Я решил актуализировать старую и очень популярную статью про настройку своего почтового сервера. Статья будет про установку postfix, dovecot, mysql базы, postfixadmin, roundcube, dkim на базе CentOS 8. Последнее время инфраструктура информационных систем стремительно изменяется, но классические почтовые сервера по прежнему актуальны и востребованы.

Содержание:

Цели статьи

1. Рассказать о настройке DNS для почтового сервера.
2. Установить и настроить базовый функционал почтового сервера на базе Centos 8 с помощью Postfix и Dovecot.
3. Настроить панель управления почтовым сервером Postfixadmin.
4. Настроить Web интерфейс Roundcube, а так же его плагины acl, managesieve.
5. Рассказать о способах борьбы со спамом средствами postfix.

Данная статья является частью единого цикла статьей про сервер Centos.

Введение

У меня есть аналогичная статья про настройку postfix на centos 7. Она в целом не утратила актуальность, кроме ссылок на старые версии софта. С тех пор ничего принципиально не изменилось. Если вы не хотите использовать Centos 8, то смело используйте ту статью. У 7-й версии centos еще очень долго будет поддержка, так что вам скорее всего хватит этого времени.

> Если у вас есть сомнения на тему того, какой почтовый сервер использовать для организации, посмотрите мою стать-размышления на эту тему. Возможно, она вам поможет определиться с выбором.

Как я уже сказал, настраивать почтовый сервер буду на ОС linux, а точнее на **CentOS 8**. За основу будет взят **postfix**, который присутствует в этой системе из коробки. Инструкция получится универсальной, можно использовать и для других дистрибутивов. Все основные конфиги легко переносятся на разные системы, требуя минимальной правки, в основном путей. Я без проблем по своим статьям для centos настраивал почтовый сервер на ubuntu.

Я напишу статью на самом что ни на есть реальном примере, без какой-либо правки доменов, ip и прочего, чтобы не ошибиться и показать максимально возможный реальный пример. У меня есть домен kirushin-vladimir.ru. Я буду использовать его в своей работе. Почтовый сервер будет иметь имя mail.kirushin-vladimir.ru. Всю теорию по подготовке dns к установке и настройке почтового сервера я рассказывал в предыдущей статье о почтовом сервере. Не хочу здесь повторяться. Уточню только список действий, которые вам нужно проделать с ДНС:

1. Создаем А запись в DNS — mail.kirushin-vladimir.ru.
2. Добавляем или редактируем MX запись, указывая в качестве почтового сервера mail.kirushin-vladimir.ru.

3. Просим провайдера прописать PTR для внешнего ip адреса, который будет использовать почтовый сервер. В качестве ptr записи просим установить имя нашего сервера — mail.kirushin-vladimir.ru.

Я использую бесплатный dns хостинг от Selectel. Раньше использовал сервера Яндекса, но из-за того, что они заблокированы на Украине, это создает неудобства. На картинке показан минимально необходимый набор записей, кроме PTR. Этими записями управляете не вы, а провайдер, который вам выдает ip адрес. Пока с днс все. Позже мы вернемся к этому вопросу, когда будем добавлять dkim и spf записи. Но обо всем по порядку.

Подготовим систему centos 8 к установке и настройке почтового сервера postfix. Если у вас еще нет готовой системы, то рекомендую воспользоваться моими статьями по установке и настройке centos. Отдельно потратьте время на настройку iptables. Я не буду касаться этого вопроса в данной статье, чтобы не раздувать ее второстепенными вещами. Удобнее, когда все по отдельности рассказано и описано с должной глубиной. Сваливать все в одну кучу не хочется.

По вступлению вроде все, основное рассказал. Приступим к настройке нашего почтового сервера.

Проверка DNS записей

Проверим наши настройки dns. Для этого зайдем на любой сервер по ssh, где установлен пакет **bind-utils**. Если у вас его нет, то ставьте командой ниже.

```
# dnf install bind-utils
```

Убедимся в правильности записей следующим набором проверок.

```
# dig kirushin-vladimir.ru mx | grep IN
# dig mail.kirushin-vladimir.ru | grep IN
# host 5.180.137.106
```

Сначала проверили **MX** запись. Посмотрели, на какую **A** запись она ссылается. Потом проверили эту **A** запись. В завершении убедились, что ip адрес этой **A**

записи имеет **PTR** запись с таким же именем домена.

Это идеальный вариант настроек, к которому нужно стремиться. В спецификации протокола smtp нет таких требований, но по факту, чтобы снизить риск попадания в спам, нужно настраивать все именно так. Если у вас нет возможности изменить PTR запись IP адреса, это не сильно страшно. Почта работать будет и без нее. Это проверено на практике. Но если есть возможность все настроить правильно, сделайте это.

С проверками закончили. Плавно переходим к настройке почтового сервера.

Установка postfixadmin

Начнем с установки и настройки панели управления почтовым сервером postfix — **postfixadmin**. Без него начинать что-то делать неудобно, так как управлять пользователями, ящиками, алиасами будет нечем. По своей сути postfixadmin — набор php скриптов для управления записями в mysql базе данных, которую использует сервер postfix во время своей работы. Соответственно, для работы postfixadmin нам нужен web сервер. Подробно о настройке web сервера на centos 8 читайте отдельно. Сейчас же мы быстро установим все необходимое. Привожу только команды, без комментариев. Все подробности по приведенной выше ссылке.

Подключаем репозиторий remi с дополнительными пакетами для php (нужен для php-imap), обновляем кэш.

```
# dnf install dnf-utils http://rpms.remirepo.net/enterprise/remi-release-8.rpm
# dnf makecache
# dnf module enable php:remi-7.2
```

Устанавливаем необходимые пакеты.

```
# dnf install httpd php mariadb mariadb-server php-imap php-mysqlnd php-mbstring
```

Этих пакетов со всеми зависимостями будет достаточно для установки всех необходимых компонентов веб сервера. Запускаем и добавляем в автозагрузку httpd.

```
# systemctl enable --now httpd
```

Теперь можно зайти по ip адресу через браузер. Вы должны увидеть дефолтную страницу Apache.

Если страница не открывается, то скорее всего у вас не настроен firewalld. Займитесь его настройкой, ссылку я давал в начале. Если он вам не нужен, то удалите его.

```
# dnf remove firewalld
```

Предлагаю сразу поставить phpmyadmin, с ним удобно работать с базой. В нашем случае все пользователи будут храниться в mysql, иногда может понадобиться туда заглянуть. Загружаем и распаковываем последнюю версию.

```
# cd ~
# wget https://files.phpmyadmin.net/phpMyAdmin/5.0.1/phpMyAdmin-5.0.1-all-languages.tar.gz
```

Распаковываем и перемещаем в директорию web сервера.

```
# tar xzvf phpMyAdmin-5.0.1-all-languages.tar.gz
# mkdir /var/www/html/basa
# cp -R /root/phpMyAdmin-5.0.1-all-languages/* /var/www/html/basa/
# chown -R apache. /var/www/html/basa/
```

Идем по ссылке http://5.180.137.106/basa/ и проверяем, что интерфейс phpmyadmin нормально загружается.

Запускаем базу данных mariadb, добавляем в автозагрузку, устанавливаем пароль root.

```
# systemctl enable --now mariadb
# /usr/bin/mysql_secure_installation
```

Теперь можно через phpmyadmin подключиться к mysql серверу с помощью учетной записи root.

Не забудьте ограничить тем или иным образом доступ к phpmyadmin, либо вообще отключить после настройки. Выставлять его в интернет крайне не рекомендуется. В этой панели периодически находят уязвимости, через которые можно скомпрометировать сервер.

Сразу создадим тут пользователя postfix и одноименную базу данных. Запомните учетные данные, они нам далее понадобятся. Установите кодировку базы **utf8_general_ci**, чтобы потом не возиться с этим.

Веб сервер готов, продолжаем настройку почтового сервера. Скачиваем последнюю версию postfixadmin.

```
# cd ~
# wget https://sourceforge.net/projects/postfixadmin/files/postfixadmin/postfixadmin-3.2/postfixadmin-3.2.tar.gz
```

Распаковываем и копируем в директорию web сервера.

```
# tar xzvf postfixadmin-3.2.tar.gz
# mkdir /var/www/html/padmin
# cp -R /root/postfixadmin-3.2/* /var/www/html/padmin
# chown -R apache. /var/www/html/padmin
```

Копируем дефолтный конфиг postfixadmin для того, чтобы вносить туда свои изменения.

```
# cp /var/www/html/padmin/config.inc.php /var/www/html/padmin/config.local.php
```

Убедитесь, что файл config.local.php содержит следующие исправленные параметры.

```
$CONF['configured'] = true;
$CONF['setup_password'] = 'bff0983368335032599fddsfhgjtretuiu8776yhj657de020a90ba837';
$CONF['default_language'] = 'ru';
$CONF['database_type'] = 'mysqli';
$CONF['database_host'] = 'localhost';
```

```
$CONF['database_user'] = 'postfix';
$CONF['database_password'] = '123';
$CONF['database_name'] = 'postfix';
$CONF['admin_email'] = 'root@kirushin-vladimir.ru';
$CONF['encrypt'] = 'md5crypt';
$CONF['domain_path'] = 'YES';
$CONF['domain_in_mailbox'] = 'YES';
$CONF['transport_default'] = 'virtual';
$CONF['show_footer_text'] = 'YES';
$CONF['footer_text'] = 'Return to http://5.180.137.106/padmin/public/';
$CONF['footer_link'] = 'http://5.180.137.106/padmin/public/';
$CONF['default_aliases'] = array (
 'abuse' => 'root',
 'hostmaster' => 'root',
 'postmaster' => 'root',
 'webmaster' => 'root'
);
```

Обращаю внимание на выделенный параметр. Он указывает на то, в каком виде хранить пароли пользователей в базе данных. Конечно, хранить обычным текстом без шифрования это дурной тон и может быть опасно. Я указал хранение в шифрованном виде. Но если мы говорим о небольшой компании без публичного доступа к серверу, можно использовать нешифрованные пароли. Для этого указываем значение параметра **cleartext**. Я сам так часто делаю просто из соображений удобства. Объясню, в чем удобство.

К примеру, у пользователя несколько устройств подключены к почте и он забыл свой пароль. Админ при создании почему-то тоже его никуда не записал, или забыл, или потерял. Вам придется сбросить пароль и перенастроить все устройства. Если же у вас пароль хранится в открытом виде, вы просто смотрите в базу и говорите пользователю пароль. Пароли в открытом виде удобно просто выгрузить дампом из базы, если понадобится кому-то все учетки передать. Но тут как посмотреть :) С одной стороны плюс, с другой минус — кто-то очень просто может спереть все ваши пароли. В общем, тут от ситуации зависит, решайте сами, как вам удобнее хранить пароли.

