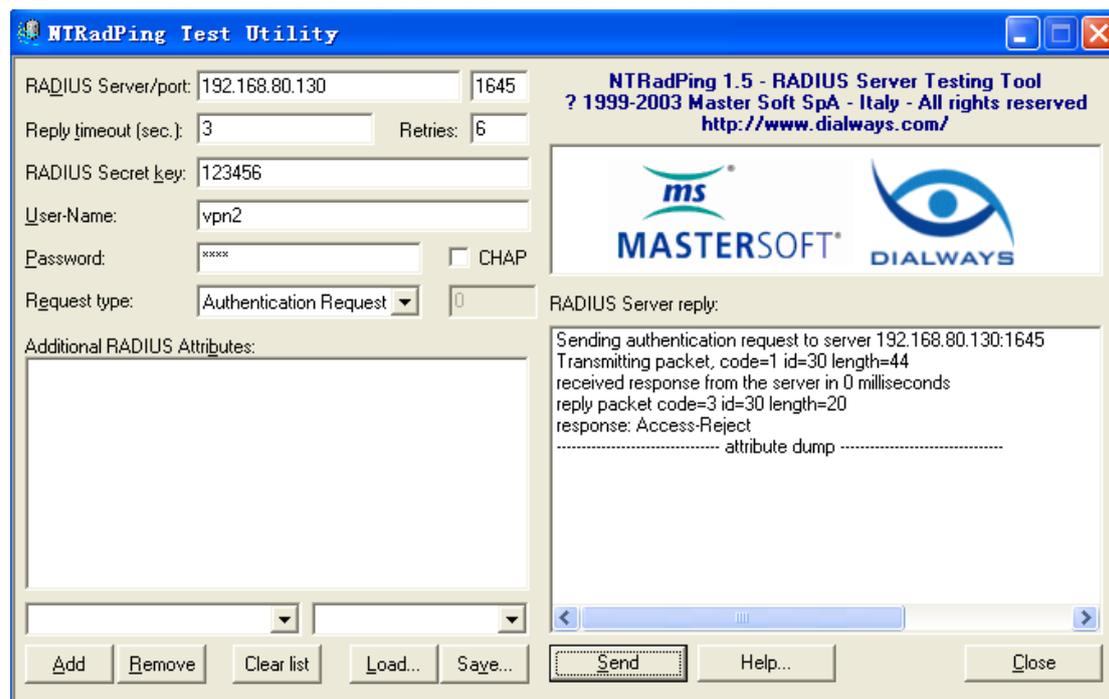


2. Windows pc 上使用软件 ntradping 进行测试

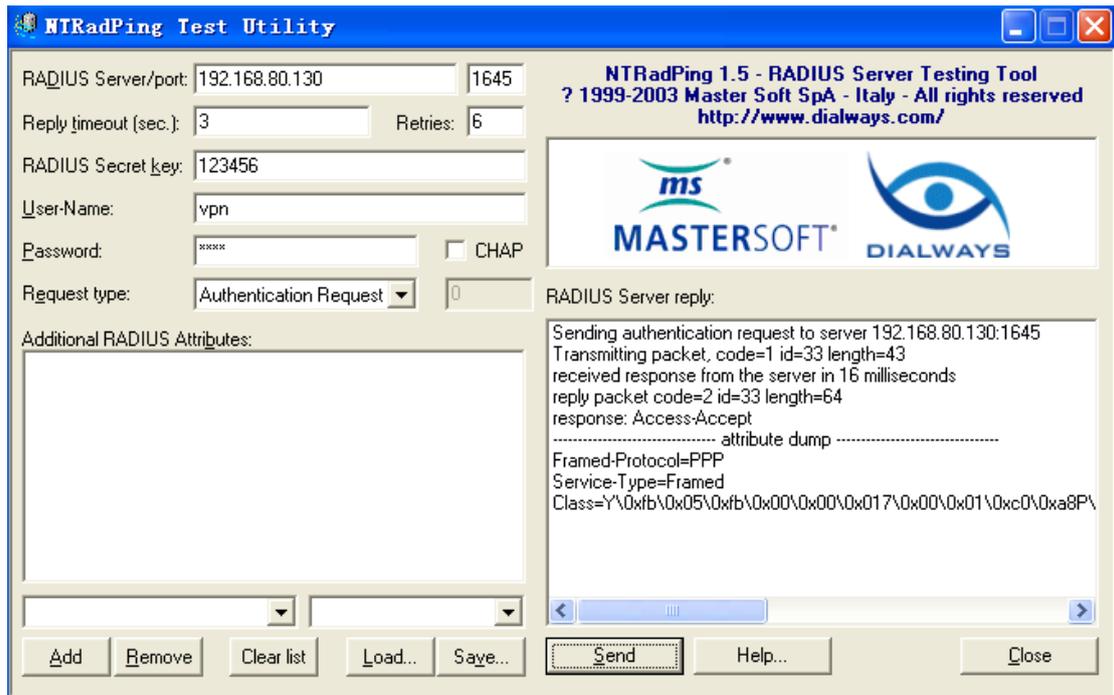
提示：你需要为你的 pc 在 radius 服务器上也新建一个 RADIUS 客户端

