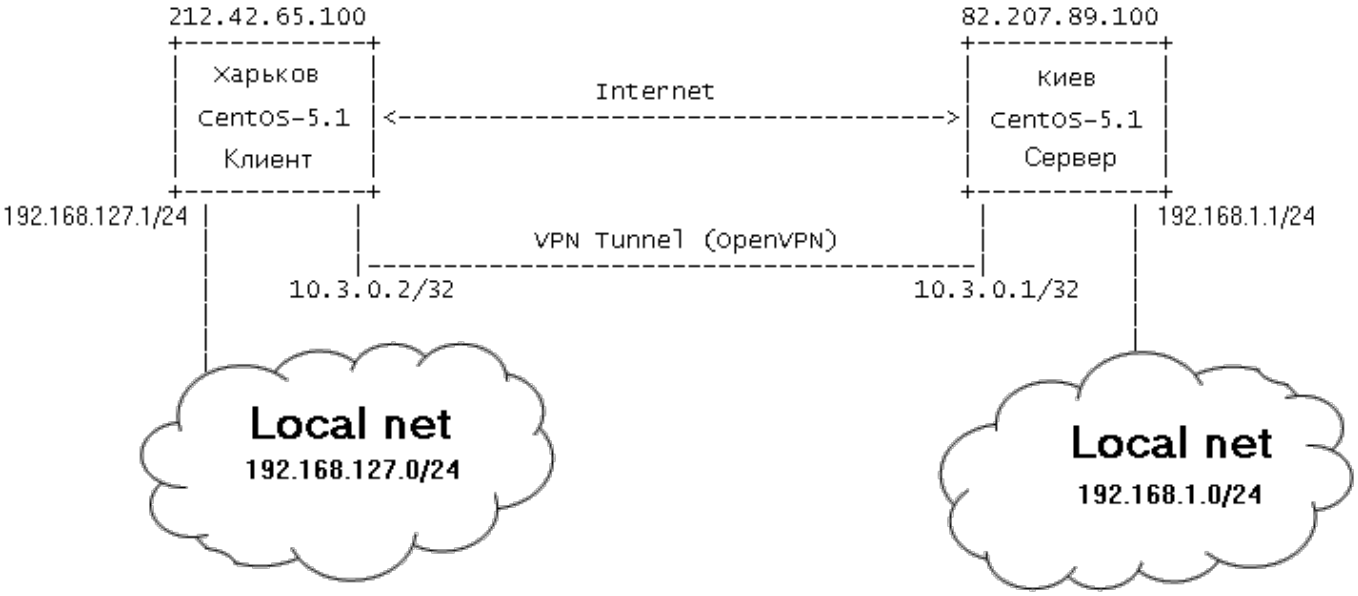
OpenVPN: создание туннеля point to point

Введение

Передо мной была поставлена задача - обеспечить безопасную работу пользователей, которые работают на удаленном сервере терминалов с помощью стандартного клиента rdp - «Подключение к удаленному рабочему столу». Под безопасностью будем понимать шифрование передаваемых данных между сервером и клиентом. Так же неплохо было бы сжимать данные, для экономии трафика и разгрузки канала.

Для лучшего понимания задачи, ниже прилагается схема данной задачи.



Нам надо обеспечить подключение клиентов из Харьковского филиала к серверу терминалов, расположенному в Киевском офисе. Сервер терминалов имеет статический адрес - 192.168.1.2/24.

На linux системах, пожалуй, самым распространенным ПО для реализации таких задач является OpenVPN.

OpenVPN - это полноценный SSL VPN, который реализует сетевые расширения безопасности OSI уровня 2 и 3, используя индустриальный стандарт - SSL/TLS протокол. Поддерживает множество методов аутентификации клиентов: основанных на сертификатах, smart картах, логине/пароле.

OpenVPN мощный и очень гибкий VPN демон. Поддерживает SSL/TLS безопасность, ethernet bridging, TCP или UDP туннельный транспорт через прокси или NAT, поддерживает динамические IP адреса и DHCP, масштабируемость до сотни или тысяч пользователей, портирован на все основные платформы и ОС. Ниже приведен список поддерживаемых платформ:

- Linux 2.2+
- Solaris
- OpenBSD 3.0+
- Mac OS X Darwin
- FreeBSD
- NetBSD
- Windows (Win 2K+)

Итак, у нас имеются следующие системы:

```
# uname -r
2.6.18-53.1.4.el5

# cat /etc/redhat-release
CentOS release 5 (Final)
```

На обоих серверах, как в Киеве, так и в Харькове, установлены одинаковые ОС - CentOS-5.1. В принципе ОС не имеет особого значения, но крайне желательно, чтобы на концах туннеля использовались одинаковые версии openvpn.

