Архитектура современных почтовых систем

Вступление

Знание основ освобождает от необходимости запоминать тысячи фактов. Попробуем подойти к построению почтовой системы, опираясь на этот принцип, и сделаем акцент не на настройку компонентов почтового сервера (об этом и так достаточно написано), а на взаимосвязь между компонентами и на протоколы, по которым они общаются между собой.

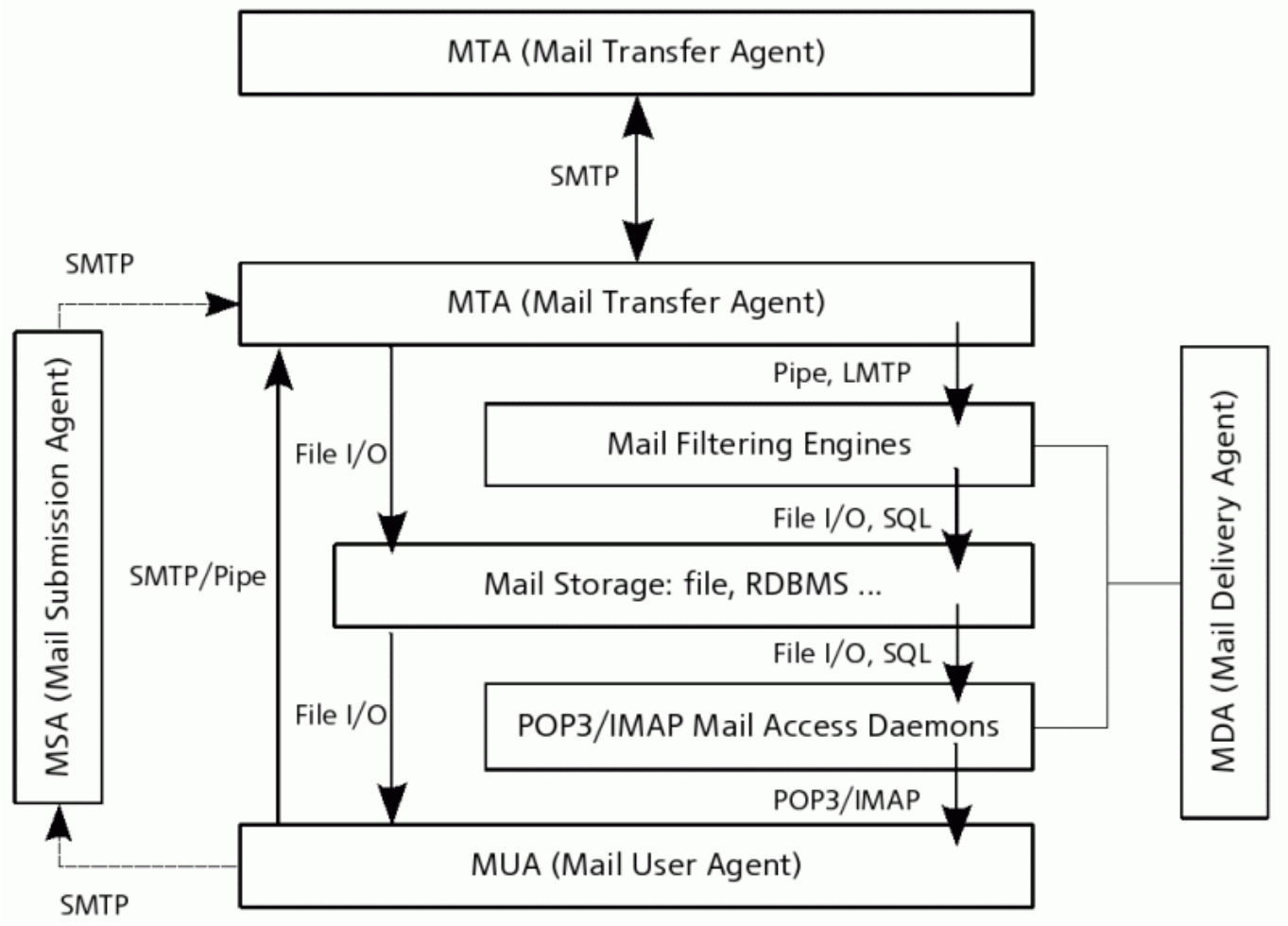
Материал этой статьи, наверное, больше подходит для учебника по основам системного администрирования, однако до сих пор такого учебника я не видел, а практика показывает, что даже многие администраторы с практическим опытом работы с трудом представляют себе общую картину того, как в целом работают сервисы электронной почты. Поэтому мы попытаемся заложить фундамент, необходимый для понимания механизмов работы почтового сервера.

Для того, чтобы упростить задачу, мы откажемся от рассмотрения таких монстров, как **Microsoft Exchange** или **Lotus Notes/Domino** , и сосредоточим внимание на открытом программном обеспечении и стандартных почтовых протоколах - их в большинстве случаев более, чем достаточно. Не будем мы рассматривать и протокол UUCP - многие из тех, кто знал, что это такое, уже успели про него забыть.

Взгляд с высоты птичьего полета

Главной отличительной чертой почтовой системы, построенной на открытом программном обеспечении, является ее модульность. Отдельные компоненты являются взаимозаменяемыми, для каждого компонента существует несколько реализаций. Замена компонентов, предназначенных для одной цели, в большинстве случаев катастрофой не является: базовая функциональность заменяемых компонентов в целом одинакова, настройки и данные тоже, как правило, можно перенести, а иногда даже использовать без изменения.

