Вячеслав Калошин

Под грифом «секретно»

Вы хотите иметь собственный почтовый сервер и не хотите, чтобы вашу корреспонденцию могли читать посторонние? А, может быть, вам нужно точно знать подлинность личности отправителя? Помочь в решении этих вопросов, а также предоставить базис для дальнейшего развития почтового сервера способна связка программ Postfix и Courier-IMAP.

Данная статья обобщает опыт, полученный за время, прошедшее с написания предыдущих вариантов ISPMail-HOWTO. Уже сменилось много дистрибутивов, сменились программы, но потребность в собственном почтовом сервере осталась. При этом к нему предъявляются следующие требования:

- почтовые пользователи никак не должны пересекаться с системными;
- должна быть возможность легко сделать резервную копию всего сервера и быстро развернуть ее в другом месте;
- все общение с сервером должно быть защищенным.

Необходимые замечания

В принципе, всем этим условиям удовлетворяют очень многие связки программных продуктов, но я выбрал Postfix и Courier-IMAP. Оба этих сервера очень легко масштабируются и не содержат труднопонимаемых для администратора мест. К тому же через пару месяцев не придется искать часами в конфигурационных файлах какой-то параметр, который необходимо немного изменить.

Также давно стало безразлично, на какой именно системе поднимать данный сервер. Совершенно без разницы, будет это Gentoo, Fedora Core или FreeBSD. Благодаря стандартизации максимум, что вам придется поменять, — это пути к файлам. В данной статье я использовал Fedora Core 2 и 3 как наиболее распространенные дистрибутивы Linux. Так же я старался соблюсти идеологию этих дистрибутивов, а это означает, что никаких программ, установленных в обход системы управления пакетами RPM, здесь описываться не будет.

И последнее напоминание: в этой статье вы не найдете пошагового описания всех действий. Если я пишу «устанавливаем MySQL», то подразумеваю, что нужно будет открыть терминал с правами суперпользователя и набрать следующие команды:

```
yum install mysql
chkconfig mysql on
/etc/init.d/mysqld start
```

Одновременно вы будете читать сообщения в лог-файлах на предмет всевозможных ошибок. Если что-либо будет отличаться от стандартной процедуры, я обязательно об этом упомяну.

Осталось сказать о конфигурации создаваемой системы и используемых именах. У меня есть домашний сервер под именем home.multik.org. К нему я подключаюсь, когда нахожусь дома, он же обрабатывает небольшой сервисный трафик от разных почтовых роботов. Сервер выходит в Интернет через неподконтрольную мне сеть, где может находиться все — от хабов до снифферов. Также в относительно «чистом» сегменте Интернета, на площадке у крупного провайдера, стоит другой мой сервер с именем multik.org. И именно он отвечает за всю почту

домена multik.org. Я хочу настроить свою систему так, чтобы никто из сети, в которой находится home.multik.org, не смог прочитать мою почтовую переписку или украсть пароли. А в других сетях, по большому счету, это никому и не нужно.

Установка MySQL

Первое, что нам необходимо больше всего, — это MySQL. В нем будут храниться все данные о пользователях и их почтовых ящиках.

После установки MySQL обычно я провожу следующие дополнительные действия:

- удаление анонимных пользователей

```
DELETE FROM mysql.user WHERE User = '';
```

- смена пароля root

```
UPDATE mysql.user SET Password = PASSWORD('newpwd')
WHERE User = 'root';
```

- создание специального пользователя, с помощью которого скрипт может удостовериться, что сервер работает

```
grant USAGE on test.* to ping;
```

Теперь, чтобы на экран при запуске MySQL не сыпались ошибки, меняем в /etc/init.d/mysqld строчку

```
RESPONSE = '/usr/bin/mysqladmin -
uUNKNOWN_MYSQL_USER ping 2>&1'
```

на

```
RESPONSE = '/usr/bin/mysqladmin -uping ping 2>&1'
```

Затем надо добавить в /etc/my.cnf одну строчку, которая разрешает серверу отображаться только на указанных адресах.

```
[mysqld]
bind-address = 127.0.0.1
```

На этом минимально необходимая настройка сервера баз данных завершена. Обо всем остальном вроде кластеров или InnoDB неплохо рассказано в документации.