Установка openvpn

К сожалению, данный пакет отсутствует в официальном репозитарии RHEL/CentOS. Так что вам придется либо устанавливать с других репозитариев, либо собрать rpm пакеты из src.rpm

Если вы хотите использовать сжатие данных, то перед началом сборки openvpn надо будет собрать и установить пакет lzo

```
# rpm -ivh http://www.sys-adm.org.ua/srpms/lzo-2.02-2.src.rpm
# cd /usr/src/redhat/SPECS
# rpmbuild -ba --target=i686 lzo.spec
Building target platforms: i686
Building for target i686
Executing(%prep): /bin/sh -e /var/tmp/rpm-tmp.9651
+ umask 022
...
...
...
Wrote: /usr/src/redhat/SRPMS/lzo-2.02-2.src.rpm
Wrote: /usr/src/redhat/RPMS/i686/lzo-2.02-2.i686.rpm
Wrote: /usr/src/redhat/RPMS/i686/lzo-devel-2.02-2.i686.rpm
Executing(%clean): /bin/sh -e /var/tmp/rpm-tmp.61481
+ umask 022
+ cd /usr/src/redhat/BUILD
+ cd lzo-2.02
+ rm -rf /var/tmp/lzo-2.02-2-root-root
```

```
+ exit 0
```

После успешной сборки, устанавливаем пакет

```
# cd /usr/src/redhat/RPMS/i686/
# rpm -ivh lzo-2.02-2.i686.rpm lzo-devel-2.02-2.i686.rpm
Preparing...                ########################################### 
[100%]
   1:lzo                    ########################################### [
50%]
   2:lzo-devel              ########################################### 
[100%]
```

Теперь у нас все готово и мы можем собирать сам openvpn.

```
# rpm -ivh http://www.sys-adm.org.ua/srpms/openvpn-2.0.9-1.src.rpm
# cd /usr/src/redhat/SPECS
# rpmbuild -ba --target=i686 openvpn.spec
Building target platforms: i686
Building for target i686
Executing(%prep): /bin/sh -e /var/tmp/rpm-tmp.95164
+ umask 022
...
...
...
Wrote: /usr/src/redhat/SRPMS/openvpn-2.0.9-1.src.rpm
Wrote: /usr/src/redhat/RPMS/i686/openvpn-2.0.9-1.i686.rpm
Executing(%clean): /bin/sh -e /var/tmp/rpm-tmp.2390
+ umask 022
+ cd /usr/src/redhat/BUILD
+ cd openvpn-2.0.9
+ '[' /var/tmp/openvpn-root '!=' / ']'
+ rm -rf /var/tmp/openvpn-root
+ exit 0
```

После успешной сборки, устанавливаем пакет

```
# cd /usr/src/redhat/RPMS/i686/
# rpm -ivh openvpn-2.0.9-1.i686.rpm
Preparing...                ########################################### 
[100%]
   1:openvpn                ########################################### 
[100%]
```

На этом установку необходимого ПО можно считать завершенной, теперь переходим непосредственно к настройке.

Создание и инициализация PKI

Так как для шифрования туннеля мы будем использовать TLS (SSL/TLS + сертификаты для аутентификации и обмена ключей) + pre-shared static key, то сначала создадим все необходимые нам ключи и сертификаты.

Как это сделать вы можете прочитать в моей статье SSL Howto . Так же в составе самого openvpn идет набор скриптов существенно облегчающих создание всех необходимых ключей и сертификатов. Вот именно этими скриптами мы и воспользуемся.

Прежде всего, инициализируем нашу PKI (public key infrastructure)

```
# cd /usr/share/doc/openvpn-2.0.9/
# cp -R easy-rsa/ /etc/openvpn/
```

В документации к openvpn рекомендуют делать именно полную копию папки easy-rsa в другое место, например **/etc/openvpn.** И производить все изменения в этой локальной копии. Чтобы при обновлении версии openvpn ваши изменения не пропали.

Перед началом работы определим значение некоторых переменных, облегчающих процесс создания сертификатов. Для этого правим файл **/etc/openvpn/easy-rsa/vars.** В данном файле хранятся значения по умолчанию.

```
# /etc/openvpn/easy-rsa/vars | grep -v ^# | grep -v ^
export D=`pwd`
export KEY_CONFIG=$D/openssl.cnf
export KEY_DIR=$D/keys
echo NOTE: when you run ./clean-all, I will be doing a rm -rf on $KEY_DIR
export KEY_SIZE=2048
export KEY_COUNTRY=UA
export KEY_PROVINCE=Ukraine
export KEY_CITY=Kharkov
export KEY_ORG="SysAdm"
export KEY_EMAIL="hostmaster@sys-adm.org.ua"
```

После этого инициализируем наши переменные

```
# cd /etc/openvpn/easy-rsa/
# . ./vars
# ./clean-all
```

