Визитка

РАШИД АЧИЛОВ, поклонник FreeBSD с многолетним опытом использования ее в совмещенных с Windows сетях и сторонник Open Source. Администратор сетей и средств защиты крупной торговой сети

Построение корпоративных VPN

Использование IPSec для связи с программным роутером

В четвертой части статьи рассматривается применение IPSec для подключения программного роутера с различными адресами с использованием авторизации через текстовые пароли и сертификаты. Роутер работает под управлением FreeBSD, но конфигурация racoon может быть перенесена в любой UNIX

Подключение со статическим адресом

В качестве программного шлюза будет выступать компьютер под управлением FreeBSD 8.1-RELEASE, на котором установлен ipsec-tools, уже применявшийся в предыдущей части [1]. На схеме он обозначен адресом 170.70.70.1 (см. рис. 1). Мы выбираем более устойчивый, чем применяемый в D-Link 3DES, алгоритм шифрования AES (принятый в США в качестве национального стандарта) как для установления соединения, так и для самого соединения, благо возможность есть.

Вот файл racoon.conf программного шлюза (общие секции опущены, секция remote anonymous{} отсутствует):

```
remote 212.20.5.1
{
        exchange_mode main;
        doi ipsec_doi;
        situation identity_only;
        nonce_size 16;
        lifetime time 24 hour;
        initial_contact on;
        proposal_check strict;
        ike_frag on;
        passive off;
        dpd_delay 0;
        script "linkup_leaf.sh" phase1_up;
        script "linkdown_leaf.sh" phase1_down;
        proposal {
                encryption_algorithm aes;
                hash_algorithm sha1;
                authentication_method pre_shared_key;
                dh_group 2;
                lifetime time 7200 sec;
        }
}
sainfo anonymous
{
        pfs_group 2;
        lifetime time 7200 sec;
        encryption_algorithm camellia,aes,blowfish,3des;
        authentication_algorithm hmac_sha1,hmac_md5;
        compression_algorithm deflate;
}
```

Обратите внимание – мы не используем DPD!

Скрипты для программного шлюза попроще – программа, выполняемая при прекращении соединения, кроме общей части, содержит только:

```
# Записать в журнал факт срабатывания
safe_logger "Deleting SA records for $REMOTE_ADDR"
# Удалить все записи SA, связанные с данными адресами
cmdfile=`mktemp $connected_spool/$REMOTE_ADDR.XXXXXX`
echo "deleteall $esg_address $REMOTE_ADDR esp;" >> $cmdfile
echo "deleteall $REMOTE_ADDR $esg_address esp;" >> $cmdfile
setkey -f $cmdfile
# Удалить собственно рабочий файл
rm -f $cmdfile
```

а скрипт, срабатывающий при установлении соединения, и вообще одну строчку:

```
# Отметить в журнале факт срабатывания
safe_logger "Link up with $REMOTE_ADDR"
```

Где при этом задаются остальные настройки, и как вообще начинается установление соединения?

Поскольку периферийный программный шлюз всегда «знает», к какому узлу ему подключаться, в задании параметров анонимного подключения нет необходимости. Поэтому мы описываем только подключение от него к центральному шлюзу, и все настройки, что на сервере задаются таким образом, чтобы обеспечить их динамическое использование. здесь просто записываются статически. В /etc/rc.conf:

```
network_interfaces="<другие_интерфейсы> gif0"
ifconfig_gif0="inet 10.87.11.1 10.87.1.1 ↵
        netmask 255.255.255.255 mtu 1450"
gif_interfaces="gif0"
gifconfig_gif0="170.70.70.1 212.20.5.1"
static_routes="vpn81"
route_vpn81="-net 10.87.0.0 -netmask 255.255.0.0 10.87.1.1"
```

В /etc/ipsec.conf (вот мы и нашли применение для него):

```
flush;
spdflush;
spdadd 10.87.11.0/24 10.87.11.0/24 any -P out none;
spdadd 10.87.11.0/24 10.87.11.0/24 any -P in none;
spdadd 10.87.11.0/24 10.87.0.0/16 any -P out ipsec ↵
        esp/tunnel/170.70.70.1-212.20.5.1/require;
spdadd 10.87.0.0/16 10.87.11.0/24 any -P in ipsec ↵
        esp/tunnel/212.20.5.1-170.70.70.1/require;
```

Таким образом, немедленно после загрузки уже будут существовать нужные нам записи SP (о том, что шиф-

роваться должен любой трафик между локальной сетью шлюза и любыми сетями 10.87.0.0/16, любыми, а не только 10.87.1.0/24! Потому что если в SP поставить 10.87.1.0/24 вместо 10.87.0.0/16, то, например, ping 10.87.10.10 уже зашифрован не будет), создан и сконфигурирован туннельный интерфейс, и настроена маршрутизация.

