China-pub.com
下载

# 第12章 电子邮件

自第一个网络诞生以来，网络的主要用途便是收发电子邮件。最初，电子邮件只是一项简单的服务，把一台机器上的文件复制到另一台机器，并把它添加到接收端的邮箱文件内。尽管网络随着其复杂多样的路由要求和信息量的日益增长，要求更精心的设计，但电子邮件的基本原理始终是差不多的。

目前，邮件交换标准种类繁多。因特网上的站点采用的标准包含在 RFC-822内，后来又增加了一些RFC，说明如何采用与机器无关的方式，传输特殊字符等等。另外一些标准则针对目前的“多媒体邮件”传输，用于处理包含在多媒体邮件中的图像和声音。标准 X.400是由CCITT（国际电报电话咨询委员会）制定的。

目前，针对Linux系统，已有相当多的邮件传输程序得到了实施。其中最有名的是伯克利分校的sendmail，它适合于大量的平台，作者是 Eric Allman，目前，Eric又回到sendmail小组进行下一阶段的开发。现在，sendmail 5.56c有两个移植版本，其中一个将在第 14章介绍。目前正处于开发阶段的 sendmail版本是 8.9.3，有关详情，可参考 http://send.mail/org和http://sendmail.com。

常和Linux一起使用的邮件代理是smail 3.1.28，Curt Landon Noll和Ronald S.Karr编写并申请版权。下面，我们将其简称为 smail。虽然还有许多截然不同的邮件代理，但我们不打算对它们进行讨论。

与历史悠久的sendmail相比，smail显得尚不成熟。在处理小型站点的邮件信息时，由于此类站点没有复杂的路由要求，所以其过人之处表现不出来。但对于大型的站点，sendmail是绝对的赢家，因为它的配置方案要灵活得多。

smail和sendmail都支持一个必须自定义的配置文件集。除了能使邮件子系统正常运行的基本信息外（比如主机名），还有大量需要调整的参数。最初，sendmail的主要配置文件非常难以理解。它看起来就像是你的猫一直按住 Shift键，在你的键盘上小睡一样。和 sendmail的相比，smail配置文件的结构要好得多，而且易于理解，但用户不能调整邮件发送机的行为。然而，对小型的UUCP和因特网站点来说，两个配置文件的设置过程几乎是一样的。

本章，我们将为大家讲解何为邮件消息，以及作为管理员必须执行哪些操作。第 13章和第14章将指导大家初次设置 smail和sendmail。这里提供的信息足以应付小型站点的邮件系统运作，但还有更多、更新鲜的选项，你可以试试它们。

本章最后将简要介绍如何设置 elm，它是许多Linuxish系统上采用的常见邮件用户代理。

关于Linux上特有的电子邮件，其详情可参见 Linux Electronic Mail HOWTO文档，它是Guylhem Aznar(邮件地址guylhem@oeil.qc.ca),位于http://metalab.unc.edu/ LDP/HOWTO/Mail-HOWTO.html。elm、smail和sendmail的源代码也包含在许多文档内，在设置它们时遇到的问题，都可在这些文档内得以解决。如果想查找电子邮件的常见信息，可参考这方面的 RFC。它们列在本书最后的参考书目内。

China-pub.com
下载



## 12.1 何谓邮件消息

邮件消息一般由消息主体（发送者写的文本内容）和指定接收者、传输载体等的特殊数据组成，后者和我们平常看到的信封上的内容相似。

管理性数据分为两类：第一类数据是指与传输媒体相关的所有数据，比如发送人和收信人的地址。因此，这类数据也被称为信封。它可以随邮件消息的投递，通过传输软件进行转换。

第二类数据是处理邮件消息所需的所有数据，它不是某个传输机制特有的，比如说消息的主线，收信人清单和发送消息的数据等。许多网络中，都将其加入邮件消息内，形成所谓的邮件头。这正逐渐成为一个标准。邮件头从邮件主体偏移一个空行（一般习惯于在邮件消息内增加一个签名或.sig，其中通常包含作者信息和一段笑话或作者的座右铭。它从邮件消息偏移含有“ - ”的一行）。