Установка и первичная настройка Postfix

К сожалению, тот вариант сборки Postfix, который присутствует во всех версиях Fedora Core, не подходит по одной-единственной причине — в нем нет поддержки SQL. Поэтому нужно забрать с ближайшего зеркала пакет SRPM и установить его. Запускаем любимый текстовый редактор и правим следующие две строчки в файле postfix.spec:

```
%define MYSQL 1
Epoch: 3
```

Первая, как легко догадаться, включает поддержку MySQL, а вторая служит для предотвращения обновления Postfix средствами дистрибутива. Иначе, как только команда поддержки Fedora выпустит новый релиз (который опять будет без поддержки SQL), он тут же затрет нашу версию. Собираем (rpmbuild -ba postfix.spec) и устанавливаем Postfix. В процессе установки придется удалить без проверки зависимостей (--nodeps) sendmail из-за множества завязанных на предоставляемый им сервис программ.

На этом установка Postfix завершена, и можно плавно переходить к настройке. Открываем файл /etc/postfix/main.cf и редактируем следующие параметры:

```
myhostname = home.multik.org
mydomain = home.multik.org
inet_interfaces = all
mydestination = $myhostname, localhost
mynetworks_style = host
in_flow_delay = 1s
smtpd_banner = $myhostname ESMTP $mail_name
```

В конце файла добавляем:

```
strict_rfc821_envelopes = yes
broken_sasl_auth_clients = yes
transport_maps = mysql:/etc/postfix/transport.cf
virtual_mailbox_base = /
virtual_uid_maps = mysql:/etc/postfix/ids.cf
virtual_gid_maps = mysql:/etc/postfix/gids.cf
virtual_mailbox_maps = mysql:/etc/postfix/aliases.cf
virtual_maps = mysql:/etc/postfix/remote_aliases.cf
relay_domains = $transport_maps
smtpd_recipient_restrictions =
permit_mynetworks,permit_sasl_authenticated,check_relay_domains
disable_vrfy_command = yes
smtpd_sasl_auth_enable = yes
smtpd_sasl_local_domain = home.multik.org
smtpd_sasl_security_options = noanonymous
smtpd_sasl_application_name = smtpd
local_transport = virtual
local_recipient_maps = $virtual_mailbox_maps
smtpd_data_restrictions = reject_unauth_pipelining
```

Прошу обратить ваше внимание на то, что в параметре smtpd_recipient_restrictions нет абсолютно никаких упоминаний о всевозможных RBL-списках. Дело в том, что лично я совершенно не горю желанием отдавать управление своим почтовым трафиком различным неадекватным личностям, которые ведут эти списки. Против действий спамеров на стороне почтового сервера отлично работают команды «in_flow_delay», «smtpd_data_restrictions», «strict_rfc821_envelopes». Чтобы найти описание того, как они это делают, обратитесь к документации.

Теперь нам необходимо создать какое-то количество дополнительных конфигурационных файлов, в которых будет

точно описано, где именно Postfix должен искать информацию о пользователях и доменах. Синтаксис этих файлов довольно прост, так что я оставлю их без комментариев.

▸ transport.cf:

```
user = postfix
password = postfix
dbname = mail
table = transport
select_field = transport
where_field = domain
hosts = localhost
```

▸ ids.cf:

```
user = postfix
password = postfix
dbname = mail
table = aliases
select_field = id
where_field = alias
hosts = localhost
```

▸ gids.cf:

```
user = postfix
password = postfix
dbname = mail
table = aliases
select_field = gid
where_field = alias
hosts = localhost
```

▸ aliases.cf:

```
user = postfix
password = postfix
dbname = mail
table = aliases
select_field = maildir
where_field = alias
hosts = localhost
```

▸ remote_aliases.cf:

```
user = postfix
password = postfix
dbname = mail
table = remote_aliases
select_field = rcpt
where_field = alias
hosts = localhost
```

Вам необходимо будет заменить здесь и далее пароли на более подходящие. Я же оставляю их в открытом виде для наглядности.

