

АРХИТЕКТУРА postfix



В последнее время очень часто почтовые системы создаются на основе Postfix. Данный пакет программного обеспечения стал довольно популярен среди администраторов всех видов UNIX-систем. Причина такого хода событий тривиальна – Postfix характеризуется как феноменально простая в освоении, удобная в настройке и чрезвычайно устойчивая система. Целью создания этого продукта была надежда полностью заменить sendmail. Являясь монолитным сервером, sendmail небезопасен, потому что одна-единственная ошибка может скомпрометировать всю программу.

АНДРЕЙ БЕШКОВ

Давайте посмотрим, как выглядит изнутри почтовая система, построенная на основе Postfix, и чем она отличается от других систем, предназначенных для выполнения тех же задач. Добиться этого можно, сосредоточив свое внимание на детальном изучении анатомии этой программы. Очень надеюсь, что прилагающаяся к статье схема поможет нам в этом.

Во-первых, хотелось бы сказать, что костяк системы построен на использовании нескольких полурезидентных программ. Каждая из них в соответствии с философией UNIX выполняет только одну задачу, но делает это хорошо. В отличие от стандартного подхода, практикуемого qmail, при котором главная программа вызывает вспомогательные по мере необходимости и позволяет им умирать, когда они уже не нужны, резидентность служебных программ позволяет сэкономить на создании новых процессов. Впрочем, если грамотно распределить этапы обработки почты для каждой программы и несильно дробить весь процесс, то подход, исповедуемый qmail-подобными системами, будет вполне допустим.

В то же время нет необходимости в большом количестве одновременно работающих программ, самостоятельно выполняющих одно и то же действие, для находящихся в обработке писем. Их все можно заменить одним или несколькими резидентными экземплярами нужной подсистемы, обрабатывающими по очереди все поступающие запросы. Соответственно любая служебная программа при необходимости может запросто обратиться с запросами к любому другому компоненту почтовой системы.

Такое жесткое деление на подсистемы позволяет легко отключать те модули, в которых вы не нуждаетесь. Например, на пограничном межсетевом экране нет необходимости держать включенным модуль, принимающий SMTP-соединения. Postfix имеет довольно сложное внутреннее устройство. На момент последнего релиза исходный код насчитывал 30 000 строк после удаления комментариев. Но в то же время все построено очень логично, и новому пользователю не приходится вникать в особенности реализации сразу после установки системы. Необходимость изучать систему глубже возникает только тогда, когда хочется сделать что-то нестандартное. Но и тут нас ожидают приятные неожиданности, благодаря своему логичному дизайну перевести систему в любой режим работы оказывается достаточно просто. В таком большом комплексе приходится как можно тщательнее заботиться о безопасности системы. Для этого применяются следующие меры:

- Использование наименьших привилегий. Postfix может быть легко ограничен с помощью chroot и работать от имени самого бесправного пользователя.
- Ни одна из служебных программ не использует set-uid бит.
- Между программами, отвечающими за доставку почты, и пользовательскими процессами, работающими в системе, отсутствуют отношения родитель-ребенок. Это позволяет свести на нет попытку использования эксплоитов, основанных на том, что злонамеренный родительский процесс может передавать дочернему специально измененные переменные среды, сигналы, открытые файлы.

- Все данные, приходящие из внешних источников по умолчанию, считаются потенциально опасными, поэтому должны быть проверены и отфильтрованы жесточайшим образом.
- Память для текстовых фрагментов и служебных буферов выделяется динамически, что позволяет значительно снизить вероятность успешного использования ошибок переполнения буфера. Слишком большие текстовые массивы обрабатываются после разрезания на фиксированные фрагменты. После завершения всех нужных действий они снова склеиваются воедино. Такой подход позволяет существенно уменьшить требования программы к наличию оперативной памяти.
- Количество объектов, находящихся в памяти, строго ограничено. Соответственно даже под большой нагрузкой Postfix не будет потреблять слишком много системных ресурсов. Ведь скорость обработки почтовых сообщений вряд ли вырастет от того, что мы, к примеру, вместо десяти будем использовать двадцать буферов для обработки писем. Тут уже ограничителем выступает скорость аппаратных компонентов самой системы, а не количество объектов для обработки почты, хранящейся на диске.