现在，许多邮件传输软件都采用 RFC-822中勾划的消息头格式。其原意是为 ARPANET上的使用制定一个标准，但由于被设计成不依赖于任何一个环境，所以，稍作改动，它就可以适用于其他的网络，包括许多基于 UUCP的网络在内。

但是，RFC-822只是最常见的标准之一。随着日益增长的需求，比如数据加密、国际化字符集支持和多用途Internet邮件扩展（MIME），越来越多的标准也相应出台。

所有这些标准中，邮件头由若干行组成，中间用一个新行字符分开。一行又由一个字段名（从第一列开始）、字段本身，偏移一个冒号和空格字符或制表符组成。字段名不同，相应的字段格式和含义也会有所不同。如果下一行以标签开头，头字段就可能继续出现在下一行。字段的排列顺序是任意的。

一个典型的邮件头可能是这样的：

```
From brewhq.swb.de!ora.com!andyo Wed Apr 13 00:17:03 1994
Return-Path: <brewhq.swb.de!ora.com!andyo
Received: from brewhq.swb.de by monad.swb.de with uucp
        (Smail3.1.28.1 #6) id m0pqqlT-00023aB; Wed, 13 Apr 94 00:17
Received: from ora.com (ruby.ora.com) by brewhq.swb.de with smtp
        (Smail3.1.28.1 #28.6) id <m0pqoQr-0008qhC>; Tue, 12 Apr 94 2
Received: by ruby.ora.com (8.6.8/8.6.4) id RAA26438; Tue, 12 Apr 94
Date: Tue, 12 Apr 1994 15:56:49 -0400
Message-Id: <199404121956.PAA07787@ruby
From: andyo@ora.com (Andy Oram)
To: okir@monad.swb.de
Subject: Re: Your RPC section
```

通常，所有必要的头字段都是由你使用的接口生成的，比如 elm、pine、mush或mailx。但是，有些字段是可选的，也可以是用户添加的。比如， elm允许你对部分消息头进行编辑。其他的则由邮件传输软件添加。表 12-1列出了部分常见的头字段及其含义。

**表12-1 常见头字段及其含义**

| | |
|---|---|
| From | 其中包含发件人的邮件地址和真名 |
| To | 收件人的邮件地址 |
| Subject | 以简短的几句话，说明邮件内容。至少说明其主题 |
| Date | 发件日期 |
| Reply-To | 指定地址，发件人要求收件人直接回复到这个地址。如果你有若干个账户，但常用的只有一个时，这个字段便特别有用。该字段可选 |

China-pub.com 下载

(续)

| | |
|---|---|
| Organization | 发邮件的这台机器所属公司。如果是私有的，就可以令其为空，或插入“私用”。该字段可选 |
| Message-ID | 发件系统上邮件传输生成的一个字串，是该邮件消息独有的 |
| Received | 对你的邮件进行处理的每个站点（包括发件人和收件人的机器）都会将这一字段插入邮件头，给出其站点名、消息 ID、以及它收到该消息的时间和日期、该消息来自的站点，所使用的传输软件等等。这样一来，你就可以跟踪邮件消息经过的路径。假如发现问题，还可据此向出问题的那一部分的管理员投诉 |
| X-anything | 假如邮件头以X-字样开头，那么任何邮件相关程序都不能对它进行苛刻的检查。它用实现一些特殊的、附加的特性，这些特性尚未在 RFC里正式实施，或者根本就不打算制订到 RFC规范里。例如，Linux Activists邮件列表便采用了这种形式，频道（channel）由“X-Mn-Key:”头字段选择 |

该结构的一个例外是它的第一行。它以关键字“ From ”开头，后面跟一个空格或制表符，而非冒号。为将其与普通的 From:字段明确区分开，通常也把它写作 From_。其中包含了该邮件经过的路由——采用 UUCP的bang-path风格（下面还会详细解释）；包含了最后一台机器收到它的时间和日期；并包含了一个可有可无的部分，指出该邮件是从哪个主机收到的。由于该字段会由处理过该邮件的每一个系统重新生成一遍，所以有时会在信封数据的下面进行一番小计。

