Полезные команды почтового сервера Exim

Exim — это так называемый *MTA* (*Mail Transfer Agent*), агент передачи сообщений, в просторечии - почтовик или почтовый сервер, использующийся в операционных системах Unix. Распространяется по свободной лицензии *GPL*, то есть доступен для распространения, использования и модификации. Exim, весьма распространен и в некоторых операционных системах является почтовым сервером по умолчанию.

ID сообщений и spool файлы

Идентификаторы сообщений в очередях *Exim*, представляют из себя буквенно-цифровые последовательности в верхнем и нижнем регистрах: XXXXXX-YYYYYY-ZZ и используются большинством команд администрирования очереди и логгирования в *Exim*. Для каждого сообщения создаются три файла в *spool* директории (зачастую это: */var/spool*). Если вы работает с данными файлами вручную, без использования описанных ниже команд и утилит, убедитесь что обработали все три файла, например, не оставив в очереди *exim* только один из них, удалив остальные. Каталог */var/spool/exim/msglog* содержит файлы со служебной информацией для каждого сообщения и именуются в соответствии с *ID* этого сообщения. Каталог */var/spool/exim/input* содержат файлы заголовков и данных, к *ID* сообщения в имени, добавлены суффиксы **-H** и **-D**, соответственно. Кроме того в этих директориях могут появляться хэшированные подкаталоги для работы с большими почтовыми очередями.

Получение базовой информации по Exim

Вывести количество сообщений в очереди:

```
root@localhost# exim -bpc
```

Печать списка сообщений в очереди. Выводятся, время постановки в очередь, размер, ID сообщения, отправитель, получатель:

```
root@localhost# exim -bp
```

Печать суммарной информации. Выводимые колонки: количество, объем, старейшее, последнее, домен.

```
root@localhost# exim -bp | exiqsumm
```

Чем в данный момент занимается Exim:

```
root@localhost# exiwhat
```

Тестирование маршрута доставки до указанного адреса:

```
root@localhost# exim -bt alias@localdomain.com
user@thishost.com
    <-- alias@localdomain.com
  router = localuser, transport = local_delivery
root@localhost# exim -bt user@thishost.com
user@thishost.com
  router = localuser, transport = local_delivery
root@localhost# exim -bt user@remotehost.com
  router = lookuphost, transport = remote_smtp
  host mail.remotehost.com [1.2.3.4] MX=0
```

Эмитировать *SMTP* транзакцию из командной строки, как если-бы сообщение пришло с указанного IP адреса. При этом будет показано прохождение и срабатывание проверок, фильтров и листов доступа (*ACL*). На самом деле, никакое сообщение никуда доставлено не будет.

```
root@localhost# exim -bh 192.168.11.22
```

Листинг всех настроек конфигурации *exim*:

```
root@localhost# exim -bP
```

Поиск очереди с помощью утилиты exiqgrep

Стандартная поставка сервера *Exim* включает в себя утилиту для поиска по очередям — *exiqgrep*, это самый оптимальный путь для решения данной задачи. Если вы используете конвейер команд, например из *exim -bp* в *awk*, *grep*, *cut* и т.д., вы просто усложняете себе жизнь. Различные ключи команды *exiqgrep*, позволяют достаточно тонко настроить критерии поиска. Ключ **-f** используется для поиска сообщений конкретного отправителя

```
root@localhost# exiqgrep -f [luser]@domain
```

Ключ **-r** используется для поиска сообщений для определенного адресата

```
root@localhost# exiqgrep -r [luser]@domain
```

Ключ **-o** указывает искать сообщения, старее, указанного количества секунд. В примере, сообщения старее 1 дня:

```
root@localhost# exiqgrep -o 86400 [...]
```

Ключ **-y** ищет сообщения свежее указанного количества секунд. В примере, найти сообщения, пришедшие в течении последнего часа:

```
root@localhost# exiqgrep -y 3600 [...]
```

Ключ **-s** позволяет искать по размеру сообщения, совпадающего с заданным регулярным выражением:

```
root@localhost# exiqgrep -s '^7..$' [...]
```

Для поиска только среди заблокированных(замороженных) сообщений, используйте ключ **-z**, или **-x** для поиска только среди не заблокированных. Еще несколько ключей отвечающих за вывод результатов поиска Вывести только *ID* сообщения, в одном из вышеупомянутых вариантов поиска

```
root@localhost# exiqgrep -i [ -r | -f ] ...
```

Печатать счетчик сообщений при одном из вышеприведенных вариантов поиска:

```
root@localhost# exiqgrep -c ...
```

Вывести только идентификатор всей очереди:

```
root@localhost# exiqgrep -i
```

Управление очередями сообщений

Основной бинарник *Exim* (*/usr/sbin/exim*), используется с различными ключами для управления сообщениями в очереди. Многие ключи, подразумевают указание одного или более *ID* сообщения в командной строке, как раз тут вам и пригодится команда *exiqgrep -i*, которая была упомянута выше. Запуск очереди:

```
root@localhost# exim -q -v
```