У меня распространены ситуации, когда я удаленно администрирую сервера, а на месте эникеи работают с пользователями. Чаще всего это не очень аккуратные и ответственные люди, иначе они бы работали с серверами :) Они часто забывают записать пароль, путают что-то и т.д. В итоге, когда один человек увольняется и приходит другой, оказывается, что найти пароли на некоторые ящики просто невозможно. Тут очень выручает возможность посмотреть пароль в базе. Я новому админу либо пароль говорю, либо весь дамп сразу отдаю, пусть работает.

Если вы сами работаете с сервером и все аккуратно ведете, записываете, например, в keepass все пароли от почтовых ящиков, то смело шифруйте все пароли, так будет спокойнее.

Последние 2 параметра **domain_path** и **domain_in_mailbox** указывайте по своему усмотрению. В файле конфигурации в комментариях расписано, за что они отвечают и в чем отличие. Мне кажется, удобно хранить директории именно в таком виде, как я указал. Получится следующий путь до ящика, если у вас архив почты будет жить, к примеру, в директории */mnt/mail* — /mnt/mail/kirushin-vladimir.ru/root@kirushin-vladimir.ru. Если доменов будет несколько, в такой иерархии удобно работать.

С параметрами разобрались. Сохраняем конфиг. Еще один маленький нюанс. Для работы панели управления нужна директория *templates_c* с правами на запись для web сервера. Сделаем ее и дадим права.

```
# mkdir /var/www/html/padmin/templates_c
# chown -R apache. /var/www/html/padmin/templates_c
```

Идем по адресу http://5.180.137.106/padmin/public/setup.php и начинаем установку postfixadmin. Первым делом идет проверка всех необходимых для установки и работы компонентов. Для продолжения установки у вас должна быть такая картинка.

Если все в порядке с настройками, сразу же начнется создание структуры базы данных. Из-за этого первая загрузка страницы будет длиться долго. Это нормально. В самом низу страницы будет интерфейс добавления администратора панели управления. Создайте его.

При первом добавлении учетной записи вы введете пароль для поля **Setup password**. После этого получите ошибку, так как хэш этого пароля не совпадает с тем, что вы ставили в конфиге postfixadmin ранее. Вам будет выведен правильный хэш. Добавьте его в конфиг config.local.php.

```
$CONF['setup_password'] = '555e082774c7b3584af0a09e45657532:43ce9c34ca00f681a002c2ea40354d7f1d68768a';
```

После этого добавляйте учетную запись администратора панели еще раз, используя тот же самый пароль установки, хэш от которого вы добавили. Попробуйте залогиниться под этой учетной записью по адресу *http://5.180.137.106/padmin/public/*.

Теперь нам нужно добавить домен в панель управления. Идем в раздел Список доменов -> Новый домен и добавляем свой домен.

При создании домена были добавлены стандартные алиасы, получателя для которых мы указали еще в конфиге — ящик root@kirushin-vladimir.ru. Создание таких алиасов требование стандартов, но по факту, кроме спама, вы скорее всего ничего не будете получать по этим адресам. Так что их создание оставляйте на свое усмотрения. Я обычно их не делаю, так как ящик для этих алиасов все равно не читаю.

Далее создадим почтовый ящик администратора — root@kirushin-vladimir.ru. Для этого идем в раздел *Обзор -> Создать* ящик и заполняем поля.

Вы получите ошибку, что невозможно отправить сообщение. Это нормально, так как непосредственно почтового сервера у нас еще нет, только панель управления для него. Сам ящик на диске создан тоже не будет, но запись в базе данных появится. Это можно проверить через phpmyadmin.

Как вы видите, пароль зашифрован. На этом установку и настройку postfixadmin завершаем. Интерфейс для управления почтовым сервером мы подготовили. Теперь можно заняться непосредственно настройкой postfix.

Настройка postfix

Сердце нашего почтового сервера на linux — postfix. В дистрибутиве centos 8 минимальной установки он по-умолчанию отсутствует. Так что сначала устанавливаем postfix.

```
# dnf install postfix postfix-mysql
```

Приводим конфиг postfix */etc/postfix/main.cf* к следующему виду.

```
soft_bounce = no
queue_directory = /var/spool/postfix
command_directory = /usr/sbin
daemon_directory = /usr/libexec/postfix
data_directory = /var/lib/postfix
mail_owner = postfix

myhostname = mail.kirushin-vladimir.ru
mydomain = kirushin-vladimir.ru
myorigin = $myhostname

inet_interfaces = all
inet_protocols = ipv4

mydestination = localhost.$mydomain, localhost
unknown_local_recipient_reject_code = 550
mynetworks = 127.0.0.0/8, 10.1.4.22/32

alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases

smtpd_banner = $myhostname ESMTP $mail_name

debug_peer_level = 2
# Строки с PATH и ddd должны быть с отступом в виде табуляции от начала строки
debugger_command =
    PATH=/bin:/usr/bin:/usr/local/bin:/usr/X11R6/bin
    ddd $daemon_directory/$process_name $process_id & sleep 5

sendmail_path = /usr/sbin/sendmail.postfix
newaliases_path = /usr/bin/newaliases.postfix
mailq_path = /usr/bin/mailq.postfix
```

```
setgid_group = postdrop
html_directory = no
manpage_directory = /usr/share/man
sample_directory = /usr/share/doc/postfix-2.10.1/samples
readme_directory = /usr/share/doc/postfix-2.10.1/README_FILES

relay_domains = mysql:/etc/postfix/mysql/relay_domains.cf
virtual_alias_maps = mysql:/etc/postfix/mysql/virtual_alias_maps.cf,
 mysql:/etc/postfix/mysql/virtual_alias_domain_maps.cf
virtual_mailbox_domains = mysql:/etc/postfix/mysql/virtual_mailbox_domains.cf
virtual_mailbox_maps = mysql:/etc/postfix/mysql/virtual_mailbox_maps.cf

smtpd_discard_ehlo_keywords = etrn, silent-discard
smtpd_forbidden_commands = CONNECT GET POST
broken_sasl_auth_clients = yes
smtpd_delay_reject = yes
smtpd_helo_required = yes
smtp_always_send_ehlo = yes
disable_vrfy_command = yes

smtpd_helo_restrictions = permit_mynetworks,
 permit_sasl_authenticated,
 reject_non_fqdn_helo_hostname,
 reject_invalid_helo_hostname

smtpd_data_restrictions = permit_mynetworks,
 permit_sasl_authenticated,
 reject_unauth_pipelining,
 reject_multi_recipient_bounce,

smtpd_sender_restrictions = permit_mynetworks,
 permit_sasl_authenticated,
 reject_non_fqdn_sender,
```

```
 reject_unknown_sender_domain

smtpd_recipient_restrictions = permit_mynetworks,
 permit_sasl_authenticated,
 reject_non_fqdn_recipient,
 reject_unknown_recipient_domain,
 reject_multi_recipient_bounce,
 reject_unauth_destination,

smtp_tls_security_level = may
smtp_tls_loglevel = 1
smtpd_tls_security_level = may
smtpd_tls_loglevel = 1
smtpd_tls_received_header = yes
smtpd_tls_session_cache_timeout = 3600s
smtp_tls_session_cache_database = btree:$data_directory/smtp_tls_session_cache
smtpd_tls_key_file = /etc/postfix/certs/key.pem
smtpd_tls_cert_file = /etc/postfix/certs/cert.pem
tls_random_source = dev:/dev/urandom
smtpd_tls_mandatory_ciphers = low
smtpd_tls_ciphers = low
smtpd_tls_mandatory_protocols = !SSLv2,!SSLv3
smtp_tls_mandatory_protocols  = !SSLv2,!SSLv3
smtp_tls_ciphers = low
smtp_tls_mandatory_ciphers = low
smtp_tls_protocols = !SSLv2,!SSLv3
smtp_tls_policy_maps = hash:/etc/postfix/tls_policy_maps
# фиксировать в логе имена серверов, выдающих сообщение STARTTLS, поддержка TLS для которых не включена
smtp_tls_note_starttls_offer = yes

# Ограничение максимального размера письма в байтах
message_size_limit = 20000000
smtpd_soft_error_limit = 10
```

```
smtpd_hard_error_limit = 15
smtpd_error_sleep_time = 20
anvil_rate_time_unit = 60s
smtpd_client_connection_count_limit = 20
smtpd_client_connection_rate_limit = 30
smtpd_client_message_rate_limit = 30
smtpd_client_event_limit_exceptions = 127.0.0.0/8
smtpd_client_connection_limit_exceptions = 127.0.0.0/8

maximal_queue_lifetime = 1d
bounce_queue_lifetime = 1d

smtpd_sasl_auth_enable = yes
smtpd_sasl_security_options = noanonymous
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/dovecot-auth

# Директория для хранения почты
virtual_mailbox_base = /mnt/mail
virtual_minimum_uid = 1000
virtual_uid_maps = static:1000
virtual_gid_maps = static:1000
virtual_transport = dovecot
dovecot_destination_recipient_limit = 1

sender_bcc_maps = hash:/etc/postfix/sender_bcc_maps
recipient_bcc_maps = hash:/etc/postfix/recipient_bcc_maps

compatibility_level=2
```

Я выделил жирным имя домена и путь для директории с почтовыми ящиками. Не забудьте поменять эти параметры на свои. Сохраняем конфиг и продолжаем настройку. В таком виде сервер еще не готов. Нужно теперь создать все то, что описано в файле конфигурации. Создаем папку для файлов с

конфигурацией подключения к mysql и сами файлы подключения.

```
# mkdir /etc/postfix/mysql && cd /etc/postfix/mysql
```

```
# mcedit relay_domains.cf

hosts = localhost
user = postfix
password = 123
dbname = postfix
query = SELECT domain FROM domain WHERE domain='%s' and backupmx = '1'
```

```
# mcedit  virtual_alias_domain_maps.cf

hosts = localhost
user = postfix
password = 123
dbname = postfix
query = SELECT goto FROM alias,alias_domain WHERE alias_domain.alias_domain = '%d' and alias.address = CONCAT('%u', '@',
alias_domain.target_domain) AND alias.active = 1
```

```
# mcedit virtual_alias_maps.cf

hosts = localhost
user = postfix
password = 123
dbname = postfix
query = SELECT goto FROM alias WHERE address='%s' AND active = '1'
```

```
# mcedit virtual_mailbox_domains.cf
```

```
hosts = localhost
user = postfix
password = 123
dbname = postfix
query = SELECT domain FROM domain WHERE domain='%s' AND backupmx = '0' AND active = '1'
```

```
# mcedit virtual_mailbox_maps.cf

hosts = localhost
user = postfix
password = 123
dbname = postfix
query = SELECT maildir FROM mailbox WHERE username='%s' AND active = '1'
```

Создадим файл tls_policy_maps, с помощью которого можно вручную задавать версию tls протокола для общения серверов, либо полностью отключать его. Это иногда бывает нужно, если подключаешься к очень старому серверу, который не поддерживает современные протоколы. Приходится использовать нешифрованное подключения.

```
# mcedit /etc/postfix/tls_policy_maps
```

```
93.175.56.122           none
77.221.87.91            encrypt protocols=TLSv1
```

```
# postmap /etc/postfix/tls_policy_maps
```

Так же не стоит забывать, что в современных ОС зачастую на уровне всей системы запрещено использование старых протоколов шифрования, таких как tls 1.0 или ssl3. Чтобы разрешить их, недостаточно настроек почтового сервера. Надо менять системные настройки. Как это сделать я рассказываю в отдельной статье — Centos 8 и TLS 1.0 и 1.1.