之所以要保留这个 From_字段，是为了保持与某些较老的邮件发送程序的“向后兼容”。但实事求是地说，它真正的用处并不大，只是邮件的用户接口必须依赖它，在用户的邮箱中，标注出一封信件的开头。要注意的是，邮件正文有时也会以“ From ”起头，所以为避免与之冲突，通常应该在其前面加一个换码字符“ > ”。

## 12.2 邮件如何发送

通常，邮件的编写是用 mail、mailx之类的邮件程序接口来完成的，也有人用更为复杂的邮件程序接口，比如elm、mush和pine。这类程序称为“邮件用户代理”，简称MUA。如果需要发送一条电子邮件消息，这个接口程序大多会将其交给另一个程序进行投递。这就是所谓的“邮件传输代理”，简称MTA。有些系统上，本地投递和远程投递采用的邮件传输代理是不一样的；而其他一些系统上，则可能采用同一个代理。远程投递所用的命令通常称为 rmail，本地投递所用的则称为lmail（如果有的话）。

当然，本地邮件的投递，不仅仅是把收到的消息添加到收件人的邮箱内。通常情况下，本地MTA可以识别别名（设置本地收件地址，使其指向另一个地址），和转发邮件（将用户的邮件重新定向到其他目的地）。另外，无法投递的邮件则必须返回，也就是说，随错误消息一起返回发件人。

对于远程邮件的投递，所用的传输程序和链接的属性有关。如果邮件必须在一个使用TCP/IP的网络上传递，则常用 SMTP。SMTP即简单邮件传输协议，其定义见 RFC-788和RFC-821。SMTP通常直接链接到发件人的计算机，和远程主机的 SMTP daemon（程序）协商消息的传输。

在UUCP网络中，邮件的投递一般不会直接进行，而是通过大量的中间系统，转发到目标主机。要想通过UUCP链接发送邮件，发送端的MTA在转发系统上利用uux，执行rmail，并将作为标准输入的邮件消息发给这个转发系统。

由于这一切是专门针对每条邮件消息来进行的，因此，可能使主邮件中间站负荷过多，

China-pub.com
下载


占据着大量磁盘空间的上百个文件将导致UUCP假脱机队列发生混乱（这是因为磁盘空间一般分为1024个字节的若干个数据块。即使一条消息只有 400个字节，也会占据1KB的空间）。因此，有些MTA允许你把准备发给远程系统的若干条消息统统装入一个单独的批处理文件内。这个批处理文件内包含的是SMTP命令，如果采用直接SMTP链接，本地主机一般会执行这些命令。这就是所谓的BSMTP或批处理SMTP。然后，批处理文件被发给远程系统上的 rsmtp或bsmtp程序，这样一来，远程系统就像对待一条普通的 SMTP链接一样，对这一批处理文件进行处理。

## 12.3 邮件地址

至于电子邮件，其地址由处理邮件的计算机名和能够为其系统识别的用户身份信息组成。后者可以是收件人的登录名，也可以是别的信息。其他邮件地址定址方案，比如 X.400，采用的是一个常用的“ 属性 ”集，用于在X.500目录服务器内查找收件人的主机。

计算机名的解释（也就是说，你的邮件消息最终将在哪个站点停留）和将这个计算机名和收件人的用户名组合在一起的方式和你所处的网络有极大的关系。

RFC-822标准中，规定因特网站点采用user@host.domain表示法，host.domain表示主机的完整资格域名。中间的@叫作“ at ”。由于这种表示法不涉及到目标主机的路由，而是给出主机名（唯一性的），所以这就叫作绝对地址。

在最初的UUCP环境中，比较流行path! host! user这种表示法，其中“ path ”表示消息在抵达目标主机之前，必须经历的一系列主机。这种结构叫作 bang路径表示法，因为感叹号（!）就叫作一个“ bang ”。如今，许多基于UUCP的网络都有经过修正的RFC-822，都能识别这类地址。