Принципиальная схема работы электронной почты и способы взаимодействия между ее основными компонентами представлены на следующем изображении:



Как создаются почтовые сообщения

С точки зрения пользователя почтовой системы существует только один компонент - это MUA (Mail User Agent), или, другими словами, его почтовый клиент (Mozilla, Outlook, The Bat!, KMail, mutt, pine, mailx, а также Web-приложения аналогичного назначения), предназначенный для создания, отправки, получения и чтения почтовых сообщений.

Формат почтовых сообщений описан в RFC 2822 (**Internet Message Format**) и в серии RFC с 2045 по 2049, которые посвящены формату MIME - **Multipurpose Internet Mail Extensions** (Format of Internet Message Bodies, Media Types, Message Header Extensions for Non-ASCII Text, Registration Procedures, Conformance Criteria and Examples).

Если коротко, то любое почтовое сообщение состоит из заголовков и тела, разделенных пустой строкой. Каждый заголовок, в свою очередь, состоит из имени и значения, разделенных двоеточием.

Следующие заголовки считаются обязательными:

- **From** - адрес и, возможно, полное имя отправителя
- **To** - адрес и, возможно, полное имя того, кому адресовано письмо (адресатов может быть несколько)

- **Subject** - тема письма
- **Date** - локальные дата и время отправления письма

Другими часто используемыми заголовками являются:

- **Cc (carbon copy)** - кому отправить копию письма (адресатов может быть несколько), при этом и основному адресату, и дополнительным, будет об этом известно
- **Received** - путь прохождения письма
- **Content-Type** - информация о том, каким образом письмо должно быть отображено

Многие MUA позволяют указать заголовок Bcc (Blind carbon copy) - его получатели письма не увидят никогда. Получатели, упомянутые в To и Cc, ничего не узнают о получателях, упомянутых в Bcc, а последние ничего не узнают друг о друге, но, тем не менее, они получат письмо и будут недоумевать: как в их почтовом ящике оказалось сообщение, в числе адресатов которого они не упомянуты? Этот заголовок используется преимущественно спамерами.

Имена заголовков могут содержать только 7-битные ASCII-символы. Значения заголовков не ограничены символами ASCII, но при наличии не ASCII-символов они должны использовать MIME-кодирование в форме **«=?charset?encoding?encoded text?=».**

Существуют следующие типы кодирования:

- 7bit - до 998 октетов на строку из диапазона [1..127]\{CR, LF}. Используется по умолчанию.
- quoted-printable - используется в первую очередь для US-ASCII-символов, но также содержит символы из других диапазонов.
- base64 - используется для двоичных данных.
- 8bit - до 998 октетов на строку из диапазона [1..255]\{CR, LF}. Этот тип кодирования в заголовках почтовых сообщений использовать нежелательно, о причинах мы поговорим позже.
- binary - произвольная последовательность октетов (фактически отсутствие какого бы то ни было кодирования). Этот тип кодирования использовать нельзя.

Таким же точно образом кодируется тело письма, при этом кодировка и тип кодирования указывается в заголовках Content-Type и Content-Transfer-Encoding. Вот пример типичного почтового сообщения:

```
Message-ID: <436F19FC.7050901@mail.domain1.com>
Date: Mon, 07 Nov 2005 12:10:20 +0300
From: User 1 <user1@domain1.com>
User-Agent: Mozilla Thunderbird 0.6 (X11/20040511)
X-Accept-Language: en-us, en
MIME-Version: 1.0
To:  user2@domain2.com
Subject: =?KOI8-R?Q?=F4=C5=D3=D4?=
Content-Type: text/plain; charset=KOI8-R; format=flowed
Content-Transfer-Encoding: 8bit
```

```
Привет!
```

Тело почтового сообщения может состоять из нескольких частей (которые используются для передачи вложений, не обязательно текстовых). Вот пример такого сообщения:

```
Message-ID: <436F2097.5060703@mail.domain1.com>
Date: Mon, 07 Nov 2005 12:38:31 +0300
From: User 1 <user1@domain1.com>
User-Agent: Mozilla Thunderbird 0.6 (X11/20040511)
X-Accept-Language: en-us, en
MIME-Version: 1.0
To:  user2@domain2.com
Subject: =?KOI8-R?Q?=F4=C5=D3=D4?=
Content-Type: multipart/mixed;
 boundary="------------070102080309020306010600"

This is a multi-part message in MIME format.
--------------070102080309020306010600
Content-Type: text/plain; charset=KOI8-R; format=flowed
Content-Transfer-Encoding: 8bit

Привет!


--------------070102080309020306010600
Content-Type: text/plain;
 name="file.txt"
Content-Transfer-Encoding: 7bit
Content-Disposition: inline;
 filename="file.txt"

Text file content

--------------070102080309020306010600--
```

Как происходит отправка почтовых сообщений

После создания сообщения MUA должен передать его MSA (Mail Submission Agent). В RFC 2476 (**Message Submission**) MSA описан как сервис, принимающий клиентские подключения на порту 678 по TCP/IP, и выполняющий первичную проверку почтовых сообщений на соответствие стандартам, авторизацию пользователей и блокирование UCE (Unsolicited Commercial Email - мы привыкли обозначать эту корреспонденцию словом «спам») еще на этапе отправки. Затем MSA должен передать письмо MTA (Mail Transfer Agent) - сервису, принимающему клиентские подключения на порту 25 по TCP/IP, который, в свою очередь, уже должен заняться доставкой письма непосредственно адресату. И в первом, и во втором случае должен использоваться протокол SMTP, описанный в RFC 2821 (**Simple Mail Transfer Protocol**) и RFC 1869 (**SMTP Service Extensions**), но MUA и MTA не должны общаться напрямую друг с другом.

