

第15章 NNTP：网络新闻传送协议

15.1 概述

NNTP，即网络新闻传送协议，在协作的主机之间发布新闻文章。NNTP是一个使用TCP的应用协议，RFC 977[Kantor and Lapsley 1986]对它进行了详细描述。[Barber 1995]对它的一般实现进行了扩展。RFC 1036 [Horton and Adams 1987]对新闻文章中的各种首部字段进行了说明。

网络新闻起源于ARPANET上的邮件列表，随后发展成为 Usenet新闻系统。邮件列表今天还很流行，但如果纯粹从容量来看，网络新闻在过去的十年有很大的增长。从图 13-1中可以看出，NNTP有跟电子邮件一样多的分组数。[Paxson 1994a]中提到，从1984年以来网络新闻的流量保持了每年约75%的增长。

Usenet不是一个物理的网络，而是建立在多个不同类型物理网络上的一个逻辑网。多年以前，在Usenet上流行的交换网络新闻的手段是通过电话线拨号（为了省钱通常在几个小时以后），而在今天，Internet是绝大多数新闻发布的主要渠道。[Salus 1995]中的第15章详细讲述了Usenet的历史。

图15-1是一个典型的新闻系统的概况。一台作为组织的新闻服务器的主机在磁盘上保留

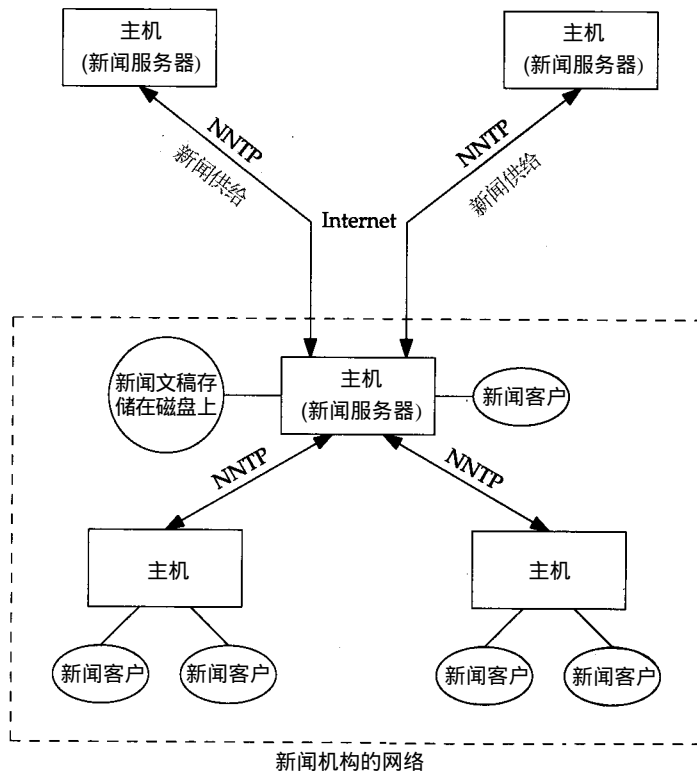


图15-1 典型的新闻系统

了所有新闻文章。这个新闻服务器通过 Internet 与其他的新闻服务器通信，互相供给新闻。新闻服务器之间的通信使用 NNTP 协议。新闻服务器有各种不同的实现，INN(InterNetNews)正成为 Unix 平台上最流行的新闻服务器程序。

组织中的其他主机通过访问新闻服务器来阅读新闻文章和选择新闻组粘贴新闻。我们把这些客户程序称为“新闻客户”。这些客户程序与新闻服务器之间的通信也采用 NNTP 协议。另外，如果新闻客户与新闻服务器在同一主机上，客户也用 NNTP 阅读和粘贴新闻。

