第一部分　Linux网络管理员指南

Olaf Kirch　著

第1章 网络基础

自世上出现“电信”或“远程通信”技术那一天起，“连网”（Networking）的概念恐怕便已产生了。想想生活在石器时代的人，若相距较远，那么可通过鼓声，相互间传递消息。现在，假定穴居人A想邀请穴居人B参加投掷石块的游戏。但是，他们离得实在太远了，以至于B听不到A的鼓声。那么，穴居人A该采取何种对策呢？他面临着三种选择：1)步行到B处；2)做一只更大的鼓；3)请求住在A和B之间的穴居人C，帮助转发消息。最后一种选择便叫做“连网”！

当然，人类进化到今天，我们再也不会使用老祖宗那些简陋的工具。今天，我们有了计算机，它们通过电缆、光纤、微波以及其他媒介，相互间可以“交谈”。相距遥远的人，可使用计算机，约定在星期六玩一场足球比赛。后面，我们将探讨达成这一目标的途径及方式。但是，不打算涉及通信线缆的问题。当然，足球比赛也请暂时抛在一边吧！

在此，有两种类型的网络是我们感兴趣的：以UUCP为基础的网络，以及以TCP/IP为基础的网络。UUCP和TCP/IP均属于“协议套件”或者“软件包”的类别，它们提供了在两台计算机之间传输数据的途径。在本章，我们打算来看看这两种类型的网络，并讨论它们的基本原理。

我们将“网络”（Network）定义成一系列“主机”（Host）的集合，不同主机相互间能够通信。通常，这种通信要依赖某些“专用主机”或“服务器”（Server），在参与通信的两部主机之间，对数据进行转发或“中继”（Relay）。当然，最常见的主机便是计算机，但也并不全是。例如，X终端或智能打印机也属于主机的一种类型。小范围内的主机集合亦称作“站点”（Site）。

无论人还是主机，相互间想实现通信，不采用某种形式的语言或代码，那是根本不可能的。在计算机网络中，这些语言统称为“协议”（Protocol）。但是，不要把它们想象成书写的那种“协议”。相反，它们实际是一些高度规范化的代码，严格约束着通信双方的行为。用最简单的话来说，计算机网络中采用的协议就是一系列非常严格的规则，规定着两个或更多主机相互间如何交换消息。

1.1 UUCP网络

UUCP是“Unix到Unix拷贝”（Unix-to-Unix Copy）的简称。开始时，它以一个软件包的形式出现，通过串行线路传输文件，对文件传输进行安排和调度，并发起程序在远程站点的执行。自70年代末首次问世以来，尽管它在许多方面发生了重大变化，但就其提供的服务来说，却变化不大。它仍然主要应用于广域网（WAN）环境。在这种环境中，主机的连接要通过拨号电话线路来进行。

UUCP是由美国贝尔实验室于1977年开发出来的，用于在他们的Unix开发站点之间实现通信。到1978年年中，这个网络总共已连接了超过80个站点。除了运行非常原始的电子邮件（E-mail）服务之外，还支持远程打印功能。然而，该系统的用途还是发布新软件和错误修正

文件（补丁文件）。今天，UUCP已经不再局限在那样的环境内。在这个网络内，目前运行着各式各样的平台，比如AmigaOS，DOS，Atari的TOS等等。它们有的提供免费服务，有的则提供商业服务。

UUCP网络最大一个缺点便是带宽较低。一方面，电话设备紧紧限死了最大的传输速度；另一方面，UUCP中极少存在永久性的连接。相反，各主机以固定的周期，通过拨号方式建立与对方的连接。所以大多数时候，邮件消息往往不能一下子送至目的地；而是先躺在某个主机的硬盘里，等待下一次连接建立的时候再发送出去。

尽管存在这些限制，目前世界上仍运行着为数众多的UUCP网络。它们主要由一些业余人员负责维护，有时收取低廉的价格，让私人用户接入网络。UUCP之所以仍然受到某些人的青睐，主要原因便是和通过专线永久性接入Internet相比，它需要的费用便宜得多。要想使您的计算机成为UUCP的一个节点，需要的全部家当只有一部Modem、一个适合UUCP运行的软硬件环境以及另外一个UUCP节点（愿意为你提供邮件和新闻转发服务，亦即你的“上游节点”）。

如何使用UUCP

UUCP的工作方式非常简单。从它的名字（Unix到Unix拷贝）便可知道，它主要负责将文件从一个主机拷贝（复制）到另一个主机。但除此以外，它还允许在远程主机上采取一些特定的行动。

假定你的机器有权访问一个名为Swim的主机，现在想让它为自己执行lpr这条打印命令。如何做到呢？可在自己的命令行键入下述命令，让这本书在Swim机器上打印出来：

```
$ uux -r swim!lpr !netguide.dvi
```