Создание базы данных пользователей

Для начала создадим все необходимые таблицы в базе данных.

```
create database mail;
grant insert,select,delete,update on mail.* to postfix@localhost
identified by 'postfix'; use mail;
create table transport (domain varchar(255) PRIMARY KEY,
transport char(8));
create table aliases (id int(6), gid int(6), alias varchar(255) PRI-
```

Настройка антивируса

Почтовый доктор

Согласитесь, было бы очень неплохо, если бы наша почтовая система могла еще и защитить пользователей от приходящих периодически почтовых вирусов. Как известно, абсолютное большинство всех известных вирусов составляют почтовые черви. Соответственно, если мы прикроем основной путь их распространения, то сможем избавиться и от большей части головной боли, связанной с этим «достижением науки и техники». В качестве защиты почтовой системы от вирусов можно избрать один из многих вариантов. Наиболее приемлемыми, на наш взгляд, являются следующие продукты:

- **KAV (www.kav.ru);**
- **ClamAV (www.clamav.net);**
- **Dr.Web (www.antivir.ru);**

На чем именно остановить свой выбор, решать вам. Мы предпочли высоконадежную и многофункциональную систему Dr.Web. Этот российский программный продукт прекрасно работает на весьма перегруженных почтовых системах таких компаний как Yandex.ru или Mail.ru. Кроме того, данный пакет выгодно отличается от конкурентов невысокой ценой, более чем приемлемой для большинства российских пользователей.

Программный комплекс Dr.Web по проверке почты состоит из двух компонентов — антивирусного демона, который выполняет процедуру проверки, а также специальной программы, передающей данные ему на проверку. Демон устанавливается на сервер и слушает выделенный ему порт или Unix-сокет (в зависимости от конфигурации). Сам плагин — небольшая программа, умеющая общаться как с демоном, так и с почтовым сервером. Его помещают на пути приема почты, и он передает демону все данные на проверку. Демон возвращает статус проверенных данных, после чего плагин принимает решение о том, как ему лучше поступить с тем или иным принятым письмом. Для развертывания всей системы вам потребуется сначала установить демон:

rpm -Uvh drweb-4.31.4-glibc.2.3.i586.rpm

А затем развернуть пакет с фильтром в корневую директорию системы:

```
MARY KEY,maildir varchar(255),password varchar(128), info
varchar(128));
create table remote_aliases (alias varchar(255) PRIMARY KEY,
rcpt varchar(255));
```

Затем определим домены, обслуживаемые данным сервером:

```
insert into transport values ('home.multik.org','virtual:');
```

И заведем первого пользователя:

```
insert into aliases values
(1000,12,'multik@home.multik.org','/var/spool/vmail/home.mul-
tik.org_multik/','password','multik@home.multik.org account');
```

1000 — это тот UID, который будет использоваться Postfix при сохранении писем на диск.

В заключение создадим каталог для хранения почты:

```
mkdir /var/spool/vmail
chown nobody.mail /var/spool/vmail
chmod 777 /var/spool/vmail
```

Вам необходимо сменить права доступа к каталогу в соответствии с принятой у вас политикой безопасности. В любом случае Postfix при создании почтовых ящиков пользователей сам установит на них минимально необходимые права, так что нарушения безопасности не произойдет в любом случае.

Первое письмо

Все предварительные настройки осуществлены. Теперь настал момент истины — запуск сервера и отправка первого письма:

```
/etc/init.d/postfix start
newaliases
mail multik@home.multik.org
```

В лог-файлах вы должны наблюдать нечто похожее:

```
Mar 13 14:49:14 home postfix/pickup[12804]: 96BC718E74B:
uid = 0 from = <root>
Mar 13 14:49:14 home postfix/cleanup[12811]: 96BC718E74B:
message-id = <20050313114914.96BC718E74B@home.multik.org>
Mar 13 14:49:14 home postfix/qmgr[12805]: 96BC718E74B:
from = <root@home.multik.org>, size = 300, nrcpt = 1 (queue active)
Mar 13 14:49:14 home postfix/virtual[12817]: 96BC718E74B:
to = <multik@home.multik.org>, relay = virtual, delay = 0,
status = sent (delivered to maildir)
Mar 13 14:49:14 home postfix/qmgr[12805]: 96BC718E74B:
removed
```

Если все именно так, значит первоначальная настройка закончена. Если же нет, то советую открыть браузер, отправиться на сайт www.google.com и искать там способы решения возникшей проблемы. Для получения более подробной информации о производимых сервером Postfix операциях рекомендую обратиться к параметру debug_peer_level, расположенному в main.cf.