На практике отдельных реализаций MSA не существует, а большинство реализаций MTA способны также выполнять функции MSA. Более того, для MSA практически никогда не конфигурируется порт 678, а все почтовые сообщения от MUA принимаются непосредственно на порт 25.

Поведение MTA после того, как он получил почтовое сообщение от MUA или MSA, зависит от настроек самого MTA, а также от домена, которому принадлежит почтовый адрес получателя. В простейшем случае (в отсутствии постоянного подключения к Интернет, постоянного реального ip-адреса и dns-имени - в сегодняшних реалиях такое происходит довольно редко) MTA вообще не берет на себя ответственность за пересылку письма, а просто отдает ее вышестоящему MTA, который для него является релеем (relay - MTA, через который производится пересылка). Релей может определить список сетей/хостов и/или список логинов/паролей, которым разрешено пересылать через него свои почтовые сообщения. Домены, обслуживаемые релеем, как правило, являются исключением: для них сообщения принимаются от кого угодно.

Релей, не устанавливающий никаких ограничений на пересылку почтовых сообщений, называется открытым релеем (open relay). Все открытые релеи ждет одна и та же участь:

- Сначала их услугами начинают активно пользоваться спаммеры
- Затем релей попадает в какой-либо черный список, и все MTA, выполняющие фильтрацию по черным спискам (а их сейчас большинство), перестают принимать от него почтовые сообщения

MTA, принимающий на себя ответственность за пересылку, сначала проверяет, обслуживает ли он домен адресата. В случае отрицательного решения MTA предпринимает попытку найти другой MTA, обслуживающий этот домен. Для этого он с помощью DNS-запроса получает список MX-записей домена, каждая из которых содержит приоритет в виде целого числа - чем оно меньше, тем MTA «главнее». В первую очередь предпринимается попытка отправить почтовое сообщение на главный MTA домена, а в случае его недоступности - по очереди на следующие за ним по приоритету (резервные) до тех пор, пока сообщение не будет отправлено. Резервные MTA могут передать сообщения на главный после восстановления его работоспособности, а могут выполнить доставку сообщения в почтовый ящик адресата самостоятельно.

Анатомия протокола SMTP

Давайте отправим тестовое письмо, используя протокол SMTP. Он является текстовым протоколом, поэтому для отправки сообщения вместо полнофункционального MUA мы можем использовать telnet (это необходимо для лучшего понимания общих принципов работы протокола SMTP, а также может быть очень полезно при отладке MTA):

```
# telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 mail.domain1.com ESMTP Postfix
```

```
EHLO host1.domain1.com
250-mail.domain1.com
250-PIPELINING
250-SIZE 10240000
250-ETRN
250-STARTTLS
250-AUTH PLAIN
250 8BITMIME
MAIL FROM: user1@domain1.com
250 Ok
RCPT TO: user2@domain2.com
250 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
Hello!
.
250 Ok: queued as 24D501771C
QUIT
221 Bye
Connection closed by foreign host.
```

Немного о том, что происходит в telnet-сессии. Получив приглашение, мы представляемся с помощью команды EHLO, и в ответ получаем список расширений, поддерживаемых тем MTA, к которому мы подключились. Сейчас нас не интересуют расширения, нам просто нужно отправить сообщение. Мы указываем отправителя с помощью команды MAIL FROM: и получателя с помощью команды RCPT TO:. После команды DATA мы вводим содержимое самого письма и завершаем его точкой на новой строке. С помощью команды QUIT мы отключаемся от MTA.

Обратите внимание: содержимое письма, мягко говоря, не соответствует никаким стандартам и совсем не похоже на те примеры «правильных» писем, которые мы приводили. Тем не менее, MTA принял письмо. Если до отправки письма включить прослушивание SMTP-трафика с помощью ethereal, а затем обработать экспортированные текстовые логи средствами grep и sed, мы получим следующие протоколы обмена.