这样一来，便可指示uux（来自UUCP套件的一个命令）为Swim安排一项作业。在这个作业中，包括一个输入文件，名为netguide.dvi；另外，还包括将该文件送给lpr的请求。其中，-r参数告诉uux不要马上呼叫远程系统，而是将这个作业暂存下来，等稍后建立了一次连接再说。这个过程称作“缓冲”（Spooling）。

UUCP的另一个特点便是允许经由几个主机，对作业及文件进行转发，只要各主机相互间能够协作。现在，假定上例的主机Swim建立了与Groucho的一条UUCP链路，后者维护着一个大型的应用程序档案库。那么，为了将文件tripwire-1.0.tar.gz下载到自己的站点，需执行下述命令：

```
$ uucp -mr swim!groucho!~/security/tripwire-1.0.tar.gztrip.tgz
```

创建的这个作业会请求Swim帮自己从Groucho处取得文件，并将其发至自己的站点。在自己的站点，UUCP会将文件保存为trip.tgz，并发一封电子邮件，通知文件已经到达。整个过程分三步走：第一步，从自己的站点将作业发给Swim。第二步，在Swim处，下一步请求建立同Groucho主机的连接，并从它那里下载回指定的文件。最后一步是将文件从Swim实际传回自己的主机（站点）。

对UUCP网络来说，目前它提供的最重要的一种服务便是电子邮件和新闻。以后，我们还会讲述这方面的问题。在这里，仅对它们进行一番简要的介绍。

电子邮件，简称E-mail、email，使我们能直接与远程主机交换消息或信件，而不用知道如何访问这些主机。将一条消息（信件）从自己的站点引导至目标站点的任务完全是由邮件控制系统执行的。在UUCP环境中，邮件传输通常需要在一个相邻的主机上执行rmail命令，

将邮件正文和接收方的地址传递给它。随后， rmail会将邮件转发至另一个主机……以此类推，直到最后抵达目标主机。在本书第12章，还会对此详述。

至于“新闻”（News），可想像成一种分布式的电子公告板系统。通常说到“新闻”的时候，是指通过Usenet新闻组发布的新闻，它目前是应用得最广的新闻交换网络，总共有大约12万个成员站点。Usenet的起源可追溯到1979年。当时，与新出的Unix-V7一道发布了UUCP之后，三位应届毕业生产生了一个想法，打算在Unix社区中实现常规的信息交换。他们设计了一些脚本，形成了世界上第一个网络新闻系统的雏形。1980年，这个网络在北卡州的两所大学里，成功地联通了Duke，UNC和PHS等大学，同时连通了北加州的两家大学。以此为基础， Usenet逐渐发展壮大。尽管它起源于UUCP网络，但现在已跨越了多种形式的网络，并不单单UUCP一种。

在Usenet中，最基本的信息单元便是“文章”或者“帖子”。所有文章都要投递到与主题对应的新闻组内。按主题分类，各个新闻组构成了一个层次分明的结构。由于每天发布的帖子数量众多，大多数站点（亦称“新闻组服务器”）只接收自己选择的一部分新闻组。即便这样，每天平均也会增加近60MB的新帖子。

在UUCP世界中，通常先从要求的新闻组中收集好所有文章，然后通过一条 UUCP链路发送出去。如果数量较大，一次传不完，便打包后，分成数次传输。打包（压缩）的新闻送至接收站点，由它们执行rnews命令，进行解包和其他处理。

最后，UUCP也是许多通过拨号连接的文件下载站点的一种传输媒介。这些站点通常允许公共访问，让人们免费下载需要的文件或软件。通常要通过 UUCP拨入这些站点，以一名“访客”（Guest）的身份登录，然后从公开的档案区下载需要的文件。这些 Guest账号通常以guest或anonymous（匿名）作为登录用户名，然后用 uucp/nuucp或类似的信息作为登录密码使用。

1.2 TCP/IP网络

尽管UUCP特别适合希望以低廉费用建立拨号网络连接的用户，但在其他许多情况下，这种“存储转发”技术也显得非常不灵活。比如在局域网中，数量不多的机器位于同一幢建筑物中，甚至位于同一层楼、同一个房间里。这些机器相互连接在一起，营造一个能够协同工作的环境。在这种情况下，我们需要在不同的主机间共享文件，或者在不同的机器上运行分布式的应用程序。

所有这些任务都要求以一种全然不同的方式来建立网络连接。此时，不再是将整个文件都“存储”起来，同时为其赋予一个作业说明，而是将所有数据都分割为较小的数据包（Packet），立即转发至目标主机。抵达目标主机后，再在那里重新“组装”回原来的样子。这种类型的网络叫作“包交换”网络。它的最基本的一个应用便是通过网络运行分布式的应用（程序）。当然，这样做的代价会比 UUCP高出许多 —— 这主要反映在软件和协议的复杂程度上。