Теперь в MySQL прописываем необходимые синонимы:

```
mysql> insert into mail.remote_aliases values('root@home.mul-
tik.org', 'multik@home.multik.org');
```

Для настройки «боевого» сервера очень рекомендую прописать следующие синонимы, полезные для работы различных

---

tar xzvf drweb-postfix-4.31-linux.tar.gz

Остается внести изменения в файл /etc/postfix/master.cf, добавив в него строку:

filter unix - n n -- pipe
flags = R user = drweb argv = /opt/drweb/drweb-postfix -f ${sender} -- ${recipient}

Также нужно заменить строку:

smtp inet n - n - - smtpd
на
smtp inet n - n - NN smtpd -o content_filter = filter:dummy

В нашем случае значение NN можно изменить на «-». Для того чтобы получить наиболее актуальную на сегодняшний день версию антивирусной базы, потребуется добавить в директорию /etc/cron.hourly специальный скрипт, отвечающий за регулярные обновления. Для этого выполним следующие действия:

$ touch drweb.update
/etc/cron.hourly

Далее в полученный файл drweb.update впишем следующие строки:

#!/bin/bash
/opt/drweb/update/update.pl

Затем сохраним изменения и дадим команду:

chmod 777
/etc/cron.hourly/drweb.update

Демонстрационная версия антивирусного пакета Dr.Web доступна для скачивания на сайте www.antivir.ru/download, но, к сожалению, она имеет ряд существенных ограничений. Например, не производится проверка на вирусы в архивах, не обрабатываются заголовки и не проводится лечение. Приходят только лишь уведомления о том, что в таком-то письме программой был обнаружен вирус.

В общем-то, потратив совсем немного времени, можно получить высокотехнологичную и очень удобную систему обработки почтовой корреспонденции, которая будет обладать всеми обязательными для нее свойствами — надежностью, удобством в использовании и простотой в обслуживании.

почтовых служб всех доменов, которые будет обслуживать сервер: MAILER-DAEMON, postmaster, abuse, noc.

Теперь можно настраивать авторизацию.

Приводим все содержимое файла /usr/lib/sasl2/smtpd.conf к следующему виду:

```
pwcheck_method: auxprop
mech_list: PLAIN CRAM-MD5 DIGEST-MD5 LOGIN
allowanonymouslogin: no
allowplaintext: no
sasl_auxprop_plugin: sql
sql_engine: mysql
sql_database: mail
sql_user: postfix
sql_passwd: postfix
sql_select: select password from aliases where alias = "%u@%r"
```

Теперь, если попробовать отправить письмо, то в логах однозначно возникнет следующая ошибка:

```
SASL authentication problem: unable to open Berkeley db
/etc/sasldb2: No such file or directory
```

Исправляем ее созданием фиктивного пользователя и последующим его удалением:

```
saslpasswd2 -c test
saslpasswd2 -d test
```

Теперь можно открыть любимый почтовый клиент и попытаться отправить письмо самому себе. Главное — не забыть при этом поставить галочки, указывающие на то, что почтовому серверу необходима авторизация. Если вы все сделаете правильно, почтовый клиент запросит у вас имя пользователя и пароль (в нашем случае это multik@home.multik.org и password соответственно), а затем отправит письмо. Внимательно поглядев на заголовки письма, вы должны увидеть там примерно следующие строки:

```
Received: from 192.168.1.118 (ppp83-237-195-35.pppoe.mtu-
net.ru [83.237.195.35])
by multik.org (Postfix) with ESMTP id 8259E8F80D4
for <multik@multik.ru>; Sun, 13 Mar 2005 04:05:12 -0800
(PST)
```

Обратите внимание на присутствие ESMTP во второй строке. Эти буквы означают, что сервер не просто так взял и передал письмо, а с определенным «понятием». К сожалению, пока точного представления о его внутренней работе в данном случае получить невозможно. Но не отчаивайтесь, мы это исправим чуть позже.