После того, как мы произвели первоначальную подготовку, переходим непосредственно к созданию самих сертификатов. Вначале нам надо создать корневой CA (root CA) с помощью которого мы будем подписывать все наши сертификаты.

```
# ./build-ca
Generating a 2048 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) [UA]:
State or Province Name (full name) [Ukraine]:
Locality Name (eg, city) [Kharkov]:
Organization Name (eg, company) [SysAdm]:
Organizational Unit Name (eg, section) []:SysAdm Security Center
Common Name (eg, your name or your server's hostname) []:Root CA
Email Address [hostmaster@sys-adm.org.ua]:
```

После того, как мы создали root CA, теперь генерируем ключ и сертификат для сервера.

```
# ./build-key-server gw1-kv.sys-adm.org.ua
Generating a 2048 bit RSA private key
.........................++++++
.....................++++++
writing new private key to 'gw1-kv.sys-adm.org.ua.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) [UA]:
State or Province Name (full name) [Ukraine]:
Locality Name (eg, city) [Kharkov]:Kiev
Organization Name (eg, company) [SysAdm]:
Organizational Unit Name (eg, section) []:Kiev VPN Server
Common Name (eg, your name or your server's hostname) []:gw1-kv.sys-
adm.org.ua
Email Address [hostmaster@sys-adm.org.ua]:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:1234567
An optional company name []:www.sys-adm.org.ua
Using configuration from /etc/openvpn/easy-rsa/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'UA'
stateOrProvinceName   :PRINTABLE:'Ukraine'
```

```
localityName          :PRINTABLE:'Kiev'
organizationName      :PRINTABLE:'SysAdm'
organizationalUnitName:PRINTABLE:'Kiev VPN Server'
commonName            :PRINTABLE:'gw1-kv.sys-adm.org.ua'
emailAddress          :IA5STRING:'hostmaster@sys-adm.org.ua'
Certificate is to be certified until Mar  5 20:04:49 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
```

Теперь генерируем ключ и сертификат для второй точки нашего туннеля.

```
# ./build-key gw1-kh.sys-adm.org.ua
Generating a 2048 bit RSA private key
.....................++++++
......++++++
writing new private key to 'gw1-kh.sys-adm.org.ua.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) [UA]:
State or Province Name (full name) [Ukraine]:
Locality Name (eg, city) [Kharkov]:
Organization Name (eg, company) [SysAdm]:
Organizational Unit Name (eg, section) []:Kharkov VPN Server
Common Name (eg, your name or your server's hostname) []:gw1-kh.sys-
adm.org.ua
Email Address [hostmaster@sys-adm.org.ua]:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:1234567
An optional company name []:www.sys-adm.org.ua
Using configuration from /etc/openvpn/easy-rsa/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'UA'
stateOrProvinceName   :PRINTABLE:'Ukraine'
localityName          :PRINTABLE:'Kharkov'
organizationName      :PRINTABLE:'SysAdm'
organizationalUnitName:PRINTABLE:'Kharkov VPN Server'
commonName            :PRINTABLE:'gw1-kh.sys-adm.org.ua'
emailAddress          :IA5STRING:'hostmaster@sys-adm.org.ua'
```

```
Certificate is to be certified until Mar  5 20:08:15 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
```

Отличие скриптов **build-key <name>** и **build-key-server <name>** заключается в том, что при генерации сертификата с помощью скрипта build-key-server мы указываем дополнительную секцию в openssl.cnf - **[server].** В данной секции мы добавляем в наш будущий сертификат поле **nsCertType=server.** Данный метод позволяет бороться с т.н. "Man-in-the-Middle" атаками.

Если вы создавали сертификаты, используя статью SSL Howto, то перед созданием сертификата для сервера вам необходимо будет отредактировать ваш openssl.cnf и добавить в самый конец следующие строки

```
[ server ]
basicConstraints=CA:FALSE
nsCertType = server
nsComment = "Kiev VPN Server Certificate"
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
```

А при генерации самого сертификата использовать следующую команду:

```
# openssl ca -out gw1-kv.sys-adm.org.ua.crt -config ./openssl.cnf -infiles
gw1-kv.sys-adm.org.ua.csr -extensions server
```

Теперь генерируем т.н. параметры Diffie Hellman'a , которые будут использоваться только на стороне сервера.

```
# ./build-dh
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
...
...
...
```

И последнее, что нам осталось сделать - это сгенерировать static pre-shared key.

```
# openvpn --genkey --secret secret.key
```

Для лучшего понимания ниже я привожу таблицу, в которой указано где используются ключи и сертификаты, которые мы создали

Имя файла	Кем используется	Назначение	Защита
ca.crt	сервер и клиент	Root CA certificate	Нет

Имя файла	Кем используется	Назначение	Защита
ca.key	только на машине, производящей подпись сертификатов	Root CA key	Да
dh1024.pem	только сервер	Параметры Diffie Hellman'a	Нет
secret.key	сервер и клиент	Shared secret key	Нет
gw1-kv.sys-adm.org.ua.crt	только сервер	Server Certificate	Нет
gw1-kv.sys-adm.org.ua.key	только сервер	Server Key	Да
gw1-kh.sys-adm.org.ua.crt	только клиент	Client Certificate	Нет
gw1-kh.sys-adm.org.ua.key	только клиент	Client Key	Да