在软件左上方中输入认证信息 -> 接着点击下方的 send 按钮

A radius 服务器正常 但认证失败（用户没有权限/密码错误等）



B radius 认证成功



至此 radius 服务器搭建完成

Popauth.pl

```
#!/usr/bin/perl
# Write by ELM
# wzk [A/T] wenzk [D/O/T] net
# http://www.wenzk.net
#
# POP3 Auth script for OpenVPN
# Usage:
# save this file to /etc/openvpn/ as popauth.pl
# run£° chmod +x /etc/openvpn/popauth.pl
# add the next line to you OpenVPN config file
# auth-user-pass-verify /etc/openvpn/popauth.pl via-env
# restart your OpenVPN Server
```

```
use Net::POP3;
```

```
# POP Server Address
```

```
$POPHOST = "localhost";
```

```
$USERNAME = $ENV{username};
```

```
$PASSWORD = $ENV{password};
```

```
# Constructors
```

```
$pop = Net::POP3->new($POPHOST);
```

```
$pop = Net::POP3->new($POPHOST, Timeout => 60);
```

```
if ( defined $pop->login($USERNAME, $PASSWORD) ) {
```

```
    $result = 0;
```

```
    } else {
```

```
        $result = 1;
```

```
    }
```

```
$pop->quit;
```

```
exit $result;
```

centos yum 光盘源

2008-09-04 18:42

CentOS 4 下 Yum 安装本地光盘软件 首先 要先挂载光盘，把光盘挂载到 media/CentOS/, media/cdrom/, 或者 media/cdrecorder/这样就不用修改配置文件了。

命令：

```
mount /dev/cdrom /media/cdrom
```

如果想挂载到其他目录，只要修改下 baseurl=file:///自己挂载的目录，就可以了。配置文件在/etc/yum.repos.d/CentOS-Media.repo。

接着 把/etc/yum.repos.d/CentOS-Base.repo 改下名，命令如下：

QUOTE:

```
#mv /etc/yum.repos.d/CentOS-Base.repo  
/etc/yum.repos.d/CentOS-Base.repo.bak
```

检查下配置文件的有没设置正确，命令如下：

QUOTE:

```
# CentOS-Media.repo  
#  
# This repo is used to mount the default locations for a CDROM / DVD on  
# CentOS-4. You can use this repo and yum to install items directly off  
the  
# DVD ISO that we release.  
#  
# To use this repo, put in your DVD and use it with the other repos too:  
# yum --enablerepo=c4-media [command]  
#  
# or for ONLY the media repo, do this:  
#  
# yum --disablerepo=\* --enablerepo=c4-media [command]
```

[c4-media]

name=CentOS-\$releasever - Media

baseurl=file:///media/cdrom/

<file:///media/cdrecorder/>

gpgcheck=1

enabled=0

gpgkey=file:///usr/share/doc/centos-release-4/RPM-GPG-KEY-centos4

其实配置文件中已经有教怎么做的了，我好像是多此一举了。

最后 就是要安装软件咯，命令如下：

```
yum --disablerepo=\* --enablerepo=c4-media install mysql-server
```

```
yum --disablerepo=\* --enablerepo=c4-media install mysql-devel
```

```
yum --disablerepo=\* --enablerepo=c4-media install curl
```

```
yum --disablerepo=\* --enablerepo=c4-media install php-mysql
```

```
yum --disablerepo=\* --enablerepo=c4-media install compat-libstdc++-33
```

```
yum --disablerepo=\* --enablerepo=c4-media install libtool
```

```
rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY*
```

```
yum --disablerepo=\* --enablerepo=c4-media install mysql-server  
mysql-devel curl php-mysql compat-libstdc++-33 libtool net-snmp php-gd  
php-snmp perl-DateManip perl-MD5 net-snmp-utils freetype sudo mod_ssl
```

分类: [技术归档](#)

字号: [大](#) [中](#) [小](#)

一、前言

具体的调用流程是:

win 的 openvpn 客户端通过 auth-user-pass 认证模式 (即通过 ca.crt ta.key server.crt) 到 openvpn 服务器, 调用插件 radiusplugin (/usr/local/openvpn/radiusplugin.so , /usr/local/openvpn/radiusplugin.cnf), radiusplugin 调用 radiusd(name=127.0.0.1、sharedsecret=testing123 和 radius 的服务端口), 通过 radiusd 的 clients.conf 实现服务器的本身 127.0.0.1 密钥模式调用 radiusd, 再通过 radiusd.conf 配置 sql 方式的认证, 再到 radiusd 的 sql.conf (通过 rlm_sql_mysql 调用) 到 mysql 数据库

具体的安装是上述调用的逆向过程

二、安装 freeradius 以及 rlm_sql_mysql.so

1、 编译安装

(1)、编译安装 freeradius

```
wget ftp://ftp.freeradius.org/pub/radius/freeradius-1.1.7.tar.gz
```

```
tar zxvf freeradius-1.1.7.tar.gz
```

```
cd freeradius-1.1.7
```

```
./configure
```

```
cp libltdl/ltdl.h src/include/
```

```
make
```

```
make install
```

注释：如果没有 `cp libltdl/ltdl.h src/include/` 在 `make` 的时候会有 2 个错误，应该是路径的问题

(2)、安装 `rlm_sql_mysql` 相关

由于我的 `mysql` 是编译好的二进制包安装的，所以系统没有 `rlm_sql_mysql.so` 这个东西

所以要重新编译出 `rlm_sql_mysql.so` 这个东西来。

```
ln -s /usr/local/mysql/bin/mysql_config /sbin/mysql_config

cd /usr/local/src/freeradius-1.1.7/src/modules/rlm_sql/drivers/rlm_sql_mysql

./configure --with-mysql-dir=/usr/local/mysql --with-mysql-lib-dir=/usr/local/mysql/lib \

--with-mysql-include-dir=/usr/local/mysql/include

make

make install
```

这里的编译参数是根据每个人的实际情况设定的，这一步的编译要在 `freeradius` 编译安装后再做。

```
vi /etc/ld.so.conf

/usr/local/lib

ldconfig
```