Установка и настройка сервера IMAP

Теперь отправляемся на сайт www.courier-mta.org/imap, скачиваем последнюю версию Courier-IMAP и, согласно располагающемуся на страницах ресурса FAQ, собираем исходные коды в RPM-пакет. Единственное, что необходимо поправить в spec-файле, — это зависимости от openldap и fam. Первое лично мне вовсе не нужно, а второе лучше будет заменить на gamin.

В общем, благодаря хорошей документации установка ни разу не вызвала у меня каких-либо затруднений.

Теперь переходим в каталог /etc/authlib/ и настраиваем систему авторизации у Courier.

В файле authdaemonrc необходимо убрать из списка модулей все кроме authmysql:

```
authmodulelist = "authmysql"
```

А в файле authmysqlrc не забудьте прописать несколько важных переменных:

```
MYSQL_SERVER          localhost
MYSQL_USERNAME        postfix
MYSQL_PASSWORD        postfix
MYSQL_SOCKET          /var/lib/mysql/mysql.sock
MYSQL_PORT            3306
MYSQL_OPT             0
MYSQL_DATABASE        mail
MYSQL_USER_TABLE      aliases
MYSQL_CLEAR_PWFIELD   password
DEFAULT_DOMAIN        home.multik.org
MYSQL_UID_FIELD       id
MYSQL_GID_FIELD       gid
MYSQL_LOGIN_FIELD     alias
MYSQL_HOME_FIELD      maildir
MYSQL_MAILDIR_FIELD   maildir
MYSQL_NAME_FIELD      info
```

Далее переходим в каталог /usr/lib/courier-imap/etc и очень внимательно просматриваем все конфигурационные файлы. Мне совершенно был не нужен протокол POP3, поэтому в соответствующих местах я изменил приведенные ниже строчки следующим образом:

```
POP3DSSLSTART = NO
POP3_STARTTLS = NO
POP3DSTART = NO
```

Еще раз все проверяем и запускаем Courier-IMAP.

```
/etc/init.d/courier-imap restart
```

Но при попытке соединения почтового клиента с сервером в лог-файле у меня появилась вот такая ошибка:

```
Mar 13 16:06:11 home imapd: connection, ip = [::ffff:127.0.0.1]
Mar 13 16:06:11 home imapd: authdaemon: s_connect() failed:
No such file or directory
```

После непродолжительного раздумья я решил, что сам по себе authdaemon не запустится, выполнил команду «/etc/init.d/courier-authlib start» и снова попробовал подключиться, но уже с включенным выводом отладочной информации:

```
Mar 13 16:39:14 home authdaemond: Authenticated:
sysusername = <null>, sysuserid = 1000, sysgroupid = 12,
homedir = /var/spool/vmail/home.multik.org_multik/,
address = multik@home.multik.org, fullname = multik@
home.multik.org account, maildir = <null>, quota = <null>,
options = <null>
```

Как видим, все прошло успешно, и сервер нашел все необходимые ему данные.

Система электронных подписей

На этом можно было бы, в принципе, и заканчивать статью, если бы не третий пункт обозначенных в самом начале требований. Его можно снять с повестки дня только одним способом — шифрованием и соответствующей авторизацией. Для выполнения этой роли как нельзя лучше подходят технологии, описываемые буквами SSL и TLS.

А для этих технологий нужны, в свою очередь, ключи и сертификаты. Конечно, можно прилично заплатить той же компании VerySign и получить все готовое, как говорится, на блюдечке, но очень жалко денег. Поэтому попробуем сделать самоподписанный сертификат, единый для всех сервисов, предоставляемых сервером.

Начнем мы непосредственно с самого «центра сертификатов» и создадим все необходимое:

```
openssl genrsa -des3 -out multik-ca.key 2048
openssl req -new -x509 -days 1825 -key multik-ca.key
-out multik-ca.crt
```

После выполнения этих двух команд мы будем иметь в multik-ca.crt сертификат со сроком действия на пять лет. Следующим шагом станет создание сертификатов серверов и их подпись нашим удостоверяющим центром.