MUA отправителя ↔ MTA отправителя:

```
Response: 220 mail.domain1.com ESMTP Postfix
Command: EHLO host1.domain1.com
Response: 250-mail.domain1.com
Response: 250-PIPELINING
Response: 250-SIZE 10240000
Response: 250-ETRN
Response: 250-STARTTLS
Response: 250-AUTH PLAIN
Response: 250 8BITMIME
Command: MAIL FROM: user1@domain1.com
Response: 250 Ok
Command: RCPT TO: user2@domain2.com
```

```
Response: 250 Ok
Command: DATA
Response: 354 End data with <CR><LF>.<CR><LF>
Message: Hello!
Response: 250 Ok: queued as 24D501771C
Command: QUIT
Response: 221 Bye
```

MTA отправителя ↔ MTA получателя:

```
Response: 220 mail.domain2.com ESMTP Sendmail 8.13.5/8.13.1; Mon, 7 Nov 2005
16:29:00 +0300 (MSK)
Command: EHLO mail.domain1.com
Response: 250-mail.domain2.com Hello mail.domain1.com [xxx.xxx.xxx.xxx],
pleased to meet you
Response: 250-ENHANCEDSTATUSCODES
Response: 250-PIPELINING
Response: 250-8BITMIME
Response: 250-SIZE
Response: 250-DSN
Response: 250-ETRN
Response: 250-AUTH DIGEST-MD5 CRAM-MD5 LOGIN PLAIN
Response: 250-STARTTLS
Response: 250-DELIVERBY
Response: 250 HELP
Command: MAIL FROM:<user1@domain1.com> SIZE=361
Response: 250 2.1.0 <user2@domain2.com>... Sender ok
Message: Received: from host1 (host1.domain1.com [xxx.xxx.xxx.xxx])
Message:     by mail.domain1.com (Postfix) with ESMTP id 24D501771C
Message:     for <user2@domain2.com>; Mon,  7 Nov 2005 16:30:58 +0300 (MSK)
Message: Message-Id: <20051107133058.24D501771C@mail.domain.com>
Message: Date: Mon,  7 Nov 2005 16:30:58 +0300 (MSK)
Message: From: user1@domain1.com
Message: To: undisclosed-recipients:;
Message:
Message: Hello!
Message: .
Message: QUIT
Response: 250 2.0.0 jA7DT0o5090086 Message accepted for delivery
Response: 221 2.0.0 mail.domain2.com closing connection
```

Наше однострочное сообщение обзавелось всеми необходимыми заголовками, и теперь вполне соответствует стандартам - MTA также выполнил часть функций MSA. Тем не менее, полагаться на это поведение нельзя - другие MTA или тот же самый, но с более строгими настройками, вправе отказать нам в приеме такого сообщения.

Что еще нужно знать об отправке

Протокол SMTP не позволяет однозначно идентифицировать отправителя сообщения, однако

существует возможность потребовать от отправителя авторизоваться - для этого служит расширение AUTH, описанное в RFC 2554 (**SMTP Service Extension for Authentication**). Для реализации этого расширения MTA используют механизм SASL, описанный в RFC 2222 (**Simple Authentication and Security Layer**), который позволяет использовать различные способы передачи и хранения логина и пароля, в том числе и те, которые используют не сам пароль, а его хэш.

Существует также возможность шифровать SMTP-трафик с помощью технологий TLS/SSL (Transport Security Layer / Secure Socket Layer), для чего могут использоваться 2 механизма:

- Устаревший протокол SMTPS - фактически это тот же самый SMTP, но шифруется весь трафик, начиная от начального приветствия MTA, при этом используется порт 465
- Расширение STARTTLS, описанное в RFC 2487 (**SMTP Service Extension for Secure SMTP over TLS**) - если клиент MTA (MUA, MSA или другой MTA) поддерживают его, то отправка сообщения может выглядеть, например, так (при этом используется порт 25 - тот же самый, что и для обмена незашифрованными сообщениями):

```
Response: 220 mail.domain1.com ESMTP Postfix
Command: EHLO host1.domain1.com
Response: 250-mail.domain1.com
Response: 250-PIPELINING
Response: 250-SIZE 10240000
Response: 250-ETRN
Response: 250-STARTTLS
Response: 250-AUTH PLAIN
Response: 250 8BITMIME
Message: STARTTLS
Response: 220 Ready to start TLS
Message: \026\003\001\000S\001\000\000O\003\001\000\033\2144\341\024A\361\
322EP\370yF\257\370x\273?\031<+\326\355\327\337X\347\207\305P\234\000\
000(\0009\0008\0005\0003\0002\000\004\000\005\000/\000\026\000\023\376\377\0
00
Message: \000\025\000\022\376\376\000\t\000d\000b\000\003\000\006\001\000
```

Расширение 8BITMIME, описанное в RFC 1652 (**SMTP Service Extension for 8bit-MIME transport**) позволяет использовать тип MIME-кодирования 8bit. Оно не является обязательным, поэтому использовать заголовки, закодированные таким образом, нежелательно.

Мы не будем рассматривать все расширения SMTP, скажем только, что каждому из них посвящен отдельный RFC, а искать их проще по названиям расширений в общем списке RFC.

Кроме использования протокола SMTP существует более простой способ отправить почтовое сообщение - это так называемая локальная отправка, поддерживаемая почти всеми MTA. Общепринятым способом локальной отправки является использование pipe-интерфейса программы sendmail, существуют MUA, которые поддерживают этот способ отправки (KMail, mutt, pine), а mailx вообще никаких других способов, кроме этого, не поддерживает. Простейший shell-скрипт, выполняющий локальную отправку, выглядит так:

```
#!/bin/sh
```

```
/usr/sbin/sendmail -t << EOF
From: user1@domain1.com
To: user2@domain2.com
Subject: Test Message
Hello!
EOF
```

Кто занимается транспортировкой сообщений

До сих пор мы говорили только об интерфейсах, предоставляемых MTA, но даже не произносили их названий. Только в протоколах ethereal видно, что на узле mail.domain1.com в качестве MTA используется Postfix, а на узле mail.domain2.com - Sendmail. Это именно та модульность, о которой говорилось выше: существует множество различных MTA, все они реализуют приблизительно одну и ту же функциональность, используют одни и те же протоколы, а как уж они устроены внутри - это их личное дело. Соответственно, дело администратора - выбрать тот MTA, который проще и гибче настраивается, надежнее и быстрее работает, да и вообще, более симпатичен.