不同的客户操作系统平台上有十几种新闻阅读器(客户程序)。最原始的 Unix 新闻客户程序是 Readnews，接着是 Rn 和它的变种：Rrn 是一个支持远程操作(remote)的版本，它允许客户和服务在不同的主机上；Trn 的意思是“线索(threaded)Rn”，它可以使用多条线索在一个新闻组中讨论；Xrn 是 Rn 的 X11 窗口系统的版本。GNUS 是一个内含 Emacs 编辑器的流行的新闻阅读器。也有一些通用的 Web 浏览器，例如 Netscape，在浏览器中内置访问新闻服务器的接口，这样就不需要单独的新闻客户程序。就像不同的电子邮件程序提供许多不同的用户接口一样，每一种不同的新闻客户程序也提供不同的用户接口。

不管使用哪种客户程序，对新闻服务器来说，这些不同的新闻客户程序的相同点是：都使用 NNTP 协议，这正是我们在本章要讨论的。

15.2 NNTP 协议

NNTP 使用 TCP 协议，知名的 NNTP 服务的端口号是 119。NNTP 也像其他的 Internet 应用(HTTP, FTP, SMTP, 等等)一样，客户发送 ASCII 命令给服务器，服务器返回数值的响应码，后面跟着可选的 ASCII 数据(取决于客户的命令)。命令和响应都以回车加换行结束。

考察这种协议最简单的办法就是用 Telnet 程序来连接一台主机上的 NNTP 端口，当然，这台主机运行了 NNTP 服务器程序。但是，通常我们必须从一台能被服务器主机识别的主机上运行客户程序，典型的情况就是从同一组织网络中的一台主机。例如，我们通过 Internet 从其他网络的主机上来登录本地的新闻服务器，会收到如下的错误信息：

```
vangogh.cs.berkeley.edu % telnet noao.edu nntp
Trying 140.252.1.54...
Connected to noao.edu.
Escape character is '^]'.
502 You have no permission to talk. Goodbye.
Connection closed by foreign host.
```

由 Telnet 客户程序输出
由 Telnet 客户程序输出
由 Telnet 客户程序输出
由 Telnet 客户程序输出

输出的第 4 行是由 NNTP 服务器输出的，响应码是 502。当 TCP 连接被建立后，NNTP 服务器收到客户的 IP 地址，将它与配置中允许的 IP 地址进行比较。

在下面的例子中，我们从一台“本地”主机连接到新闻服务器。

```
sun.tuc.noao.edu % telnet noao.edu nntp
Trying 140.252.1.54...
Connected to noao.edu.
Escape character is '^]'.
200 noao InterNetNews NNRP server INN 1.4 22-Dec-93 ready (posting ok).
```

这次从服务器来的响应码为 200(命令 OK)，响应行中余下的是服务器的有关信息。返回信息的最后是“posting ok”或“no posting”，这取决于是否允许客户粘贴新闻(这个由系统管理员根据客户的 IP 地址来控制)。

我们注意到服务器的响应信息中提到，这个服务器是 NNRP(Network News Reading Protocol)服务器，而不是INND(InterNetNews daemon)服务器。先是INND服务器接收客户的请求，查找客户的IP地址。如果客户的IP地址是被允许的，并且客户不是一个已知的、提供新闻的主机，那么NNRP服务器被激活，替代INND服务器，假定客户是想要读新闻而不是想要给服务器提供新闻。这就可以将新闻供给服务器（约10 000行C代码）与新闻阅读服务器（约5 000行C代码）分别来实现。

图15-2列出了数字响应码中第1位和第2位的含义。这与FTP中的用法也很相似(卷1的319页)。

应 答	说 明
1yz	报告情况的消息
2yz	命令执行成功
3yz	迄今为止命令执行成功；发送余下的命令
4yz	命令正确，但因为某些原因不能执行
5yz	命令未实现，或命令不正确，或遇到了严重的程序差错
x0z	连接、设置和杂项消息
x1z	新闻组选择
x2z	文章选择
x3z	分发功能
x4z	发送
x8z	非标准扩展
x9z	调试用输出