Сначала для multik.org:

```
openssl genrsa -des3 -out multik-server.key 1024
openssl req -new -key multik-server.key -out multik-server.csr
openssl x509 -req -in multik-server.csr -out multik-server.crt
-sha1 -CA multik-ca.crt -CAkey multik-ca.key
-CAcreateserial -days 1825
```

Затем для home.multik.org:

```
openssl genrsa -des3 -out home-multik-server.key 1024
openssl req -new -key home-multik-server.key
-out home-multik-server.csr
openssl x509 -req -in home-multik-server.csr
-out home-multik-server.crt -sha1 -CA multik-ca.crt
-CAkey multik-ca.key -CAcreateserial -days 1825
```

Описание этого процесса вы сможете найти в документации к программе openssl и на многочисленных тематических ресурсах.

Итак, с серверами мы разобрались, теперь подошла очередь почтового клиента:

```
openssl genrsa -des3 -out multik-client.key 1024
openssl req -new -key multik-client.key -out multik-
client.csrEnter pass phrase for multik-client.key:
openssl x509 -req -in multik-client.csr -out multik-client.crt
-sha1 -CA multik-ca.crt -CAkey multik-ca.key -CAcreateserial
-days 1825
```

Единственное отличие от серверных сертификатов состоит в том, что нам необходимо теперь привести все к поддерживаемому большинством программ формату PKCS12:

```
openssl pkcs12 -export -in multik-client.crt -inkey multik-
client.key -name "Viacheslav Kaloshin cert" -out multik-client.p12
```

Последний шаг в создании сертификатов — снятие паролей с серверных сертификатов. Это необходимо для того, чтобы сервер не запрашивал постоянно пароль при каждом запуске или перезагрузке. Необходимо будет позаботиться и о том, чтобы полученные файлы не были доступны кому попало:

```
cp multik-server.key multik-server.key.orig
openssl rsa -in multik-server.key.orig -out multik-server.key
cp home-multik-server.key home-multik-server.key.orig
openssl rsa -in home-multik-server.key.orig -out home-multik-
server.key
```

В качестве финального аккорда положим файлы в какое-то одно определенное место. Я, например, выбрал для этого каталоги сервера apache — /etc/httpd/conf/ssl.crt и ssl.key.

Теперь добавляем в main.cf для обоих почтовых серверов строки следующего содержания:

```
smtpd_use_tls = yes
smtpd_tls_auth_only = yes
smtpd_tls_key_file = /etc/httpd/conf/ssl.key/home-multik-server.key
smtpd_tls_cert_file = /etc/httpd/conf/ssl.crt/home-multik-server.crt
smtpd_tls_CAfile = /etc/httpd/conf/ssl.crt/multik-ca.crt
smtpd_tls_loglevel = 3
smtpd_tls_received_header = yes
smtpd_tls_session_cache_timeout = 3600s
tls_random_source = dev:/dev/urandom
```

Перезагружаем Postfix, затем включаем в настройках почтового клиента использование TLS и пробуем отправить письмо. В моем случае в лог-файле получилось следующее:

```
Mar 14 12:39:01 home postfix/smtpd[6851]: setting up TLS
connection from home.multik.org[127.0.0.1]
Mar 14 12:39:11 home postfix/smtpd[6851]: TLS connection
established from home.multik.org[127.0.0.1]: TLSv1 with cipher
```

```
DHE-RSA-AES256-SHA (256/256 bits)
Mar 14 12:39:11 home postfix/smtpd[6851]: 8D9C918E772:
client = home.multik.org[127.0.0.1], sasl_method = CRAM-MD5,
sasl_username = multik@home.multik.org
```

А в заголовках письма можно было прочитать строки вот такого содержания:

```
Received: from [127.0.0.1] (home.multik.org [127.0.0.1])
    (using TLSv1 with cipher DHE-RSA-AES256-SHA
(256/256 bits))
    (No client certificate requested)
    by home.multik.org (Postfix) with ESMTP id 8D9C918E772
    for <multik@home.multik.org>; Mon, 14 Mar 2005
12:39:11 +0300 (MSK)
```

В процессе передачи письма был установлен зашифрованный с помощью TLS канал, но нам по-прежнему так и не видно, кто же отправил сообщение.

В принципе, такое поведение почтового сервера можно назвать правильным. Ни он, ни почтовый клиент ничего не знают про ключи пользователя. Поэтому следующий шаг необходимо посвятить исправлению этого недостатка. Для этого импортируем наши сертификаты в почтовый клиент. В Thunderbird это делается с помощью разделов меню «Options → Advanced → Certificates → Manage → Certificates → Authorities» и нажатия кнопки «Import». Аналогичным образом надо поступить со вкладкой «Your certificates» и файлом multik-client.p12.