Для чего нужны первые две записи SP? Ведь они, если присмотреться, бессмысленные? А вот и нет. Если этих записей не будет, то весь трафик, в том числе и локальный (например, ping 10.87.11.10), будет зашифрован и отправлен в туннель!

Ну вот, можно запускать racoon на программном шлюзе.

Запустили. Ничего не произошло – racoon прочитал свой конфигурационный файл и перешел в фоновый режим, никто никуда не обращается, никто ни с кем не устанавливает – и даже не пытается! – соединение... В чем дело?

Не торопитесь думать, что что-то было сделано не так. Вспомните, какие действия мы только что произвели.

- Мы описали, какие пакеты в системе должны быть зашифрованы при отправке их наружу и какие пакеты следует ожидать зашифрованными при приеме их.
- Мы создали IP-в-IP-туннель, настроив как его внешние (реальные) адреса, так и внутренние (виртуальные). Данный туннель в системе представляется виртуальным интерфейсом.
- Мы настроили маршрутизацию, так что система знает, куда ей направлять нужные пакеты.

Таким образом, у системы есть вся необходимая информация, для того чтобы доставить пакет. Почему же ничего не произошло? Потому что нет самого пакета. Нет собственно данных, которые необходимо шифровать, доставлять и т.д. Стоит запустить, скажем, ping 10.87.1.10, и параллельно запущенный на порту 500 tcpdump покажет прохождение сначала IPSec Фазы 1, потом IPSec Фазы 2, а потом и ping пойдет. При этом несколько первых неизбежно теряются за то время, когда идет установление соединения.

Впрочем, есть и другой способ – racoonctl. Racoonctl – это программа для управления запущенной копией программы racoon через adminsock. Она позволяет принудительно установить или разорвать соединение, перезагрузить конфигурационный файл, обслуживать базу SA, дублируя в этом отношении setkey. Самые интересные возможности racoonctl – это как раз установление и разрыв соединений. По команде racoonctl vpn-connect (vc) выполняется подключение к удаленной стороне, по команде racoonctl vpn-disconnect (vd) – разрыв. При этом при соединении пройдет только Фаза 1, проведение Фазы 2 все равно будет отложено до появления первого пакета с данными. Особой ценности в этой программе нет, все необходимое управление базой SA выполняется через команду setkey, но в скриптах использовать можно.

Если вы все же собираетесь использовать racoonctl, помните, что путь к административному сокету в этой программе задается жестко во время сборки параметром STATEDIR (по умолчанию в FreeBSD /var/db/racoon), причем без модификации Makefile в любом пути последним элементом должен быть racoon. Параметр adminsock секции listen{} должен совпадать с этим значением, иначе racoonctl не будет работать.

Обратите внимание – мы не используем DPD! Почему?

Как только время действия SA истечет, она будет удалена. Запущенный мониторинг DPD тут же обнаружит, что peer dead, и преспокойненько удалит SP, указывающие, что шифровать, и, когда системе понадобится передать очередной пакет, который по идее должен быть зашифрован, SP у нее уже не будет, соединение с удаленным racoon не установится, а данные никуда не отправятся. Убедившись в этом, я всюду DPD отключил.