目前，这两类地址还不能混用。以 hostA!user@hostB这个地址为例，它不能清楚地说明“ @ ”符号优先于路径，还是路径优先于这个符号：是必须将邮件消息发给 hostB，再由其投递到hostA!user呢，还是将其发给hostA，再由它转发到user@hostB？

同时采用这两类地址符号的地址称为混合地址（hybrid address）。最典型的就是前面那个地址。一般把它解释为“ @ ”符号先于路径。因此，前面的地址意思是先向hostB发送邮件消息。

然而，RFC-822中，还有一种指定路由定址的方法：<@hostA,@hostB:user@hostC>表示主机hostC上的用户地址，hostC指的是通过hostA和hostB（按这个顺序），最后到达的目标主机。这类地址通常称为“ route-addr ”地址。

此外，还有“ % ”地址符：user%hostB@hostA，邮件消息首先被发送到hostA，再由后者将最右面的（这里，没有最右面的主机了）百分号沿伸到“ @ ”符号。现在的地址就变成了user@hostB，邮件程序将你的消息转发到hostB，后者再将其投递给相应的用户。这类地址有时被称为“ Ye Olde ARPANET Kludge ”，一般不鼓励大家采用这类地址。但事实上，许多邮件传输代理都会产生这类地址。

其他网络中仍然存在不同的定址方案。比如，基于 DECnet的网络，它采用两个冒号来作为分隔符，生成这样的地址：host::user（从RFC-822环境，试着抵达DECnet地址时，可能会用到host::user@relay，“ relay ”代表大家熟知的Internet-DECnet relay）。最后，X.400标准利用的是另一种截然不同的定址方案，它用若干个属性 -值对组成的集合来描述收件人，比如国家、公司等。

China-pub.com 下载

FidoNet上，每个用户是用诸如2:320/204.9此类的代码识别的，它由四个代表区域的号码(2代表欧洲)、网(320代表巴黎)，节点(本地hub)和端点(各个用户的计算机)组成。Fidonet地址可以映射为RFC-0822内的地址；上面的地址可写成Thomas.Quinot@p9.f204.n320.z2.fidonet.org。看来，还)是域名好记呀！

使用这些不同类型的定址方案时，有几点需要注意，详情情况将在随后的几个小节内进行讨论。但是对RFC-822环境来说，最好采用user@host.domain之类的绝对地址，其他的则不常用。

## 12.4 邮件路由的工作原理

将邮件消息导向到收件人主机的过程就叫作路由。除了找出发送站点到目标站点的路径外，路由还涉及到错误检查和传输速度和成本的优化。

UUCP站点和因特网站点处理路由的方式是有很大区别的。因特网上，将邮件数据导向收件人主机（只要能得知其IP地址）的主要任务是由IP层来完成的，而UUCP区内，路由必须由用户提供或由邮件传输代理生成。

### 12.4.1 因特网上的邮件路由

因特网上，完全依靠目标主机来执行邮件路由的选择。默认情况是通过查找目标主机之IP地址，将消息直接投递给它，把真正的路由数据留在IP传输层。

对许多站点来说，都想将所有封装好的邮件导向一个具有较强处理能力的邮件服务器，这类服务器能够对所有的邮件通信进行处理，并在本地就开始进行分发。为宣布这项服务器，站点须在DNS数据库内，为其本地域出版一条所谓的“ MX ”记录。MX代表“ 邮件交换者 ”，基本上可表示服务器主机愿意充当该域内所有主机的邮件转发者。 MX记录还可用于处理主机通信，这些主机本身没有接入因特网，比如UUCP网络或公司内网内用于传输机密信息的主机。

同时，MX记录还有一个与之关联的优先级。这个首选项是一个正整数值。如果一个主机对应若干个邮件交换者，邮件传输代理就会试着将邮件消息传给优先级最低的交换者，失败之后，再尝试优先级较高的交换者。如果本地主机本身就是目标地址的邮件交换者，它就会禁止向任何一台优先级高于它自己的MX主机转发邮件；其目的是为了避免邮件循环。