目前，许多系统（包括非Linux站点）选择的协议都是TCP/IP。在这一节里，我们打算就其基本概念作一番论述。

1.2.1 TCP/IP网络入门

TCP/IP的历史可追溯到1969年由美国国防部高级研究计划局（DARPA）投资进行的一个研究项目。项目的宗旨是建立一个实验性网络，名叫 ARPANET（阿帕网）。1975年，项目完

成，网络正式投入运行。

1983年，新协议套件TCP/IP被接纳为正式标准，网络上的所有主机都必须使用它。当这个阿帕网最终进化成Internet后（阿帕网于1990年正式退出历史舞台），TCP/IP的应用范围已不仅仅在Internet之内。它最引人瞩目的应用是在局域网中；但随着快速数字电话设备比如ISDN的问世，TCP/IP也昭示着拨号网络的未来。

为便于下文对TCP/IP的讨论，我们在此打算以位于芬兰某地的Groucho Marx大学（GMU）为例。在这所大学中，大多数系都运行有自己的局域网（ LAN）。一些系和别的系共享一个网络，另一些系则同时运行着几个网络。所有网络相互间都是连接起来的（互联）。整个校园网通过一条高速链路，接入Internet。

假定我们的机器挂接的是数学系的 LAN，名字叫做Erdos。要想访问物理系中名为 Quark的一个主机，需要执行下述命令：

```
$ rlogin quark.physics
Welcome to the Physics Department at GMU
(ttyq2) login:
```

在提示行，需要输入自己的登录名： Andres，以及正确的密码。随后，便可获得在 Quark机器上的一个外壳，可在其中键入任何命令，就像自己亲身坐在那个系统的控制台前一样。退出这个外壳后，便可返回自己机器（本机）的提示行。

刚才，我们已试验了由TCP/IP提供的一种实时的交互式应用：远程登录！

登录进Quark的时候，有时也想运行一个以 X11为基础的应用，比如一个函数演算程序，或者一个PostScript格式预览程序。要想告诉应用程序你希望将它的窗口显示在自己主机的屏幕上，必须设置DISPLAY环境变量：

```
$ export DISPLAY=erdos.maths:0.0
```

如果现在启动程序，它便会同你的 X服务器联系，而不是同 Quark的服务器联系，并将所有窗口都显示在自己的屏幕上。当然，这要求你在 Erdos机器上运行X11。这里的关键在于，TCP/IP协议允许Quark和Erdos两部主机来回传送 X11数据包，从而营造在单个系统上运行的“ 假象 ”。此时的网络永远是“ 透明 ”的。

在TCP/IP网络中，另一个非常重要的应用是 NFS，亦即“ 网络文件系统 ”（Network File System）。它是让网络“ 透明 ”的另一种形式。 NFS主要用来让我们“ 复制 ”其他主机的目录结构，令其看起来就像本机的文件系统。举个例子来说，所有用户的“ 主目录 ”（Home Directory）都可存放在一台中心服务器中。从这台机器，局域网内的所有主机都复制它的目录。这样一来，用户实际可登录进入网内的任何一台机器，并发现自己处在一模一样的主目录中。类似地，可将那些要求占用大量磁盘空间的应用程序（如 TeX）安装到一台机器，然后将它的目录导出至其他机器。到本书第 10章，我们还会对NFS进行详细论述。

当然，这些仅仅是通过 TCP/IP网络能够做到的一些事情的例子。事实上，用它能做到的事情几乎是无限的。

接下来，我们打算就 TCP/IP的工作原理做一番详细探讨。只有掌握了这方面的知识，才能对自己机器的配置方法做到心中有数。首先从硬件开始，再从它慢慢地延伸开去。

1.2.2 以太网

目前在LAN中广泛采用的硬件形式叫做Ethernet，即“ 以太网 ”。其中包含了一条电缆，主

机通过接头、分路器或者收发器加以连接。简单以太网的安装费用十分低廉，最常见的传输速度是10Mbps（每秒10兆位，而非10兆字节），但也有逐步向100Mbps（百兆网）过渡的趋势。

以太网可划分为三种实施形式。根据采用的电缆规格，可分别称为粗缆、细缆以及双绞线网络。其中，细缆和粗缆以太网使用的都是同轴电缆，只是线缆粗细以及与主机的连接方式有所区别。细缆采用的是BNC接头，俗称T头，需要“拧”入计算机背后的一个圆柱形接头（做在网卡上）。粗缆要求在线缆上打一个小孔，然后用一个“Vampire Tap”连接一个收发器。随后，一个或多个主机建立与这个收发器的连接。细缆和粗缆以太网电缆分别最多可以长达200和500米，所以也叫做10Base-2和10Base-5网络。双绞线电缆则由两个铜芯线对绕合而成，类似于普通的电话线。但是，它通常要求新增额外的硬件。人们也把它叫做 10Base-T网络，其中的“T”代表Twisted pair（双绞线）。