Настройка openvpn

При запуске демона openvpn он считывает файлы с расширением conf из папки **/etc/openvpn.** Имя файла не имеет значение. Поэтому мы создаем собственный конфигурационный файл со следующим содержимым на стороне сервера:

```
# cat /etc/openvpn/kiev-kharkov.conf | grep -v ^
dev tun
local 82.207.89.100
ifconfig 10.3.0.1 10.3.0.2
port 1194
proto udp
user nobody
group nobody
comp-lzo
ping 15
ping-restart 45
persist-key
persist-tun
log /var/log/openvpn.log
status /var/log/openvpn-status.log
verb 3
tls-server
ca ca.crt
cert gw1-kv.sys-adm.org.ua.crt
key gw1-kv.sys-adm.org.ua.key
dh dh1024.pem
tls-auth secret.key 0
```

Также копируем следующие файлы в папку **/etc/openvpn** и выставляем необходимые права.

- ca.crt,
- secret.key,
- dh1024.pem,
- gw1-kv.sys-adm.org.ua.crt,
- gw1-kv.sys-adm.org.ua.key

```
# cd /etc/openvpn/
# chmod 600 ca.crt secret.key dh1024.pem
# chmod 600 gw1-kv.sys-adm.org.ua.crt
# chmod 600 gw1-kv.sys-adm.org.ua.key
```

Производим аналогичную процедуру и на стороне клиента. Создаем собственный конфигурационный файл со следующим содержимым:

```
# cat /etc/openvpn/kharkov-kiev.conf | grep -v ^
dev tun
remote 82.207.89.100
ifconfig 10.3.0.2 10.3.0.1
port 1194
proto udp
user nobody
group nobody
comp-lzo
ping 15
ping-restart 45
persist-tun
persist-key
log /var/log/openvpn.log
status /var/log/openvpn-status.log
verb 3
tls-client
ca ca.crt
cert gw1-kh.sys-adm.org.ua.crt
key gw1-kh.sys-adm.org.ua.key
tls-auth secret.key 1
ns-cert-type server
```

Копируем следующие файлы в папку /etc/openvpn и выставляем необходимые права.

- ca.crt,
- secret.key,
- gw1-kh.sys-adm.org.ua.crt,
- gw1-kh.sys-adm.org.ua.key

```
# cd /etc/openvpn/
# chmod 600 ca.crt secret.key
# chmod 600 gw1-kh.sys-adm.org.ua.crt
# chmod 600 gw1-kh.sys-adm.org.ua.key
```

На этом настройку openvpn можно считать завершенной. Осталось только запустить сам демон на обоих серверах и настроить автоматический запуск при старте системы.

Выполняем следующие команды на стороне сервера

```
[root@gw1-kv openvpn]# chkconfig --level 35 openvpn on
[root@gw1-kv openvpn]# service openvpn start
Starting openvpn:                                          [  OK  ]
```

И на стороне клиента

```
[root@gw1-kh openvpn]# chkconfig --level 35 openvpn on
[root@gw1-kh openvpn]# service openvpn start
Starting openvpn:                                          [  OK  ]
```