我们以Foobar公司为例。该公司打算让一台名为mailhub的机器来处理他们的邮件。所以，他们的DNS数据库内就有一条这样的MX记录：

```
foobar.com IN   MX 5 mailhub.foobar.com
```

该记录宣布mailhub.foobar.com作为foobar.com的邮件交换者，其优先级为5。希望将邮件消息投递给joe@greenhouse.foobar.com的主机，将在DNS数据库内查找foobar.com，并发现这条MX记录位于mailhub。如果没有找到优先级低于5的MX，这条邮件消息就会投递到mailhub，然后，再由后者将其分发到greenhouse。

上面的例子只是简单说明了MX的工作原理。如果想更多地了解因特网上的邮件路由，请参考RFC-974。

### 12.4.2 UUCP网络内的邮件路由

UUCP网络内的邮件路由比因特网上的复杂得多，因为传输软件自身是不会执行任何路由

China-pub.com
下载



的。早期，所有的邮件都必须利用 bang路径进行寻址。bang路径指定一系列邮件必须经由的主机，中间用感叹号隔开，最后才是用户名。如果你想为发给 Janet的邮件寻址，而她是Moria机器的用户，就应该采用这条路径： eek!swim!moria!janet。这样，就会把邮件从你的主机发送到Eek，再由Eek发给Swim，最后再投递到Moria。

这种路由的不足是它要求你必须记住网络拓扑、必须有快速链接等等。更糟糕的是，一旦网络结构发生了变化，比如链接已被删除或主机已发生增减，都可能导致邮件投递失败，究其原因，仅仅是你不知道这一变化。最后，如果你搬到另一个地方，很可能将不得不更新所有路由。

采用源路由的另一个必要原因是主机名指代不明。比如，我们以两个名为 moria的站点为例，一个位于美国，另一个位于法国。 moria!janet指的究竟是哪个站点呢？只有指定抵达moria的路径之后，才能得知。

解决主机名指代不明的第一步是：成立 UUCP映射工程。它位于 Rutgers大学，对所有正式的UUCP主机名及其UUCP邻居和地理位置进行注册，以保证主机名不会发生雷同。映射工程收集的这些信息作为 Usenet Maps出版，并定期通过 Usenet向外发布（用 UUCP映射工程注册的站点映射信息则通过新闻组 comp.mail.maps发布；其他一些组织可能也会发布自己的网络映射信息）。一个典型的映射系统条目（在删除批注之后）如下所示：

```
moria
    bert(DAILY/2),
    swim(WEEKLY)
```

这个条目是说Moria分别有一条指向Bert和Swim的链接，前者每天拨打前两次，后者则是一周一次。接下来详情讲解映射（map）文件的格式。

利用映射中提供的连接信息，便可自动生成从主机到任何一个目标站点的完整路径。这些信息通常都保存在 paths（路径）文件内，该文件有时也称为路径别名数据库（ pathalias database）。以通过Ernie抵达Bert的连接映射状态为例，从前面摘录的映射针对 Moria生成的路径别名如下：

```
moria ernie!bert!moria!%s
```

现在，如果你指定目标地址 janet@moria.uucp，你的MTA就会从上面选出一条路由，并采用bert!moria!janet这个地址向Ernie发送消息。

但是，根据 Usenet映射构建路径文件并不是上策。因为映射中提供的连接信息通常变动很大，而且有时已经过时。因此，只有主要主机才会采用完整的 UUCP全球映射来构建自己的路径文件。许多站点只是在其邻近维护自己的路径信息，将目标站点不在其数据库内的邮件消息发送给一个更“ 聪明 ”的主机，后者具有更完整的路径信息。这种方案称为“ 智能主机路由 ”（smart-host routing）。只有一条UUCP邮件链接的主机（也就是所谓的叶子站点）没有自己的路由信息；完全依靠它们的智能主机进行路由。

### 12.4.3 UUCP和RFC-822

迄今为止，要解决UUCP网络内邮件路由问题，最好的办法是修改 UUCP网络内的域名系统。当然，你是不能在 UUCP网络上查询域名服务器的。但不管怎样，许多 UUCP站点都有与