Продолжаем настройку почтового сервера postfix. Редактируем файл */etc/postfix/master.cf*. Нам надо добавить строки, касающиеся настройки Submission

для того, чтобы почтовый сервер работал на 587 порту. Смартфоны очень часто при настройке используют этот порт по-умолчанию, где-то даже без возможности изменить эту настройку. Приводим секцию, отвечающую за эту работу к следующему виду.

```
submission inet n       -       n       -       -       smtpd
  -o syslog_name=postfix/submission
  -o smtpd_tls_security_level=encrypt
  -o smtpd_sasl_auth_enable=yes
  -o smtpd_tls_auth_only=yes
  -o smtpd_reject_unlisted_recipient=no
  -o smtpd_recipient_restrictions=
  -o smtpd_relay_restrictions=permit_sasl_authenticated,reject
  -o milter_macro_daemon_name=ORIGINATING
```

Обращаю внимание на пробел в начале строки, начиная со второй. Его надо обязательно оставить. Добавляем еще настройки для того, чтобы наш сервер поддерживал протокол SSL/TLS и слушал порт 465

```
smtps inet n - n - - smtpd
 -o syslog_name=postfix/smtps
 -o smtpd_tls_wrappermode=yes
 -o smtpd_sasl_auth_enable=yes
 -o smtpd_recipient_restrictions=permit_mynetworks,permit_sasl_authenticated,reject
 -o smtpd_relay_restrictions=permit_mynetworks,permit_sasl_authenticated,defer_unauth_destination
 -o milter_macro_daemon_name=ORIGINATING
```

В этот же файл добавляем еще одну настройку, которая будет указывать postfix, что доставкой почты у нас будет заниматься dovecot, который мы настроим следом. Добавляем в master.cf в самый конец.

```
dovecot unix - n n - - pipe
 flags=DRhu user=vmail:vmail argv=/usr/libexec/dovecot/deliver -f ${sender} -d ${recipient}
```

Сгенерируем самоподписанные ssl сертификаты для нашего почтового сервера. Позже отдельным пунктом я расскажу как использовать полноценные

сертификаты. Они не всем нужны, поэтому показываю быструю настройку postfix на использование своих сертификатов, которые уже указаны в конфиге postfix.

Создаем директорию и сами сертификаты:

```
# mkdir /etc/postfix/certs
# openssl req -new -x509 -days 3650 -nodes -out /etc/postfix/certs/cert.pem -keyout /etc/postfix/certs/key.pem
```

Для генерации вам зададут несколько вопросов по поводу данных о сертификате. В принципе, можете там писать все, что угодно. Вот мои данные.

Создадим файлы для информации о ящиках, куда будет собираться вся входящая и исходящая почта.

```
# mcedit /etc/postfix/recipient_bcc_maps
```

```
@kirushin-vladimir.ru all_in@kirushin-vladimir.ru
```

```
# mcedit /etc/postfix/sender_bcc_maps
```

```
@kirushin-vladimir.ru all_out@kirushin-vladimir.ru
```

Создаем индексированные базы данных из этих файлов. Это нужно делать каждый раз, после изменения.

```
# postmap /etc/postfix/recipient_bcc_maps /etc/postfix/sender_bcc_maps
```

Теперь создайте два почтовых ящика all_in@kirushin-vladimir.ru и all_out@kirushin-vladimir.ru через postfixadmin.

Немного поясню по этим ящикам — для чего они нужны. Изначально я их делал, когда пользователи использовали протокол pop3 без сохранения писем на сервере. Это позволяло организовать бэкап всей переписки. Эти ящики очень быстро заполняются и занимают огромный объем, поэтому их обязательно

надо чистить. Я просто скриптами регулярно собирал всю почту в архивы с именами в виде дат. Если нужно было какое-то письмо найти, то просто распаковывал нужный архив. Пример прямого поиска писем по дате я показывал в статье про обслуживание почтовой базы.

В случае с imap роль бэкапа отпадает, так как вся почта хранится на сервере. Но эти ящики все равно бывают полезны, когда пользователь, к примеру, удалил какое-то важное письмо и потом делает вид, что его и не было. Если это письмо пришло только сегодня и еще не успело улететь в бэкап, то кроме записи в логах об этом письме, вы не увидите само содержимое. А с такими ящиками все сразу будет понятно, и вопросы отпадут. Последнее применение — служба безопасности. Если у вас есть кто-то, кому положено читать всю переписку, то реализовать этот функционал можно таким простым способом.

Все основные настройки для postfix мы сделали. Некоторые из них завязаны на работу с dovecot, который мы еще не настроили. Поэтому больше postfix не трогаем, не перезапускаем. Идем настраивать dovecot — imap сервер нашей почтовой системы.

Настройка dovecot

Займемся настройкой dovecot — сервер доставки почты пользователю по протоколам pop3 и imap. Я не вижу причин использовать pop3. Он неудобен по сравнению с imap. Чаще всего pop3 отключаю вовсе. Но это уже на ваше усмотрение. Приведу пример с настройкой обоих протоколов. Помимо основного функционала по доставке почты, я настрою несколько полезных плагинов. Расскажу о них поподробнее:

- **Sieve** — выполняет фильтрацию почты по заданным правилам в момент локальной доставки на почтовом сервере. Удобство такого подхода в том, что вы один раз можете настроить правило сортировки, и оно будет работать во всех клиентах, которыми вы будете получать почту по imap. Правила создаются, хранятся и исполняются на самом сервере.
- **Acl** — позволяет пользователям расшаривать папки в своем почтовом ящике и предоставлять доступ к этим папкам другим пользователям. Не часто видел, чтобы этот функционал настраивали и использовали. Думаю, просто по незнанию. По мне так очень удобный и полезный функционал.

Часто вижу, что люди настраивают плагин **quota**, который позволяет ограничивать максимальный размер почтового ящика. Я лично в своей работе его не использую. Возможно, когда у тебя клиентов сотни и тысячи это имеет значение и надо обязательно настроить ограничение. Когда же ящиков меньше, нет смысла напрягать людей постоянной чисткой. Сейчас диски стоят не так дорого. Мне кажется, проще и дешевле увеличить место на сервере, нежели постоянно беспокоить пользователей необходимостью чистки ящика. Лучше ограничить максимальный размер письма, скажем 20-ю мегабайтами. Тогда сильно забить ящик даже при большом желании быстро не получится. А почта все-таки важный инструмент в работе. Мне кажется, ее лучше хранить как можно дольше.

Есть еще один полезный плагин **expire**, который позволяет удалять устаревшие письма в определенных папках. Например, удалять все письма старше 30-ти дней в корзине и папке со спамом. Но реально пользоваться им не получается по простой причине. Разные почтовые клиенты создают различные папки для корзины и спама. Thunderbird создает папки с латинскими именами trash и spam, outlook с русскими, которые на почтовом сервере преобразуются в

кодировку UTF7, мобильные клиенты тоже используют разные имена папок. В итоге нет единообразия, плагин полноценно не работает.

Я рассказал об этих плагинах для наводки. Сам их не настраиваю, но если вам захочется реализовать описанный функционал, можете сами разобраться и настроить.

Небольшую теорию я дал, теперь переходим к практике. Устанавливаем необходимые для dovecot пакеты.

```
# dnf install dovecot dovecot-mysql dovecot-pigeonhole
```