Серверная часть

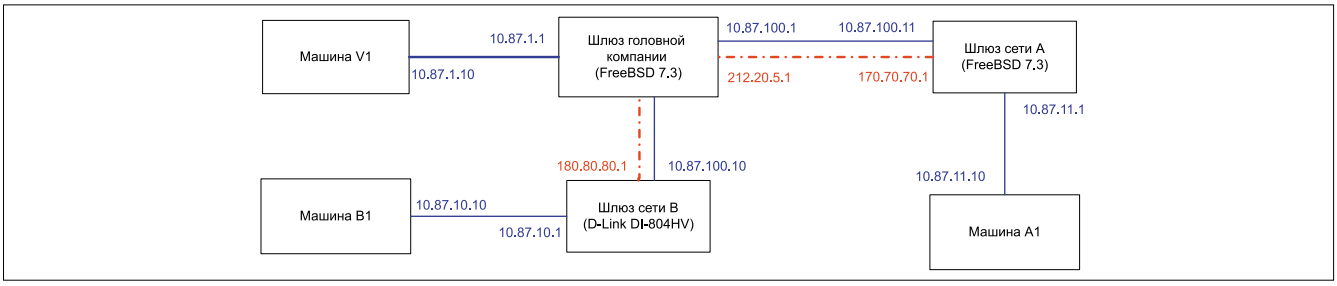
Здесь есть еще один момент, на который мы не обратили внимание. В секции racoon.conf sainfo anonymous{} для программного шлюза мы задали параметры для SA, отличающиеся тем, что заданы в настройках sainfo anonymous{} в конфигурационном файле сервера, поэтому соединение установлено не будет. Но для устройств D-Link нам нужна sainfo anonymous{} именно такого вида, поэтому добавим в racoon.conf сервера описание для программного шлюза:

Здесь ничего особо интересного – немножко другой (по сравнению с параметрами для D-Link) алгоритм шифрования, немножко другой алгоритм хэша, да длительность увеличена до двух часов.

```
remote 170.70.70.1 inherit anonymous
{
        exchange_mode main;
        script "linkup.sh" phase1_up;
        proposal {
                encryption_algorithm aes;
                hash_algorithm sha1;
                authentication_method pre_shared_key;
                dh_group 2;
                lifetime time 7200 sec;
        }
}
```

Этот набор параметров описывает настройки для SA, создаваемой только для туннеля из любой сети 10.87.0.0/16 в сеть 10.87.11.0/24, для других туннелей она использовать-

Рисунок 1. Схема имитационной модели VPN



ся не будет. Здесь время действия SA задается 7200 секунд (2 часа), после чего соединение будет установлено заново в случае необходимости. Напоминаю, что «необходимостью» будет появление любых данных, которые должны быть переданы из сети источника в сеть приемника.

```
sainfo address 10.87.0.0/16 any address 10.87.11.0/24 any
{
        pfs_group 2;
        lifetime time 7200 sec;
        encryption_algorithm camellia,aes,blowfish,3des;
        authentication_algorithm hmac_sha1,hmac_md5;
        compression_algorithm deflate;
}
```

Подключение с динамическим адресом

Рассмотрим вариант подключения программного шлюза когда невозможно получить постоянный действительный («белый») IP-адрес, т.е. он может быть действительным, но выделяться динамически, а может быть и недействительным (т.е. принадлежать согласно RFC1918 к диапазонам для Интранета). Установление соединения делается в этом случае так же, как и при подключении D-Link [1] – используется xAUTH, который передает логин и пароль. Логин и пароль проверяются PAM-модулями, если в этом есть необходимость, и передаются в скрипт, который выполняет обычные действия по управлению интерфейсами и маршрутами. Помнить надо то, что пароль для указываемого логина записывается в файл psk.txt в формате: <логин> <пароль>, и в качестве метода аутентификации следует указать xauth_psk_server на сервере и xauth_psk_client на программном шлюзе.

Секцию sainfo anonymous{} мы повторно приводить не будем – она аналогична упомянутой выше, приведем только настройки remote 212.20.5.1{} для программного шлюза и remote anonymous{} для сервера.

```
remote 212.20.5.1
{
        exchange_mode aggressive;
        doi ipsec_doi;
        situation identity_only;
        nonce_size 16;
        lifetime time 24 hour;
        initial_contact on;
        proposal_check strict;
        nat_traversal on;
        ike_frag on;
        passive off;
        dpd_delay 0;
        xauth_login "baybabay";
        proposal {
                encryption_algorithm aes;
                hash_algorithm sha1;
                authentication_method xauth_psk_client;
                dh_group 2;
                lifetime time 7200 sec;
        }
}
```