China-pub.com 下载

其内部路由对应的小型区域。在这些映射中，这些域声明了一台或两台主机作为自己的邮件网关，所以域内的每台主机都不必有自己的映射条目。网关对流入或流出该域的所有邮件进行处理。该域内部的路由方案在外界看来，是不可见的。

这对前面提到的智能主机路由方案来说，也非常有用。全局路径信息只由网关进行维护；该域内的次要主机只负责一个小型的手工写入的路径文件，该文件中列出了域内的路由和指向邮件中转器的路由。甚至于邮件网关，也不必再有针对全球范围内每台独立 UUCP主机的路由信息。对邮件网关来说，除了有完整的、用于自己提供服务的域的路由信息外，只需要在其数据库内有指向整个域的路由。比方说，下面的路径别名条目将把 sub.org域内站点的所有邮件路由到Smurf：

```
.sub.org swim!smurf!%s
```

地址是claire@jones.sub.org的所有邮件都将投到swim，后者的地址是smurf!jones!claire。

域名空间的分层式结构允许邮件服务器采用多种特定路径。例如，位于法国的一个系统，可能有用于“ fr ”子域的特定路由，但它会把目标为“ us ”域内的主机的所有邮件路由到美国的特定系统。通过这种方式，基于域的路由将大大减少路由数据库的规模并减少路由管理上的开支。

但是，在UUCP环境内采用域名的好处主要是顺应 RFC-822的需要，后者要求 UUCP网络和因特网之间必须容易沟通。目前，许多 UUCP域都有一条具有因特网网关的链接，由该网关充当其智能主机。通过因特网发送消息更为快捷，而且路由信息也更为可靠，因为因特网主机可用DNS来代替Usenet映射。

对基于UUCP的域来说，为了邮件能通过因特网抵达自己域内的主机，它们通常都有自己的因特网网关为其声明一条 MX记录。比如，假设 Moria属于orcnet.org域。Gcc2.groucho.edu充当其因特网网关。因此， Moria将采用gcc2作为自己的智能主机，发到域外的所有邮件都将通过因特网得以投递。另一方面， gcc2将为orcnet.org域声明一条MX记录，并将目标为orcnet站点的所有进入邮件都投递到 Moria。

现在，唯一的问题是 UUCP传输程序不能处理完整资格域名。大多数 UUCP站点的站点名都限制在8个字符之内，有的甚至更少，而且是不包括文字或数字的名字。多数情况下，点 . 是绝不可用的。

所以，需要一些映射进行RFC822名和UUCP主机名之间的转换。映射方法与实施方案有关。对FQDN（完整资格域名）和UUCP主机名之间的映射来说，常见方法是利用路径别名文件：

```
moria.orcnet.org  ernie!bert!moria!%s
```

该文件将根据指定完整资格域名的一个地址，产生一个纯粹的 UUCP样式的bang路径。有些邮件服务器会为此提供一个特殊文件；比如 sendmail采用的就是uucpxtable。

从UUCP网络向因特网发送邮件时，有时还需要逆向转换（俗称 domainizing，定域）。只要发件人在其目标地址中采用的是完整资格域名，这个问题便可迎刃而解。作法是：在把邮件消息转发给智能主机时，不要从信封地址中删除域名。但是，还有些 UUCP站点不属于任何一个域。它们通常会通过增加伪域 UUCP的方式，进行定域。

## 12.5 路径别名和映射文件格式

路径别名数据库提供基于 UUCP网络内的主要路由信息。典型的条目如下（站点名和路径

China-pub.com
下载


用制表符隔开）：

```
moria.orcnet.org      ernie!bert!moria!%s
moria                 ernie!bert!moria!%s
```

该条目使发给Moria的所有邮件消息都通过Ernie和Bert进行投递。对Moria的完整资格域名和其UUCP主机名来说，如果邮件服务器无法单独将它们映射为对应的域名空间，这两者都是必须指定的。