Теперь пришло время скомандовать Postfix спрашивать у клиентов, кто они такие:

```
smtpd_tls_loglevel = 1
smtpd_tls_ask_ccert = yes
relay_clientcerts = hash:/etc/postfix/relay_clientcerts
```

В файле relay_clientcerts должны находиться отпечатки клиентских сертификатов, которым данный сервер доверяет.

Отпечаток снимается следующей командой:

```
openssl x509 -fingerprint -in multik-client.crt
```

Нас интересует последняя строчка из показанного списка:

```
MD5
Fingerprint = 90:B6:12:52:57:D3:35:93:5D:85:91:4A:04:0F:DA:5B
```

Добавляем в файл relay_clientcerts следующее:

```
90:B6:12:52:57:D3:35:93:5D:85:91:4A:04:0F:DA:5B multik
```

Все, что написано после отпечатка, сервером никак не воспринимается, поэтому можно воспользоваться этим местом для описания отпечатков. А теперь создадим хеш отпечатков и перезагрузим сервер:

```
postmap relay_clientcerts
postfix reload
```

Попробуем снова отправить письмо, внимательно просматривая лог-файл. У меня, к примеру, появились такие строчки:

```
Mar 14 12:50:48 home postfix/smtpd[6995]: Verified:
subject_CN = Viacheslav Kaloshin, issuer = multik.org
```

А в заголовках письма было написано следующее:

```
Received: from [127.0.0.1] (home.multik.org [127.0.0.1])
    (using TLSv1 with cipher DHE-RSA-AES256-SHA
(256/256 bits))
    (Client CN "Viacheslav Kaloshin", Issuer "multik.org" (verified OK))
    by home.multik.org (Postfix) with ESMTP id 5D87E18E779
    for <multik@home.multik.org>; Mon, 14 Mar 2005
12:50:48 +0300 (MSK)
```

Связка «почтовый клиент — почтовый сервер» заработала. Теперь, мало того что никто не сможет прочитать отправленное мною письмо по пути, так еще и во всех мыслимых местах останутся свидетельства, что это его отправил именно я.

Переходим к процессу получения писем. Courier-IMAP предъявляет немного другие требования к хранению открытых и личных частей ключа — их необходимо хранить в одном файле:

```
cp home-multik-server.key /etc/cert/home-multik-server.pem
cat home-multik-server.crt >> /etc/cert/home-multik-server.pem
```

Указываем в imapd-ssl путь к этим файлам:

```
TLS_TRUSTCERTS = /etc/cert/trusted.crt
TLS_CERTFILE = /etc/cert/home-multik-server.pem
TLS_VERIFYPEER = PEER
```

Перезагружаем IMAP-сервер. При попытке соединения в логах появятся следующие строки:

```
Mar 14 12:59:13 home imapd-ssl: couriertls: connect:
error:140890B2:SSL routines:SSL3_GET_CLIENT_CERTIFICATE:no certificate returned
```

Это говорит о том, что соединение установилось, но сервер ничего не знает про сертификат клиента. Так добавим же его:

```
cat multik-client.crt > /etc/cert/trusted.crt
```

Снова попробуем соединиться, установив необходимые галочки в настройках почтового клиента. Самое главное теперь — проверить, что клиент будет запрашивать шифрованное соединение TLS с портом 143, а не просто соединится через SSL по 465-му порту. Ведь Courier напрямую никуда не пишет о параметрах соединения, поэтому приходится догадываться.

Если все прошло успешно, в логах появятся строки такого вида:

```
Mar 14 15:22:33 home imapd-ssl: Connection, ip = [::ffff:127.0.0.1]
Mar 14 15:22:33 home imapd-ssl: LOGIN,
user = multik@home.multik.org, ip = [::ffff:127.0.0.1],
protocol = IMAP
```

Как видим, несмотря на то что соединение установлено через IMAP, в логах появляются сообщения об imap-ssl. Это и есть косвенное свидетельство наличия шифрования в канале. Если нужно более весомое доказательство, воспользуйтесь любым сниффером и просмотрите трафик. Первой же после установки соединения должна стать команда «starttls».