Обратите внимание на параметр exchange_mode, на значение xauth_login, который задает имя пользователя (пароль прописан в psk.txt), и на настройку authentication_method.

```
remote anonymous
{
        exchange_mode main,aggressive;
        doi ipsec_doi;
        situation identity_only;
        nonce_size 16;
        lifetime time 24 hour;
        initial_contact on;
        proposal_check strict;
        generate_policy unique;
        ike_frag on;
        passive on;
        script "linkdown.sh" phase1_down;
        proposal {
                encryption_algorithm aes;
                hash_algorithm sha1;
                authentication_method xauth_psk_server;
                dh_group 2;
                lifetime time 7200 sec;
        }
}
```

Фактически вся разница – это смена authentication_method на xauth_psk_server. В anonymous, как правило, ставится как раз xauth_psk_server, чтобы иметь возможность обработать логин xauth, а для конкретных устройств записываются собственные секции. Ну, еще отключен скрипт, срабатывающий на начало Фазы 1, – вместо этого отработает программа, вызываемая через PAM

Подключение с использованием сертификатов

Для D-Link подключение с применением сертификатов невозможно – устройство их не поддерживает, можно использовать только PSK. Программный шлюз в отличие от D-Link позволяет устанавливать связь, авторизуя удаленную сторону с помощью сертификатов, без применения PSK.

Команды по генерации сертификатов мы приводить не будем – о них уже писалось многократно, как минимум в [1, 2]. Предположим, что компьютер центрального узла называется glance.shelton.net, и соответствующие ему файлы называются glance.key и glance.crt. Компьютер же программного шлюза называется babay.shelton.net, и соответствующие ему файлы называются babay.key и babay.crt.

Кроме этого, существует Объединенный Контрольный Файл УЦ (Merged Control File, MCF) – файл, содержащий в себе PEM-формат сертификата УЦ, за которым непосредственно вплотную следует список отзыва данного УЦ. Такое решение используется во многих программах, не имеющих собственной настройки для указания отдельного списка отзыва. Формируется MCF простейшей командой:

```
cat caserv.crt cacrl.pem > camerged.pem
```

где caserv.crt – сертификат УЦ, а cacrl.pem – список отзыва. Сертификаты узлов glance.crt и babay.crt подписаны УЦ, сертификат которого находится в файле camerged.pem.

Напомним также, что согласно конфигурационному файлу racoon.conf все находится в /etc/ssl/certs, в двух подкаталогах certs и rootca. В первом хранятся собственно сертификаты и их ключи, во втором – сертификат УЦ, список отзыва УЦ и перечень некоторых доверенных корневых сертификатов. Приведем только секции remote 212.20.5.1{} для программного шлюза и remote 170.70.70.1{} для центрального сервера.

```
remote 212.20.5.1
{
        exchange_mode main;
        doi ipsec_doi;
        situation identity_only;
        my_identifier asn1dn;
        certificate_type x509 "babay.crt" "babay.key";
        ca_type x509 "/etc/ssl/rootca/camerged.pem";
```

```
        peers_certfile x509 "glance.crt";
        verify_cert on;
        nonce_size 16;
        lifetime time 24 hour;
        initial_contact on;
        proposal_check strict;
        nat_traversal on;
        ike_frag on;
        passive off;
        dpd_delay 0;
        script "linkup_leaf.sh" phase1_up;
        script "linkdown_leaf.sh" phase1_down;
        proposal {
                encryption_algorithm aes;
                hash_algorithm sha1;
                authentication_method rsasig;
                dh_group 2;
                lifetime time 7200 sec;
        }
}
```

Обратите внимание на **exchange_mode** – при применении сертификатов можно использовать main даже при наличии динамических адресов.

My_identifier asn1dn – указывает на то, что идентификатор необходимо прочитать из сертификата, который задается параметром certificate_type. Первый параметр – это тип сертификата (x509), второй и третий – соответственно имя файла сертификата и имя файла ключа, относящегося к данному сертификату. Все файлы, имена которых не начинаются с «/», ищутся относительно path certificate.

Ca_type – указывает, где брать сертификат УЦ, которым подписан сертификат нашего узла. Имя данного файла указано с полным путем. Кроме сертификата УЦ, из него racoon также получает CRL.