Тестирование

Если у вас все правильно настроено, то после запуска демонов на обоих концах тоннеля в log-файле сервера должны быть примерно такие строки

```
# cat openvpn.log
Fri Mar 7 23:34:08 2008 OpenVPN 2.0.9 i686-redhat-linux [SSL] [LZO] [EPOLL]
built on Feb 29 2008
Fri Mar 7 23:34:08 2008 Diffie-Hellman initialized with 1024 bit key
Fri Mar 7 23:34:08 2008 Control Channel Authentication: using 'secret.key'
as a OpenVPN static key file
Fri Mar 7 23:34:08 2008 Outgoing Control Channel Authentication: Using 160
bit message hash 'SHA1' for HMAC authentication
Fri Mar 7 23:34:08 2008 Incoming Control Channel Authentication: Using 160
bit message hash 'SHA1' for HMAC authentication
Fri Mar 7 23:34:08 2008 LZO compression initialized
Fri Mar 7 23:34:08 2008 Control Channel MTU parms [ L:1542 D:166 EF:66 EB:0
ET:0 EL:0 ]
Fri Mar 7 23:34:08 2008 TUN/TAP device tun0 opened
Fri Mar 7 23:34:08 2008 /sbin/ifconfig tun0 10.3.0.1 pointopoint 10.3.0.2
mtu 1500
Fri Mar 7 23:34:08 2008 Data Channel MTU parms [ L:1542 D:1450 EF:42 EB:135
ET:0 EL:0 AF:3/1 ]
Fri Mar 7 23:34:08 2008 Local Options hash (VER=V4): '38909f4f'
Fri Mar 7 23:34:08 2008 Expected Remote Options hash (VER=V4): '6bc625ed'
Fri Mar 7 23:34:08 2008 GID set to nobody
Fri Mar 7 23:34:08 2008 UID set to nobody
Fri Mar 7 23:34:08 2008 UDPv4 link local (bound): 82.207.89.100:1194
Fri Mar 7 23:34:08 2008 UDPv4 link remote: [undef]
Fri Mar 7 23:34:12 2008 TLS: Initial packet from 212.42.65.100:1194,
sid=b84137e2 c3e555c6
Fri Mar 7 23:34:12 2008 VERIFY OK: depth=1,
/C=UA/ST=Ukraine/L=Kharkov/O=SysAdm/OU=SysAdm_Security_Center/CN=Root_CA/ema
ilAddress=hostmaster@sys-adm.org.ua
Fri Mar 7 23:34:12 2008 VERIFY OK: depth=0,
/C=UA/ST=Ukraine/O=SysAdm/OU=Kharkov_VPN_Server/CN=gw1-kh.sys-
adm.org.ua/emailAddress=hostmaster@sys-adm.org.ua
Fri Mar 7 23:34:12 2008 Data Channel Encrypt: Cipher 'BF-CBC' initialized
```

```
with 128 bit key
Fri Mar 7 23:34:12 2008 Data Channel Encrypt: Using 160 bit message hash
'SHA1' for HMAC authentication
Fri Mar 7 23:34:12 2008 Data Channel Decrypt: Cipher 'BF-CBC' initialized
with 128 bit key
Fri Mar 7 23:34:12 2008 Data Channel Decrypt: Using 160 bit message hash
'SHA1' for HMAC authentication
Fri Mar 7 23:34:12 2008 Control Channel: TLSv1, cipher TLSv1/SSLv3 DHE-RSA-
AES256-SHA, 2048 bit RSA
Fri Mar 7 23:34:12 2008 [gw1-kh.sys-adm.org.ua] Peer Connection Initiated
with 212.42.65.100:1194
Fri Mar 7 23:34:14 2008 Initialization Sequence Completed
```

При этом в системе должен был появиться новый интерфейс - tun0

```
# ifconfig tun0
tun0      Link encap:UNSPEC  HWaddr
00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          inet addr:10.3.0.1  P-t-P:10.3.0.2  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

А на стороне клиента следующие записи в log-файле

```
# cat openvpn.log
Sat Mar 8 01:49:27 2008 OpenVPN 2.0.9 i686-redhat-linux [SSL] [LZO] [EPOLL]
built on Mar 7 2008
Sat Mar 8 01:49:27 2008 Control Channel Authentication: using 'secret.key'
as a OpenVPN static key file
Sat Mar 8 01:49:27 2008 Outgoing Control Channel Authentication: Using 160
bit message hash 'SHA1' for HMAC authentication
Sat Mar 8 01:49:27 2008 Incoming Control Channel Authentication: Using 160
bit message hash 'SHA1' for HMAC authentication
Sat Mar 8 01:49:27 2008 LZO compression initialized
Sat Mar 8 01:49:27 2008 Control Channel MTU parms [ L:1542 D:166 EF:66 EB:0
ET:0 EL:0 ]
Sat Mar 8 01:49:27 2008 TUN/TAP device tun0 opened
Sat Mar 8 01:49:27 2008 /sbin/ifconfig tun0 10.3.0.2 pointopoint 10.3.0.1
mtu 1500
Sat Mar 8 01:49:27 2008 Data Channel MTU parms [ L:1542 D:1450 EF:42 EB:135
ET:0 EL:0 AF:3/1 ]
Sat Mar 8 01:49:27 2008 Local Options hash (VER=V4): '6bc625ed'
Sat Mar 8 01:49:27 2008 Expected Remote Options hash (VER=V4): '38909f4f'
Sat Mar 8 01:49:27 2008 GID set to nobody
Sat Mar 8 01:49:27 2008 UID set to nobody
Sat Mar 8 01:49:27 2008 UDPv4 link local (bound): [undef]:1194
```

```
Sat Mar 8 01:49:27 2008 UDPv4 link remote: 82.207.89.100:1194
Sat Mar 8 01:49:27 2008 TLS: Initial packet from 82.207.89.100:1194,
sid=cf756109 bb37aa09
Sat Mar 8 01:49:27 2008 VERIFY OK: depth=1,
/C=UA/ST=Ukraine/L=Kharkov/O=SysAdm/OU=SysAdm_Security_Center/CN=Root_CA/ema
ilAddress=hostmaster@sys-adm.org.ua
Sat Mar 8 01:49:27 2008 VERIFY OK: nsCertType=SERVER
Sat Mar 8 01:49:27 2008 VERIFY OK: depth=0,
/C=UA/ST=Ukraine/O=SysAdm/OU=Kiev_VPN_Server/CN=gw1-kv.sys-
adm.org.ua/emailAddress=hostmaster@sys-adm.org.ua
Sat Mar 8 01:49:27 2008 Data Channel Encrypt: Cipher 'BF-CBC' initialized
with 128 bit key
Sat Mar 8 01:49:27 2008 Data Channel Encrypt: Using 160 bit message hash
'SHA1' for HMAC authentication
Sat Mar 8 01:49:27 2008 Data Channel Decrypt: Cipher 'BF-CBC' initialized
with 128 bit key
Sat Mar 8 01:49:27 2008 Data Channel Decrypt: Using 160 bit message hash
'SHA1' for HMAC authentication
Sat Mar 8 01:49:27 2008 Control Channel: TLSv1, cipher TLSv1/SSLv3 DHE-RSA-
AES256-SHA, 2048 bit RSA
Sat Mar 8 01:49:27 2008 [gw1-kv.sys-adm.org.ua] Peer Connection Initiated
with 82.207.89.100:1194
Sat Mar 8 01:49:28 2008 Initialization Sequence Completed
```

При этом в системе должен был появиться новый интерфейс - tun0