2、构建 `radius` 数据库(建立相关用户和密码)

```
/usr/local/mysql/bin/mysql -uroot -pXXXX

mysql>create database radius;

mysql>GRANT ALL PRIVILEGES ON radius.* TO radius@localhost
IDENTIFIED BY 'AAAA';

mysql>\q

cd /usr/local/src/freeradius-1.1.7

/usr/local/mysql/bin/mysql -uroot -pXXXX radius
< ./doc/examples/mysql.sql
```

3、配置 radiusd.conf 以及 sql.conf

(1)、radiusd.conf 配置

```
vi /usr/local/etc/raddb/radiusd.conf
```

```
proxy_requests = no
```

```
authorize {
```

```
preprocess
```

```
chap
```

```
mschap
```

```
suffix
```

```
eap
```

```
sql
```

```
# pap
```

```
# files
```

```
}
```

注释：使用 sql，authorize 中的 file 必须被注释掉。

```
preacct {  
  
    preprocess  
  
    acct_unique  
  
    suffix  
  
#    files  
  
}
```

```
accounting {  
  
    detail  
  
    unix  
  
    radutmp  
  
    sql  
  
}
```

(2)、sql.conf 配置

```
vi /usr/local/etc/raddb/sql.conf
```

```
driver = "rlm_sql_mysql"
```

```
server = "localhost"
```

```
login = "radius"
```

```
password = "AAAA"
```

```
radius_db = "radius"
```

配置 NAS 信息：

```
vi /usr/local/etc/raddb/clients.conf
```

```
#本地帐号，用于测试
```

```
client 127.0.0.1 {
    secret          = testing123
    shortname       = localhost
    nastype         = other
}
```

注释:

secret 这里是针对 client 127.0.0.1 通讯密钥

```
#mysql -u root -p
```

```
mysql> use radius;
```

建立组信息:

```
mysql> insert into radgroupreply (groupname, attribute, op, value)
values ('user', 'Auth-Type', ':=', 'Local');
```

```
mysql> insert into radgroupreply (groupname, attribute, op, value)
values ('user', 'Service-Type', ':=', 'Framed-User');
```

```
mysql> insert into radgroupreply (groupname, attribute, op, value)
values ('user', 'Framed-IP-Address', ':=', '255.255.255.255');
```

```
mysql> insert into radgroupreply (groupname, attribute, op, value)
values ('user', 'Framed-IP-Netmask', ':=', '255.255.255.0');
```

建立用户信息:

```
mysql> insert into radcheck (username, attribute, op, value) values
('test', 'User-Password', ':=', 'test');
```

```
mysql> insert into radcheck (username, attribute, op, value) values
('sense', 'User-Password', ':=', '123456');
```

将用户加入组中:

```
mysql> insert into usergroup (username, groupname) values
('test', 'user');
```

注释:

对于 radcheck 这个表 attribute 的项有好几种设置的,我们设置的是认证的密码模式明码 User-Password,也可以设置成加密 Crypt-Password,如果是加密模式需要 libcrypt 支持。

同时对于表 radgroupreply 中项 attribute, 的含义不是太明白:
Auth-Type 、 Service-Type 、 Framed-IP-Address、 Framed-IP-Netmask、
Acct-Interim-Interval

测试:

```
radiusd -x &
```

这个时候控制台会被占用

另开一个控制台进行测试

```
ln -s /usr/local/bin/radtest /sbin/radtest
```

```
/sbin/radtest test test localhost 0 testing123
```

Sending Access-Request of id 204 to 127.0.0.1 port 1812

User-Name = "test"

User-Password = "test"

NAS-IP-Address = 255.255.255.255

NAS-Port = 0

rad_recv: Access-Accept packet from host 127.0.0.1:1812, id=204, length=38

Service-Type = Framed-User

Framed-IP-Address = 255.255.255.255

Framed-IP-Netmask = 255.255.255.0

/sbin/radtest sense 123456 localhost 0 testing123

Sending Access-Request of id 212 to 127.0.0.1 port 1812

User-Name = "sense"

User-Password = "123456"

NAS-IP-Address = 255.255.255.255

NAS-Port = 0

rad_recv: Access-Accept packet from host 127.0.0.1:1812, id=212, length=20

由于 test 加入了组所以信息量多一点

以上测试说明 freeradius-mysql 工作正常!

说明 (freeradius\rlm_sql_mysql.so\mysql\) 正常!

三、 装 OpenVPN

1、其中我的 openssl 我是系统自带的,所以不要再次安装

```
rpm -aq|grep openssl
```

```
openssl-devel-0.9.8b-8.3.e15
```

```
openssl-0.9.8b-8.3.e15
```

2、安装 openvpn2.09

```
cd /usr/local/src
```

```
wget http://openvpn.net/release/openvpn-2.0.9.tar.gz
```

```
tar zxvf openvpn-2.0.9.tar.gz
```

```
cd openvpn-2.0.9
```

首先是看一下 openvpn.spec 里面对于 liblzo1 和 openssl 的版本要求，对照自己的版本看符合要求吗

检查 pam-devel 包是否安装，否则从系统盘安装改软件包，一般系统安装时都已经安装了

这一部分的 pam 不一定要安装，因为 openvpn 不是直接调用 mysql 的，如果是 openvpn 直接调用 mysql 需要安装 pam。（openvpn 直接通过 mysql，跟通过 radius 调用 mysql 主要是在计费统计上有很大区别，认证什么的差不多）

```
rpm -qa | grep pam
```

```
pam_passwdqc-1.0.2-1.2.2
```

```
pam-0.99.6.2-3.14.e15
```

```
pam_pkcs11-0.5.3-23
```

```
pam_krb5-2.2.11-1
```

```
pam-devel-0.99.6.2-3.14.e15
```

```
pam_smb-1.1.7-7.2.1
```

```
pam_ccreds-3-5
```