尽管在粗缆以太网中增加一个主机显得比较麻烦，但在增加主机的同时，不会造成网络的中断。而要想在细缆网络中新增一个主机，则必须让网络服务暂停几分钟，因为必须对线缆进行处理（通常需要割断），插入新的接头。

大多数人都愿意选择细缆连接，因其造价相当低廉：一张网卡最多 100元人民币，线缆每米1~3元，其他便不再需要任何费用。但是，假如网络的规模较大，粗缆以太网就显得更为恰当。例如，GMU大学数学系的以太网采用的便是粗缆连接，所以每次有一个主机加入网络时，不会造成整个网络的中断。

以太网技术的一个缺陷是电缆长度（布线长度）有限，所以只适用于 LAN的建设。但是，利用转发器、网桥或路由器，几个独立的以太网网段也可以相互连接到一起。其中，转发器（Repeater）的作用最简单，只是在两个或更多的网段之间复制信号，使各个网段表面上似乎合并成了一个统一的以太网。

以太网工作起来就像一个总线系统，其中的一个主机可将单位长度多达 1500字节的数据包（或“帧”）发给同一以太网内的另一个主机。每个主机都分配有一个 6字节的地址，这种地址需要固化到网卡内部，但也可以通过专门的软件更改。通常，我们用两位十六进制数字的一个序列来表达这种地址，每对数字之间用冒号分隔，例如 aa:bb:cc:dd:ee:ff。

由一个主机发出的数据包可被网内其他所有主机“看到”。但是，只有目标主机才能实际地接收并处理它。假如两个主机同时试图发送数据，便会发生“冲突”。解决这种冲突的办法便是两个主机都取消这一次发送，各自等待随机的一小段时间，再进行发送数据包的尝试。

1.2.3 其他类型的硬件

在规格较大的网络中，比如像前例提到的 GMU大学校园网，以太网通常并非唯一的一种网络硬件安装方式。在 GMU大学，各个系的 LAN都同校园干线连接，后者是一条光纤线缆，运行的是FDDI（光纤分布数据接口）。FDDI采用一种全然不同的形式来实现数据的传送。其基本原理是发送大量“令牌”，令其在整个网络内循环。只有某个主机拿到了一个令牌，才有权将数据送入网络（随那个令牌一道）。FDDI的主要优点在于，它可达到很高的数据传输速度，通常100Mbps是没有问题的，而且布线长度可以高达 200公里。

对于远距离网络链接，通常还有另一种设备类型可供选择，它建立在一种名为 X.25标准的基础上。在美国，有许多所谓的“公共数据网络”（Public Data Network，PDN），比如Tymnet，它们提供的便是这种服务；而在德国，Datex-P网络提供的也是这种形式的服务。X.25要求安装

一种特殊的硬件，名为PAD，亦即“包汇编／反汇编器”（Packet Assembler／Disassembler）。X.25定义了一系列网络协议，由自己专用，但通常也用于连接运行 TCP/IP和其他协议的网络。由于IP包不能直接映射（转换）为X.25格式（反之亦然），所以其他协议的数据包只是简单地封装在X.25包里面，再通过网络传输。

通常，无线电爱好者利用自己的专门设备将计算机连成网络；这种技术称为“包无线电”，或称“火腿无线电”。这些爱好者理所当然也叫做“火腿一族”。火腿无线电使用的协议叫做AX.25，自X.25衍生而来。

另一种技术需要使用速度较慢、但价格便宜的串行线路，用于拨号访问。这要求另一种协议来进行数据包的传送，比如SLIP或PPP，后文还会详细讲述。

1.2.4 网际协议

当然，我们的网络连接并不仅仅限于一个以太网。在最理想的情况下，我们希望能任意使用一个网络，无论它运行于什么硬件形式之上，也无论它由多少个子网组成。比如，在像GMU校园网这样规模较大的网络中，通常包含了数量众多的独立以太网，它们需要以某种形式，相互连接到一起。在GMU大学，数学系运行着两个以太网。其中一个网络包含的都是快速机器，由教授和研究生使用；另一个网络的机器速度较慢，由学生平时上机使用。两个网络都同时接入FDDI校园干线。

这种连接是由一个专用主机控制的，名为“网关”。网关的作用是在两个以太网和光纤线缆之间，通过复制数据包的方式，对数据包的接收和发送进行处理。举个例子来说，如果你在数学系，想访问物理系的局域网上 Quark，然而连网软件不能直接向Quark发送数据包，因为它们不在同一个以太网上。因此必须依靠网关来“转发”数据包。然后，网关（命名为Sophus）再把数据包转发到物理系的、它的同级网关 Niels，由Niels把数据包转发到目标主机。