Кратко перечислим несколько наиболее распространенных MTA:

- **Sendmail** - старожил, самый первый MTA, до сих пор сохраняющий свои позиции, отчасти по инерции (в BSD-системах и во многих дистрибутивах Linux именно он устанавливается по умолчанию), отчасти благодаря возможности очень гибкой настройки. Однако процедура конфигурирования, монолитная архитектура, работа с полномочиями суперпользователя и регулярно обнаруживаемые уязвимости вызывают множество нареканий, поэтому его использование уже становится дурным тоном.
- **Qmail** - первая более защищенная и модульная альтернатива Sendmail, за взлом которой учреждена до сих пор никем не полученная премия. Однако этот MTA так и не получил широкого распространения из-за лицензионных ограничений, благодаря которым многие дистрибутивы Linux не включают его в свой состав. Фактически, уже несколько лет Qmail не развивается автором, и чтобы получить нормально работающую систему на его основе, придется собирать его с множеством патчей.
- **Postfix** - еще одна альтернатива Sendmail, получившая значительно большее распространение благодаря вменяемой лицензионной политике, модульной архитектуре, высокой производительности, простой процедуре конфигурирования и использованию в некоторых дистрибутивах Linux и в Mac OS X по умолчанию. В журнале уже публиковался цикл статей Андрея Бешкова, посвященных Postfix, поэтому детально рассматривать этот MTA мы не станем.
- **Exim** - монолитный MTA подобно Sendmail, однако с безопасностью у него дело обстоит гораздо лучше. Кроме того, Exim претендует на звание MTA с максимальной функциональностью и самой логичной системой конфигурирования, которое, впрочем, требует больших усилий по сравнению с аналогичной процедурой для Postfix. Exim используется по умолчанию в Debian Linux. В журнале уже публиковались статьи, посвященные этому MTA, поэтому останавливаться на нем мы не будем, а пойдем дальше.

Доставка почтовых сообщений

Количество MTA, через которые пройдет письмо, пока не найдет своего адресата, в принципе не ограничено. На практике в большинстве случаев достаточно двух MTA, если домены отправителя и получателя обслуживаются разными MTA (и между ними есть прямой TCP/IP маршрут), или одного в противном случае. Задачей MTA после получения письма для своего домена является сохранение письма в постоянное хранилище, откуда его сможет прочесть адресат с помощью своего MUA. Доставкой письма в это хранилище занимается очень широкий класс ПО, который носит общее название MDA (Mail Delivery Agent).

Когда-то в процедуре доставки ничего сложного не было. Использовалась исключительно push-технология: MTA посредством собственного локального MDA доставлял почтовые сообщения куда-нибудь в /var/mail или /var/spool/mail, а MUA, появившиеся в те времена (mailx, mutt, pine), читали сообщения непосредственно оттуда - при этом они обязаны были находиться на одной машине с MTA или монтировать свои почтовые ящики, например, по NFS. Затем более популярными стали pop-технологии: /var/mail и /var/spool/mail выродились в специализированные серверные хранилища почты с собственными специализированными средствами доставки почты, появились сервисы, предоставляющие доступ к этим хранилищам по протоколам POP3 и IMAP по запросу самого MUA - причем новые MUA (Mozilla, Outlook, The Bat!) по другому себе жизнь уже не мыслят и о /var/mail и /var/spool/mail ничего не знают.

Форматы серверных почтовых хранилищ

На сегодняшний день существует два стандартных формата серверных хранилищ почты (**mbox** и **Maildir** - их стандартность проявляется в том, что доставлять в них почту и извлекать ее оттуда могут различные MDA) и несколько нестандартных (например, файловое хранилище Cyrus, похожее по идеологии на Maildir, но несовместимое с ним, и хранилище DBMail, использующее БД MySQL или PosgreSQL - если использовать эти хранилища, то вместе с их преимуществами мы одновременно резко ограничим себя в выборе MDA для доставки и извлечения почты). Впрочем, в любом случае мы говорим об открытых хранилищах, формат которых документирован (в худшем случае в исходных кодах MDA), поэтому нет никаких препятствий к тому, чтобы написать конвертор из одного хранилища в другое (и такие конверторы есть).

Нестандартные хранилища (Cyrus и DBMail) и MDA, работающие с этими хранилищами, лучше будет рассмотреть в отдельных статьях (статью о DBMail планируется опубликовать в следующем номере журнала, статья о Cyrus IMAP уже публиковалась), а сейчас мы сосредоточим свое внимание на стандартных хранилищах: mbox и Maildir.