(1)、lzo

```
wget
```

<http://www.oberhumer.com/opensource/lzo/download/lzo-2.02.tar.gz>

```
tar zxvf lzo-2.02.tar.gz
```

```
cd lzo-2.02
```

```
./configure
```

```
make
```

```
make install
```

这一部分是支持 openvpn 的压缩功能

(2)、安装 openvpn

```
cd openvpn-2.0.9
```

```
./configure --prefix=/usr/local/openvpn  
--with-lzo-headers=/usr/local/include/lzo \  
  
--with-lzo-lib=/usr/local/lib  
--with-ssl-headers=/usr/include/openssl \  
  
--with-ssl-lib=/usr/lib
```

```
make
```

```
make install
```

注释：以上路径请根据自己系统的配置调整

(3)、生成密钥

```
cd /usr/local/src/openvpn-2.0.9
```

```
cp -rf ./easy-rsa/ /usr/local/openvpn/
```

```
cd /usr/local/openvpn/easy-rsa/2.0
```

```
source ./vars
```

```
./clean-all
```

```
./build-ca
```

```
./build-key-server server
```

```
./build-dh
```

```
./build-key c11
```

```
cd keys

/usr/local/openvpn/sbin/openvpn --genkey --secret ta.key

cd /usr/local/openvpn

mkdir ssl

cp -a ./easy-rsa/keys/2.0/ca.crt ./ssl/

cp -a ./easy-rsa/keys/2.0/dh1024.pem ./ssl/

cp -a ./easy-rsa/keys/2.0/ta.key ./ssl/

cp -a ./easy-rsa/keys/2.0/server.crt ./ssl/

cp -a ./easy-rsa/keys/2.0/server.key ./ssl/
```

3、安装 RadiusPlugin

(1)、安装

```
cd /usr/local/src/

wget wget http://www.nongnu.org/radiusplugin/radiusplugin\_v2.0b\_beta2.tar.gz

tar zxvf radiusplugin_v2.0b_beta2.tar.gz

cd radiusplugin_v2.0b_beta2

make

cp /usr/local/src/radiusplugin_v2.0b_beta2/radiusplugin.so
/usr/local/openvpn/

cp /usr/local/src/radiusplugin_v2.0b_beta2/radiusplugin.cnf
/usr/local/openvpn/

vi /usr/local/openvpn/radiusplugin.cnf
```

这里 2 个文件的位置，是自己定义的，根据这个定义的位置，后面相关配置要设定这里路径的

(2)、配置 radiusplusin 插件的配置文件 radiusplugin.cnf

```
# The NAS identifier which is sent to the RADIUS server
NAS-Identifier=OpenVpn

# The service type which is sent to the RADIUS server
Service-Type=5

# The framed protocol which is sent to the RADIUS server
Framed-Protocol=1

# The NAS port type which is sent to the RADIUS server
NAS-Port-Type=5

# The NAS IP address which is sent to the RADIUS server
NAS-IP-Address=127.0.0.1

# Path to the OpenVPN configfile. The plugin searches there for
# client-config-dir PATH    (searches for the path)

# status FILE                (searches for the file, version
must be 1)

# client-cert-not-required (if the option is used or not)

# username-as-common-name  (if the option is used or not)

OpenVPNConfig=/usr/local/openvpn/server.conf

# Support for topology option in OpenVPN 2.1
```

If you don't specify anything, option "net30" (default in OpenVPN) is used.

You can only use one of the options at the same time.

If you use topology option "subnet", fill in the right netmask, e.g. from OpenVPN option "--server NETWORK NETMASK"

#subnet=255.255.255.0

If you use topology option "p2p", fill in the right network, e.g. from OpenVPN option "--server NETWORK NETMASK"

#p2p=10.10.0.1

Ich benutze die Default Option

Allows the plugin to overwrite the client config in client config file directory,

default is true

overwriteccfiles=true

Path to a script for vendor specific attributes.

Leave it out if you don't use an own script.

vsascript=/root/workspace/radiusplugin_v2.0.5_beta/vsascript.pl

Path to the pipe for communication with the vsascript.

Leave it out if you don't use an own script.

vsanamedpipe=/tmp/vsapipe

A radius server definition, there could be more than one.

The priority of the server depends on the order in this file. The first one has the highest priority.

```
server
{
    # The UDP port for radius accounting.
    acctport=1813

    # The UDP port for radius authentication.
    authport=1812

    # The name or ip address of the radius server.
    name=127.0.0.1

    # How many times should the plugin send the if there is no
response?
    retry=1

    # How long should the plugin wait for a response?
    wait=1

    # The shared secret.
    sharedsecret= testing123
}
```

注释：这里的 sharedsecret= testing123 跟 radiusd 的设置相关，注意 /usr/local/etc/raddb/clients.conf 中的 client 127.0.0.1 的设置，（这个 clients.conf 是调用 radius 的客户端配置）

```
client 127.0.0.1 {
    secret = testing123
```

```
    shortname = localhost

    nastype = other

}
```

同时重要的是要在正式应用的时候 同时修改这 2 个地方的 testing123, 这个就是 radius 的密钥

4、配置 openvpn 的服务器设置文件 server.conf

```
cp /usr/local/src/openvpn-2.0.9/sample-config-files/server.conf
/usr/local/openvpn/server.conf
```

```
vi /usr/local/openvpn/server.conf
```