直接将数据导向远程主机的方法称作“路由”，数据包常常被称作“数据报”。为使一切简单化，数据报交换由一个独立的、与所用硬件无关的协议进行管理。这个协议就是 IP，或者说“网际协议”。我们将在第2章，就IP和路由进行全面论述。

IP的主要好处是：从物理上把不同的网络变成了同一个网络。这就是“联网”技术，而这种“变形网”也就是internet（网间网）。注意，internet和Internet之间，有明显的区别。后者是一个特定的、官方命名的全球化“网间网”。

当然，IP同时还需要一个与硬件无关的定址方案。也就是每台主机对应一个独一无二的32位数，这个数便是该主机的IP地址。IP地址通常用“点分四段”的格式来表示，每一段以十进制表示其四字节中的一个字节，段间用句点分隔开。比如，Quark可能有“0x954C0C04”这样的一个IP地址，它对应的是149.76.12.4，前三个字节由InterNIC注册服务中心分配，标识主机接入的网络，剩下的字节用来标识该主机本身。

大家还会注意到，我们现在有三种地址类型：其一是主机名，比如说 Quark；其二是IP地址；最后是硬件地址，比如那个 6字节的以太网地址。所有这些都必须彼此相符，只有这样，在输入rlogin quark时，联网软件才能给出Quark的IP地址；而且，在“网际协议”向物理系以太网投递数据时，必须找出其 IP地址对应的那个以太网地址 —— 这个地址是最容易搞混淆的。

我们不打算在此深入下去，因为第 2章为大家准备了丰富的IP大餐。目前，大家只需记住的一点是：寻找地址的过程叫做“ 主机名解析 ”（即把主机名映射成 IP地址）和“ 地址解析 ”（把IP地址映射成硬件地址）。

1.2.5 串行线路网际协议

串行线路网际协议 (SLIP)，又叫做“ 串行线路接口协议 ”。是一种允许通过拨号连接进行IP数据包传输的数据链路协议，使得计算机或局域网可以连接到因特网或其他网络中。在串行线路上，常用的标准一般是 SLIP或Serial Line IP。SLIP的修正版称为CSLIP，或者压缩过的SLIP，执行IP头的压缩，以便更好地利用串行链路提供的、相对较低的带宽。另一个不同的串行链路协议是PPP（参见第7章），或者说“ 点到点传输协议 ”（Point-to-Point）。和SLIP相比，PPP的特性更多，其中还包括一个链路协商段。但 PPP领先于SLIP主要表现在没有对 IP数据报的传输进行限制，而且可以发送任何一种类型的数据报。

1.2.6 传输控制协议

当然，从目前看来，从一个主机向另一个主机发送数据报只是任务之一。如果你登录到Quark，还打算在Erdos上的登录协议命令（ rlogin）进程和Quark上的外壳进程之间建立一条可靠的连接，那么，发送端必须把准备收发的信息分成若干个数据包，再由接收端把多个数据包重新组合成一个字符流。这样不仅琐碎，还会额外增加许多工作量。

关于IP，不容忽视的另一点是：不可靠。假设你的以太网上有 10名用户，他们都通过GMU的FTP服务器开始下载XFree86的最新版本。那么所产生的通信量对网关而言，是难以应付的，因为速度太慢了，而且内存也非常的紧张。现在，如果你碰巧向 Quark发送一个数据包，Sophus可能会出现缓冲区空间暂时短缺，导致不能转发数据包的情况。 IP解决此类问题的办法是“ 丢弃 ”它。因此，这个包必定会被丢失。所以，通信主机的责任就是在出现错误的情况下，检查数据的完整性、数据是否完全，以及另行转发。

然而，这是由另一个协议 —— TCP协议（也叫做传输控制协议） —— 来完成的。TCP在IP之上建立了一个可靠的服务。其本质特征是利用 IP，向大家勾划了主机和远程机器上的两个进程之间的一条简单连接，这样一来，就大可不必担心你的数据实际上是沿着哪条路由，以及怎样路由的了。从本质上来说， TCP连接的原理其实就像一条“ 双向 ”管道，两个进程可以边读取，边写入。也可把它想像成“ 打电话 ”。

TCP利用连接涉及的两台主机之IP地址和各主机上的所谓“ 端口 ”号，来标识此类连接的端点。端口可被视为网络连接的“ 附着点 ”（attatchment point）。如果把TCP连接想像成“ 打电话 ”，那么IP地址就是区号（对应一个地区或城市），而端口号就是本地代号（对应各家各户的电话机）。

在“ rlogin ”这一例子中，客户机应用程序（ rlogin）打开Erdos上的一个端口，并将它连接到Quark上的513端口， rlogind服务器将被告知对这个端口进行监听。这样便建立了一条TCP连接。利用这条连接， rlogind执行了认证进程，然后衍生出外壳，该外壳的输入和输出被重新导向TCP连接，这样一来，你在自己机器上输入的任何“ 登录协议命令 ”都会通过 TCP数据流得以传递，并被当作标准输入传给外壳。