如果打算把目标为某域内主机的所有邮件消息都导向其邮件中转器，还需要在路径别名数据库内指定一条路径，给出作为目标域的域名，前面加上点 .。比如，如果sub.org域内的所有主机都可通过swim!smurf抵达，其路径别名条目可能会像这样：

```
\&.sub.org swim!smurf!%s
```

只有运行的站点不需要太多路由时，才可编写路径别名文件。如果必须为大量的主机进行路由，最好采用pathalias命令，根据映射文件创建一个别名文件。映射的维护要简单得多，因为只须通过编辑系统的映射条目，重建映射文件，便可增加或删除系统条目。尽管 Usenet映射工程出版的映射不再适用于路由，但小型的 UUCP网络则可能在自己的映射集内提供路由信息。

映射文件主要由站点列表组成，该列表列出了每个系统轮询或被其轮询的站点。系统名始于第一列，然后是用句点隔开的链接列表。如果新行以制表符开头的话，这个列表可能会持续若干行。每个链接则由站点名，和紧随其后的双括号中的成本组成。这个成本是一个算术表达式，由数字和象征性的成本值构成。以 #号开头的行则忽略不计。

以Moria例，它每天对swim.twobirds.com轮询两次，对bert.sesame.com每周轮询一次。而且，指向bert的链接只采用了一个非常慢的2 400bps的modem。Moria公布的映射条目如下：

```
moria.orcnet.org
        bert.sesame.com(DAILY/2),
        swim.twobirds.com(WEEKLY+LOW)

moria.orcnet.org = moria
```

最后一行将令其知晓其UUCP主机名。注意，它必须是DAILY/2（一天2次），因为一天拨打两次事实上将使该链接的成本减半。

对路径文件内列出的任何一个目标站点来说，利用源于此类映射文件的信息， pathalias命令能够计算出到这些目标站点的最优路由，并根据这些路由的信息，生成一个路径别名数据库。

pathalias还提供了另外两个特性：比如隐藏站点（也就是说，令站点只能通过网关进行访问）等等。关于pathalias命令以及链接成本的完整列表，可参见相关的手册页。

映射文件的批注中，一般还包含站点说明。指定批注的格式非常严格，所以一般从映射文件中获取。比如说，一个名为 uuwho的程序可以利用根据映射文件建立的数据库，以比较清晰的方式显示出这些信息。

在将自己的站点注册为一个组织（把映射文件分发给其成员）时，一般都必须找出这样的映射条目。

下面是一个映射条目示例（事实上，取自我自己站点）：

China-pub.com 下载

```
#N      monad, monad.swb.de, monad.swb.sub.org
#S      AT 486DX50; Linux 0.99
#O      private
#C      Olaf Kirch
#E      okir@monad.swb.de
#P      Kattreinstr. 38, D-64295 Darmstadt, FRG
#L      49 52 03 N / 08 38 40 E
#U      brewhq
#W      okir@monad.swb.de (Olaf Kirch); Sun Jul 25 16:59:32 MET DST
#
monad   brewhq(DAILY/2)
# Domains
monad = monad.swb.de
monad = monad.swb.sub.org
```

前两个字符之后的空格字符是制表符。条目中许多字段的含义都是相当清楚的；可从你注册的域收到一分详细说明。L字段最有趣；它给出了你所处位置的经度／纬度，并用于制定PostScript地图，显示各个国家的所有站点乃至全世界的每个站点（要知道，它们的数目是相当惊人的，定期在news.lists.ps-maps公布）。

## 12.6 elm的配置

elm代表“ electronic mail ”（电子邮件），是最理想的命名工具之一。它提供了一个非常有用的全屏接口。我们不打算讨论elm的用法，但将详细讨论其配置选项。

理论上讲，可以运行未经配置的elm，如果运气好的话，一切都会正常运行。但有几个选项是必须设置的，尽管它们用的地方较少。

启动时，elm读取elm.rc文件内的配置参数集，该文件位于 /usr/lib/elm。然后，再试着读取根目录下的.elm/elmrc文件。通常，这个文件不需要你自行编写。你从 elm的“ 选项 ”菜单中，选定“ Save ”选项，便可建立它了。