注释：注意这里的 server.conf 要与 /usr/local/openvpn/radiusplugin.cnf 中 OpenVPNConfig 的设置一致

这里的 openvpn 服务器端可以有多个配置文件, 每一个配置等于开启了一个单独的 vpn 服务, 但是要单独设置每个服务器端配置文件. /build-key-server ser*** 同时不同的客户端. /build-key cli*** 对应于相应 ser****的 unit name, 同时客户端要拷贝不同的 ser***. crt

```
port 1194

proto udp

# Which device

dev tun

;fast-io

user nobody

group nogroup

persist-tun
```

```
persist-key

server 10.14.0.0 255.255.0.0

management 127.0.0.1 7505

float

username-as-common-name

;client-config-dir ccd

client-cert-not-required

client-to-client

#push "redirect-gateway def1"

push "dhcp-option DNS 172.21.41.15"

ping-timer-rem

keepalive 10 120

# Use compression

comp-lzo

# Strong encryption

tls-server

tls-auth ssl/ta.key 0

dh ssl/dh1024.pem

ca ssl/ca.crt

cert ssl/server.crt

key ssl/server.key

max-clients 200
```

```
plugin /usr/local/openvpn/radiusplugin.so  
/usr/local/openvpn/radiusplugin.cnf
```

```
verb 3
```

```
mute 10
```

```
status /var/log/openvpn/status.log 1
```

```
log /var/log/openvpn/openvpn.log
```

```
mkdir /usr/local/openvpn/ccd
```

```
mkdir /var/log/openvpn
```

这里我定义 10.14 段主要是不想跟别人定义的冲突：

```
#让客户端发起的所有 IP 请求都通过 OPENVPN 服务器
```

#这一句还是不用的好，没必要全部从 vpn 走，一般通过 dns 把内网的通过这里就可以了

#除非对方本地上网是被限制的，那么可以考虑开放这个配置，所有的通过 vpn 上网

```
push "redirect-gateway def1"
```

```
#定义客户端的 dns 服务器地址，设置本地的服务器内网地址就可以了
```

```
push "dhcp-option DNS 172.21.41.15"
```

注释：（这一部分暂时用不到，就是端到端的服务配置）

关于服务器端内网可以访问客户端内网的设置

```
# 使服务器子网内机器可以访问客户端子网内机器
```

```
# 仅用于路由模式
```

```
# 假设：客户端子网网段 192.168.1.0
```

首先, 在服务器配置文件中添加下面这两行

```
# client-config-dir ccd
```

```
# 和 route 192.168.1.0 255.255.255.0
```

然后在服务器端 ccd 目录下创建一个文件, 文件名是客户端的公共名, 这里的公共名是客户证书的公共名

文件内容是:

```
# iroute 192.168.1.0 255.255.255.0
```

```
;client-config-dir ccd
```

```
;route 192.168.1.0 255.255.255.0
```

ccd 目录通常在配置文件目录下面建立, 公共名 (common name) 在生成证书, 回答问题时填上的, 跟证书的文件名一至.

route 192.168.1.0 255.255.255.0 表示在服务器端增加访问客户端的路由 (192.168.1.0/24 指的是客户端网段)

我现在的配置还暂时用不到这个设置项, 以上只是参考, 感觉有些特殊的地方还是用的到的!

client-to-client 如果让 Client 之间可以相互看见, 去掉本行的注释掉, 否则 Client 之间无法相互访问

duplicate-cn 是否允许一个 User 同时登录多次, 去掉本行注释后可以使用同一个用户名登录多次

plugin /usr/local/opensvpn/radiusplugin.so
/usr/local/opensvpn/radiusplugin.cnf 说明使用的插件,

client-cert-not-required #不请求客户的 CA 证书

username-as-common-name #使用客户提供的 Username 作为 Common Name

四、开启 radius 和 opensvpn 服务

1、建立 opensvpn 的 shell 执行脚本

```
cp  
/usr/local/src/opensvpn-2.0.9/sample-scripts/opensvpn.init /etc/init.d  
/
```

```
ln -s /usr/local/openvpn/sbin/openvpn /usr/sbin/openvpn
```

```
vi /etc/init.d/openvpn.init
```

```
work=/usr/local/openvpn
```

2、开启 openvpn 服务器进程

```
/etc/init.d/openvpn.init start
```

这个时候要看一下 ifconfig 有没有 tun0 设备，如果有就基本 ok 了！

如果有错误，请看 /var/log/messages 和 /var/log/openvpn/openvpn.log 的信息

如果关闭 openvpn 使用：

```
/etc/init.d/openvpn.init stop
```

在测试的时候，由于配置出错，会碰到 /etc/init.d/openvpn.init stop 关闭不了，使用

```
killall openvpn
```

 来关闭

3、开启 radius 服务

这个做到这里容易忘记，我就忘了，还查了一阵子错误

```
radiusd &
```

五、windows 下 client.opvn 客户端配置文件：

```
# Which device
```

```
dev tun
```

```
persist-key
```

```
persist-tun
```

```
# Our remote peer
```

```
nobind
```

```
remote *.*.*.* 1194
```

#出现用户、密码的认证

auth-user-pass

ns-cert-type server

tls-auth ta.key 1

#对于客户端建立了 vpn 连接后，通过 vpn 远程走的 route 设定（这里 172.16.0.0 255.248.0.0 是所有的内网包括从内部路由走出去的其他公司的内网，可以添加多个网段。）

route 172.16.0.0 255.240.0.0

Use compression

comp-lzo

Strong encryption

verb 3

mute 10

六、服务器端的路由和防火墙的调整

把以下的 iptables 命令放在靠前的规则中

#我这里定义\$IPT 是 /sbin/iptables

#对于 udp 1813 和 udp 1812 端口，因为 radius 是 127.0.0.1 本地的调用，所以不要另外再开许可

#打开 openvpn 连接端口 udp 1194

```
$IPT -A INPUT -p udp -m udp --dport 1194 -j ACCEPT
```