Итак, разобравшись с основными концепциями дизайна системы, перейдем к детальному рассмотрению ее архитектуры. Во главе всего стоит демон master, который обычно запускается командой postfix при старте системы и работает постоянно до тех пор, пока действует почтовая система. Этот супердемон отвечает за запуск по требованию всех остальных демонов и перезапуск тех из них, которые преждевременно завершили свою работу из-за каких-либо проблем. В его обязанности также входит следить, чтобы количество дочерних процессов не превышало ограничения установленных в файле master.cf, и чтобы каждый из них жил не дольше, чем определено настройками почтовой системы. В дальнейшем в статье по мере возможности вместо слова «демон» я буду употреблять слово «процесс». Думаю, так будет удобнее всего. Главное, не забывать, что наши демонизированные процессы в отличие от обычных процессов никогда не завершаются самостоятельно, а лишь по приказу от master. Если те или иные процессы простаивают из-за отсутствия работы, то по истечении определенного тайм-аута они будут принудительно завершены в целях экономии памяти. Между собой все они общаются либо с помощью UNIX-сокетов, либо через FIFO. Все каналы обмена находятся в специально защищенной директории. Несмотря на такие предосторожности, все данные, получаемые от разных подсистем самого Postfix и внешних сущностей, обязательно подвергаются добавочным проверкам.

Потихоньку разобравшись с вопросом, кто здесь главный, давайте посмотрим, какими путями письма появляются внутри почтовой системы.

Самый частый случай – когда новые письма приходят через сетевое соединение. Первым делом их получает SMTP-демон. Если администратор включил соответствующую возможность, то сначала будет произведена проверка, не попадает ли письмо под ограничения UCE-филь-

трации. UCE (unsolicited commercial email) – невостреванная коммерческая почта, или, проще говоря, спам. Для борьбы с ним можно применять следующие меры:

- Фильтрация по служебным заголовкам или телу сообщений.
- Ограничения IP-адресов и имен хостов клиентов, которым разрешено отправлять почту через нашу систему.
- Принуждение клиентских программ начинать каждую сессию с команды EHLO. Большинство инструментов, используемых спамерами для групповой рассылки, не обучены следовать этому стандарту.
- Требование использовать почтовые адреса, полностью соответствующие стандарту RFC 821.
- Ограничения, накладываемые на почтовый адрес отправителя и получателя сообщения.
- Проверка IP-адреса отправителя по черному списку RBL.

После этого с полученным письмом начинает работать процесс cleanup, занимающийся его зачисткой. Он проводит первоначальные проверки на правильность оформления и формат входящего сообщения, позволяя защитить остальную систему от многих атак, рассчитанных на переполнение буфера. В случае необходимости добавляет в письмо недостающие служебные заголовки и с помощью процесса по имени trivial-rewrite приводит адреса к виду: пользователь@поддомен.домен. В postfix пока нет полноценного языка, позволяющего гибко управлять процессом переписывания адресов, поэтому вместо него используются служебные таблицы canonical и virtual для хранения правил, управляющих перезаписью.

После такой обработки письмо сохраняется на диск в формате файла почтовой очереди и кладется в директорию, где хранится очередь incoming, обычно это директория /var/spool/postfix/incoming/. Как вы могли догадаться по названию, эта очередь используется только для обработки входящих сообщений. Затем процесс cleanup уведомляет процесс queue manager, управляющий всеми очередями о прибытии новой почты.

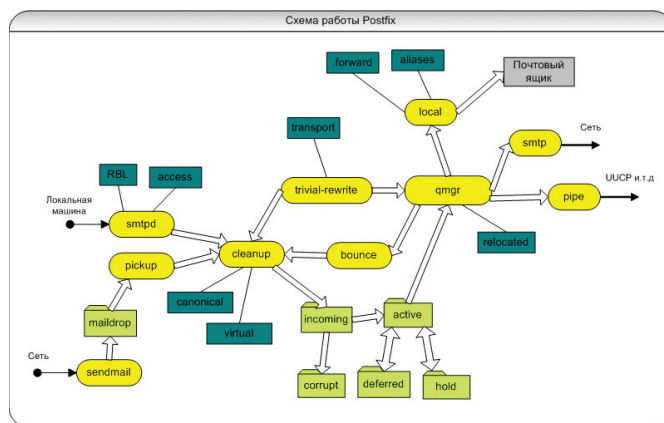
Следующий случай – письмо, отправленное из нашей локальной системы с помощью какого-либо скрипта или, например, с помощью вот такой команды:

```
echo "mail text" | mail tigrisha@unreal.net -s "test subject"
```