Изначально конфиг dovecot разбит на отдельные сегменты и лежат они в директории */etc/dovecot/conf.d*. Каждый файл — отдельный функционал. Мне не нравится прыгать по файлам, поэтому я храню все в едином общем файле конфигурации */etc/dovecot/dovecot.conf*. С ним мы и будем работать. Обращаю внимание, что это только вопрос удобства. Можете оставить все файлы отдельности, если вам так привычнее. Приводим конфиг к следующему виду.

```
listen = *

mail_plugins = mailbox_alias acl
protocols = imap pop3 sieve lmtp

mail_uid = 1000
mail_gid = 1000

first_valid_uid = 1000
last_valid_uid = 1000

auth_verbose = yes
log_path = /var/log/dovecot/main.log
info_log_path = /var/log/dovecot/info.log
debug_log_path = /var/log/dovecot/debug.log

ssl_protocols = !SSLv3
ssl = required
verbose_ssl = yes
```

```
ssl_cert = </etc/postfix/certs/cert.pem
ssl_key = </etc/postfix/certs/key.pem

ssl_cipher_list = ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-
GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-
SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-
SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-
AES256-SHA:DHE-RSA-AES256-SHA:ECDHE-RSA-DES-CBC3-SHA:ECDHE-ECDSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-
SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:AES:CAMELLIA:DES-CBC3-SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!MD5:!PSK:!aECDH:!EDH-
DSS-DES-CBC3-SHA:!EDH-RSA-DES-CBC3-SHA:!KRB5-DES-CBC3-SHA
ssl_prefer_server_ciphers = yes

disable_plaintext_auth = no

mail_location = maildir:/mnt/mail/%d/%u/

auth_default_realm = kirushin-vladimir.ru

auth_mechanisms = PLAIN LOGIN

service auth {
 unix_listener /var/spool/postfix/private/dovecot-auth {
 user = postfix
 group = postfix
 mode = 0666
 }
unix_listener auth-master {
 user = vmail
 group = vmail
 mode = 0666
 }

unix_listener auth-userdb {
```

```
 user = vmail
 group = vmail
 mode = 0660
 }
}

service lmtp {
 unix_listener /var/spool/postfix/private/dovecot-lmtp {
 user = postfix
 group = postfix
 mode = 0600
 }

 inet_listener lmtp {
 address = 127.0.0.1
 port = 24
 }
}

userdb {
 args = /etc/dovecot/dovecot-mysql.conf
 driver = sql
 }

passdb {
 args = /etc/dovecot/dovecot-mysql.conf
 driver = sql
 }

auth_master_user_separator = *

plugin {
 auth_socket_path = /var/run/dovecot/auth-master
```

```
 acl = vfile
 acl_shared_dict = file:/mnt/mail/shared-folders/shared-mailboxes.db
 sieve_dir = ~/.sieve/
 mailbox_alias_old = Sent
 mailbox_alias_new = Sent Messages
 mailbox_alias_old2 = Sent
 mailbox_alias_new2 = Sent Items
}

protocol lda {
 mail_plugins = $mail_plugins sieve
 auth_socket_path = /var/run/dovecot/auth-master
 deliver_log_format = mail from %f: msgid=%m %$
 log_path = /var/log/dovecot/lda-errors.log
 info_log_path = /var/log/dovecot/lda-deliver.log
 lda_mailbox_autocreate = yes
 lda_mailbox_autosubscribe = yes
# postmaster_address = root
}

protocol lmtp {
 info_log_path = /var/log/dovecot/lmtp.log
 mail_plugins = quota sieve
 postmaster_address = postmaster
 lmtp_save_to_detail_mailbox = yes
 recipient_delimiter = +
}

protocol imap {
 mail_plugins = $mail_plugins imap_acl
 imap_client_workarounds = tb-extra-mailbox-sep
 mail_max_userip_connections = 30
}
```

```
protocol pop3 {
 mail_plugins = $mail_plugins
 pop3_client_workarounds = outlook-no-nuls oe-ns-eoh
 pop3_uidl_format = %08Xu%08Xv
 mail_max_userip_connections = 30
}

service imap-login {
 service_count = 1
 process_limit = 500
 }

service pop3-login {
 service_count = 1
 }

service managesieve-login {
 inet_listener sieve {
 port = 4190
 }
}

service stats {
    unix_listener stats-reader {
        user = vmail
        group = vmail
        mode = 0660
    }

    unix_listener stats-writer {
        user = vmail
        group = vmail
        mode = 0660
```

```
    }
}

namespace {
 type = private
 separator = /
 prefix =
 inbox = yes

 mailbox Sent {
 auto = subscribe
 special_use = \Sent
 }
 mailbox "Sent Messages" {
 auto = no
 special_use = \Sent
 }
 mailbox "Sent Items" {
 auto = no
 special_use = \Sent
 }
 mailbox Drafts {
 auto = subscribe
 special_use = \Drafts
 }
 mailbox Trash {
 auto = subscribe
 special_use = \Trash
 }
 mailbox "Deleted Messages" {
 auto = no
 special_use = \Trash
 }
```

```
 mailbox Junk {
 auto = subscribe
 special_use = \Junk
 }
 mailbox Spam {
 auto = no
 special_use = \Junk
 }
 mailbox "Junk E-mail" {
 auto = no
 special_use = \Junk
 }
 mailbox Archive {
 auto = no
 special_use = \Archive
 }
 mailbox Archives {
 auto = no
 special_use = \Archive
 }
}

namespace {
 type = shared
 separator = /
 prefix = Shared/%%u/
 location = maildir:%%h:INDEX=%h/shared/%%u
 subscriptions = yes
 list = children
}
```

Обращаю внимание на параметры *ssl_protocols = !SSLv2 !SSLv3* и *ssl = required*. Они актуальны до версии Dovecot v2.2. В версии 2.3 и выше будет ошибка

на этот параметр. Его надо будет заменить на *ssl_min_protocol*. В официальных репах centos как обычно старый софт, но возможно через некоторое время появится новая версия и конфиг надо будет поправить. Подробнее в документации dovecot.

Создаем группу и пользователя с указанными в конфиге uid 1000.

```
# groupadd  -g 1000 vmail
# useradd -d /mnt/mail/ -g 1000 -u 1000 vmail
# chown vmail. /mnt/mail
```

Создаем конфигурационные файлы для доступа к mysql базе.

```
# mcedit /etc/dovecot/dovecot-mysql.conf
```

```
driver = mysql
default_pass_scheme = CRYPT
connect = host=127.0.0.1 dbname=postfix user=postfix password=123
user_query = SELECT '/mnt/mail/%d/%u' as home, 'maildir:/mnt/mail/%d/%u' as mail, 1000 AS uid, 1000 AS gid,
concat('*:bytes=', quota) AS quota_rule FROM mailbox WHERE username = '%u' AND active = '1'
password_query = SELECT username as user, password, '/mnt/mail/%d/%u' as userdb_home, 'maildir:/mnt/mail/%d/%u' as
userdb_mail, 1000 as userdb_uid, 1000 as userdb_gid, concat('*:bytes=', quota) AS userdb_quota_rule FROM mailbox WHERE
username = '%u' AND active = '1'
```

Обращаю внимание на параметр шифрования паролей. Если пароли не зашифрованы, то параметр будет **PLAIN**.

Создадим директорию и файлы для логов.

```
# mkdir /var/log/dovecot
# cd /var/log/dovecot && touch main.log info.log debug.log lda-errors.log lda-deliver.log lmtp.log
# chown -R vmail:dovecot /var/log/dovecot
```

Создаем служебную папку для плагина acl.

```
# mkdir /mnt/mail/shared-folders
# chown -R vmail. /mnt/mail
```

На этом основная настройка почтового сервера на базе postfix и dovecot завершена. Можно запускать службы и проверять работу системы.

```
# systemctl restart postfix
# systemctl start dovecot
# systemctl enable postfix
# systemctl enable dovecot
```

Обе службы должны запуститься без ошибок. Если это не так, то разбирайтесь в чем проблема. Скорее всего где-то ошиблись в конфигах.

Проверка работы почтового сервера

Самый простой и быстрый способ проверить работу почтового сервера — отправить на него письмо. Я буду отправлять со своего почтового адреса zeroxzed@gmail.com на адрес root@kirushin-vladimir.ru. Вот что должно быть в логе, если у вас все правильно настроено и почтовый сервер нормально работает.

```
# cat /var/log/maillog
```

```
Feb 10 16:07:21 mail postfix/smtpd[12646]: connect from mail-ua1-f44.google.com[209.85.222.44]
Feb 10 16:07:22 mail postfix/smtpd[12646]: Anonymous TLS connection established from mail-ua1-f44.google.com[209.85.222.44]:
TLSv1.3 with cipher TLS_AES_128_GCM_SHA256 (128/128 bits)
Feb 10 16:07:22 mail postfix/smtpd[12646]: 7CB35120E58: client=mail-ua1-f44.google.com[209.85.222.44]
Feb 10 16:07:22 mail postfix/cleanup[12651]: 7CB35120E58: message-id=<CAHWPLcMu2=VCy_ZmNJH4QG1NeR=EoxcFaxjaCvFXcX+-3Ko-
fw@mail.gmail.com>
Feb 10 16:07:22 mail postfix/qmgr[12645]: 7CB35120E58: from=<zeroxzed@gmail.com>, size=4233, nrcpt=2 (queue active)
Feb 10 16:07:22 mail postfix/smtpd[12646]: disconnect from mail-ua1-f44.google.com[209.85.222.44] ehlo=2 starttls=1 mail=1
rcpt=1 data=1 quit=1 commands=7
Feb 10 16:07:22 mail postfix/pipe[12652]: 7CB35120E58: to=<all_in@kirushin-vladimir.ru>, relay=dovecot, delay=0.45,
```

```
delays=0.29/0.02/0/0.14, dsn=2.0.0, status=sent (delivered via dovecot service)
Feb 10 16:07:22 mail postfix/pipe[12653]: 7CB35120E58: to=<root@kirushin-vladimir.ru>, relay=dovecot, delay=0.47,
delays=0.29/0.03/0/0.14, dsn=2.0.0, status=sent (delivered via dovecot service)
Feb 10 16:07:22 mail postfix/qmgr[12645]: 7CB35120E58: removed
```

Пояснять тут нечего, по логу все понятно. Было подключение от сервера mail-ua1-f44.google.com[209.85.222.44] с использованием протокола шифрования и шифров TLSv1.3 with cipher TLS_AES_128_GCM_SHA256.

Письмо было доставлено в указанный ящик и в общий ящик для сбора всей входящей почты. В директории */mnt/mail* была создана директория с именем домена kirushin-vladimir.ru, а в ней созданы 2 папки с именами ящиков:

- all_in@kirushin-vladimir.ru
- root@kirushin-vladimir.ru

Директории с почтовыми ящиками создаются в момент получения первого письма в ящик. Непрочитанное письмо помещается в директорию /new в почтовом ящике. После прочтения переносится в /cur.

Попробуем теперь подключиться к почтовому ящику по imap, прочитать письмо и отправить ответ. Настроим любой почтовый клиент для проверки работы настроенного почтового сервера. Я для этих целей буду использовать Thunderbird. Из всех почтовых клиентов мне он нравится больше всего. В основном из-за его портированной версии. Указываем следующие настройки.

Так как мы используем самоподписанный сертификат ssl, почтовый клиент предупредит нас о том, что серверу нельзя доверять.

Нас это не пугает, добавляем сертификат в список доверенных и продолжаем работать. Позже получим и настроим нормальный сертификат. На этом этапе чаще всего возникают проблемы. Если не получается подключиться к ящику, смотрите логи postfix и dovecot. Можете показать ошибки в комментариях, я подскажу, в какую сторону смотреть.

Я подключился к почтовому ящику и увидел тестовые письма. Отвечу на одно из них и посмотрю в логе, как прошла отправка. У меня еще раз выскочило окно с предупреждением о небезопасном сертификате. Еще раз добавляю его в исключения. Это нормально, сертификат проверяется во время получения

почты в dovecot, а во время отправки в postfix. Так что нужны 2 подтверждения. Отправляю письмо еще раз и смотрю лог.

```
# cat /var/log/maillog
```

```
Feb 10 17:26:05 mail postfix/submission/smtpd[13896]: connect from unknown[176.118.159.81]
Feb 10 17:26:05 mail postfix/submission/smtpd[13896]: Anonymous TLS connection established from unknown[176.119.158.81]:
TLSv1.3 with cipher TLS_AES_128_GCM_SHA256 (128/128 bits)
Feb 10 17:26:06 mail postfix/submission/smtpd[13896]: 06F62120E56: client=unknown[176.118.159.81], sasl_method=PLAIN,
sasl_username=root@kirushin-vladimir.ru
Feb 10 17:26:06 mail postfix/cleanup[13901]: 06F62120E56: message-id=<a8d829a4-8e24-347b-0181-fe534ac9039f@kirushin-
vladimir.ru>
Feb 10 17:26:06 mail postfix/qmgr[13803]: 06F62120E56: from=<root@kirushin-vladimir.ru>, size=3333, nrcpt=2 (queue active)
Feb 10 17:26:06 mail postfix/submission/smtpd[13896]: disconnect from unknown[176.118.159.81] ehlo=2 starttls=1 auth=1
mail=1 rcpt=1 data=1 quit=1 commands=8
Feb 10 17:26:06 mail postfix/pipe[13904]: 06F62120E56: to=<all_out@kirushin-vladimir.ru>, relay=dovecot, delay=0.22,
delays=0.07/0.02/0/0.13, dsn=2.0.0, status=sent (delivered via dovecot service)
Feb 10 17:26:06 mail postfix/smtp[13905]: Untrusted TLS connection established to gmail-smtp-
in.l.google.com[64.233.162.26]:25: TLSv1.3 with cipher TLS_AES_256_GCM_SHA384 (256/256 bits)
Feb 10 17:26:06 mail postfix/smtp[13905]: 06F62120E56: to=<zeroxzed@gmail.com>, relay=gmail-smtp-
in.l.google.com[64.233.162.26]:25, delay=0.87, delays=0.07/0.03/0.42/0.35, dsn=2.0.0, status=sent (250 2.0.0 OK  1581344766
w2si182048lfl.121 - gsmtp)
Feb 10 17:26:06 mail postfix/qmgr[13803]: 06F62120E56: removed
```

Все в порядке. Видно подключение с моего ip, успешную sasl авторизацию, формирование письма на сервере, присваивание ему message-id, отправка копии письма в ящик для сбора исходящей почты и отправка оригинала в ящик получателя. Все этапы прошли без ошибок.