1.2.7 用户数据报协议

当然，TCP/IP联网中，TCP并不是唯一的用户协议。虽然它非常适合于 rlogin之类的应用程序，但万万不能用于NFS这样的应用程序。相反地，应该用它的兄弟协议 —— UDP（也叫做“用户数据报协议”）。和TCP协议一样，UDP也允许应用程序和远程机器某个特定端口上的服务取得联系，但它不能为此而建立连接。取而代之的是，可通过它向目标服务发送独立的数据包。

假设你已经通过本系的中心 NFS服务器Galois装入了TeX目录结构，并且打算查看一个关于如何使用 LaTeX的文档。启动编辑器，先在整个文件中读取。但是，要和 Galois建立一条TCP连接、发送并再次发布这个文件，会花相当长的时间。相反地，如果向 Galois发出请求，编辑器则会把该文件分成两个 UDP数据包发送，这种方式要快得多。但美中不足的是， UDP协议不会处理数据包的丢失或中断。若出现这种情况，就由 NFS对此进行处理。

1.2.8 端口问题

端口可以看作是网络连接的附着点。如果一个应用程序想提供一项特定的服务，它就会把自己附着在一个端口上，等待客户机（即所谓的“在该端口上监听”）。想利用该项服务的客户机便在其本地主机上分配一个端口，并连接到远程主机的服务器端口。

端口的一个重要属性是：客户机和服务器之间的连接一旦建立，服务器的另一个副本可能就会附着在服务器端口，监听更多的服务器。也就是说，允许若干个并发登录利用同一个端口513登录到同一个主机。TCP能够把这些连接区分出来，因为它们来自不同的端口或主机。举个例子来说，如果你两次均从 Erdos主机登录进入Quark，第一个rlogin客户机就会采用1023这个本地端口，第二个则采用1022端口。然而，两者均是连接到Quark上同一个513端口上的。

上面的例子说明了端口可用做集合点，客户机通过它连接到另一个特定的端口，以获得特定的服务。为了让客户机知道正确的端口号，两个系统的主管必须在端口号的分配问题上达成一致。对那些用得较广的服务来说（比如 rlogin），其端口号就必须进行集中管理。这是由IETF（因特网工程任务组）来完成的， IETF定期发布一个标题为“已分配号”的 RFC。该RFC对分配给“众所周知”的服务的端口号进行了说明。 Linux采用的是一个文件，名为/etc/services，该文件把服务名映射为端口号。我们将在第 8章对此进行详述。

值得注意的是，尽管 TCP和UDP连接和端口有很大关系，但其端口号之间不会有冲突。在我们前面的例子中，这便意味着 TCP端口513有别于UDP端口513。事实上，这些端口被用作两类不同服务的访问点，也就是 rlogin（TCP）和rwho（UDP）。

1.2.9 套接字库

在操作系统中，执行所有任务的软件和前面所讲的协议通常是内核的一部分。全世界最流行的编程接口是 Berkeley Socket Library（伯克利套接字库）。其名字源于一个流行的比方，即把端口视作套接字，与端口之间的连接视作插拔。它提供了 (bind(2))调用，借此指定远程主机、传输协议和一个程序可以连接或监听的服务（利用连接 (2)、监听(2)和接受(2)）。但是，这个套接字库过于普通，不仅提供了一个基于 TCP/IP套接字的类（AF_INET套接字），还提供了一个可处理本地与远程机连接的类（ AF_UNIX类）。有的实施方案中还可以对其他类进行处理，比如XNS（Xerox连网系统）协议和X.25。

Linux操作系统中，套接字库是标准 libc C-library中的一部分。当前，它只提供了对AF_INET和AF_UNIX套接字的支持，但人们正在努力，提供对 Novell连网协议的支持，以便最后能增添更多的套接字类。

1.3 连网

由于世界各地程序员的共同协作， Linux才成为可能。没有如此广泛的协作， Linux不会有如此夺人的魅力。所以早期开发阶段，寥寥数人着手提供连网功能的工程，其艰难程度简直难以想象。UUCP实施的运作几乎从零开始，基于 TCP/IP连网的工作大约开始于 1992年秋，当时Ross Biro和其他人创建的即是后来人们熟知的“ Net-1 ”。

1993年5月， Ross中止了这项活动的实施， Fred van Kempen开始着手一项新的实现，重新编写主要的代码，这就是 Net-2。1992年夏，首次公开发行 Net-2d（作为0.99.10内核的一部分），而且从此以后， Net-2d得到了几个人的维护和延伸，最值得一提的是 Alan Cox。在对代码进行繁重而庞杂的调试修改之后， Alan Cox在1.0发布之后，把Net-2改成Net-3。