#许可 tun 设备的，如果你开的是 tap 设备进行相应的更改

```
$IPT -A INPUT -i tun0 -j ACCEPT
```

```
$IPT -A OUTPUT -o tun0 -j ACCEPT
```

```
$IPT -A FORWARD -i tun0 -j ACCEPT
```

```
$IPT -t nat -A POSTROUTING -s 10.14.0.0/24 -o $LOCAL_IFACE -j SNAT
--to-source $LOCAL_IP
```

七、自安装客户端的生成步骤

1、下载客户端制作程序

<http://www.openvpn.se/files/nsis/nsis205.exe>

http://www.openvpn.se/files/install_packages_source/openvpn_install_source-2.0.9-gui-1.0.3.zip

2、安装 nsis205.exe

(1)、生成正确的配置源

在windows下解压缩[openvpn_install_source-2.0.9-gui-1.0.3.zip](http://www.openvpn.se/files/install_packages_source/openvpn_install_source-2.0.9-gui-1.0.3.zip)

进入 openvpn_install_source-2.0.9-gui-1.0.3/openvpn/目录下，建立 config，

将正确的客户端的配置文件和密匙放到这个目录下（一个配置文件、3个跟密匙有关的文件）：

client.ovpn、 ta.key 、 ca.crt、 server.crt

注释：

clint.ovpn 就是 client.conf!

(2)、编辑生成正确的配置文件

编辑 openvpn_install_source-2.0.9-gui-1.0.3 目录下 openvpn-gui.nsi 文件

查找 ;File "\${HOME}\config\Office.ovpn"

替换成

File "\${HOME}\config\client.ovpn"

File "\${HOME}\config\ca.crt"

File "\${HOME}\config\server.crt"

File "\${HOME}\config\ta.key"

(3)、生成自动安装程序

运行 NSIS Menu 的

compiler -- star MakNsiSW

菜单 files--load script

导入 openvpn_install_source-2.0.9-gui-1.0.3/目录下的
openvpn-gui.nsi

导入这个 openvpn-gui.nsi 就会在目录
openvpn_install_source-2.0.9-gui-1.0.3 下自动生成
openvpn_install_source-2.0.9-gui-1.0.3-install.exe

在客户端运行这个程序就能得到正确配置的客户端！

参考：

http://www.roessner-net.com/VPN_RADIUS_MYSQL.howto.txt

<http://www.linuxfly.org/read.php?86>

<http://www.xiaohui.com/dev/server/20070514-install-openvpn.htm>

http://www.xxlinux.com/linux/article/development/database/20060707/2522_2.html

<http://blog.chinaunix.net/u/2389/>

Ethernet Bridging

译者：温占考（Email: wzk<AT>wenzk<DOT>net），来源：OpenVPN.net，转载请注明译者和出处及版权信息，并且不能用于商业用途，违者必究。

由于英语水平有限，如有错误，还望指正！

时间：2006-06-19 22:00 于 沈阳

桥接概述

查看[FAQ](#)了解路由对比桥接的概述。

以太网桥接实质上是把一个以太网接口和一个或多个虚拟 TAP 接口桥接在一个网桥接口上。以太网桥表现为使用软件模拟一个真实的以太网交换机。以太网桥可以理解为一种能够用来连接一个机器上的多个以太网接口（无论是物理的还是虚拟的）使它们共享一个 IP 子网的软件交换机。

把分布在两个独立网络的一个物理的以太网 NIC 和一个 OpenVPN 所使用的 TAP 接口桥接起来，这样就可以逻辑上合并两个以太网，就像它们在一个以太网子网内。

配置桥接

这个例子将指导你配置一个 OpenVPN 服务器端的桥接。多个客户将会连接到这个网桥上，并且每个客户机的 TAP 接口将会分配到一个服务器端 LAN 网段的 IP 地址。

下面是两个实现客户 IP 地址分配的方法：

- 让 OpenVPN 使用 **server-bridge** 指示来管理其客户的 IP 地址池，或者
- 在 LAN 内配置 DHCP 服务器让其为 VPN 客户分配 IP 地址。

在这个例子中，我们将使用第一个方法即 OpenVPN 服务器管理其客户在 LAN 子网内的 IP 地址池，和 DHCP 服务器使用的地址池分开（如果存在的话）。两种方法在[FAQ item](#)都有详细描述。

在我们的例子中，我们将使用下面的网桥配置：

设置	网桥启动 参数	取值
以太网接口	eth	eth0
本地 IP 地址	ip	192.168.8.4

本地子网掩码	eth_netmask	255.255.255.0
本地广播地址	eth_broadcast	192.168.8.255
VPN 客户地址池		192.168.8.128 to 192.168.8.254
虚拟网桥接口	br	br0
虚拟 TAP 接口	tap	tap0

第一步是按照[HOWTO](#)中的“开始使用VPN并且测试其连通性(Starting up the VPN and testing for initial connectivity)”章节。然后，根据你是在Linux或Windows下配置桥接继续配置。

在 Linux 上配置桥接服务器

首先，确认你已经安装了 **bridge-utils** 软件包。

编辑下面的[bridge-start](#)（启动网桥）脚本。根据你要桥接的物理网络接口设置**br**，**tap**，**eth**，**eth_ip**，**eth_netmask**，和 **eth_broadcast** 参数。确认使用一个内部网络接口并且是连接在一个受防火墙保护的LAN内。你可以使用Linux的**ifconfig**命令获得网络接口的一些必要的信息来完成**bridge-start**（启动网桥）参数。