Mbox - это целое семейство не совсем совместимых друг с другом форматов, различные модификации которых используются как в хранилищах почты на серверах, так и локально в MUA. Все эти форматы объединяет то, что все сообщения хранятся в одном файле, начинаются с поля From и отделены друг от друга пустой строкой. В процессе доставки новые сообщения дописываются в конец mbox-файла. Если вернуться к нашему примеру письма от user1@domain1.com к user2@domain2.com, то после его сохранения в конце mbox-файла (скорее всего это файл будет называться /var/mail/user2 или /var/spool/mail/user2) добавится следующее:

```
From user1@domain1.com Sun Oct 23 16:26:52 2005
Return-Path: <user1@domain1.com>
Received: from mail.domain1.com (mail.domain1.com [xxx.xxx.xxx.xxx])
        by mail.domain2.com (8.13.5/8.13.1) with ESMTP id j9NCQqTZ012628
        for <user2@domain2.com>; Sun, 23 Oct 2005 16:26:52 +0400 (MSD)
        (envelope-from user1@domain1.com)
Received: from host1.domain1.com (host1.domain.com [xxx.xxx.xxx.xxx])
        by mail.domain1.com (Postfix) with ESMTP id CEA8840F0
        for <user2@domain2.com>; Sun, 23 Oct 2005 16:33:25 +0400 (MSD)
From: user1@domain1.com
To: user2@domain2.com
Subject: Test Message
Message-Id: <20051023123325.CEA8840F0@mail.domain1.com>
Date: Sun, 23 Oct 2005 16:33:25 +0400 (MSD)

Hello!
```

Хранение всех сообщений в одном файле приводит к тому, что писать приложения, осуществляющие одновременное добавление новых сообщений и редактирование уже существующих, не так просто, как хотелось бы, не слишком надежно (возможностей повредить mbox предостаточно), а иногда и просто невозможно (блокировка mbox на запись при доставке сообщений делает невозможной параллельную доставку). Именно это и было причиной появления другого формата - Maildir.

Maildir - это каталог с тремя подкаталогами внутри: tmp, new, cur. При доставке сообщения оно помещается в файл в подкаталоге tmp, имя файла формируется из текущего времени, имени хоста, идентификатора процесса, создавшего этот файл, и некоторого случайного числа - таким образом, гарантируется уникальность имен файлов. После записи в файл всего сообщения создается жесткая ссылка на этот файл в каталоге new, а текущая ссылка из tmp удаляется - это делается для того, чтобы никакой другой процесс не смог прочитать содержимое сообщения до тех пор, оно не будет записано полностью. По такому же алгоритму при чтении сообщения (это может делать как MUA, так и другой MDA, предоставляющий доступ к Maildir по протоколу POP3 или IMAP) оно перемещается в каталог cur, при этом название файла изменяется: к нему добавляются пометки о прочтении, ответе, удалении и т.д.

Maildir++ - это дальнейшее усовершенствование Maildir с поддержкой вложенных каталогов IMAP (они должны начитаться с .) и квот.

Кто занимается доставкой сообщений в серверные хранилища

Все MTA содержат в своем составе локальные MDA для доставки в mbox, а иногда и в Maildir, но часто для доставки выгоднее бывает использовать внешние MDA общего назначения, позволяющие выполнить дополнительные операции с почтовыми сообщениями: переложить в другой почтовый ящик, переслать на другой почтовый адрес, передать другой программе по pipe-интерфейсу для дальнейшей обработки и т.д. Самыми распространенными внешними MDA общего назначения являются:

- **Procmail** - в основном используется для доставки сообщений в mbox (хотя существует

патч для поддержки Maildir).
- **Maildrop** - является частью проекта Courier, может доставлять почту в mbox, но чаще используется для доставки почты в Maildir. Более эффективно расходует системные ресурсы, чем Procmail, поддерживает виртуальных пользователей, информация о которых может храниться в Berkley DB, LDAP, MySQL и PostgreSQL.

И Procmail, и Maildrop используют pipe-интерфейс для приема почты от MTA: для каждого почтового сообщения создается отдельный процесс MDA, получающий сообщение на stdin. Это не слишком экономный способ обработки почты, особенно при больших объемах, поэтому MTA предлагают еще один способ получения сообщений - протокол LMTP, описанный в RFC 2033 (**Local Mail Transfer Protocol**). MDA, использующий LMTP, принимает сообщения по TCP/IP или UNIX-сокету подобно MTA, поэтому создавать отдельный процесс на каждое сообщение не нужно. Cyrus и DBMail предлагают MDA, использующие LMTP, для доставки в собственные хранилища, а вот для mbox и Maildrop таких MDA нет.

Кроме фильтрующих MDA общего назначения в процессе доставки сообщения в серверное хранилище также могут использоваться специализированные контент-фильтры. Такие фильтры могут выполнять функцию отсеивания спама и вирусов, в этом качестве часто используются SpamAssassin и SpamOracle в связке с популярными антивирусами. Они могут использовать как pipe-интерфейс, так и интерфейс LMTP. Последний вариант, как правило, является более производительным, но затрудняет использование контент-фильтра в MDA общего назначения - остается использовать MDA из комплекта MTA.