Во многих UNIX-системах в качестве почтовой программы и по сей день традиционно используется sendmail. Соответственно большинство скриптов, написанных ранее и используемых сейчас априори, считает, что в системе установлена именно эта разновидность почтового сервера. Чтобы не разрушить совместимость с огромным количеством программного обеспечения при переходе к использованию Postfix, в систему вместе с прочими файлами устанавливается программа, заменяющая sendmail. Таким образом, получается, что команда mail передает письмо программе sendmail, установленной Postfix, а та в свою очередь вызывает привилегированную программу postdrop. Ну а последняя уже кладет файлы с письмами в служебную директорию maildrop, которая обычно находится в /var/spool/postfix/maildrop/. Затем письмо подхватывает

процесс pickup и аналогично SMTP-демону проверяет соответствие послания установленным форматам, в результате чего письмо попадает в лапы свободного на данный момент процессу cleanup. В случае если письмо не поддается коррекции, то оно немедленно уничтожается. Стоит отметить, что отправитель не получит никаких извещений о данном прискорбном факте. Если же ничего аномального не было найдено, то сообщение беспрепятственно попадет в очередь incoming, а queue manager, как обычно, получит сигнал о появлении новых писем.

Идем дальше. Новая почта также может появиться в случае, если письмо невозможно доставить адресату и настройки системы диктуют в данном случае отправлять оповещения отправителю. При таком повороте событий процесс bounce или defer генерирует письмо с извещением о проблеме. Адресовано оно, конечно же, отправителю первоначального сообщения. Другим источником возникновения писем может быть настройка Postfix, заставляющая его отправлять администратору уведомления в случае проблем с SMTP или нарушения тех или иных политик, которые продиктованы настройками почтовой системы. Соответственно в обоих случаях, чтобы доставить письмо адресату, приходится, как обычно, отдать его процессу cleanup и затем отправить в очередь incoming.



Доставка почтовых сообщений

Мы разобрались с тем, какими путями письма появляются внутри почтовой системы. Теперь все, что находится в очереди incoming, нужно доставить получателям. Давайте посмотрим, как это происходит. Получив от cleanup уведомление о поступлении новой почты демон queue manager, управляющий очередями, перекладывает письмо в очередь active. Кстати, стоит заметить, что эта очередь очень маленького размера. В ней может находиться всего лишь несколько писем, которые в данный момент находятся в процессе доставки. Сделано это по двум причинам. Во-первых, для того чтобы при большой нагрузке менеджер очередей не потреблял слишком много памяти. Вторая причина состоит в том, что в случае, когда принимающая сторона не в состоянии получать письма с той скоростью, с которой Postfix старается их передать, приходится притормозить скорость отправки. С малой очередью это делать гораздо проще. Положив письмо в очередь active, демон queue manager создает запрос к процессу trivial-rewrite, с помощью которого удается определить точку на

значения сообщения. В ответ можно будет лишь узнать, локальный ли получатель или он живет на удаленной системе. Добавочную информацию о маршрутизации почты можно получить из файла `transport`. В зависимости от полученных данных `queue manager` входит в контакт с одним из следующих агентов доставки:

- **Local** – используется для доставки почты внутри локальной системы. Умеет работать со стандартными для UNIX почтовыми ящиками. В случае если для данного пользователя не заведено системных псевдонимов, в файле псевдонимов обычно это `/etc/aliases` и нет файлов локального перенаправления `.forward`, которые любой пользователь может создать в своей домашней директории, то письмо сразу же попадает в целевой почтовый ящик. В противном случае письмо будет передано туда, куда они указывают. Возможности локального агента доставки на этом не заканчиваются. Одновременно могут работать несколько агентов локальной доставки, но в то же время параллельная доставка нескольких писем в один почтовый ящик обычно не практикуется. Несмотря на то, что агент локальной доставки может самостоятельно справиться со всеми проблемами, по желанию администратор может задействовать механизмы, позволяющие передоверить доставку в почтовый ящик внешним программам. Примером такой программы может служить широко известный многим администраторам `prosmtp`.
- **Virtual** – агент виртуальной доставки представляет собой очень урезанную версию агента локальной доставки. Поэтому он может работать только с почтовыми ящиками в формате `mailbox`. В нем также отсутствует поддержка системной базы псевдонимов и файлов `.forward`. За счет таких ограничений данный агент доставки считается самым безопасным из всех. Благодаря своей природе он может легко работать с почтой, предназначенной для нескольких виртуальных доменов, располагающихся на одной и той же машине.
- **SMTP-клиент**, являющийся еще одним агентом, вступает в действие в тот момент, когда нужно доставить письмо пользователю удаленной системы. Обычно `queue manager` передает ему следующие данные: имя файла очереди, адрес получателя, хост или домен, куда нужно доставить почту, адрес отправителя. Первым делом с помощью DNS-запроса нужно получить список MX-серверов для целевого домена и отсортировать его по приоритету. Следующим шагом мы начинаем пробовать каждый адрес до тех пор, пока не найдем тот, который находится в рабочем состоянии. Обычно доставка идет одновременно сразу для нескольких доменов, поэтому в системах, через которые проходит большой поток почты, можно увидеть несколько SMTP-клиентов, работающих параллельно. Если доставка завершилась удачно, то SMTP-клиент модифицирует файл почтовой очереди так, чтобы было понятно, что он обработан. В противном случае данный клиент уведомляет `queue manager` либо о фатальной ошибке, либо о временных затруднениях. Проблемы первого типа могут быть вызваны отсутствием нужного пользователя удаленной системы, а вторые, к примеру, неполадками в сети или неработоспособностью принимающего сервера. В первом случае процесс