Технология TLS в IMAP не очень распространена, и с некоторыми версиями клиентов шифрации трафика может не получиться, признаком чего станет появление в логах строк вида:

```
error:1408F10B:SSL routines:SSL3_GET_RECORD:wrong
version number
```

Я так и не смог проследить тенденцию, но даже разные сборки одного и того же клиента могут вести себя совершенно по-разному, поэтому выяснить, какой клиент поддерживает TLS и IMAP, вы можете сами опытным путем.

Шифрование канала между серверами

Последний шаг, который нам необходимо сделать, — это установить зашифрованное соединение между двумя почтовыми серверами. Здесь мы пойдем уже по проторенной дорожке, так как основные моменты были выяснены ранее.

В конфигурационный файл home.multik.org добавим строки:

```
smtp_use_tls = yes
smtp_tls_key_file = /etc/httpd/conf/ssl.key
/home-multik-server.key
smtp_tls_cert_file = /etc/httpd/conf/ssl.crt
/home-multik-server.crt
smtp_tls_CAfile = /etc/httpd/conf/ssl.crt/multik-ca.crt
```

Они командуют Postfix пытаться установить защищенное соединение, используя указанные ключи. Отправляем письмо и смотрим в заголовки. У меня, к примеру, появилось следующее:

```
Received: from home.multik.org (unknown [195.166.171.193])
      (using TLSv1 with cipher DHE-RSA-AES256-SHA
(256/256 bits))
      (Client CN "home.multik.org", Issuer "multik.org"
(verified OK))
      by multik.org (Postfix) with ESMTP id B04E28F80D4
      for <multik@multik.org>; Mon, 14 Mar 2005 04:53:02
-0800 (PST)
Received: from [127.0.0.1] (home.multik.org [127.0.0.1])
      (using TLSv1 with cipher DHE-RSA-AES256-SHA
(256/256 bits))
      (Client CN "Viacheslav Kaloshin", Issuer "multik.org" (veri-
fied OK))
      by home.multik.org (Postfix) with ESMTP id 0326B18E77D
      for <multik@multik.org>; Mon, 14 Mar 2005 15:53:01
+0300 (MSK)
```

Увидев нечто аналогичное у себя на мониторе, вы можете быть уверены в том, что каждый шаг в процессе приема или отправки писем на ваших почтовых серверах действительно контролируется только вами, и больше никем, что никакой нехороший человек не вклинится и не прочитает вашу личную переписку и что все отправители корреспонденции являются именно теми персонами, за которых себя и выдают.

Совершенству нет предела

На этом создание почтовой системы можно считать завершенным. Поставленные в самом начале статьи задачи мы с успехом выполнили: сервер исправно работает и может быть в дальнейшем усовершенствован.

Например, самым логичным было бы добавить антивирус (ClamAV, KAV или же Dr.Web, установка и настройка которого вкратце описана выше в блоке «Почтовый доктор»), а также поддержку технологии защиты от спама SPF или SpamAssassin. Но это уже тема совсем другой статьи.

Почтовый веб-интерфейс

Вездесущий e-mail

Чтобы работа с почтой была удобной на любом компьютере, можно настроить доступ к почтовому серверу через веб-интерфейс. Так как мы выбрали основным почтовым протоколом IMAP, то для организации веб-доступа лучше всего подойдет бесплатный программный пакет Squirrelmail. Процедура его установки весьма проста. В данном случае нам потребуется выполнить команду:

yum install squirrelmail

По умолчанию веб-интерфейс для работы с почтой будет доступен по адресу:

http://<hostname>/webmail

Для получения более красивого адреса имеется несколько способов. Первый — создать виртуальный домен вида mail.company.com. За поддержку такого адреса будет отвечать ваш сервер доменных имен и веб-сервер Apache. Другой вариант — прописать в настройках /etc/httpd/conf.d/squirrelmail.conf пункт «Alias» с более коротким именем, например /mail. Все, теперь у нас есть полностью работоспособный почтовый сервер с возможностью доступа к нему через веб-интерфейс. Более подробную информацию по настройке Squirrelmail можно получить на сайте www.squirrelmail.org.