Запуск очереди только для локальных доставок:

```
root@localhost# exim -ql -v
```

Удалить сообщение из очереди:

```
root@localhost# exim -Mrm  [  ... ]
```

Очистит все заблокированные сообщения из очереди:

```
root@localhost# exipick -zi |  xargs exim -Mrm
```

Очистит все сообщения из очереди:

```
root@localhost# exipick -i |  xargs exim -Mrm
```

Заблокировать(заморозить) сообщение:

```
root@localhost# exim -Mf  [  ... ]
```

Разблокировать сообщение:

```
root@localhost# exim -Mt  [  ... ]
```

Доставить сообщение, вне зависимости от состояния блокировки или времени повторной доставки:

```
root@localhost# exim -M  [  ... ]
```

Доставить сообщение, только если достигнуто время для повторной доставки:

```
root@localhost# exim -Mc  [  ... ]
```

Принудительно остановить сообщение с формулировкой "отменено администратором":

```
root@localhost# exim -Mg  [  ... ]
```

Удалить все заблокированные сообщения:

```
root@localhost# exiqgrep -z -i |  xargs exim -Mrm
```

Удалить все сообщения, старше 5 дней (86400 * 5 = 432000 секунд):

```
root@localhost# exiqgrep -o 432000 -i |  xargs exim -Mrm
```

Заблокировать все письма от указанного отправителя:

```
root@localhost# exiqgrep -i -f luser@example.tld |  xargs exim -Mf
```

Просмотреть заголовки сообщений:

```
root@localhost# exim -Mvh
```

Просмотреть тело сообщений:

```
root@localhost# exim -Mvb
```

Просмотр логов сообщения:

```
root@localhost# exim -Mvl
```

Добавить получателя в сообщение:

```
root@localhost# exim -Mar
```

[
...]
Редактировать отправителя сообщения:

```
root@localhost# exim -Mes
```

Листы контроля доступа (Access Control List, ACL)

Exim предоставляет возможность использовать, *листы контроля доступа* (*ACL*) на различных этапах *SMTP* передачи. Условия *ACL* назначаются в конфигурационном файле *exim.conf*. Начать имеет смысл с *HELO*.

```
# Назначить ACL для использования после команды HELO
acl_smtp_helo = check_helo

# Условия проверки check_helo для ACL:
check_helo:

    deny message = Gave HELO/EHLO as "friend"
    log_message = HELO/EHLO friend
    condition = ${if eq {$sender_helo_name}{friend} {yes}{no}}

    deny message = Gave HELO/EHLO as our IP address
    log_message = HELO/EHLO our IP address
    condition = ${if eq {$sender_helo_name}{$interface_address} {yes}{no}}

    accept
```

> Используйте проверку HELO на свой страх и риск. Сейчас HELO не имеет такого большого значения в общей схеме SMTP, так что не стоит сильно доверять его содержимому. Не только спамеры используют строку HELO, вы будете удивлены сколько нормальных сообщений может приходить с кривым HELO. Спамерам не составит труда отправлять сообщения с легальной строкой HELO а не писать там "я спамер", так что много спама вы на этом не отсеете.

Далее, вы можете провести проверку по отправителю или удаленному хосту. В примере показано как фильтровать по содержимому, идущему после команды *RCPT TO*. Если прием сообщения будет отклонен в этом месте, вы получите больше данных в логах, нежели при блокировке по *MAIL FROM*.

```
# Назначит проверку содержимого после RCPT TO
acl_smtp_rcpt = check_recipient

# Условия для check_recipient ACL
check_recipient:

    # [...]

    drop hosts = /etc/exim_reject_hosts
    drop senders = /etc/exim_reject_senders

    # [ Probably a whole lot more... ]
```

В приведенном примере для блокировки используются два текстовых файла. В файл */etc/exim_reject_hosts*, значения добавляются в виде: имя_хоста/IP_адрес, в файл */etc/exim_reject_senders* в виде адреса отправителя, по одной записи в строке. Кроме того, можно сканировать содержимое сообщения, на предмет совпадения с регулярным выражением. Имейте в виду, это дает дополнительную нагрузку на процессор, особенно на больших сообщениях.

```
# Назначить ACL для использования после команды DATA
acl_smtp_data = check_message

# Условия проверки для check_messages ACL
check_message:

    deny message = "Sorry, Charlie: $regex_match_string"
    regex = ^Subject:: .*Lower your self-esteem by becoming a sysadmin

    accept
```

Исправление SMTP аутентификации для pine

В случае, если *pine* не может использовать аутентификацию на сервере *Exim*, возвращая сообщение "*unable to authenticate*", без запроса на ввод пароля, нужно добавить в *exim.conf* следующие строки.

```
  begin authenticators

  fixed_plain:
  driver = plaintext
  public_name = PLAIN
  server_condition = "${perl{checkuserpass}{$1}{$2}{$3}}"
  server_set_id = $2
>  server_prompts = :
```

Некоторое время назад, данная проблема имела место быть в *CPanel*, на текущий момент этот недочет исправлен.