Расскажу, куда еще надо смотреть для отладки почтовой системы. Да и не только отладки, во время работы периодически придется разбираться, куда ушло то или иное письмо, кто и когда подключался к ящику. Разные ситуации бывают. В файле */var/log/dovecot/lda-deliver.log* содержится информация обо всех пришедших письмах — когда, от кого и в какой ящик было положено.

```
Feb 10 16:13:30 lda(root@kirushin-vladimir.ru): Info: mail from zeroxzed@gmail.com: msgid=<CAHWPLcNe-
a7J3EqDOHJv0iD73NBRfUHNbTwPuZ3RChH=giHtHA@mail.gmail.com> saved mail to INBOX
```

```
Feb 10 17:26:06 lda(all_out@kirushin-vladimir.ru): Info: mail from root@kirushin-vladimir.ru:
msgid=<a8d829a4-8e24-347b-0181-fe534ac9039f@kirushin-vladimir.ru> saved mail to INBOX
```

В */var/log/dovecot/info.log* информация о подключениях к почтовым ящикам — кто, когда, откуда и каким способом авторизовывался на сервере.

```
Feb 10 17:26:06 imap-login: Info: Login: user=<root@kirushin-vladimir.ru>, method=PLAIN, rip=176.118.159.81,
lip=5.180.137.106, mpid=13908, TLS, session=<l20RhTmeXAawd55R>
Feb 10 17:26:06 imap(root@kirushin-vladimir.ru): Info: Logged out in=3131 out=590
Feb 10 17:26:13 imap-login: Info: Login: user=<root@kirushin-vladimir.ru>, method=PLAIN, rip=176.118.159.81,
lip=5.180.137.106, mpid=13910, TLS, session=<ExF+hTmeXwawd55R>
```

Остальное уже не так полезно. Сами посмотрите, что собирается в остальных лог файлах.

На текущий момент почтовый сервер на базе postfix работоспособен. В таком виде им без проблем можно пользоваться. Но функционал полностью не раскрыт. Использовать плагины sieve и acl через удаленные почтовые клиенты неудобно. Проще всего их настроить через web почту roundcube. Установим эту web панель на наш почтовый сервер.

Установка web интерфейса roundcube

Установим и настроим самый популярный web интерфейс для postfix — **roundcube**. Вообще говоря, его не обязательно ставить на почтовый сервер. Более того, я бы рекомендовал его ставить в другое место. Если в инфраструктуре имеется отдельный веб сервер, лучше поставить roundcube туда. Но в рамках этой статьи, я все буду ставить на один сервер для простоты и наглядности. Если у вас все получится на одном сервере, потом сможете не спеша все разнести по разным/

Скачиваем исходники последней lts версии roundcube. Я обычно использую именно lts версию, хотя она достаточно старая. Но моя практика показывает, что именно в ней все работает стабильно и сразу, без лишней возни. К примеру, когда я готовил эту статью, у меня не получилось настроить работу автоответчика, о котором я расскажу отдельно ниже. Я его включал, а он просто не работает. За разумный срок (1 час) я не смог выяснить причину этого и опять откатился на lts версию, где все заработало сразу.

Если есть желание, можете попробовать последнюю версию. Может все, что вам надо, заработает. Но в целом, там во всех версиях все примерно одинаковое. В новых версиях есть адаптивный скин, но лично мне он все равно не понравился, я использую стандартный larry.

```
# cd ~
# wget https://github.com/roundcube/roundcubemail/releases/download/1.2.9/roundcubemail-1.2.9-complete.tar.gz
```

Распаковываем и перемещаем на веб-сервер.

```
# tar xzvf roundcubemail-1.2.9-complete.tar.gz
# mkdir /var/www/html/webmail
# cp -R /root/roundcubemail-1.2.9/* /var/www/html/webmail
# chown -R apache. /var/www/html/webmail
```

Не забудьте проверить в момент установки номер актуальной версии roundcube и заменить его в приведенных выше командах.

Устанавливаем дополнительные модули php.

```
# dnf install php-pear php-mcrypt php-intl php-ldap php-pear-Net-SMTP php-pear-Net-Sieve php-pear-Mail-Mime php-pear-Net-IDNA2
```

Последние три у меня почему-то не установились из репозитория Remi, хотя они там есть. Какой-то нюанс с работой новых репозиториев в Centos 8, с которым я еще не разобрался. Как разберусь полностью, сделаю отдельную статью по этому поводу. Пока просто напрямую их установил. Если кто-то понимает, почему они не ставятся обычным образом, прошу подсказать. Я не стал разбираться, чтобы не затягивать публикацию статьи.

```
# dnf install http://rpms.remirepo.net/enterprise/8/remi/x86_64/php-pear-Net-Sieve-1.4.4-1.el8.remi.noarch.rpm
# dnf install http://rpms.remirepo.net/enterprise/8/remi/x86_64/php-pear-Mail-Mime-1.10.6-1.el8.remi.noarch.rpm
# dnf install http://rpms.remirepo.net/enterprise/8/remi/x86_64/php-pear-Net-IDNA2-0.2.0-1.el8.remi.noarch.rpm
```

Указываем часовой пояс в php. Добавляем в */etc/php.ini*:

```
date.timezone = Europe/Moscow
```

Перезапускаем apache и php-fpm.

```
# systemctl restart httpd php-fpm
```

Идем по адресу http://5.180.137.106/webmail/installer/ и проверяем окружение. Вы увидите несколько незначительных замечаний. На них можно не обращать внимание, если установщик позволяет нажать кнопку **NEXT**.

На следующем этапе нам надо указать настройки подключения к mysql базе. Предварительно ее следует создать через phpmyadmin. Я создал пользователя roundcube и такую же базу с полными правами пользователя на нее. Эти параметры указал в настройках.

Так же на этой странице нужно будет указать несколько параметров:

- smtp_server — localhost
- smtp_port — 25
- Use the current IMAP username and password for SMTP authentication — галочка должна стоять
- language — ru_RU
- skin — larry (мне он просто больше нравится, потом можно поменять)
- Выбираем плагины — acl, managesieve, userinfo. Остальные на свое усмотрение.

Жмем **CREATE CONFIG**. Должны увидеть сообщение:

```
The config file was saved successfully into RCMAIL_CONFIG_DIR directory of your Roundcube installation.
```

Жмем **CONTINUE**. Открывается страница с проверкой настроек. Нам нужно инициализировать базу данных. Для этого жмите на соответствующую кнопку.

Тут проверять отправку почты неудобно, можно этого не делать. Зайдем в почтовый ящик и там все проверим. Если что, конфиг потом все равно можно вручную отредактировать. Папку */var/www/html/webmail/installer* удаляем. Заходим в почтовый ящик через roundcube — http://5.180.137.106/webmail/ Набирать нужно полное имя ящика и пароль. Если все сделали правильно, должны попасть в свой почтовый ящик.

Можете попробовать написать и отправить письмо. В настройках можно изменить некоторые параметры web интерфейса. Если будут какие-то ошибки, можно посмотреть лог самого roundcube — */var/www/html/webmail/logs/errors.log* Он достаточно информативен. В этой же директории будет вестись лог всех отправленных писем через web интерфейс.

Настройка фильтра почты sieve

Sieve очень удобная штука, но вот хорошего интерфейса для управления через почтовый клиент я не знаю. Существует плагин для thunderbird, который так и называется sieve. Но лично мне он не понравился вообще, так как предлагает писать правила определенным кодом. Для этого надо знать синтаксис, тратить время. Можете сами на него посмотреть — https://github.com/thsmi/sieve.

К счастью, есть удобный способ писать правила фильтрации для sieve через roundcube. Там это реализовано отдельным плагином **managesieve**, который мы активировали во время установки. Его конфиг находится в директории */var/www/html/webmail/plugins/managesieve*. Скопируйте дефолтный конфиг и проследите, чтобы там были следующие настройки.

```
# cp config.inc.php.dist config.inc.php
```

```
$config['managesieve_port'] = 4190;
$config['managesieve_host'] = 'localhost';
$config['managesieve_usetls'] = false;
```

Для создания правила фильтрации, зайдите в свой почтовый ящик через roundcube. Переходите в раздел *Настройки -> Фильтры* и создавайте новое правило.

После этого письма будут обрабатываться правилом сразу после поступления в почтовый сервер, в независимости от вашего подключения к ящику. В ящике в папке *.sieve* появилась запись с настроенным правилом. Можете познакомиться с синтаксисом написания правил. Он не сложный.

Настройка автоответчика

В roundcube есть замечательная возможность настроить автоответчик в почтовом ящике. Это актуально, к примеру, если вы уходите в отпуск. Вы можете сами настроить автоответчик, который будет отправлять письмо с указанной вами информацией всем, от кого будут приходить письма в ваш ящик. Возможность эта реализована на базе того же плагина managesieve. По-умолчанию она отключена. Активировать ее нужно вручную.

Для того, чтобы модуль автоответчика заработал, отредактируйте конфигурационный файл плагина. Для этого открываем его в моем случае по следующему адресу:

```
# mcedit /var/www/html/webmail/plugins/managesieve/config.inc.php
```

Изменяем там параметр:

```
$config['managesieve_vacation'] = 1;
```

После этого достаточно просто обновить веб интерфейс roundcube, и появятся новые настройки по адресу *Настройки -> Отпуск.*

На вкладке *Дополнительные настройки* есть возможность настроить различные полезные действия, в том числе пересылку входящей почты вашему заместителю. По своей сети настройка автоответчика это просто создание еще одного правила sieve для всей входящей почты.