私用elmrc文件的选项集也可用于全局 elm.rc文件。你自己的私用 elmrc文件中的多数设置优先于全局文件内的设置。

### 12.6.1 全局elm选项

全局elm.rc文件中，必须设置和你的主机名相关的选项。比如，在 VirtualBrewery公司网络内，vlager主机所用的全局elm.rc文件就应该包含下面的内容：

```
#
# The local hostname
hostname = vlager
#
# Domain name
hostdomain = .vbrew.com
#
# Fully qualified domain name
hostfullname = vlager.vbrew.com
```

这些选项设置了elm的本地主机名。尽管该信息较少使用，但仍应该设置这些选项。注意，

China-pub.com
下载


只有在全局配置文件内设置它们时，这些选项才会生效；如果是在你自己的私用 elmrc文件内配置的，它们就会被忽略。

### 12.6.2 国家特有字符集

近来，人们一直期望对 RFC-822标准进行修改，使之能支持诸如纯文本、二进制数据、PostScript文件等等之类的不同类型的信息。涉及到这些不同类型支持的标准字符集和 RFC常被称为多用途因特网邮件扩展 (Multipurpose Internet Mail Extensions,MIME)。它们使收件人能够得知邮件消息中是否采用了除标准 ASCII之外的其他字符。比如邮件消息中采用了法语重音或德语元音等。这是通过扩展 elm来支持的。

Linux内部用以代表字符的字符集通常称为 ISO 8859-1，这是它的标准名。它也被称为 Latin-1。对采用该字符集内字符的任何一条邮件消息来说，其邮件头都应该有下面这一行：

```
Content-Type: text/plain: charset=iso-8859-1
```

收件系统应该识别出这个字段，并采取相应的措施显示邮件消息。针对文本／纯文本消息的默认设置是us-ascii字符值。

为了能够显示带有 ASCII和其他字符集的邮件消息， elm必须知道如何打印这些字符。默认情况下， elm在收到这样的邮件消息（除了 us-ASCII字符之外，其中还有一个 charset字段，或除了文本／纯文本以外，还有内容类型的消息）时，它将试着利用一个名为 metamail的命令，显示这一消息。需要 metamail命令显示的邮件消息将在屏幕上的带有 M的第一列中出现。

由于Linux原始字符集是ISO-8859-1，所以没必要调用 metamail命令来显示使用了该字符集中字符的邮件消息。如果 elm得知显示端能够识别ISO 8859-1，它就不会采用 metamail命令，而是直接在屏幕上显示邮件消息。这是通过在全局 elm.rc配置文件内设置下面的选项来完成的：

```
displaycharset = iso-8859-1
```

注意，即使在你根本不可能收发除了 ASCII之外其他字符的邮件时，仍然应该设置这个选项。这是因为发送此类邮件的人们通常会对他们的邮件服务器进行配置，通过默认设置，将恰当的Content-Type:field放入邮件头，不管它们是否发送其中只包含 ASCII字符的消息。

但是，仅仅在 elm.rc文件内设置这个选项是不够的。因为，在用其内置的 pager显示邮件消息时， elm将针对每个字符，调用一个库函数来判断该字符是否能够打印。默认情况下，这个函数只能将ASCII认作是可以打印的，而把其他所有字符显示为“ ？”。所以，要解决这个问题，就应该把环境变量LC_CTYPE设置为ISO8859-1，后者将要求库函数把 Latin-1也认作是可以打印的。自 libc-4.5.8以来，已经可以支持这一设置和其他特性了。

在发送的邮件中，如果其中包含取自 ISO 8859-1的字符，就应该保证在 elm.rc文件内设置另外两个变量：

```
charset =  iso-8859-1
textencoding = 8bit
```

这两个变量使 elm在邮件头，把这个字符集当作 ISO-8859-1报告出来，并将它当作一个 8位值进行发送（默认情况是将所有的字符拆为 7位值）。

当然，这些选项还可以在私用的 elmrc文件而不是全局 elmrc文件内进行设置。