图15-2 三位响应码中第1位和第2位的含义

我们发给新闻服务器的第一个命令是 help，help命令会将这个新闻服务器支持的所有命令列出来。

```

help
100 Legal commands          100是响应吗
  authinfo user Name|pass Password
  article [MessageID|Number]
  body [MessageID|Number]
  date
  group newsgroup
  head [MessageID|Number]
  help
  ihave
  last
  list [active|newsgroups|distributions|schema]
  listgroup newsgroup
  mode reader
  newgroups yymmdd hhmmss ["GMT"] [<distributions>]
  newnews newsgroups yymmdd hhmmss ["GMT"] [<distributions>]
  next
  post
  slave
  stat [MessageID|Number]
  xgtitle [group_pattern]
  xhdr header [range|MessageID]
  xover [range]
  xpat header range|MessageID pat [morepat...]
  xpath xpath MessageID
Report problems to <usenet@noao.edu>

```

这一行只有一个句点，表示服务器响应的结束

因为客户无法确认服务器返回的信息到底有多少行，所以协议要求服务器以一个只包含句号的行来结束返回。如果某一行恰好就是要以句号开头，那么服务器会在前面再加上一个句号再发送，客户收到这行后先去掉这个句号。

下面我们来看一下 list 命令。如果不带任何参数执行 list 命令，它列出所有本服务器上每一个新闻组的名字，后面跟着这个组中最后一篇新闻和组中第一篇新闻的编号，最后是“y”或“m”，表示是否允许这个组粘贴新闻或只是一个普通的组。

list

```
215 Newsgroups in form "group high low flags".      215是响应码
alt.activism 0000113976 13444 y
alt.aquaria 0000050114 44782 y
```

还有许多行没有显示出来

```
comp.protocols.tcp-ip 0000043831 41289 y
comp.security.announce 0000000141 00117 m
```

还有许多行没有显示出来

```
rec.skiing.alpine 0000025451 03612 y
rec.skiing.nordic 0000007641 01507 y
```

这一行只有一个句点，表示服务器响应的结束

当然，215是响应码，而不是新闻组的编号。这个例子中的服务器向客户返回了 4 238 个新闻组的情况，共 175 833 字节的 TCP 数据。返回的新闻组信息没有按字母排序。

在新闻客户端上通过较慢的拨号线从一个新闻服务器获取这样一个列表，通常会感到很慢。例如，如果数据传输速率是 28 800 bit/s，那么这个过程将花费约 1 分钟(实际测量时，使用这样一个调制解调器，并在数据发送时进行压缩，大约需 50 秒)。在以太网上，这个过程所需时间不到 1 秒。

group 命令用来指定某一新闻组作为客户的“当前”新闻组。下面的命令就是把 comp.protocols.tcp-ip 设为当前新闻组。

group comp.protocols.tcp-ip

```
211 181 41 289 43 831 comp.protocols.tcp-ip
```

服务器以响应码 211(命令执行成功)开头，后面跟着这个组中新闻总数的估计值(181)，然后是本组中第一篇新闻文章的编号(41 289)、最后一篇新闻文章的编号(43 831)和新闻组的名字。在新闻文章的起始和结束编号之间的差值(43831 - 41289 = 2542)通常大于新闻文章数(181)。一方面是因为有些文章是典型的 FAQ(Frequently Asked Questions)，它们的失效时间(通常为 1 个月)比起其他大多数新闻(很少的几天，取决于服务器硬盘的容量)要长得多。另一个原因是这些文章能被显式地删除。

下面我们用 head 命令来看一篇特殊文章(编号为 43 814)的首部内容。

head 43814

```
221 43814 <3vtrje$ote@noao.edu> head
Path: noao!rstevens
From: rstevens@noao.edu (W. Richard Stevens)
Newsgroups: comp.protocols.tcp-ip
Subject: Re: IP Mapper: Using RAW sockets?
Date: 4 Aug 1995 19:14:54 GMT
Organization: National Optical Astronomy Observatories, Tucson, AZ, USA
Lines: 29
Message-ID: <3vtrje$ote@noao.edu>
```