Если указать файл, не содержащий CRL, в журналах мы увидим сообщение об ошибке, но работе это не помешает.

```
Jan 28 01:12:20 babay racoon: WARNING: unable to get
certificate CRL(3) at depth:0 SubjectName:/C=RU/
ST=Novosibirskaya/L=Novosibirsk/O=LLC ASK/OU=IT dep/
CN=glance.askd.gmbh/emailAddress=root@glance.askd.gmbh
```

Peers_certifile – позволяет вместо использования сертификата, передаваемого удаленной стороной, указать локальный файл сертификата. Это имеет смысл при проверке идентификаторов, прописанных в сертификатах. В качестве параметров указываются тип и имя файла.

Verify_cert on – включает проверку сертификатов.

Ну и метод аутентификации **rsasig** – аналог pre_shared_key, только вместо парольных фраз используются сертификаты. Если необходимо применение еще и xAUTH, добавляется параметр xauth_login, и authentication_method меняется на xauth_rsa_client.

```
remote 170.70.70.1 inherit anonymous
{
        exchange_mode main;
        my_identifier asn1dn;
        certificate_type x509 "glance.crt" "glance.key";
        ca_type x509 "/etc/ssl/rootca/camerged.pem";
        peers_certfile x509 "babay.crt";
        verify_cert on;
        nat_traversal on;
        script "linkup.sh" phase1_up;
        proposal {
                encryption_algorithm aes;
                hash_algorithm sha1;
                authentication_method rsasig;
                dh_group 2;
                lifetime time 7200 sec;
        }
}
```

В чем ценность подключения с применением сертификатов? При работе со статическими адресами они позволяют отойти от использования текстовых паролей и добавляют дополнительные возможности по ограничению круга устройств, с которыми устанавливается соединение.

Например, добавляется:

```
peers_identifier asn1dn "C=RU, L=Novosibirsk, O=MyOrg, ↵
       OU=*, CN=*";
verify_identifier on;
```

И никто с сертификатами других организаций не сможет подключиться, знай он хоть двадцать паролей. При работе же с динамическими или меняющимися адресами обеспечивается точно такой же, как и для статических адресов, уровень безопасности. Ну и понятно, что такой способ подключения позволяет контролировать круг лиц, которые смогут подключаться с динамическими адресами – если программный шлюз не имеет сертификата, выданного корпоративным УЦ (а использует, например, самоподписанный сертификат или сертификат, подписанный некоторым другим УЦ), то соединение никогда не будет установлено – в журналах сервера будут постоянные сообщения о том, что сертификат удаленной стороны не может быть проверен

```
Jan 28 16:58:15 glance racoon: ERROR: unable to get local
issuer certificate(20) at depth:
0 SubjectName:/C=RU/ST=Novosibirskaya oblast/L=Novosibirsk/
O=LLC ASK/OU=IT/C
N=babay.askd.gmbh/emailAddress=root@babay.askd.gmbh
Jan 28 16:58:15 glance racoon: ERROR: the peer's certifi-
cate is not verified.
```

И хотя для программного шлюза состояние, когда у него имеется динамический или постоянно меняющийся адрес, – это, как правило, ситуация исключительная, например вывод в Интернет через 3G-модем при аварии основного подключения, здесь мы закладываем то, чем будем широко пользоваться при рассмотрении подключения «листьев» – клиентов под управлением ОС Windows, Symbian и Windows Mobile.

***

Охватить все или хотя бы основные возможности IPSec достаточно сложно в рамках одной-двух-трех статей. В данной статье мы рассмотрели только подключение программных маршрутизаторов с различными вариантами адресации и аутентификации. Используя эту и предыдущую части статьи, уже можно построить основной скелет VPN – собрать в единую сеть все точки присутствия (офисы, магазины, склады). Но это пока не позволяет подключать произвольных клиентов с ноутбуками, нетбуками или мобильными устройствами. Об этом – в продолжении темы. EOF

1. Ачилов Р. Построение корпоративных VPN. Использование IPSec для связи с аппаратным роутером. //«Системный администратор», № 3, 2011 г. – С. 28-32.
2. Ачилов Р. SSL-сертификаты сайтов. Назначение и использование. //«Системный администратор», №3, 2010 г. – С. 64-68.
3. Ачилов Р. Используем SSL для защиты корпоративной почты. //«Системный администратор», №9, 2010 г. – С. 48-52.