Запись в лог файл заголовка Subject

Один из самых полезных хаков конфигурации *Exim*. Добавление в *exim.conf* приведенной строки, позволит писать в лог файл строку *subject*, писем проходящих через сервер. Это сильно помогает при решении проблем в процессе настройки сервера, а так-же дает дополнительные критерии для отсева спама.

```
log_selector = +subject
```

Отключение identd

Честно говоря не думаю что протокол *identd* был когда либо очень полезен. *Identd* опирается на подключающийся хост, что-бы подтвердить идентификацию (*System UID*), удаленного пользователя, владельца процесса, устанавливающего сетевое соединение. Это может быть в определенной степени полезно в мире системных оболочек и IRC пользователей, но уж никак не на нагруженном почтовом сервере, где пользователем процесса зачастую является просто какой-нибудь "mail", и за этим может быть любой другой MTA. В итоге куча накладных расходов, нулевой результат, лишь задержки *identd* запросов и как следствие отказы и таймауты. Можно запретить *Exim* делать подобные запросы, установив таймаут 0 секунд в *exim.conf*.

```
rfc1413_query_timeout = 0s
```

Отключение блокировки вложений

Что-бы отключить блокировку исполняемых вложений, что *CPanel* делает по умолчанию, правда не предоставляя при это контроля "для каждого домена", следующий блок нужно добавить в начало файла */etc/antivirus.exim*:

```
if $header_to:  matches "example\.com| example2\.com"
then
  finish
endif
```

Поиск в журнальных файлах с помощью exigrep

Утилита *exigrep* (не путайте с *exiqgrep*, использующейся для поиска в очереди), используется для поиска по лог файлам. Например *exigrep* может вывести все записи из лог файла с совпадающим *ID* сообщения, что довольно удобно, учитывая что каждое сообщение занимает 3 строки в лог файле. Поиск сообщений отправленных с определенного IP адреса:

```
root@localhost# exigrep '<= .* \[12.34.56.78\] '  /path/to/exim_log
```

Поиск сообщений отправленных на определенный IP адрес:

```
root@localhost# exigrep '=> .* \[12.34.56.78\]'  /path/to/exim_log
```

Данный пример ищет сообщения содержащие символы "=>", и отправленные на адрес "*user@domain.tld*", далее по конвейеру, результат передается команде *grep*, которая из полученного результата выбирает строки, содержащие "<=" с информацией об отправителе, почтовом адресе, IP адресе, размере сообщения, ID сообщения и заголовок *subject*, если логгирование этой строки включено.

```
root@localhost# exigrep '=> .*user@domain.tld'  /path/to/exim_log |  fgrep '<='
```

Генерировать из лог файла и показать статистику *Exim*:

```
root@localhost# eximstats /path/to/exim_mainlog
```

То-же что и выше но с более подробными данными:

```
root@localhost# eximstats -ne -nr -nt /path/to/exim_mainlog
```

Аналогично но за определенный день:

```
root@localhost# fgrep YYYY-MM-DD /path/to/exim_mainlog | eximstats
```

В качестве дополнения Удалить все сообщения в очереди, содержащие в теле, определенную строку:

```
root@localhost# grep -lr 'a certain string' /var/spool/exim/input/ | \
                sed -e 's/^.*\/\([a-zA-Z0-9-]*\)-[DH]$/\1/g' | xargs exim -Mrm
```

Командой выше, мы проверяем содержимое каталога */var/spool/exim/input/*, в поисках файлов очереди, содержащих определенную строку в теле сообщения, поскольку команда *exiqgrep* не умеет просматривать тело сообщений. Если вы решите удалить найденные файлы напрямую, ЭТО БУДЕТ НЕ ПРАВИЛЬНО, используйте предназначенные для этого команды *exim*. Если вывод используемой команды слишком длинный, например ID сообщений при *exiqgrep -i*, которые нужно передать дальше по конвейеру команде *exim*, может быть превышено количество аргументов командной строки вашей системной оболочки. В этом случае передавайте результат поиска через конвейер, команде *xargs*, которая будет обрабатывать результаты ограниченными порциями. Например удалим тысячи сообщений, отправленных с адреса *joe@example.com*:

```
root@localhost# exiqgrep -i -f '' | xargs exim -Mrm
```

После того как вы внесли изменения в файл конфигурации, необходимо перезапустить exim, или послать рабочему процессу сигнал SIGHUP, что-бы он перечитал конфигурационный файл и изменения вступили в силу. Предпочтительней естественно отправить сигнал, нежели перезапускать приложение.

```
root@localhost# kill -HUP `cat /var/spool/exim/exim-daemon.pid`
```

Revision #1
Created 23 February 2023 00:01:11 by 3err0
Updated 23 February 2023 00:02:32 by 3err0