References: <3vtdhb\$jnf@oclc.org>
NNTP-Posting-Host: gemini.tuc.noao.edu

应答的第一行带有响应码 221(命令执行成功), 后面是 10 行的首部, 最后是只含有句号的
一行。

大多数首部字段无需解释, 但消息的 ID 看上去有些让人迷惑。INN 试图按下面的
格式生成唯一的消息 ID 格式: 当前时间, 一个 \$ 符号, 进程 ID, 一个 @ 符号, 本地主
机的完整域名。时间和进程号的数值都以 32 进制的数字串输出: 数字由每 5 位二进制
数为 一组, 每一组用字母: 0...9a...v 来表示。

接着我们用 body 命令返回同一篇文章的主体。

```
body 43814
222 43814 <3vtrje$ote@noao.edu> body
> My group is looking at implementing an IP address mapper on a UNIX
```

文章中还有 28 行没有列出来

新闻的首部和主体可以用一个命令 (article) 获取, 但绝大多数新闻客户是先取得文章
的首部, 允许客户根据新闻的主题进行选择, 然后只取回用户所选取的文章的主体。

我们用 quit 命令来终止到服务器的连接。

```
quit
205
Connection closed by foreign host.
```

服务器的响应是数字代码: 201。我们的客户程序 Telnet 显示服务器关闭了连接。

整个客户与服务器的交互过程只使用单个的、由客户发起的 TCP 连接。但是连接上大部
分数据都是由服务器发向客户的。连接的持续时间以及数据的交换量均取决于用户阅读新闻
时间的长短。

15.3 一个简单的新闻客户

下面我们通过使用一个简单的新闻客户程序, 进行简要的新闻会话来看一下 NNTP 命令与
响应之间的交互。我们使用最老的新闻阅读器中的一种: Rn, 它简单而且容易使用, 同时选
用它还因为它带有调试选项 (-D16 命令行选项, 假定客户程序编译时打开了调试选项)。这
让我们可以看到客户发出的 NNTP 命令以及相应的服务器的响应。我们用黑体字来表示客户端
的命令。

- 1) 第一个命令是 list, 在上一节中我们看到从服务器返回了约 175 000 字节, 每行表示
一个新闻组。同时 Rn 也把用户想要阅读的新闻组以及在这组中最后读过的新闻的编号
的列表保存在文件 .newsrsc (在用户的主目录中) 中。例如, 某一行:

```
comp.protocols.tcp-ip: 1-43815
```

通过把文件中保存的、最后读过的新闻的编号与现有最新的新闻的编号进行比较, 客
户就知道本组中是否还有没读过的新闻。

- 2) 然后客户检查是否有新的新闻组建立。

```
NEWGROUPS 950803 192708 GMT
231 New newsgroups follow.
```

231 是响应码

Rn在用户的主目录的文件 `.rnlst` 中保存了最近一次通报新的新闻组的时间。这个时间成为 `newsgroups` 命令的参数 (NNTP 命令和命令的参数都与大小写无关)。在这个例子中保存的时间是：格林尼治时间 1995 年 8 月 3 日, 19:27:08。服务器返回为空 (在返回码 231 与只包含句点的行中没有其他的内容)，指示没有新的新闻组建立。如果有新的新闻组建立，客户程序会询问用户是否要加入这个组。

- 3) 接着 Rn 将显示前 5 个新闻组中未读新闻的编号，并询问是否要阅读第一个新闻组：
`comp.protocols.tcp-ip`。我们以一个等于号响应，让 Rn 返回一个对该组所有文章的一行摘要，然后我们可以选择想要阅读的文章 (可以用 `.rninit` 文件对 Rn 进行配置，让 Rn 按照我们期望的方式给出每一篇新闻文章的摘要。书的作者配置的摘要包括文章编号、主题、文章的行数和文章的作者)。 `group` 命令是由 Rn 发出的，设置当前的新闻组。

```
GROUP comp.protocols.tcp-ip
211 182 41289 43832 comp.protocols.tcp-ip
```

第一篇未读文章的首部和主体可以用以下命令获得：

```
ARTICLE 43815
220 43815 <3vtq8o$5p1@newsflash.concordia.ca> article
.
文章未列出
```