现在就运行 **bridge-start** 脚本。脚本将会创建一个永久[译者注：永久意为系统重新启动前永久，默认虚拟接口在 OpenVPN 退出后就消失]的 **tap0** 网络接口并且已经和以太网接口桥接成功了。

下一步，我们将要编辑[OpenVPN server configuration file](#)（OpenVPN服务器端配置文件）来使能桥接配置。

注释掉 **dev tun** 这行行并且替换成：

```
dev tap0
```

注释掉以 **server** 开始的行并且替换成：

```
server-bridge 192.168.8.4 255.255.255.0 192.168.8.128 192.168.8.254
```

现在设置 Linux 防火墙使之允许数据包通过新建的 **tap0** 和 **br0** 接口：

```
iptables -A INPUT -i tap0 -j ACCEPT
```

```
iptables -A INPUT -i br0 -j ACCEPT
```

```
iptables -A FORWARD -i br0 -j ACCEPT
```

OpenVPN 桥接现在能够通过下面顺序来启动和停止：

- 运行 **bridge-start**
- 运行 `openvpn`
- 停止 `openvpn`
- 运行 **bridge-stop**

到目前为止，桥接的配置就完成了，你可以[continue where you left off in the HOWTO](#)（根据 HOWTO 继续完成剩下部分）。

在 Windows XP 中配置桥接服务器

这个配置在桥接端需要 Windows XP 或更高版本系统。据我所知，Windows 2000 不支持桥接，然而 Windows 2000 系统可以是一个桥接网络的客户，而 OpenVPN 另一端的桥接是在 Linux 或 Windows XP 系统上实现的。

在 Windows 上安装 OpenVPN 时，安装程序将自动创建一个名为“本地连接 2”的 TAP-Win32 适配器。转到控制面板中的网络连接下并且改名为“tap-bridge”。

接下来使用鼠标选择 **tap-bridge** 和以太网适配器，点击鼠标右键，并且选择**桥接（Bridge Connections）**。这样就在控制面板中创建了一个新的**桥适配器**图标。

在桥适配器的 TCP/IP 属性里配置 IP 地址 192.168.8.4 子网掩码 255.255.255.0。

接下来，编辑[OpenVPN server configuration file](#)（OpenVPN 服务端配置文件）使能桥接配置。

注释 `dev tun` 这行且替换成：

```
dev tap
```

```
dev-node tap-bridge
```

注释以 `server` 开始的行并且替换成：

```
server-bridge 192.168.8.4 255.255.255.0 192.168.8.128 192.168.8.254
```

如果你的系统是 XP SP2，进入防火墙配置控制面板，并且禁用桥适配器和 TAP 适配器上的包过滤功能。

到目前为止，桥接的配置就完成了，你可以[continue where you left off in the HOWTO](#)（根据 HOWTO 继续完成剩下部分）。

配置桥接客户端

在[sample OpenVPN client configuration](#)（OpenVPN客户配置）基础上进行配置。注释**dev tun**的行且替换成：

dev tap

最后，确认客户端的配置文件和服务器端的配置文件是否一致。需要着重注意的是**proto** (udp or tcp)选项是否一致。同样如果使用了**comp-lzo** 和 **fragment** 确认客户端和服务器端是否一致。

以太网桥接说明

使用以太网桥接配置时，第一步是构造以太网桥--一种虚拟网络接口是其他网络接口的一个容器，可以是物理的 NIC 也可以是虚拟的 TAP 接口。以太网桥接口必须在 OpenVPN 启动之前创建。

没有通用的网桥配置方法--每个 OS 有自己的配置方法（见下面例子）。

网桥接口一旦被创建，并且以太网接口加入到网桥中，OpenVPN 就可以启动了。

- 网桥接口是一种由一个或多个以太网接口组成的虚拟网络接口，这些以太网接口可以是物理网络接口或 OpenVPN 所使用的虚拟 TAP 接口。
- 当你配置以太网桥接口时，你需要手工设置网桥接口的 IP 地址和子网掩码并且不要在 OpenVPN 配置文件中**ifconfig** 选项来配置。这是因为与 TUN/TAP 接口不一样，OpenVPN 程序不能配置网桥接口的 IP 地址和子网掩码。
- OpenVPN 配置文件应该通过 **dev** 选项来指定已经加入网桥 TAP 接口，而不是指定网桥接口的名字。
- 对于 Windows 系统，使用 **dev-node** 选项来指定已经加入网桥 TAP-Win32 适配器（**dev-node** 这个名字是网络连接控制面板上显示的名字）。
- 在 Linux/BSD/Unix 系统中，对于 **dev tap** 选项，使用你已经加入网桥的 TUN/TAP 的序号如 **dev tap0**。
- 如果你的 OpenVPN 工作在点对点模式，去掉 **ifconfig** 选项，如果你使用的是客户/服务器模式，在服务器端使用 **server-bridge** 选项。
- 桥接的时候，你必须手工设置网桥接口的 TCP/IP 设置。例如在 Linux 中，可以使用 **ifconfig** 命令配置而在 Windows XP 中可以通过配置网络连接面板中网桥接口的 TCP/IP 属性来实现（在 Windows XP 和更高版本的网络连接面板中可以通过鼠标操作来实现桥接）。
- 确定只有桥接的 TAP 接口和内部以太网接口是被防火墙保护的。不要把 TAP 接口和连接互联网的网络接口进行桥接，这样可能存在潜在的安全漏洞。
- **local** 和 **remote** 所使用的地址不能是桥接子网内的地址--否则将会出现路由环路。
- 理解以太网桥接很重要的一点就是每个加入网桥的网络接口将会失去原有的配置如 IP 地址和子网掩码。只有网桥接口的 TCP/IP 设置有效。
- 手工配置网桥时常见的错误是在为网桥配置 IP 地址和子网掩码之前把主要的以太网网络适配器加入到网桥中。这就导致主要的以太网卡“丢失”了原有的配置，但是所属的网桥接口还没有配置，所以这就造成了以太网网络接口的连接就丢失了。