Общие папки по imap

Рассмотрим настройку необычного и полезного функционала в виде общих папок. С их помощью один пользователь почтового ящика может предоставить другому пользователю доступ к папке внутри своего почтового ящика. Где и как использовать этот функционал, каждый может придумать сам, в зависимости от своих потребностей. Мне кажется это удобным в том случае, когда создан какой-то общий ящик, на который только поступает информация и нет необходимости писать ответ от его имени. То есть по сути работает как обычный почтовый алиас. Но в случае с алиасом и несколькими почтовыми ящиками, письмо падает в каждый ящик. Если таких писем и получателей много, то идет большое дублирование одного и того же письма в рамках почтового сервера. Если сделать ящик и расшарить на нем папку, подключить ее всем пользователям, то дублирования почты не будет. Каждый сможет

прочитать письмо, без необходимости его доставки в каждый конкретный ящик.

Настроим общую папку imap. Хотя настраивать нам, по сути, нечего. Мы уже все настроили ранее. Добавили соответствующие настройки в dovecot и активировали плагин **acl** в roundcube. Теперь нужно просто сделать папку и открыть ее для другого пользователя. Для этого идем в раздел *Настройки -> Папки*. Создаем там любую папку. В моем случае это папка с названием Общая. После того, как создали папку, открываем ее еще раз.

Добавляем необходимый доступ либо всем ящикам в домене, либо какому-то конкретному. Так же можно указать, какого рода это будет доступ:

- чтение
- запись
- удаление

Заходим в ящик, которому добавили общий доступ и проверяем.

Все в порядке, общая папка imap настроена и подключена. В папке */mnt/mail/shared-folders* появился файл с настроенным выше правилом.

На этом настройка пользовательского функционала закончена. В принципе, почтовый сервер полностью готов к работе. Но мы сделаем еще несколько полезных настроек на стороне сервера.

Настройка DKIM

Напишу своими словами как я понимаю работу **dkim**. С помощью dkim вся исходящая почта сервера подписывается электронной цифровой подписью, связанной с именем домена. Открытый ключ шифрования с помощью DNS публикуется в txt записи. Таким образом, удаленный сервер, при получении письма от вас, сравнивает цифровую подпись с опубликованным в dns открытым ключом вашего домена. Если все в порядке, то считает, что ваше письмо в самом деле пришло от вас, а не от мошенников. То есть с помощью этой технологии можно однозначно идентифицировать отправителя.

Не путать эту технологию с защитой от спама. Защиты тут нет. Любой спамер так же может себе сделать dkim подпись. Спамеру не составит большого

труда настроить на своем сервере dkim и отправлять спам, но подписанный электронной цифровой подписью. Теоретически, dkim помогает защититься от подделки адреса отправителя, когда письмо якобы от вас шлет совсем другой сервер. Но с этим можно бороться и другими способами. В общем, я до конца не понимаю, зачем это надо. Я много лет эксплуатировал сервера без dkim подписей и проблем это не вызывало. Но так как настроить dkim не сложно, сейчас всегда это делаю.

Установка dkim на Centos 8 на момент написания статьи имеет некоторые сложности. Дело в том, что этого пакета до сих пор нет в стабильной версии репозитория Epel, где он обычно был для предыдущих версию. Хорошая новость в том, что он уже есть в тестовой ветке. Если будете настраивать по статье, то сначала попробуйте просто установить пакет через dnf из обычного репозитория epel. Если его там не будет, то ставьте из тестовой ветки, как я.

Для настройки dkim устанавливаем соответствующий пакет:

```
# dnf --enablerepo=epel-testing install opendkim
```

Создаем директорию для хранения ключей:

```
# mkdir -p /etc/postfix/dkim && cd /etc/postfix/dkim
```

Генерируем ключи для домена:

```
# opendkim-genkey -D /etc/postfix/dkim/ -d kirushin-vladimir.ru -s mail
```

kirushin-vladimir.ru	имя почтового домена
mail	непосредственно имя сервера

На выходе получаете пару файлов — закрытый (приватный) и открытый ключ. Закрытый останется на сервере, открытый будет опубликован в dns. Переименуем их сразу, чтобы не путаться, если у вас будет несколько доменов. Ключи нужно будет делать для каждого домена.

```
# mv mail.private mail.kirushin-vladimir.ru.private
# mv mail.txt mail.kirushin-vladimir.ru.txt
```

Создаем файл с таблицей ключей, в которой будут описаны все домены. В данном случае только один.

```
# mcedit keytable
```

```
mail._domainkey.kirushin-vladimir.ru kirushin-vladimir.ru:mail:/etc/postfix/dkim/mail.kirushin-vladimir.ru.private
```

Тут же создаем еще один файл, в котором будет описано, каким ключом подписывать письма каждого домена. У нас один домен, поэтому только одна запись.

```
# mcedit signingtable
```

```
*@kirushin-vladimir.ru mail._domainkey.kirushin-vladimir.ru
```

Выставляем права доступа на все файлы:

```
# chown root:opendkim *
# chmod u=rw,g=r,o= *
```

Рисуем конфиг службы.

```
# mcedit /etc/opendkim.conf
```

```
AutoRestart Yes
AutoRestartRate 10/1h
PidFile /var/run/opendkim/opendkim.pid
Mode sv
Syslog yes
SyslogSuccess yes
LogWhy yes
UserID opendkim:opendkim
```

```
Socket inet:8891@localhost
Umask 022
Canonicalization relaxed/relaxed
Selector default
MinimumKeyBits 1024
KeyFile /etc/postfix/dkim/mail.kirushin-vladimir.ru.private
KeyTable /etc/postfix/dkim/keytable
SigningTable refile:/etc/postfix/dkim/signingtable
```

Добавляем в конфигурационный файл postfix в самый конец следующие параметры:

```
# mcedit /etc/postfix/main.cf
```

```
smtpd_milters = inet:127.0.0.1:8891
non_smtpd_milters = $smtpd_milters
milter_default_action = accept
milter_protocol = 2
```

Перезапускаем postfix и dkim, последний добавляем в автозагрузку.

```
# systemctl restart postfix
# systemctl restart opendkim.service
# systemctl enable opendkim.service
```

Теперь нам надо добавить открытый ключ в dns. Идем в консоль управления dns и добавляем новую txt запись. Ее содержание берем из файла */etc/postfix/dkim/mail.kirushin-vladimir.ru.txt*

```
# cat /etc/postfix/dkim/mail.kirushin-vladimir.ru.txt
```

```
mail._domainkey IN        TXT        ( "v=DKIM1; k=rsa; "
```

```
"p=MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDRrqwnax3Slj2dkrrf3hk1X0pfSOWxPh67KYuz2lEwH/rEQARKs1W9bx/QTV5aQQ/pPEsJiEmVXbPjP8aYp
fmSg16FA61SuVJBy2Or+xwVGjLDVkGFm1Lu/lxfViWtPaMR5PH2xTceoQfLkF+93y99KfgzTQ0UjAZcyvcPLuYdqQIDAQAB" )  ; ----- DKIM key mail
for kirushin-vladimir.ru
```

Убираем кавычки, лишние проблемы и вставляем. Должно получиться вот так:

Проверяю работу. Отправляю письмо на gmail и смотрю лог почтового сервера:

```
# cat /var/log/maillog
```

```
Feb 11 12:20:38 mail postfix/smtpd[19615]: connect from localhost[127.0.0.1]
Feb 11 12:20:38 mail postfix/smtpd[19615]: 4C1A4120F65: client=localhost[127.0.0.1], sasl_method=LOGIN,
sasl_username=root@kirushin-vladimir.ru
Feb 11 12:20:38 mail postfix/cleanup[19623]: 4C1A4120F65: message-id=<9f2b1abb24603b73584b05d89011adf5@kirushin-vladimir.ru>
Feb 11 12:20:38 mail opendkim[19547]: 4C1A4120F65: DKIM-Signature field added (s=mail, d=kirushin-vladimir.ru)
Feb 11 12:20:38 mail postfix/qmgr[19542]: 4C1A4120F65: from=<root@kirushin-vladimir.ru>, size=685, nrcpt=2 (queue active)
Feb 11 12:20:38 mail postfix/pipe[19627]: 4C1A4120F65: to=<all_out@kirushin-vladimir.ru>, relay=dovecot, delay=0.23,
delays=0.15/0.02/0/0.06, dsn=2.0.0, status=sent (delivered via dovecot service)
Feb 11 12:20:38 mail postfix/smtpd[19615]: disconnect from localhost[127.0.0.1] ehlo=1 auth=1 mail=1 rcpt=1 data=1 quit=1
commands=6
Feb 11 12:20:38 mail postfix/smtp[19628]: Untrusted TLS connection established to gmail-smtp-
in.l.google.com[64.233.162.26]:25: TLSv1.3 with cipher TLS_AES_256_GCM_SHA384 (256/256 bits)
Feb 11 12:20:39 mail postfix/smtp[19628]: 4C1A4120F65: to=<zeroxzed@gmail.com>, relay=gmail-smtp-
in.l.google.com[64.233.162.26]:25, delay=1.5, delays=0.15/0.03/0.38/0.97, dsn=2.0.0, status=sent (250 2.0.0 OK  1581412839
y133si1852705lfc.170 - gsmtp)
Feb 11 12:20:39 mail postfix/qmgr[19542]: 4C1A4120F65: removed
```

Все в порядке, электронная цифровая подпись установлена. Проверим, как гугл отреагировал на нашу подпись:

Тоже все в порядке. Подпись выполнена корректно, проверку прошла. Дополнительно, проверить корректность dkim записи в dns можно онлайн сервисом — http://dkimcore.org/c/keycheck.

Настройка SPF

Настроим еще одно средство для повышения доверия к нашей почте со стороны других серверов — **spf**. Расскажу опять своими словами для чего это нужно. Spf запись добавляется в виде txt записи в dns вашего домена. С помощью этой записи вы указываете, какие ip адреса имеют право отправлять почту от вашего имени. Если кто-то из спамеров будет использовать ваше имя домена при рассылке спама, он не пройдет проверку по spf и скорее всего будет идентифицирован как спам.

Можно указать конкретные ip адреса в записи, а можно сказать, чтобы ip адреса проверялись по спискам A и MX записей. У нас простой случай и только 1 сервер с одним ip, поэтому укажем конкретно этот ip адрес. Идем в панель управления dns и добавляем новую txt запись.

```
kirushin-vladimir.ru. TXT v=spf1 ip4:5.180.137.106 ~all
```

Больше ничего делать не надо. Можно снова отправить письмо на gmail и проверить. Обращаю внимание, что на прошлом скрине в gmail уже было указано, что проверка spf прошла, хотя txt запись еще не была создана. Гугл умный. Думаю, он автоматом сопоставил все dns записи домена и сам убедился, что отправка идет с доверенного сервера, к которому привязана A запись и MX запись.