第一篇未读文章的一行摘要显示在终端上。

- 4) 对本组中剩下的 17 个未读的新闻执行 `xhdr` 命令，然后再用 `head` 命令。如下面的例子：

```
XHDR subject 43816
221 subject fields follow
43816 Re: RIP-2 and messy sub-nets
.
HEAD 43816
221 43816 <3vtq8o$5p1@newsflash.concordia.ca> head
.
首部的 14 行未列出
```

`xhdr` 命令能接受的参数不但可以是单个文章编号，还可以是号码范围，这就是为什么服务器返回了许多行，然后以只含有句点的行结束的原因。每篇文章的一行摘要显示在终端上。

- 5) 我们敲空格键选择第一篇未读的文章，客户程序发出 `head` 命令，接着是 `article` 命令。文章便显示在终端上。对所有文章相继使用这两个命令。
 6) 当我们读完这个组的新闻后，便移到另一个组，这时客户程序又发出另一个 `group` 命令。我们向服务器请求每一篇未读新闻的一行摘要，在新的组中再一次执行上面讨论过的命令。

我们注意到的第一件事情是 Rn 发出了太多的命令。例如，为了对所有的未读文章取得一行摘要，先要发出 `xhdr` 取得摘要，然后是用 `head` 命令取得文章的首部。这两个命令中的第一个是不必要的。增加这些额外命令的原因之一是最开始这些命令是为工作在主机 (也是新闻服务器) 上的客户程序设计的，因此这些附加命令可能要快一些，因为没有网络的传输时间。使用 NNTP 访问远程新闻服务器的功能是后来加上去的。

15.4 一个复杂的新闻客户端

下面我们来看一个更复杂的新闻客户端程序：Netscape 1.1N 版的 Web 浏览器，它内置了新

闻阅读器。这个客户程序没有调试选项，所以我们只有跟踪它与服务器之间交换的 TCP 分组来看它是怎样工作的。

- 1) 当我们启动客户程序，并选择新闻阅读特性时，它读 .newsrsc 文件，并且只向服务器请求在这个文件中我们所预订的新闻组的相关内容。对每一个预订的新闻组都发出 group 命令来确定起始和结束的文章编号，并与 .newsrsc 文件中所存储的最后阅读的文章编号进行比较。这个例子中，作者在 4 000 多个新闻组中只预订了 77 个，因此共有 77 个 group 命令发向服务器。这在拨号线的 PPP 链路上仅需 23 秒，相比较而言，Rn 所使用的 list 命令要 50 秒。

如果新闻组的数量由 4 000 减少至 77，客户所花的时间应小于 23 秒。实际上，用 sock (卷 I 附录 C) 发送 77 个同样的 group 命令只需约 3 秒。看起来浏览器在这 77 个命令上叠加了其他的启动处理。

- 2) 我们选择一个有未读文章的新闻组：comp.protocols.tcp-ip，接着执行下面的命令：

```
group comp.protocols.tcp-ip
211 181 41289 43831 comp.protocols.tcp-ip
xover 43815-43831
224 data follows
43815\tping works but netscape is flaky\troot@PROBLEM_WITH_INEWS
_DOMAIN_FILE (root)\t4 Aug 1995 18:52:08 GMT\t<3vtq8o$5p1@newsfl
ash.concordia.ca>\t\t1202\t13
43816\tRe: help me to select a terminal server\tgvcnet@hnt2.hin
et.net (gvcnet)\t5 Aug 1995 09:35:08 GMT\t<3vve0c$gq5@serv.hinet
.net>\t<claude.80753760 @bauv111>\t1503\t23
.
```

指定范围内剩余文章的一行摘要

第一个命令设置当前新闻组，第二个命令向服务器请求指定文章的概况。在这个组中，43 815 是第一篇、43 831 是最后一篇未读的文章。每篇文章的一行摘要包括：文章编号、主题、作者、日期和时间、消息 ID、文章的引用、字节数和行数 (注意每个一行摘要都很长，所以上面我们把每一行都几次换行。同时我们还把分隔字段的 tab 符换成了 \t，这样便于看清楚)。

