基于用户名/密码认证和流量控制的OpenVPN系统配置 (1698.html)

9-13 1,039

一、OpenVPN的基本安装与配置

以Debian 5.0系统为例。主要包括OpenVPN服务器程序的安装和证书的生成。

1、下载安装OpenVPN

```
apt-get install openvpn
```

2、生成证书

复制生成证书的脚本：

```
cp -R /usr/share/doc/openvpn/examples/easy-rsa/ /etc/openvpn/
```

修改证书的变量：

```
cd /etc/openvpn/easy-rsa/2.0/
nano vars
```

编辑该文件，将最后几行的变量改成自己的，例如

```
export KEY_COUNTRY="CN"
export KEY_PROVINCE="BJ"
export KEY_CITY="Beijing"
export KEY_ORG="XX"
export KEY_EMAIL="xxx@xxx.com"
```

保存退出后，运行脚本设置变量，并清理：

```
source ./vars
./clean-all
```

之后就可以生成公钥和私钥证书了，一路回车默认值或yes即可：

```
./build-ca
./build-key-server server
./build-key client1
./build-dh
```

实际上，对于用户名/密码认证机制来说， `client1` 可以省略掉。

二、基于MySQL的用户名/密码认证实现

1、安装MySQL Server

```
apt-get install mysql-server
```

已安装的可以略过。

2、建立数据库

以管理员身份登录MySQL：

```
mysql -uroot -p
```

运行以下SQL命令：

```
-- 创建数据库
CREATE DATABASE openvpn;

-- 切换数据库
USE openvpn;

-- 创建用户，用户名openvpn，密码openvpn（可自行设定）
GRANT ALL ON openvpn.* TO 'openvpn'@'localhost' IDENTIFIED BY 'openvpn';

-- 创建用户数据表
CREATE TABLE IF NOT EXISTS `user` (
  `username` char(32) COLLATE utf8_unicode_ci NOT NULL,
  `password` char(128) COLLATE utf8_unicode_ci DEFAULT NULL,
  `active` int(10) NOT NULL DEFAULT '1',
  `creation` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `name` varchar(32) COLLATE utf8_unicode_ci NOT NULL,
  `email` char(128) COLLATE utf8_unicode_ci DEFAULT NULL,
  `note` text COLLATE utf8_unicode_ci,
  `quota_cycle` int(10) NOT NULL DEFAULT '30',
  `quota_bytes` bigint(20) NOT NULL DEFAULT '10737418240',
  `enabled` int(10) NOT NULL DEFAULT '1',
  PRIMARY KEY (`username`),
  KEY `idx_active` (`active`),
  KEY `idx_enabled` (`enabled`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

-- 创建日志数据表
CREATE TABLE IF NOT EXISTS `log` (
  `username` varchar(32) COLLATE utf8_unicode_ci NOT NULL,
  `start_time` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `end_time` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
  `trusted_ip` varchar(64) COLLATE utf8_unicode_ci DEFAULT NULL,
  `trusted_port` int(10) DEFAULT NULL,
  `protocol` varchar(16) COLLATE utf8_unicode_ci DEFAULT NULL,
  `remote_ip` varchar(64) COLLATE utf8_unicode_ci DEFAULT NULL,
  `remote_netmask` varchar(64) COLLATE utf8_unicode_ci DEFAULT NULL,
  `bytes_received` bigint(20) DEFAULT '0',
  `bytes_sent` bigint(20) DEFAULT '0',
  `status` int(10) NOT NULL DEFAULT '1',
  KEY `idx_username` (`username`),
  KEY `idx_start_time` (`start_time`),
  KEY `idx_end_time` (`end_time`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

3、安装pam_mysql模块

```
apt-get install libpam-mysql
```

4、配置OpenVPN的PAM

```
nano /etc/pam.d/openvpn
```

输入以下内容：

```
auth            sufficient      pam_mysql.so user=openvpn passwd=openvpn host=localhost db=op
envpn table=user usercolumn=username passwdcolumn=password where=active=1 sqllog=0 crypt=1

account         required        pam_mysql.so user=openvpn passwd=openvpn host=localhost db=op
envpn table=user usercolumn=username passwdcolumn=password where=active=1 sqllog=0 crypt=1
```

其中数据库、用户名、密码按照自己的实际情况设置。
crypt表示密码在数据库中加密存储的方式，含义如下：

> 0 (or “plain”)：不加密，明文存储。不推荐使用。
> 1 (or “Y”)：使用crypt(3)函数，相当于MySQL中的ENCRYPT()函数。
> 2 (or “mysql”)：使用MySQL的PASSWORD()函数。PAM可能与MySQL的函数不同，不推荐使用。
> 3 (or “md5”)：使用MD5。
> 4 (or “sha1”)：使用SHA1。

MD5我试用过有些问题。最后我使用的是1。
之后重启saslauthd：