Net-3随同SLIP（用于在串行线路上发送网络数据流）和 PLIP（针对并行线）一起，为各式各样的以太网插板提供了设备驱动程序。 Net-3中有一个TCP/IP实现，其性能像在本地网络环境中一样出色，甚至可以和商业 PC软件人员开发的软件一比高下。

不同的开发倾向

与此同时， Fred继续开发他的Net-2e，主要对网络层的设计进行了大量的修改。在编写代码的同时， Net-2e仍然只是一个测试版软件。关于 Net-2e，最值得一提的是它结合了 DDI，即设备驱动程序接口。DDI针对所有的连网设备和协议，提供了一个统一的访问和配置方法。

然而， TCP/IP连网的另一种实现却源于 Matthias Urlichs，他曾经为Linux和FreeBSD编写了一个ISDN驱动程序。这次，他在内核中集成了一部分 BSD连网代码。

但在可以预见的将来， Net-3似乎停步不前了。 Alan目前在从事 AX.25协议实施的开发，该协议面向“ 火腿无线电发烧友 ”。但已准备开发的内核“ 模块 ”代码仍然会为连网代码带来新的活力，却是一个不争的事实。模块允许爱好者在运行时在内核中添加驱动程序。

尽管这些不同实施都可利用同一个设备，但在内核和设备级还有明显的差异。因此，你不能通过Net-2d或Net-3，利用公用程序对正在运行 Net-2e内核的系统进行配置，反之亦然。这条规则只适用于对内核内部进行处理的命令、诸如 rlogin或telnet之类的应用程序和连网命令。

虽然如此，所有这些不同的网络版本都不会让你无所适从。除非你加入了活动开发行列，否则，就不必担心自己运行的 TCP/IP是什么版本。正式发布的内核总附带有一套连网工具，它们适用于内核中介绍的连网代码。

1.4 系统维护

本书从头到尾，都将为大家讲解安装和配置方面的问题。但提得更多的却是管理。设立一项服务之后，还必须使之能够运行。大多数服务的维护并不繁杂，但有的服务，比如邮件和新闻，则要求你执行常规任务，以保持系统的更新。我们将在后面几章中讨论这些任务。

维护系统过程中，最基本的要求是定期检查系统和应用程序前的日志文件，以了解错误情况和异常事件。一般说来，大家可能想通过编写两个管理外壳脚本来执行该任务，并从

cron周期性地运行这两个脚本。有些主要应用程序（比如 smail和C-News）的源代码中包含有这样的脚本。只须按照自己的要求和喜好，适当地增减。

任何cron作业的输出结果都会寄到一个管理性质的帐号上。默认状态下，多数应用程序都会向根帐号发出错误报告、用法统计和登录文件总结。如果你经常以根的身份登录的话，这是很有用的；另一个较好的方案是把根的邮件转发到你自己的私人帐号，设立一个邮件别名，详情参见第13章。

不管你配置自己的站点时有多细心，始终会产生问题。因此，系统维护的另一层含义是接受用户的抱怨。通常情况下，人们希望通过电子邮件把自己的意思转达给作为“ BOOT ”的系统管理员，但同时还需要其他的地址，以便负责某一特定维护的人员也能够看到这些邮件。举个例子来说，报怨邮件配置故障的电子邮件通常被发送给邮局管理员；而和新闻系统有关的问题则向新闻管理员或新闻组报告。寄给主机管理员的邮件应该重新定向到主机基础网络服务的负责人，如果你运行的是命名服务器，则重定向到 DNS命名服务。

系统安全

网络环境中，对系统进行维护的另一个不容忽视的重要原因是：避免自己的系统和用户受到攻击。管理疏忽的系统往往是那些心怀不轨的人攻击的目标。攻击手段相当多，从猜测密码到以太网偷窥。导致的恶果轻重程度不一，从恶作剧式的邮件信息到数据丢失，或者侵犯用户的隐私权。在讨论出现这些特殊情况的背景时，我们将进一步深入这一话题，同时探讨常见的防御手段。

本小节将讨论几个实例和处理系统安全的一些基本技巧。当然，我们所涉及的主题也许不够全面；它们只对一些常见问题进行了说明。因此，找本安全方面的好书来读是绝对必要的，尤其是在连网系统中。强烈推荐大家参考 Simon Garfinkel所著的《Practical UNIX Security》，该书由O ’ Reilly出版。还可以在www.oreilly.com/catalog/puis/找到此书。

提起系统安全，首先得从优良的系统管理开始。这包括了检查所有重要文件和目录的属主和许可，监视特权帐号的使用情况等等。比如， COPS程序（www.cert.org/ftp/tools/cops）将检查你的文件系统和常用配置文件是否有异常许可或其他异常情况。它还聪明地采用了一个密码套件，强制性地对用户密码实施某一特定的规定，这样，密码就很难被人破译。比如影子密码套件，它要求密码至少要有5个字母，而且还包括大、小写形式的数字和数位。