Netscape 客户程序按主题组织返回的概况，并显示未读主题的列表以及文章的作者和行数。将一篇文章及其应答组合在一起，称为编线索，因为一个议题的线索都是组合在一起的。

- 3) 对每一篇我们选择阅读的文章，执行一次 article 命令，文章便显示出来。

从上面的 Netscape 新闻客户程序的概况中可以看出，它采用两种优化措施来减少用户的等待时间。第一个措施是只向服务器请求用户所需要阅读的新闻组，而不是使用 list 命令。第二，它使用 xover 命令提供每一篇文章的摘要，而不是对组中的每一篇文章使用一次 head 和 xhrd 命令。

15.5 NNTP 的统计资料

为了理解典型的 NNTP 的用法，我们在第 14 章曾提到的主机上运行 Tcpdump，来收集 NNTP 所使用的 SYN、FIN 和 RST 报文。这个主机从一台 NNTP 新闻供给主机上获得新闻 (可能有其他备用的新闻供给主机，但是观察到的报文都是来自同一台主机)，然后分发给 10 个其他

站点。在这10个站点中只有两个使用NNTP，其他都是使用UUCP，所以我们的Tcpdump只记录到两个NNTP供给主机。两个流出的NNTP主机收到的新闻只是这台主机收到的新闻的一小部分。最后，因为这台主机属于一个Internet服务提供商，所以各式各样的客户把主机当成新闻服务器来阅读新闻。所有的客户阅读新闻均使用NNTP协议，包括同在主机上的新闻阅读进程和其他主机上的新闻阅读进程（典型的是通过PPP或SLIP连接）。Tcpdump连续运行了113小时（4.7天），共收集了1 250个连接上的信息。图15-3汇总了这些信息。

	1个输入 新闻供给	两个输出 新闻供给	新闻阅读 客户	总 计
连接数	67	32	1 151	1 250
流入字节总数	875 345 619	4 499	593 731	875 943 849
流出字节总数	4 071 785	1 194 086	56 488 715	61 754 586
总持续时间(分钟)	6 686	407	21 758	28 851
每连接流入字节数	13 064 860	141	516	
每连接流出字节数	60 773	37 315	49 078	
连接平均持续时间(分钟)	100	13	19	

图15-3 单个主机上4.7天的NNTP统计资料

我们首先注意输入新闻供给主机，它每天收到约1.86亿字节的新闻，平均每小时约为8百万字节。同时我们也可以看出，到主新闻供给主机的NNTP连接的持续时间很长：100分钟，交换了1300万字节的数据。这台主机与它的输入新闻供给主机之间的TCP连接经过一段时间的静默后由新闻服务器关闭。下次需要时再重新建立连接。

典型的新闻阅读程序使用NNTP连接约19分钟，读取约50 000字节的新闻。绝大多数NNTP流量是单向的：从主新闻供给主机流向新闻服务器，从新闻服务器流向新闻阅读客户。

站点-站点的NNTP流量之间有巨大的差异。上面的统计数据就是一个例子：这些统计数据中没有典型值。

15.6 小结

NNTP是又一个使用TCP协议的简单协议。客户发出ASCII命令（服务器支持超过20种不同的命令），服务器的响应先是响应码，然后跟着一行或多行的应答，最后以只包含句号的行结束（如果响应是可变长度）。类似其他的Internet协议，NNTP协议本身已多年没有变化，但是由客户程序提供给交互式用户的接口却变化很快。

不同新闻阅读程序之间的很多区别都取决于应用程序怎样使用协议。我们看到Rn客户程序和Netscape程序之间的不同有：确定哪些文章未读的方法不同，取得未读文章的方法也不同。

NNTP协议使用单个TCP连接维持整个的客户-服务器数据交换。这一点与HTTP协议不同，HTTP协议从服务器每获取一个文件都要建立一条TCP连接。这种差异一方面是因为NNTP客户只与一个服务器通信，而HTTP客户能同时与多个不同服务器通信。同时我们也看到绝大多数TCP连接上的NNTP协议的数据流是单向的。