- 在大多数情况下，只是在服务器端配置一个可用的桥接，而不是在客户端配置。这样一来，当客户机连接上服务器后就成为了多宿主系统了，例如：它们仍然拥有原有的以太网接口，但是连接到 OpenVPN 服务器的上层连接，它们将会有一个新的 TAP 接口和服务器端的以太网接口桥接（并且有可能是所有连接到服务器的客户机的 TAP 接口如果在服务器的配置文件里申明了 **client-to-client** 选项）。

说明 - Windows 下的以太网桥接

Check out [this HOWTO](#) by Adam Pavelec.

[Windows Notes](#) 页面有更多关于以太网桥接的说明。

说明 - Linux 下的以太网桥接，配置脚本

这些脚本将处理 Linux 下的网桥的启动和停止。在 OpenVPN 发行包的 *sample-scripts* 下可以找到。

sample-scripts/bridge-start

```
#!/bin/bash
```

```
#####
```

```
# Set up Ethernet bridge on Linux
```

```
# Requires: bridge-utils
```

```
#####
```

```
# Define Bridge Interface
```

```
# 定义网桥接口
```

```
br="br0"
```

```
# Define list of TAP interfaces to be bridged,
```

```
# for example tap="tap0 tap1 tap2".
```

定义需要桥接的 TAP 接口列表，如：

```
tap="tap0"
```

Define physical ethernet interface to be bridged

with TAP interface(s) above.

定义需要和 TAP 接口桥接的物理接口

```
eth="eth0"
```

```
eth_ip="192.168.8.4"
```

```
eth_netmask="255.255.255.0"
```

```
eth_broadcast="192.168.8.255"
```

```
for t in $tap; do
```

```
    openvpn --mktun --dev $t
```

```
done
```

```
brctl addbr $br
```

```
brctl addif $br Seth
```

```
for t in $tap; do
```

```
    brctl addif $br $t
```

```
done
```

```
for t in $tap; do
```

```
    ifconfig $t 0.0.0.0 promisc up
```

```
done
```

```
ifconfig $Seth 0.0.0.0 promisc up
```

```
ifconfig $br $eth_ip netmask $eth_netmask broadcast $eth_broadcast
```

sample-scripts/bridge-stop

```
#!/bin/bash

#####

# Tear Down Ethernet bridge on Linux

#####

# Define Bridge Interface

br="br0"

# Define list of TAP interfaces to be bridged together

tap="tap0"

ifconfig $br down

brctl delbr $br

for t in $tap; do

    openvpn --rmtun --dev $t

done
```

为什么使用 TUN 模式时, OpenVPN 的"ifconfig-pool"选项使用/30 子网(一个客户端占用 4 个私网 IP 地址)?

这是为了和 Windows 下的客户端兼容, 由于 TAP-Win32 的 TUN 模拟模式的限制, OpenVPN 为每个客户端分配一个/30 的子网, 如果连接 OpenVPN 服务器的都是非 Windows 客户端, 则可以在配置文件中使用 ifconfig-pool-linear 指令来避免这一行为。

在 OpenVPN 2.0 中, OpenVPN 服务端可以在仅有一个 tun 接口的机器上处理多个客户端。要做到这些, 可以将你在服务端上看到的点到点 PtP 连接想象为操作系统和 OpenVPN 间的一个连接。在 OpenVPN 内部还需要为每个客户端创建另外一个点到点连接。如果所有的 O/S 都支持 tun 接口上的真正的 PtP 连接, 则 OpenVPN 服务器使用一个 IP 地址而每个客户端使用另外一个 IP 地址是可以实现的。

但是, Windows 下的 TUN/TAP 驱动实现并不支持真正的 PtP 连接, 它是通过一个/30 子网来模拟的。所以, 首先在 OpenVPN 服务器和操作系统之间有一个 PtP 连接 192.168.1.1<->192.168.1.2.

然后 OpenVPN 为每一个连接的客户端分配一个/30 的子网, 第一个/30 子网(除掉服务器使用的)是:

192.168.1.4/30

192.168.1.4 -- 网络地址

192.168.1.5 -- 在 OpenVPN 服务端上的虚拟 IP 地址

192.168.1.6 -- 分配给客户端

192.168.1.7 -- 广播地址

要到达 OpenVPN 服务器后网络的其它部分, 需要给客户端添加(push)一条路由, 使流量通过 192.168.1.5 路由。因为 192.168.1.5 是 OpenVPN 服务器内部的一个虚拟 IP 地址, 用作路由的一个端点(endpoint), 所以 OpenVPN 不响应在这一地址上的 ping 包.而 192.168.1.1 是服务器操作系统上的一个真实 IP 地址, 所以它会响应 ping 包。

这的确浪费了一些 IP 地址, 但是这是使配置文件在 OpenVPN 所支持的所有操作系统中保持一致的最好方法。

TAP-Win32 驱动包括了一个给你分配 192.168.1.6 IP 地址的 DHCP 服务器, 这就是你看到 192.168.1.5 作为 DHCP 服务器地址的原因了。