`bounce` посылает отправителю оповещение о неудаче предпринятого действия и делает запись в протокол работы почтовой системы с помощью `syslogd`. А во втором письмо помещается в специальную очередь `deferred`, где оно должно дожидаться следующей попытки доставки.

- **LMTP-клиент** работает точно так же, как и SMTP-клиент, разве что протокол используется другой. LMTP специально создан для того, чтобы доставлять письмо на локальный или удаленный сервер, выделенный для хранения почтовых ящиков. В таком качестве могут выступать Courier- или Cyrus-сервер. Большим плюсом такого подхода к доставке писем является то, что один сервер Postfix может раздавать письма разным серверам почтовых ящиков. В то же время никто не мешает одному серверу почтовых ящиков получать почту от нескольких серверов postfix одновременно.
- **Pipe mailer-интерфейс**, предназначенный для работы с внешними транспортными агентами. Примером такого взаимодействия может служить работа с UUCP. В то же время никто не мешает нам привязать к данному интерфейсу свой самодельный транспорт.

Как я уже говорил ранее, в случае если доставка не удалась, менеджер очередей переправляет файл, в котором хранится письмо, в директорию, где находится очередь `deferred`. В ней складываются отложенные до лучших времен письма. На файл с письмом ставится временной штамп, находящийся в будущем, соответственно следующая обработка этого файла произойдет именно в тот момент, на который указывает штамп. Периодически менеджер очередей проверяет, не пришло ли время предпринять следующую попытку доставки отложенных сообщений. В случае если есть необходимость выполнить очередную попытку раньше, чем истечет тайм-аут, нужно воспользоваться командой `postfix flush`.

Следующим интересным для нас понятием является очередь `corrupt`. В нее попадают все поврежденные, нечитаемые или неправильно отформатированные файлы почтовых очередей. Такая предосторожность позволяет изолировать подозрительные данные до тех пор, пока администратор системы не решит, что с ними делать. Впрочем, за несколько лет непрерывной работы моего сервера мне так и не удалось увидеть ни одного случая, чтобы сообщение попало в эту очередь.

Последняя из существующих в системе очередей называется `hold`. Здесь хранятся письма, доставка которых приостановлена по тем или иным причинам. Они будут находиться в этой очереди, пока не поступит специальная команда, выводящая их из состояния паузы.

Менеджер очередей может работать в разных режимах, обычно они называются стратегиями. В то же время никто не мешает их комбинировать между собой. Давайте рассмотрим характерные особенности каждого из них.

- **leaky bucket** – дырявое ведро. Жестко ограничивает количество сообщений в очереди `active`, тем самым защищая менеджера очередей от потребления чрезмерного объема памяти. Большая часть сообщений для доставки берется из очереди `incoming` и лишь малая часть из `deferred`.

- **Fairness** – честная очередь. В случае если очередь `active` не заполнена, письма берутся равномерно из `deferred` и `incoming`. Что дает залежавшейся почте больше шансов быть доставленной, даже если система очень сильно загружена.
- **slow start** – медленный старт. Данная стратегия позволяет бороться с заторами, которые могут образоваться, если принимающая сторона слишком медленно обрабатывает входящие запросы, а очередь предназначенной ей почты слишком велика. Соответственно нужно подобрать ровно такое количество одновременных потоков доставок, чтобы сервер успевал обрабатывать их все, не теряя ни одного из-за перегрузки и следующих за этим тайм-аутов. Подстройка обычно производится с помощью плавного уменьшения количества одновременных попыток доставки.
- **round robin** – круговая сортировка. Обычно менеджер очередей сортирует очередь на доставку по пунктам назначения. С помощью этой стратегии вероятность доставки более равномерно распределяется по всем пунктам назначения, что обеспечивает честные условия конкуренции для каждого пункта.
- **exponential backoff** – экспоненциальный откат. Почта, которая не может быть доставлена с первой же попытки, попадает в очередь `deferred`. После каждой неудачной попытки доставки тайм-аут увеличивается вдвое. Таким образом, почта для недееспособных узлов не будет потреблять слишком много ресурсов отправляющей системы напрасными попытками доставки.
- **destination status cache** – кэширование статуса. Менеджер очередей поддерживает таблицу, в которой некоторое время хранятся статусы предыдущих попыток доставки. Соответственно это позволяет сэкономить на повторных попытках, обращенных на недоступные узлы.