```
# ifconfig tun0
tun0      Link encap:UNSPEC  HWaddr
00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          inet addr:10.3.0.2  P-t-P:10.3.0.1  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

На данный момент сервер и клиент должны «видеть» друг друга. Для проверки воспользуемся командой ping

```
[root@gw1-kh log]# ping -c 4 10.3.0.1
PING 10.3.0.1 (10.3.0.1) 56(84) bytes of data.
64 bytes from 10.3.0.1: icmp_seq=1 ttl=64 time=1.86 ms
64 bytes from 10.3.0.1: icmp_seq=2 ttl=64 time=1.06 ms
64 bytes from 10.3.0.1: icmp_seq=3 ttl=64 time=2.07 ms
64 bytes from 10.3.0.1: icmp_seq=4 ttl=64 time=2.85 ms

--- 10.3.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 1.062/1.964/2.854/0.639 ms
```

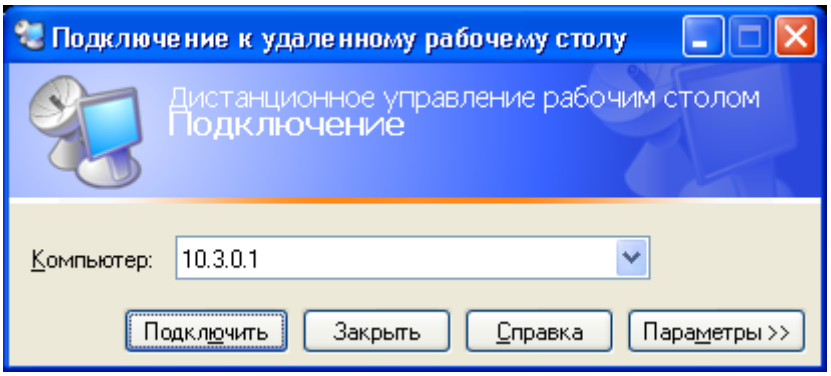
```
[root@gw1-kv log]# ping -c4 10.3.0.2
PING 10.3.0.2 (10.3.0.2) 56(84) bytes of data.
64 bytes from 10.3.0.2: icmp_seq=1 ttl=64 time=4.18 ms
64 bytes from 10.3.0.2: icmp_seq=2 ttl=64 time=1.12 ms
64 bytes from 10.3.0.2: icmp_seq=3 ttl=64 time=2.09 ms
64 bytes from 10.3.0.2: icmp_seq=4 ttl=64 time=2.11 ms

--- 10.3.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 1.128/2.378/4.184/1.116 ms
```

Для того, чтобы работал доступ к серверу терминалов необходимо внести некоторые изменения в firewall. Далее я приведу лишь минимальный набор команд, необходимый для проверки работы нашего туннеля. Будем считать, что форвардинг включен на обоих серверах и политика по умолчанию во всех цепочках - ACCEPT