Организация доступа к серверным хранилищам

Наиболее популярными MDA, предоставляющими доступ к mbox и Maildir по протоколам POP3 и IMAP, являются:

- **UW IMAP** - самый простой из всех перечисленных. Работает под inetd/xinetd, не имеет вообще никаких настроек, в качестве механизма аутентификации использует только PAM, по умолчанию работает только с mbox (хотя существует патч для поддержки Maildir). Предоставляет доступ ко всем файлам домашнем в каталоге пользователя как к каталогам IMAP, даже если они не являются mbox - обязанность отображения только нужных файлов (действительно являющихся mbox) возлагается на MUA. Да и история безопасности у него очень нехорошая.
- **Courier IMAP** - часть проекта Courier, в который также входят собственный (но не слишком распространенный) MTA, уже упоминавшийся Maildrop, менеджер списков рассылки, средства совместной работы (groupware) и средства доступа к почте через Web. В качестве формата хранилища поддерживается только Maildir. Для аутентификации и Courier IMAP, и Maildrop используют общую библиотеку Courier Authentication Library, которая поддерживает хранение логинов, паролей, пути к Maildir пользователя и квот Maildir в Berkley DB, LDAP, MySQL и PostgreSQL - таким образом почтовые пользователи не обязаны иметь системные учетные записи. Кроме того, для аутентификации возможно использование PAM.
- **Dovecot** - позиционируется как сервер POP3/IMAP, при написании которого в первую очередь учитывалась безопасность: другими словами, он относится к UW IMAP и Courier IMAP как Qmail и Postfix к Sendmail. В качестве формата хранилища поддерживается как mbox, так и Maildir, для увеличения производительности (которая всегда была слабым местом при работе с mbox) используются специальные индексы. Для аутентификации

используется уже упоминавшийся механизм SASL - его же для аутентификации используют многие MTA и Cyrus IMAP. Но в целом по функциональности Dovecot пока не дотягивает до Courier IMAP.
- **Cyrus IMAP** - наиболее продвинутая почтовая система, которая помимо уже перечисленных возможностей, доступных в других MDA, включает возможности кластеризации и проксирования, быстрый и гибкий механизм фильтрации почтовых сообщений Sieve, описанный в RFC 3028 (**Sieve: A Mail Filtering Language**). Однако все компоненты Cyrus используют собственный формат хранилища почтовых сообщений и не умеют работать со стандартными mbox и Maildir.

В тех случаях, когда поддержка протокола IMAP не требуется, возможно, использование более простых MDA, предоставляющих доступ к mbox и Maildir только по протоколу POP3: к их числу относятся Cubic Circle (cucipop), QPopper и Popa3d.

Как происходит получение сообщений

И MDA, используемые для доставки почты в mbox и Maildir, и MDA, предоставляющие доступ к ним по протоколам POP3 и IMAP, взаимозаменяемы точно так же, как и различные MTA. С точки зрения MUA абсолютно не важно, какие MDA и какой тип хранилища используются на сервере.

Протокол POP3, описанный в RFC 1939 (**Post Office Protocol - Version 3**), подразумевает, что пользователи забирают сообщения из серверного хранилища и работают с ними в локальном хранилище своего MUA. Продемонстрируем использование этого протокола с помощью telnet:

```
# telnet localhost 110
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
+OK Hello there.
USER user1
+OK Password required.
PASS pwd1
+OK logged in.
STAT
+OK 1 422
LIST
+OK POP3 clients that break here, they violate STD53.
1 422
.
RETR 1
+OK 422 octets follow.
Return-Path: <user1@domain1.com>
Received: from mail.domain1.com (mail.domain1.com [xxx.xxx.xxx.xxx])
        by mail.domain2.com (8.13.5/8.13.1) with ESMTP id j9NCQqTZ012628
        for <user2@domain2.com>; Sun, 23 Oct 2005 16:26:52 +0400 (MSD)
        (envelope-from user1@domain1.com)
Received: from host1.domain1.com (host1.domain.com [xxx.xxx.xxx.xxx])
```

```
        by mail.domain1.com (Postfix) with ESMTP id CEA8840F0
        for <user2@domain2.com>; Sun, 23 Oct 2005 16:33:25 +0400 (MSD)
From: user1@domain1.com
To: user2@domain2.com
Subject: Test Message
Message-Id: <20051023123325.CEA8840F0@mail.domain1.com>
Date: Sun, 23 Oct 2005 16:33:25 +0400 (MSD)

Hello!
.
DELE 1
+OK Deleted.
QUIT
+OK Bye-bye.
```

После аутентификации с помощью команд USER и PASS мы получаем количество сообщений в почтовом ящике и их общий размер - для этого мы используем STAT. С помощью LIST мы получаем список сообщений и их размеры. Затем мы запрашиваем текст первого (и единственного) сообщения с помощью RETR, удаляем его из серверного хранилища с помощью DELE и отключаемся от сервера - для этого служит команда QUIT.