在编写供网络访问的服务时，务必确定赋予它“ 最低特权 ”，意思是除了为它设计的任务外，不允许它执行别的任务。比如，在根用户和其他特权账号的确需要 setuid程序时，就应该为他们赋予相应的特权。同时，如果希望服务只限于特定的应用时，不要迟疑，在特定应用允许的范围内，尽可能地对它进行严格配置。例如，如果想允许通过你自己的机器启动无盘主机，你必须为之提供 TFTP（日常文件传输服务），以便它们能够从 /boot目录下下载基本配置文件。但是，如果在非限制条件下， TFTP允许全世界的用户都可以从你的系统中下载这些文件。如果不希望如此暴露，就必须把 TFTP服务限定在/boot目录下。

同样的道理，大家肯定还想过如何在自己的局域网内，限制某些主机的用户使用网络服务。这方面的详情，可参考第8章介绍的TCPD。

另一个不容忽视的要点是避免使用“ 危险 ”的软件。当然，我们平常所用的软件都可能是危险的，因为软件不可能总是十全十美的，难免存在错误，有些聪明人士肯定能够找出其

中的漏洞，进而造访你的系统。这是常有的事儿，而且根本无法杜绝。但是，和其他程序比较起来，要求特殊权限的程序似乎更缺乏安全保障，因为任何一个漏洞都可能导致难以想象的后果。所以，在网络中安装setiud程序时，一定要加倍小心，不能漏掉安装文档中的任何东西，以免无意中为“黑客”造成可乘之机。

正如天有不测风云，谁也料不到未来会发生什么事情，因为无论我们多么细心，总不能排除意外的发生。最好的办法是防患于未然。首先从检查自己的系统日志文件着手，但有些“闯入者”可能很聪明，会将自己的造访痕迹清除掉。但是，像 Tripwire（www.cert.org/ftp/tools/tripwire/）这类的工具，它们允许我们对重要的系统文件进行检查，看其中的内容或许可权是否已被篡改。Tripwire对这些文件进行各种校验和计算，并把它们保存在一个数据库内。在之后的运行过程中，将对行些校验和重新进行计算，并和数据库内保存的校验和进行比较，从而有效地防止旁人对系统数据的肆意篡改。

1.5 后续章节提要

下面几个章节将讨论如何配置 TCP/IP连网，如何运行一些主要的应用程序。在着手编辑文件之前，我们将在第2章详细讨论一下IP。已经掌握IP路由原理、知道如何执行地址解析的读者，可跳过此章。

第3章将讨论一些基本的配置问题，比如如何建立内核，如何安装自己的以太网卡等等。至于如何配置自己的串行端口，则在另外一章中讨论，因为它不仅适用于 TCP/IP连网，而且还适用于UUCP。

第5章将帮助大家针对TCP/IP连网，安装自己的机器。其中包括只启用loopback的单机主机安装题示，连接到以太网的主机安装提示。另外还要介绍几个颇有用的工具，大家可以用它们来测试并调试自己的安装结果。第6章将讨论如何配置主机名解析，如何安装名字服务器。

接下来的两章将分别介绍SLIP和PPP的配置和使用。第6章将对如何建立SLIP连接进行解释，并展示一个详细的Dip参考，该工具允许大家对某些必要的步骤进行自动化处理。第 7章将全面介绍PPP和pppd。

第8章将简要介绍一些重要网络应用程序的安装，比如 rlogin和rcp等等。第8章还将全面介绍inetd super是如何对服务进行管理的，以及如何将关系到系统安全的服务限定在信任主机范围内。

接下来的两章将讨论NIS（网络信息系统）和NFS（网络文件系统）。NIS是一个非常有用的工具，用于分发管理信息（比如局域网内的用户口令）。NFS则允许你与网络内的若干个主机共享文件系统。

第11章将广泛介绍Taylor UUCP管理，它是一个免费的UUCP套件。

本书的其余部分将介绍电子邮件和 Usenet新闻组。第12章介绍电子邮件的核心概念，比如邮件地址是什么样的，邮件处理系统如何将电子邮件发到收信人手中的等等。

第13章和第14章分别介绍smail和sendmail的安装。本章对两者都进行了介绍，因为它们各有千秋：对初学者来说， smail更易于安装， sendmail则更为灵活。

第15章和第16章介绍Usenet新闻组内的新闻管理方式，以及如何安装和使用 C-news（一个常用的软件包，用于管理Usenet新闻组）。第17章简要讨论如何安装NNTP daemon，为局域网提供新闻阅读访问。最后，第18章向大家展示如何配置和维护不同的新闻用户。