Но отправка может идти не только с почтового сервера. К примеру, может быть отдельно web сервер с интернет магазином. Он по каким-то причинам может отправлять почту сам (нет модуля для smtp отправки, не работает smtp авторизация, разработчики хотят использовать php_mail и т.д.), а не через настроенный почтовый сервер. Так часто бывает. Тогда нужно обязательно добавить в spf запись ip адрес этого web сервера, с которого будет идти отправка.

Настройка DMARC

Для настройки DMARC на самом почтовом сервере ничего делать не надо. По своей сути это просто указание другим, что делать с письмами от вас, не прошедшими проверки dkim и spf (которые являются подделками, если у вас все настроено правильно). Для этого сам принимающий почтовый сервер должен поддерживать работу в соответствии с dmark. Плюс, для вашего домена должны быть настроены правила, что делать в том или ином случае.

Есть три типа действий, которые можно настроить с помощью dmark:

1. Отклонить письмо.
2. Пометить письмо как спам.
3. Ничего не делать.

При этом можно настроить при каждом действии формирование отчета и отправку его на какой-то email адрес. Я очень осторожно отношусь к этим правилам и никогда не настраиваю блокировку или пометку о спаме. Так можно выстрелить себе в ногу и загубить всю свою почту из-за какой-нибудь ошибки. Мне кажется разумнее всего настроить пропускание всех писем, но сделать отправку отчетов на свой существующий адрес. Если вы заметите какую-то подозрительную активность по отчетам, то можно будет временно изменить настройки, чтобы помечать все поддельные письма, к примеру, как спам.

Указанные правила мы сейчас и добавим с помощью txt записи в DNS. Запись будет такая:

```
v=DMARC1; p=none; rua=mailto:dmarc@kirushin-vladimir.ru
```

Отчеты будут приходить в xml формате. Нужно будет еще потрудиться, чтобы в них разобраться :) В общем случае, я вообще не слежу за dmark. Думаю, это актуально только для крупных компаний, где есть отдельные люди, которые занимаются обслуживанием почты.

Дополнительный функционал почтового сервера postfix

Я рассмотрел и настроил наиболее актуальный с моей точки зрения функционал почтового сервера. В статье я основывался исключительно на своем опыте работы с почтой в малых и средних организациях, поэтому не претендую на истину в последней инстанции. Рекомендую осмысленно подходить к

настройке своего сервера и решать, что нужно именно вам. Будет хорошо, если кто-то укажет на мои ошибки или подскажет какие-то более удобные и эффективные приемы для решения затронутых задач.

В данном виде почтовый сервер представляет собой готовое и законченное решение, но есть еще несколько вещей, которые ему бы не помешали. Я их сейчас перечислю, а затем постараюсь постепенно писать статьи на указанные темы и ставить на них ссылки в этой теме. Вот список того, что по моему мнению нужно еще настроить на почтовом сервере:

1. Защиту от подбора паролей с помощью fail2ban.
2. Мониторинг почтового сервера postfix с помощью zabbix.
3. Сбор статистики с помощью pflogsumm или чего-то подобного.
4. Просмотр и анализ логов с помощью webmin.
5. Использование бесплатных сертификатов let’s encrypt.
6. Регулярную очистку служебных почтовых ящиков.
7. Бэкап всей почтовой базы.
8. Сбор логов с почтовых серверов в ELK Stack.

Расскажу еще почему я не настраиваю некоторые популярные программы, которые использую на почтовых серверах:

1. **Clamav** — известный антивирус. Считаю, что сейчас он не актуален, так как вирусов, от которых он способен защитить, я уже давно не видел. Сейчас вирусная эпидемия шифровальщиков. От них он не защищает.
2. **Spamassasin** — популярный бесплатный антиспам фильтр. Скажу честно, работал с ним очень мало и могу быть не объективен. Насколько я видел его настройку и работу — он требует к себе некоторого внимания, калибровки, особенно на начальном этапе. Мне обычно не хочется этим заниматься.
3. **Graylist** — эффективное средство борьбы со спамом. Я уже подробно его рассматривал, когда писал про iredmail, так что не буду повторяться. Скажу лишь, что режет спам очень эффективно и бесплатно, но есть существенные неудобства, которые по моему мнению не перекрывают плюсы. Поэтому я не использую.

В качестве антиспама я предпочитаю коммерческое решение — Kaspersky Anti-Spam. Я знаю этот продукт уже лет 10. Он действительно отлично фильтрует спам. Ложных срабатываний вообще не припоминаю, 95% спама фильтрует, может больше. Субъективно, работает лучше чем антиспам у того же gmail или яндекса. С ним вопрос спама отпадает вообще. Стоит он недорого, можно купить лицензию на меньшее количество ящиков, чем реально используется в системе. Этот вопрос никак не отслеживается и на качество работы не влияет. Но нужно понимать, что это уже нарушение лицензионного соглашения. Но можно всякие хитрости придумать, чтобы и фильтровать и не нарушать.

Борьба со спамом средствами postfix

Сначала хотел сразу все настройки postfix разместить в соответствующем разделе в едином конфиге, но потом передумал и решил все же вынести этот вопрос на отдельное рассмотрение. Возможно, не каждому захочется сразу в эту тему углубляться. Все, что рассказано выше, позволит настроить полноценный почтовый сервер, который будет успешно принимать почту и доставлять ее пользователям. Но в таком виде он будет принимать слишком много спама, но зато не будет проблем с тем, что от кого-то что-то не придет. Как ни крути, но все средства борьбы со спамом так или иначе несут накладные расходы в виде ложных срабатываний с той или иной вероятностью. Если вы решите не заморачиваться и купить Kaspersky Anti-Spam, можете этот раздел не читать. Он сам реализует все те проверки, что мы будем делать. Если же хотите своими силами бороться со спамом средствами postfix, то давайте дальше разбираться.

Я буду использовать штатные возможности postfix, позволяющие отсеять спам по тем или иным параметрам еще до получения письма. Это очень эффективный способ с точки зрения производительности. Благодаря этому, правильно настроенный на отсев спама postfix часто ставят перед exchange, чтобы снизить на него нагрузку. Сразу дам ссылки на официальную документацию с описанием параметров, которые я буду использовать:

1. smtpd_helo_restrictions
2. smtpd_sender_restrictions
3. smtpd_recipient_restrictions
4. smtpd_data_restrictions
5. smtpd_client_restrictions

Есть еще парочка — smtpd_etrn_restrictions и smtpd_end_of_data_restrictions, но я ими не пользуюсь.

> Обращаю внимание на то, что нужно очень аккуратно работать с настройками, о которых пойдет речь. Нужно четко понимать как, зачем и что вы делаете. Неверные настройки могут нарушить нормальное хождение почты. Нужно уметь анализировать лог файл почтового сервера и понимать, что там происходит.

Долго думал, как лучше всего представить информацию по этому разделу. В итоге решил просто написать часть конфига, которая отвечает за restrictions с комментариями. Более подробную информацию по каждой настройке вы можете найти в официальной документации postfix, ссылки я привел выше, либо в гугле. Данные настройки это моя многолетняя калькуляция различных параметров, собранных из черновиков и рабочих серверов. Не везде все было настроено именно в таком виде, так как ситуации бывают разные. Сейчас я постарался все собрать в одном месте и аккуратно описать. Те проверки в

борьбе со спамом, что вам не нужны, просто закомментируйте. В конце я еще пройдусь по некоторым из них и поделюсь своим опытом.

```
#Описание списков исключений
smtpd_restriction_classes = white_client_ip,
                            black_client_ip,
                            block_dsl,
                            white_client,
                            white_helo,
                            black_client,
                            mx_access

# IP адреса, которые нужно пропускать всегда
white_client_ip         = check_client_access hash:/etc/postfix/lists/white_client_ip

# IP адреса, которые нужно блокировать всегда
black_client_ip         = check_client_access hash:/etc/postfix/lists/black_client_ip

# E-mail, которые нужно пропускать всегда
white_client            = check_sender_access hash:/etc/postfix/lists/white_client

# E-mail, которые нужно блокировать всегда
black_client            = check_sender_access hash:/etc/postfix/lists/black_client

# Неправильные значения HELO, которые мы тем не менее принимаем
white_helo = check_sender_access hash:/etc/postfix/lists/white_helo
# Правила для блокировки различных динамических ip.
block_dsl               = check_client_access regexp:/etc/postfix/lists/block_dsl

# Список приватных сетей, которые не могут быть использованы в качестве IP для MX записей
mx_access               = check_sender_mx_access cidr:/etc/postfix/lists/mx_access

# Проверки на основе данных, переданных в HELO/EHLO hostname
smtpd_helo_restrictions =       permit_mynetworks,
```

```
                permit_sasl_authenticated,
                white_client_ip,
                white_helo,
                black_client_ip,
                block_dsl,
                # Отказываем серверам, у которых в HELO несуществующий или не FQDN адрес
                reject_invalid_helo_hostname,
                reject_non_fqdn_helo_hostname,
                # Запрещаем приём писем от серверов, представляющихся адресом, для которого не существует A или MX записи.
                reject_unknown_helo_hostname

# Проверки клиентского компьютера или другого почтового сервера, который соединяется с сервером postfix для отправки письма
smtpd_client_restrictions =     permit_mynetworks,
                permit_sasl_authenticated,
                # Отвергает запрос, когда клиент отправляет команды SMTP раньше времени, еще не зная, поддерживает ли
Postfix конвейерную обработку команд ESMTP
                reject_unauth_pipelining,
                # Блокируем клиентов с адресами from, домены которых не имеют A/MX записей
                reject_unknown_address,
                reject_unknown_client_hostname

# Проверки исходящей или пересылаемой через нас почты на основе данных MAIL FROM
smtpd_sender_restrictions =     permit_mynetworks,
                permit_sasl_authenticated,
                white_client,
                black_client,
                # Запрет отправки писем, когда адрес MAIL FROM не совпадает с логином пользователя
                reject_authenticated_sender_login_mismatch,
                # Отклоняем письма от несуществующих доменов
                reject_unknown_sender_domain,
                # Отклоняем письма от доменов в не FQDN формате
                reject_non_fqdn_sender,
                # Отклонение писем с несуществующим адресом отправителя
```

```
                reject_unlisted_sender,
                reject_unauth_destination,
                # Отклонять сообщения от отправителей, ящики которых не существуют, использовать аккуратно
                #reject_unverified_sender,
                mx_access

# Правила приема почты нашим сервером на основе данных RCPT TO
smtpd_recipient_restrictions =  permit_mynetworks,
                permit_sasl_authenticated,
                # Отклоняет всю почту, что адресована не для наших доменов
                reject_unauth_destination,
                # Отклонение писем с несуществующим адресом получателя
                reject_unlisted_recipient,
                # Отклоняет сообщения на несуществующие домены
                reject_unknown_recipient_domain,
                # Отклоняет сообщения если получатель не в формате FQDN
                reject_non_fqdn_recipient,
                # Отклоняем прием от отправителя с пустым адресом письма, предназначенным нескольким получателям.
                reject_multi_recipient_bounce
```

У меня во всех ограничениях первыми правилами стоят разрешения для mynetworks и авторизовавшихся пользователей. Важно понимать, что это значит и для чего сделано. Ограничения читаются последовательно в порядке их перечисления. Таким образом, мы своих пользователей пускаем мимо ограничений, а для всех остальных выполняются проверки.