Теперь посмотрим, как происходит то же самое, но протоколу IMAP, описанному в RFC 3501 (**INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1**). Он подразумевает, что все сообщения даже после прочтения хранятся на сервере с пометкой о том, что они прочитаны, соответственно пользователи могут работать с одним и тем же почтовым ящиком с разных рабочих станций с помощью различных MUA одновременно.

```
# telnet localhost 143
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
* OK [CAPABILITY IMAP4rev1 UIDPLUS CHILDREN NAMESPACE THREAD=ORDEREDSUBJECT
THREAD=REFERENCES SORT QUOTA IDLE ACL ACL2=UNION STARTTLS] Courier-IMAP
ready.
Copyright 1998-2004 Double Precision, Inc.
See COPYING for distribution information.
1 LOGIN "user1" "pwd1"
1 OK LOGIN Ok.
2 SELECT "INBOX"
* FLAGS (\Draft \Answered \Flagged \Deleted \Seen \Recent)
* OK [PERMANENTFLAGS (\* \Draft \Answered \Flagged \Deleted \Seen)] Limited
* 1 EXISTS
* 1 RECENT
* OK [UIDVALIDITY 1131530433] Ok
* OK [MYRIGHTS "acdilrsw"] ACL
2 OK [READ-WRITE] Ok
3 FETCH 1:* (FLAGS)
* 1 FETCH (FLAGS (\Recent))
3 OK FETCH completed.
4 FETCH 1 (UID RFC822.SIZE FLAGS BODY.PEEK[])
* 1 FETCH (UID 1 RFC822.SIZE 443 FLAGS (\Recent) BODY[] {443}
```

```
Return-Path: <user1@domain1.com>
Received: from mail.domain1.com (mail.domain1.com [xxx.xxx.xxx.xxx])
        by mail.domain2.com (8.13.5/8.13.1) with ESMTP id j9NCQqTZ012628
        for <user2@domain2.com>; Sun, 23 Oct 2005 16:26:52 +0400 (MSD)
        (envelope-from user1@domain1.com)
Received: from host1.domain1.com (host1.domain.com [xxx.xxx.xxx.xxx])
        by mail.domain1.com (Postfix) with ESMTP id CEA8840F0
        for <user2@domain2.com>; Sun, 23 Oct 2005 16:33:25 +0400 (MSD)
From: user1@domain1.com
To: user2@domain2.com
Subject: Test Message
Message-Id: <20051023123325.CEA8840F0@mail.domain1.com>
Date: Sun, 23 Oct 2005 16:33:25 +0400 (MSD)

Hello!
)
4 OK FETCH completed.
5 STORE 1 +FLAGS (Seen)
* FLAGS (Seen \Draft \Answered \Flagged \Deleted \Seen \Recent)
* OK [PERMANENTFLAGS (Seen \* \Draft \Answered \Flagged \Deleted \Seen)]
Limited
* 1 FETCH (FLAGS (\Recent Seen))
5 OK STORE completed.
6 CLOSE
6 OK mailbox closed.
7 LOGOUT
* BYE Courier-IMAP server shutting down
7 OK LOGOUT completed
Connection closed by foreign host.
```

После аутентификации средствами команды LOGIN мы переходим в каталог IMAP INBOX с помощью SELECT «INBOX», при этом мы получаем количество существующих и еще не прочитанных сообщений. С помощью команды FETCH 1:* (FLAGS) мы читаем список всех сообщений с флагами, а с помощью FETCH 1 (UID RFC822.SIZE FLAGS BODY.PEEK[]) - первое (и единственное) сообщение. Затем мы помечаем это сообщение как прочитанное командой STORE 1 +FLAGS (Seen), закрываем почтовый ящик и отключаемся от сервера - для этого мы используем CLOSE и LOGOUT.

Что еще нужно знать о получении сообщений

Разумеется, функциональность POP3, а особенно IMAP приведенными командами не ограничивается. Например, IMAP включает в себя также механизм подписки на каталоги IMAP, возможность совместной работы с общими каталогами под различными учетными записями (shared folders), поиск в каталогах средствами сервера, частичную загрузку сообщений (в примере мы запросили все сообщение целиком, а могли бы запросить только некоторые интересующие нас заголовки) и т.д. POP3 и IMAP поддерживают шифрование, причем эта поддержка, как и в случае SMTP, возможна в 2 вариантах:

- Использование отдельных портов, на которых шифруется весь трафик (995 для POP3 и

993 для IMAP)
- Механизм TLS: используются стандартные порты, но после подключения MUA может инициировать шифрование с помощью команды STARTTLS

Существуют специализированные MUA, предназначенные для получения почтовых сообщений по протоколам POP3/IMAP и отправки их дальше в автономном режиме. Такой механизм может быть полезен для сбора почтовой корреспонденции в один почтовый ящик из разных ящиков в разных доменах. Как и использование для MTA внешнего релея, этот механизм незаменим для почтового сервера, не имеющего постоянного подключенния к интернет, постоянного реального ip-адреса и dns-имени, и подключающегося к провайдеру периодически для приема и отправки почты.

К числу автономных MUA относятся:

- **Getmail** - после получения почты по POP3/IMAP он сохраняет ее в серверное хранилище (mbox или Maildir) либо передает ее какому-нибудь фильтрующему MDA общего назначения (Procmail или Maildrop).
- **Fetchmail** - после получения почты по POP3/IMAP он пересылает ее указанному в конфигурационном файле MTA.

Что происходит дальше

MUA замыкают круговорот почтовых сообщений. Пересылка существующих писем или создание новых - это то, с чего мы начинали. Можно вернуться к схеме, с которой начиналась статья, и к описанию MTA - теперь ему решать, что делать с письмом дальше.

Рассмотренные нами компоненты позволяют строить почтовые системы, какой угодно сложности. Именно это и называется unix way: вместо одной огромной и монстрообразной программы, пытающейся уметь все, для решения задачи используется набор строительных блоков, из которых и собирается почтовый сервер - быстро, качественно и надежно.

Данная статья впервые была опубликована в журнале «Системный администратор», декабрь 2005. Автор статьи Евгений Прокопьев.