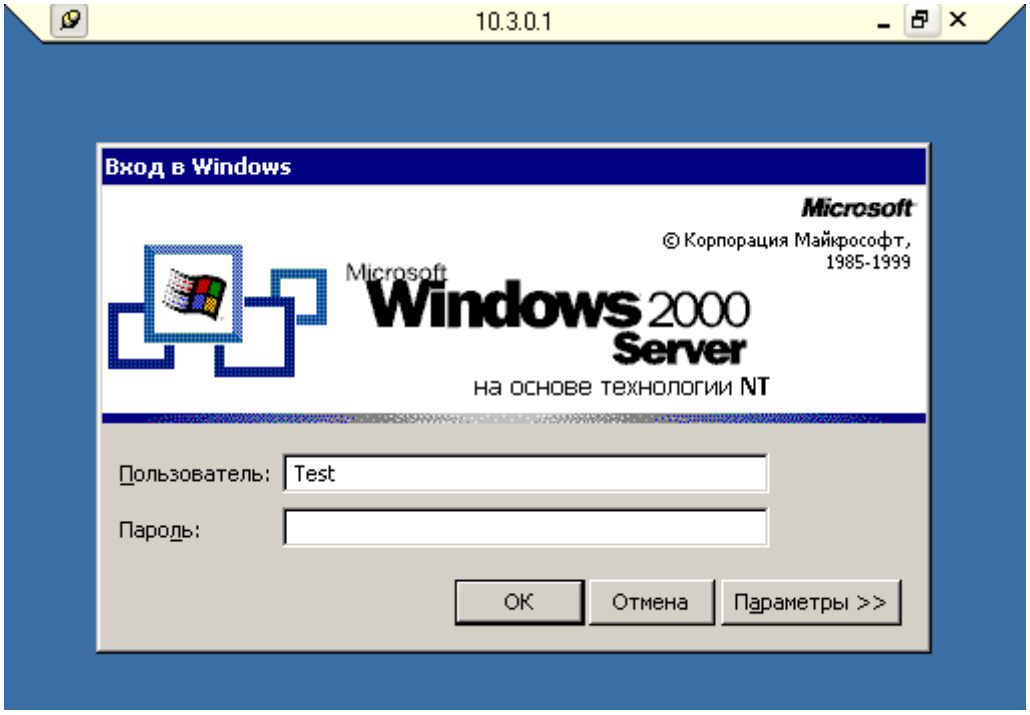
На стороне клиента выполняем следующую команду

```
# iptables -t nat -A POSTROUTING -s 192.168.127.0/255.255.255.0 -o tun0 -j
SNAT --to-source 10.3.0.2
```

Теперь запускаем клиента подключения к удаленному рабочему столу на любой машине в Харьковском офисе и в строке адреса набираем 10.3.0.1.



Если вы все правильно настроили, то вы должны соединиться с сервером терминалов.



Ради интереса можно посмотреть с помощью tcpdump, что происходит в этот момент на интерфейсе tun0 на сервере.

```
# tcpdump -npi tun0 port 3389
tcpdump: WARNING: arptype 65534 not supported by libpcap - falling back to
cooked socket
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on tun0, link-type LINUX_SLL (Linux cooked), capture size 96 bytes
23:51:14.217974 IP 10.3.0.2.netinfo-local > 10.3.0.1.ms-wbt-server: S
3790039829:3790039829(0) win 64240
23:51:14.229282 IP 10.3.0.2.netinfo-local > 10.3.0.1.ms-wbt-server: . ack 1
win 64296
23:51:14.231734 IP 10.3.0.2.netinfo-local > 10.3.0.1.ms-wbt-server: P
1:35(34) ack 1 win 64296
23:51:14.244642 IP 10.3.0.1.ms-wbt-server > 10.3.0.2.netinfo-local: P
1:12(11) ack 35 win 64206
23:51:14.277071 IP 10.3.0.2.netinfo-local > 10.3.0.1.ms-wbt-server: P
35:447(412) ack 12 win 64285
...
...
...
23:51:20.890252 IP 10.3.0.2.netinfo-local > 10.3.0.1.ms-wbt-server: . ack
14376 win 63678
23:51:20.960846 IP 10.3.0.1.ms-wbt-server > 10.3.0.2.netinfo-local: P
14376:14449(73) ack 4249 win 63402
23:51:21.018435 IP 10.3.0.1.ms-wbt-server > 10.3.0.2.netinfo-local: P
14449:14469(20) ack 4249 win 63402
23:51:21.027522 IP 10.3.0.2.netinfo-local > 10.3.0.1.ms-wbt-server: . ack
14469 win 63585
23:51:21.199395 IP 10.3.0.1.ms-wbt-server > 10.3.0.2.netinfo-local: P
14469:14569(100) ack 4249 win 63402
23:51:21.283439 IP 10.3.0.2.netinfo-local > 10.3.0.1.ms-wbt-server: . ack
```

```
14569 win 63485
23:51:22.250287 IP 10.3.0.1.ms-wbt-server > 10.3.0.2.netinfo-local: P
14569:14596(27) ack 4249 win 63402
```

Но, как правило, NAT в VPN туннелях не используется. Мы можем сделать так, что подсети 192.168.1.0/24 и 192.168.127.0/24 будут видеть друг друга «напрямую». Для этого достаточно на каждом из шлюзов прописать маршрут в соответствующую подсеть.