Теперь важные комментарии по указанным параметрам. Если бы все почтовые сервера всех системных администраторов были настроены по правилам, то эти комментарии не были бы нужны. Пройдемся по некоторым ограничениям, которые нужно включать осторожно:

- reject_invalid_helo_hostname и reject_unknown_helo_hostname — под эти правила иногда попадают почтовые серверы клиентов, которые не очень хорошо настроены. У них бывают неправильные адреса, кривые записи dns, отсутствие обратных зон и т.д. Их не много, но попадаются. Это не страшно, если вы регулярно следите за сервером. Отправитель получит сразу сообщение о том, что его письмо не дошло до вас. Если он как-то сообщит вам о проблеме, вы легко добавите его в белый список и все будет нормально. Если вам не хочется следить за сервером, лучше не указывайте эти ограничения. Но спам они отсекают не плохо. Сюда попадают все завирусованные компьютеры и сервера без нормальных настроек dns (а их чаще всего и нет).

- reject_unverified_sender — специально его закомментировал. Я тестировал этот параметр. В принципе, работает нормально, но есть, как обычно, нюансы. Поясню, что делает этот параметр. Когда вам кто-то шлет письмо, ваш сервер обращается к серверу отправителю и спрашивает его стандатрной командой, есть ли на сервере такой отправитель. Если удаленный сервер отвечает, что есть, то никаких проблем — письмо принимается. Если удаленный сервер не отвечает или говорит, что такого адресата нет — письмо отклоняем. Очевидно, что такие проверки создают дополнительную постоянную нагрузку. Это нужно учитывать. К тому же, у вас почтовый лог постоянно будет забит этими проверками, особенно, если вам приходит много спама. На каждое спамовое письмо будет идти проверка, а сервера отправителя скорее всего либо нет, либо он неправильный, либо не отвечает и т.д. Все это будет постоянно проверяться и фиксироваться. В общем, я не использую.

На время отладки ограничений, рекомендую пользоваться параметром:

```
soft_bounce = yes
```

Когда он включен, все ответы сервера с кодами ошибок 5XX, заменяются на 4XX. То есть постоянная ошибка, которая сразу отклоняет письмо, заменяется на временную, которая предлагает повторить отправку позже. Таким образом, вы увидите работу всех ограничений, но письма не будут отклонены навсегда. Сервер отправителя через некоторое время снова придет к вам с новой попыткой доставки почты. Письмо безвозвратно не отклоняется. Вы можете проанализировать работу фильтра и решить, ставить его на постоянную работу или с ним что-то не так.

Создадим теперь файлы с белыми и черными списками.

```
cd /etc/postfix/lists && touch white_client_ip black_client_ip white_client black_client white_helo block_dsl mx_access
```

Ниже пример содержания этих файлов. Вы можете менять по своему усмотрению.

```
# cat white_client_ip
195.28.34.162 OK
141.197.4.160 OK
```

```
# cat black_client_ip
205.201.130.163 REJECT You IP are blacklisted!
198.2.129.162 REJECT You IP are blacklisted!
```

```
# cat white_client
# Принимать всю почту с домена яндекс
yandex.ru OK
# Разрешить конкретный ящик
spammer@mail.ru OK
```

```
# cat black_client
# Блокировать всю почту с домена mail.ru
mail.ru REJECT You domain are blacklisted!
# Блокировать конкретный ящик
spam@rambler.ru REJECT You e-mail are blacklisted!
```

```
# cat white_helo
# Могут попадаться вот такие адреса, которые не пройдут наши проверки
ka-s-ex01.itk.local     OK
exchange.elcom.local    OK
```

```
# cat block_dsl
/^dsl.*\..*\..*/i                               553 AUTO_DSL spam
/dsl.*\..*\..*/i                                553 AUTO_DSL1 spam
/[ax]dsl.*\..*\..*/i                            553 AUTO_XDSL spam
/client.*\..*\..*/i                             553 AUTO_CLIENT spam
/cable.*\..*\..*/i                              553 AUTO_CABLE spam
/pool.*\..*\..*/i                               553 AUTO_POOL spam
/dial.*\..*\..*/i                               553 AUTO_DIAL spam
/ppp.*\..*\..*/i                                553 AUTO_PPP spam
/dslam.*\..*\..*/i                              553 AUTO_DSLAM spam
/node.*\..*\..*/i                               553 AUTO_NODE spam
/([0-9]*-){3}[0-9]*(\..*){2,}/i                 553 SPAM_ip-add-rr-ess_networks
/([0-9]*\.){4}(.*\.){3,}.*/i                    553 SPAM_ip-add-rr-ess_networks
/.*\.pppool\..*/i                               553 SPAM_POOL
```

```
/[0-9]*-[0-9]*-[0-9]*-[0-9]*-tami\.tami\.pl/i   553 SPAM_POOL
/pool-[0-9]*-[0-9]*-[0-9]*-[0-9]*\..*/i         553 SPAM_POOL
/.*-[0-9]*-[0-9]*-[0-9]*-[0-9]*\.gtel.net.mx/i  553 SPAM_POOL
/dhcp.*\..*\..*/i                               553 SPAM_DHCP
```

```
# cat mx_access
127.0.0.1      DUNNO
127.0.0.2      550 Domains not registered properly
0.0.0.0/8      REJECT Domain MX in broadcast network
10.0.0.0/8     REJECT Domain MX in RFC 1918 private network
127.0.0.0/8    REJECT Domain MX in loopback network
169.254.0.0/16 REJECT Domain MX in link local network
172.16.0.0/12  REJECT Domain MX in RFC 1918 private network
192.0.2.0/24   REJECT Domain MX in TEST-NET network
192.168.0.0/16 REJECT Domain MX in RFC 1918 private network
224.0.0.0/4    REJECT Domain MX in class D multicast network
240.0.0.0/5    REJECT Domain MX in class E reserved network
248.0.0.0/5    REJECT Domain MX in reserved network
```

По сути файлы белых и черных списков не отличаются друг от друга. Можно использовать только один файл и в нем в каждой отдельной строке указывать либо запрет, либо разрешение. Я разделил просто для удобства восприятия и редактирования. Возможно вам будет удобнее с одним файлом.

После редактирования файлов обязательно выполняем команду на перестроение базы данных. Я перестрою сразу все файлы:

```
cd /etc/postfix/lists && postmap white_client_ip black_client_ip white_client black_client white_helo block_dsl mx_access
```

Еще упомяну о таком популярном явлении в спамерских письмах, как подделка адреса отправителя. Причем не просто подделка на абы кого, а именно на ваше имя домена. Пользователь получает спам письмо и в почтовом клиенте видит, что оно отправлено с вашего домена. Только по заголовкам можно определить реального отправителя. Такой подход позволяет обходить некоторые антиспам системы, которые не фильтруют письма внутреннего домена. Бороться с подменой адреса отправителя в нашем случае очень просто. Об этом я рассказал отдельно — Запрет писем с поддельным полем From.

Приведу в завершении описания методов борьбы со спамом простой пример. Добавим в black_client почтовый адрес и отправим с него письмо.

```
# cat black_client
zeroxzed@gmail.com REJECT Your e-mail was banned!
```

```
# postmap black_client
```

Отправляем сообщение и проверяем почтовый лог.

```
# cat /var/log/maillog
```

```
Feb 11 16:21:31 mail postfix/smtpd[20115]: connect from mail-ua1-f48.google.com[209.85.222.48]
Feb 11 16:21:31 mail postfix/smtpd[20115]: Anonymous TLS connection established from mail-ua1-f48.google.com[209.85.222.48]:
TLSv1.3 with cipher TLS_AES_128_GCM_SHA256 (128/128 bits)
Feb 11 16:21:32 mail postfix/smtpd[20115]: NOQUEUE: reject: RCPT from mail-ua1-f48.google.com[209.85.222.48]: 554 5.7.1
<zeroxzed@gmail.com>: Sender address rejected: Your e-mail was banned!; from=<zeroxzed@gmail.com> to=<root@kirushin-
vladimir.ru> proto=ESMTP helo=<mail-ua1-f48.google.com>
Feb 11 16:21:32 mail postfix/smtpd[20115]: disconnect from mail-ua1-f48.google.com[209.85.222.48] ehlo=2 starttls=1 mail=1
rcpt=0/1 data=0/1 quit=1 commands=5/7
```

Вот и результат. На этом по борьбе со спамом все.

Заключение

> Не понравилась статья и хочешь научить меня администрировать? Пожалуйста, я люблю учиться. Комментарии в твоем распоряжении. Расскажи, как сделать правильно!

Проверить настроенный почтовый сервер можно с помощью онлайн сервиса https://www.mail-tester.com. Не факт, что получите максимальный бал, но все недочеты будут указаны. Критичное нужно исправить (например, если обратная зона неправильная), некритичное можно пропустить (если dkim, к примеру, не настраивали).

Кажется все написал, что знал по поводу почтового сервера на linux в небольших и средних организациях. У некоторых может возникнуть вопрос, а зачем свой почтовый сервер? Почему бы не воспользоваться средствами корпоративной почты, которую представляют популярные почтовые сервисы бесплатно? У меня есть определенный опыт на этот счет, в том числе негативный. И если некоторое время назад я считал, что свои почтовые серверы в небольших организациях уже не актуальны, то сейчас я так не думаю, поэтому и появилась эта статья.

Так же вам могут быть полезны мои материалы на тему postfix:

- Как изменить тему письма и адрес отправителя через postfix
- Настройка relayhost, отдельный для каждого домена
- Перенос почтового сервера postfix

Буду рад замечаниям по делу и советам в комментариях. Напоминаю, что данная статья является частью единого цикла статьей про сервер Centos.

Помогла статья? Есть возможность отблагодарить автора