Кроме всех перечисленных функций менеджер очередей может делать еще одну приятную мелочь. В случае если нам нужно создавать стандартные оповещения, говорящие отправителю исходного сообщения о том, что какие-то почтовые адреса или целые домены перестали существовать или переменили свои названия, мы можем воспользоваться файлом `relocated`. Я думаю, это довольно удобная возможность. К примеру, представим себе такую ситуацию: в офисе есть пользователь с адресом `Irina.Petrova@firma.ru`. По какой-то причине она решила сменить фамилию. С женщинами в жизни такое случается вообще достаточно часто. Самым типичным решением было бы завести новый адрес типа `Irina.Vasilieva@firma.ru` и поставить пользователю задачу самостоятельно оповестить о смене адреса всех, с кем она когда-либо общалась по почте. Конечно, можно создать псевдоним первого адреса, указывающий на второй, но, согласитесь, это неудобно. Мы поступим проще, а именно: запишем пару из старого и нового адреса в `/etc/postfix/relocated` и `postfix` начнет отправлять уведомления о невозможности доставки с указанием нового адреса в ответ на каждое письмо, предназначенное старому адресу.

Закончив обсуждать ключевые подсистемы `Postfix`, хо-

телось бы немного поговорить о сопутствующих утилитах, о которых мы еще не успели поговорить.

- **mailq** – программа, позволяющая смотреть список писем, находящихся в почтовых очередях. На самом деле эта программа является всего лишь интерфейсом к демону `showq`, который вызывается через команду `sendmail -bp`.
- **postsuper** – предназначена для обслуживания почтовых очередей. Одним из применений является удаление какого-либо сообщения или повторная установка его в очередь на доставку. В то же время не стоит забывать об утилите `postqueue`, которая также создана для управления очередями. Единственное различие в них это то, что для работы с `postsuper` требуются права `root`, а для `postqueue` таких широких полномочий не нужно, хотя за счет этого теряется часть функционала.
- **postalias** – используется для создания баз псевдонимов или выполнения запросов к этим базам. Для совместимости с `sendmail` существует команда `newaliases`, делающая то же самое, что и `postalias`.
- **postconf** – показывает состояние конфигурационных переменных `Postfix`.
- **postlog** – команда, которую можно использовать для записи данных в протоколы работы `Postfix`. Полезна для использования в своих собственных скриптах.
- **postcat** – данная утилита, позволяющая посмотреть содержимое файла почтовой очереди.
- **postmap** – используется для выполнения запросов к вспомогательным таблицам или для создания таких таблиц из текстовых файлов. Перевод данных из текстовой формы в табличную довольно сильно ускоряет процедуру выполнения запросов.
- **postlock** – позволяет работать с блокировками, установленными `Postfix` на файлы. Обычно применяется для написания скриптов.
- **postkick** – предназначена для отправки сигналов по каналам межпроцессового обмена внутри `Postfix`. Удобна для организации взаимодействия между внутренними процессами `Postfix` и самописными скриптами.
- **spawn** – демон, позволяющий подключить внешнюю систему фильтрации содержимого сообщений. На данный момент находится в стадии активной разработки, и хотя работает достаточно надежно, но в силу особенностей реализации создает слишком большую нагрузку на систему. Скорее всего в следующих версиях `Postfix` будет заменен на что-то лучшее.
- **proxmtpar** – сервис, позволяющий централизованно выполнять запросы ко всем служебным таблицам вместо того, чтобы каждый из процессов выполнял их самостоятельно. Еще одним применением может быть предоставление `Postfix` доступа к файлам, находящимся за рамками ограничений, накладываемых `chroot`.

Вот и подошло к концу наше повествование. Надеюсь, что данный труд поможет тем, кто использует `Postfix`, лучше понимать механизмы его жизнедеятельности. Ну а для тех, кто пока еще не задумывался о необходимости поставить его на свой сервер, эта статья, возможно, станет первым шагом к такому решению.