Прописываем маршрут в подсеть 192.168.127.0/24 на стороне сервера

```
[root@gw1-kv log]# route add -net 192.168.127.0/24 gw 10.3.0.2
[root@gw1-kv log]# route -n | grep tun0
10.3.0.2        0.0.0.0         255.255.255.255 UH    0      0        0 tun0
192.168.127.0   10.3.0.2        255.255.255.0   UG    0      0        0 tun0
```

А на стороне клиента прописываем маршрут в подсеть 192.168.1.0/24

```
[root@gw1-kh log]# route add -net 192.168.1.0/24 gw 10.3.0.1
[root@gw1-kh log]# route -n | grep tun0
10.3.0.1        0.0.0.0         255.255.255.255 UH    0      0        0 tun0
192.168.1.0     10.3.0.1        255.255.255.0   UG    0      0        0 tun0
```

Теперь производим проверку «видимости» подсети 192.168.127.0/24 с помощью команды ping. Для этого на стороне сервера выполняем следующую команду.

```
[root@gw1-kv log]# ping -c 4 192.168.127.2
PING 192.168.127.2 (192.168.127.2) 56(84) bytes of data.
64 bytes from 192.168.127.2: icmp_seq=1 ttl=127 time=2.43 ms
64 bytes from 192.168.127.2: icmp_seq=2 ttl=127 time=2.06 ms
64 bytes from 192.168.127.2: icmp_seq=3 ttl=127 time=1.81 ms
64 bytes from 192.168.127.2: icmp_seq=4 ttl=127 time=2.65 ms

--- 192.168.127.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 1.810/2.240/2.653/0.326 ms
```

При этом на стороне клиента, если посмотреть с помощью tcpdump, должно быть примерно следующее

```
[root@gw1-kh log]# tcpdump -npi tun0
tcpdump: WARNING: arptype 65534 not supported by libpcap - falling back to
cooked socket
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on tun0, link-type LINUX_SLL (Linux cooked), capture size 96 bytes
15:01:16.279286 IP 10.3.0.1 > 192.168.127.2: ICMP echo request, id 55305,
seq 1, length 64
```

```
15:01:16.293389 IP 192.168.127.2 > 10.3.0.1: ICMP echo reply, id 55305, seq
1, length 64
15:01:17.267281 IP 10.3.0.1 > 192.168.127.2: ICMP echo request, id 55305,
seq 2, length 64
15:01:17.268145 IP 192.168.127.2 > 10.3.0.1: ICMP echo reply, id 55305, seq
2, length 64
15:01:18.339259 IP 10.3.0.1 > 192.168.127.2: ICMP echo request, id 55305,
seq 3, length 64
15:01:18.341297 IP 192.168.127.2 > 10.3.0.1: ICMP echo reply, id 55305, seq
3, length 64
15:01:20.002665 IP 10.3.0.1 > 192.168.127.2: ICMP echo request, id 55305,
seq 4, length 64
15:01:20.003493 IP 192.168.127.2 > 10.3.0.1: ICMP echo reply, id 55305, seq
4, length 64
```

Ну и для чистоты совести производим проверку «видимости» подсети 192.168.1.0/24 с помощью команды ping. Для этого на стороне клиента выполняем следующую команду.

```
[root@gw1-kh log]# ping -c 4 192.168.1.2
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
64 bytes from 192.168.1.2: icmp_seq=1 ttl=127 time=4.70 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=127 time=1.93 ms
64 bytes from 192.168.1.2: icmp_seq=3 ttl=127 time=1.75 ms
64 bytes from 192.168.1.2: icmp_seq=4 ttl=127 time=2.65 ms

--- 192.168.1.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 1.752/2.761/4.705/1.173 ms
```

При этом на стороне сервера, если посмотреть с помощью tcpdump, должно быть примерно следующее

```
[root@gw1-kv log]# tcpdump -npi tun0
tcpdump: WARNING: arptype 65534 not supported by libpcap - falling back to
cooked socket
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on tun0, link-type LINUX_SLL (Linux cooked), capture size 96 bytes
13:19:46.013188 IP 10.3.0.2 > 192.168.1.2: ICMP echo request, id 62985, seq
1, length 64
13:19:46.013986 IP 192.168.1.2 > 10.3.0.2: ICMP echo reply, id 62985, seq 1,
length 64
13:19:47.027705 IP 10.3.0.2 > 192.168.1.2: ICMP echo request, id 62985, seq
2, length 64
13:19:47.030003 IP 192.168.1.2 > 10.3.0.2: ICMP echo reply, id 62985, seq 2,
length 64
13:19:48.057565 IP 10.3.0.2 > 192.168.1.2: ICMP echo request, id 62985, seq
3, length 64
13:19:48.058251 IP 192.168.1.2 > 10.3.0.2: ICMP echo reply, id 62985, seq 3,
length 64
```

```
13:19:49.089651 IP 10.3.0.2 > 192.168.1.2: ICMP echo request, id 62985, seq
4, length 64
13:19:49.091784 IP 192.168.1.2 > 10.3.0.2: ICMP echo reply, id 62985, seq 4,
length 64
```

Теперь любой клиент из подсети 192.168.127.0/24 сможет попасть на сервер терминалов, находящийся в подсети 192.168.1.0/24 просто указав в строке адреса - 192.168.1.2.

К сожалению, при такой настройке в сетевом окружении windows клиентов вы не сможете использовать NetBIOS имена компьютеров, а только их ip адреса. Если вам необходим данный функционал, то на одной из сторон можно поднять WINS сервер. Или настроить VPN в режиме моста.

На этом настройку нашей системы можно считать завершенной.