```
/etc/init.d/saslauthd restart
```

如果出现以下提示：

```
To enable saslauthd, edit /etc/default/saslauthd and set START=yes (warning).
```

说明saslauthd未配置成自动启动，则需修改 `/etc/default/saslauthd` 文件，将 `START=no` 改为 `START=yes` ，再重启服务即可。

5、测试saslauthd是否配置成功

登入MySQL数据库：

```
mysql -uopenvpn -p
```

执行以下命令：

```
USE openvpn;
INSERT INTO user(username, password) VALUES('test', ENCRYPT('123456'));
```

退出后，运行以下命令：

```
testsaslauthd -u test -p 123456 -s openvpn
```

如果显示

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

则说明配置成功。否则，请根据 `/var/log/auth.log` 日志查找原因。

6、复制OpenVPN PAM认证模块

```
cp /usr/lib/openvpn/openvpn-auth-pam.so /etc/openvpn/
```

7、编写OpenVPN配置文件。

OpenVPN服务启动时，会扫描 `/etc/openvpn` 目录中的 `.conf` 文件，对于每个文件，启动一个daemon。本系统要实现UDP、TCP登录的同时支持，我的做法是写两份配置文件，即启动两个daemon，分别负责UDP和TCP协议。

```
nano /etc/openvpn/
```

输入以下内容

```
dev tun
proto udp
port 1194

ca /etc/openvpn/easy-rsa/2.0/keys/ca.crt
cert /etc/openvpn/easy-rsa/2.0/keys/server.crt
key /etc/openvpn/easy-rsa/2.0/keys/server.key
dh /etc/openvpn/easy-rsa/2.0/keys/dh1024.pem

user nobody
group nogroup
server 10.8.0.0 255.255.255.0

keepalive 20 120
persist-key
persist-tun

# user/pass auth from mysql
plugin ./openvpn-auth-pam.so openvpn
client-cert-not-required
username-as-common-name

client-to-client

push "redirect-gateway def1"
push "dhcp-option DNS 8.8.8.8"
push "dhcp-option DNS 8.8.4.4"

comp-lzo

max-clients 15

status status/udp.log
log-append /var/log/openvpn/udp.log
verb 3
mute 5
```

其中， `# user/pass auth from mysql` 下面的几行是该认证设置的关键所在。
同理，如果想支持TCP，建立一个 `openvpn-tcp.conf` 文件，内容跟上面相同，仅仅把

```
proto udp
server 10.8.0.0 255.255.255.0
status status/udp.log
log-append /var/log/openvpn/udp.log
```

改为

```
proto tcp
server 10.10.0.0 255.255.255.0
status status/tcp.log
log-append /var/log/openvpn/tcp.log
```

即可。
同时，为日志和状态文件建立目录：

```
mkdir /etc/openvpn/status
mkdir /var/log/openvpn
```

重启OpenVPN服务：

```
/etc/init.d/openvpn restart
```

8、设置iptables

```
nano /etc/rc.local
```

在 `exit 0` 之前添加以下几行：

```
# iptables for OpenVPN
iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -o venet0 -j SNAT --to xxx.xxx.xxx.xxx
iptables -t nat -A POSTROUTING -s 10.10.0.0/24 -o venet0 -j SNAT --to xxx.xxx.xxx.xxx
```

其中 `xxx.xxx.xxx.xxx` 是你的服务器的IP地址。
然后让其生效：

```
/etc/rc.local
```

至此，一个用户名/密码认证的OpenVPN系统就配置完成了。客户端下载使用 `/etc/openvpn/easy-rsa/2.0/keys/ca.crt` 作为证书文件，用用户名、密码认证，即可连接。一个典型的客户端配置文件如下：

```
client
dev tun
proto udp
remote xxx.com 1194
resolv-retry infinite
nobind
persist-key
persist-tun
ca ca.crt
auth-user-pass
comp-lzo
verb 3
```

其中 xxx.com 替换为服务器的域名或IP地址。
用刚才建立的test/123456用户连接一下试试吧！

9、“PAM unable to dlopen(/lib/security/pam_mysql.so): /lib/security/pam_mysql.so: undefined symbol: pam_get_item”错误的解决办法

我在Debian 5中，遇到了如上的错误（ /var/log/auth.log ），导致OpenVPN提示认证失败。解决方法如下：

```
echo "/lib/libpam.so.0" >> /etc/ld.so.preload
/etc/init.d/saslauthd restart
/etc/init.d/openvpn restart
```

三、流量控制的实现

总体思路：利用OpenVPN程序在连接、断开时的脚本钩子，将用户的使用信息记录到数据库。根据数据库中的日志，判断用户是否超过流量配额，如果超过，则将用户锁定。

1、连接、断开时的脚本

建立文件 /etc/openvpn/connect.sh ，内容如下：

```
#!/bin/bash

DB='openvpn'
DBADMIN='openvpn'
DBPASSWD='openvpn'

mysql -u$DBADMIN -p$DBPASSWD -e "INSERT INTO log(username,start_time,trusted_ip,trusted_port,
protocol,remote_ip,remote_netmask,status) VALUES('$common_name',now(),'$trusted_ip',$trusted_
port,'$proto_1','$ifconfig_pool_remote_ip','$route_netmask_1',1)" $DB
```

建立文件 /etc/openvpn/disconnect.sh ，内容如下：

```
#!/bin/bash

DB='openvpn'
DBADMIN='openvpn'
DBPASSWD='openvpn'

mysql -u$DBADMIN -p$DBPASSWD -e "UPDATE log SET end_time=now(),bytes_received=$bytes_receive
d,bytes_sent=$bytes_sent,status=0 WHERE trusted_ip='$trusted_ip' AND trusted_port=$trusted_po
rt AND remote_ip='$ifconfig_pool_remote_ip' AND username='$common_name' AND status=1" $DB

mysql -u$DBADMIN -p$DBPASSWD -e "UPDATE user SET active=0 WHERE user.username IN (SELECT user
name FROM (SELECT log.username AS username, quota_bytes FROM user, log WHERE log.username='$c
ommon_name' AND log.username=user.username AND log.status=0 AND TO_DAYS(NOW())-TO_DAYS(start_
time)< =quota_cycle GROUP BY log.username HAVING SUM(bytes_received)+SUM(bytes_sent)>=quota_b
ytes) AS u);" $DB
```

将文件改为可执行属性：

```
chmod +x /etc/openvpn/connect.sh
chmod +x /etc/openvpn/disconnect.sh
```

修改OpenVPN配置文件 `openvpn-udp.conf` 、 `openvpn-tcp.conf` ，添加以下几行：

```
# record in database
script-security 2
client-connect ./connect.sh
client-disconnect ./disconnect.sh
```

其主要作用是：在用户连接时，在数据库 `log` 表中新建一条记录，记录用户的IP地址、端口号、连接时间等信息。在用户断开连接时，更新刚才添加的记录，记下用户的断开连接时间、发送数据量、接收数据量等。然后，对用户的流量进行判断，若超过配额，则将用户锁定（ `active=0` ）。
`user` 表中的 `quota_cycle` 是用户的流量计算周期， `quota_bytes` 是用户每个周期内最多允许的流量。
`connect.sh` 和 `disconnect.sh` 脚本文件中调用了OpenVPN的环境变量。OpenVPN在执行脚本时，自动各种设置了环境变量，供脚本使用。具体的环境变量可以查看这里。

2、使用cron每天对用户进行检查

以上操作在用户超过流量时自动将用户锁定。每天还应该执行一次检查，把已经恢复流量的用户解锁。可以通过cron实现此功能。建立文件 `/etc/cron.daily/openvpn` ，内容如下：

```
#!/bin/bash

DB='openvpn'
DBADMIN='openvpn'
DBPASSWD='openvpn'

mysql -u$DBADMIN -p$DBPASSWD -e "UPDATE user SET active=1" $DB

mysql -u$DBADMIN -p$DBPASSWD -e "UPDATE user SET active=0 WHERE user.username IN (SELECT user
name FROM (SELECT log.username AS username, quota_bytes FROM user, log WHERE log.username=use
r.username AND log.status=0 AND TO_DAYS(NOW())-TO_DAYS(start_time)< =quota_cycle GROUP BY lo
g.username HAVING SUM(bytes_received)+SUM(bytes_sent)>=quota_bytes) AS u);" $DB

mysql -u$DBADMIN -p$DBPASSWD -e "UPDATE user SET active=0 WHERE enabled=0" $DB
```

其思路是：先默认将所有用户解锁，然后将超过流量的用户锁定。同时，管理员可以通过 `user` 表中的 `enabled` 字段手工禁用用户。
然后给文件可执行权限：

```
chmod +x /etc/cron.daily/openvpn
```

3、修改saslauthd的缓存时间

saslauthd默认有一段较长的缓存时间，在用户通过认证后的一段时间里，可以再次通过认证而不需要重新查询数据库。这样不利于实现对超流量用户的立即锁定。
saslauthd启动时有一个 `-t` 参数，可以设置其超时时间。修改 `/etc/default/saslauthd` 文件，将

```
OPTIONS="..."
```

一行，引号最后添上 `-t 60` ，可将缓存时间设置为60秒。当然，也可直接将其设置为0，即不缓存。
重启saslauthd服务和OpenVPN，使设置生效：

```
/etc/init.d/saslauthd restart
/etc/init.d/openvpn restart
```

(vpn.jpg)

上一篇 (1675.html)

下一篇 (1742.html)

1. Pingback: payday loans ottawa
2. Pingback: direct payday loans prince george lender
3. Pingback: drugrehabcentershotline.com drug rehabilitation
4. Pingback: Blue Coaster33

搜索...