

附录C sock程序

在本书中一直使用一个称为 sock 的小测试程序，用来生成 TCP 和 UDP 数据。它既可以用作一个客户进程，也可以用作一个服务器进程。有这样一个可以从外壳程序执行的测试程序，使我们避免了为每一个我们想要研究的特征编写新的客户和服务器的 C 程序。因为本书的目的是了解网络互联协议，而不是网络编程，所以在这个附录中我们只描述这个程序和它不同的选项。

有很多与 sock 功能类似的程序。Juergen Nickelsen 写了一个称为 socket 的程序，Dave Yost 写了一个称为 sockio 的程序。两者都包含了很多类似的特征。sock 程序的某些部分也受到了 Mike Muuss 和 Terry Slattery 所写的公开域 ttcp 程序的启发。

sock 程序运行在以下四种模式之一：

1) 交互式客户：默认模式。程序和一个服务器相连，然后将标准输入的数据传给服务器，再将从服务器那里接收到的数据复制到标准输出。如图 C-1 所示。



图C-1 sock 程序作为交互式客户的默认操作

我们必须指明服务器主机的名字和想要连接的服务的名字。主机可指明为点分十进制数，服务可指明为一个整数的端口号。从 sun 到 bsd 与标准的 echo 服务器（1.12 节）相连，回显我们键入的每一个字符：

```
sun % sock bsd echo
a test line
a test line
^D
```

我们键入这一行
echo 服务器返回一个复制行
键入文件结束符来中止

2) 交互式服务器：指明 -s 选项。需要指明服务名字（或端口号）：

```
sun % sock -s 5555
```

作为一个在端口 5555 监听的服务器

程序等待一个客户的连接请求，然后将标准输入复制给客户，将从客户接收到的东西复制到标准输出。在命令行中，端口号之前可以有一个因特网地址，用来指明接收哪一个本地接口上的连接：

```
sun % sock -s 140.252.13.33 5555
```

只接受来自以太网的连接

默认的模式是接受任何一个本地接口上的连接请求。

3) 源客户：指明 -i 选项。在默认情况下，将一个 1024 字节的缓存写到网络中，写 1024 次。-n 选项和 -w 选项可以改变默认值。例如，

```
sun % sock -i -n12 -w4096 bsd discard
```

把 12 个缓存，每个包含 4096 字节的数据，送给主机 bsd 上的 discard 服务器。

4) 接收器服务器：指明 -i 选项和 -s 选项。从网络中读数据然后扔掉。

这些例子都使用了 TCP（默认情况），-u 选项指明使用 UDP。

sock 程序有许多选项，用于对程序的运行提供更好的控制。我们需要使用这些选项来产

生本书中用到的所有测试条件。

- b *n* 将*n*绑定为客户的本地端口号（在默认情况下，系统给客户分配一个临时的端口号）。
- c 将从标准输入读入的新行字符转换为一个回车符和一个换行符。类似地，当从网络中读数据时，将 回车，换行 序列转换为新行字符。很多因特网应用需要 NVT ASCII（26.4节），它使用回车和换行来终止每一行。
- f *a.b.c.d.p* 为一个UDP端点指明远端的IP地址（*a.b.c.d*）和远端的端口号（*p*）。
- h 实现TCP的半关闭机制（18.5节）。即，当在标准输入中读到一个文件结束符时并不终止。而是在 TCP连接上发送一个半关闭报文，继续从网络中读报文直到对方关闭连接。
- i 源客户或接收器服务器。向网络写数据（默认），或者如果和 -s 选项一起用，从网络读数据。-n选项可以指明写（或读）的缓存的数目，-w选项可以指明每次写的大小，-r选项可以指明每次读的大小。
- n *n* 当和-i选项一起使用时，*n*指明了读或写的缓存的数目。*n*的默认值是1024。
- p *n* 指明每个读或写之间暂停的秒数。这个选项可以和源客户（-i）或接收器服务器（-is）一起使用作为每次对网络读写时的延迟。参考-P选项，实现在第1次读或写之前暂停。
- q *n* 为TCP服务器指明挂起的连接队列的大小：TCP将为之进行排队的、已经接受的连接的数目（图18-23）。默认值是5。
- r *n* 和-is选项一起使用，*n*指明每次从网络中读数据的大小。默认是每次读1024字节。
- s 作为一个服务器，而不是一个客户。
- u 使用UDP，而不是TCP。
- v 详细模式。在标准差错上打印附加的细节信息（如客户和服务器的临时端口号）。
- w *n* 和-i选项一起使用，*n*指明每次从网络中写数据的大小。默认值是每次写1024字节。
- A 使能 SO_REUSEADDR插口选项。对于TCP，这个选项允许进程给自己分配一个处于2MSL等待的连接的端口号。对于UDP，这个选项支持多播，它允许多个进程使用同一个本地端口来接收广播或多播的数据报。
- B 使能SO_BROADCAST插口选项，允许向一个广播IP地址发送UDP数据报。
- D 使能SO_DEBUG插口选项。这个选项使得内核为这个TCP连接维护另外的调试信息（A.6节）。以后可以运行trpt(8)程序输出这个信息。
- E 如果实现支持，使能 IP_RECVDSTADDR插口选项。这个选项用于

UDP服务器，用来打印接收到的UDP数据报的目的IP地址。

- F 指明一个并发的TCP服务器。即，服务器使用fork函数为每一个客户连接创建一个新的进程。
- K 使能TCP的SO_KEEPALIVE插口选项（第23章）。
- L *n* 把一个TCP端点的拖延时间(linger time)（SO_LINGER）设置为*n*。一个为0的拖延时间意味着当网络连接关闭时，正在排队等着发送的任何数据都被丢弃，向对方发送一个重置报文（18.7节）。一个正的拖延时间（百分之一秒）是关闭网络连接必须等待的将所有正在排队等着发送的数据发送完并收到确认的时间。关闭网络连接时，如果这个拖延定时器超时，挂起的数据没有全部发送完并收到确认，关闭操作将返回一个差错信息。
- N 设置TCP_NODELAY插口选项来禁止Nagle算法（19.4节）。
- O *n* 指明一个TCP服务器在接受第一个客户连接之前暂停的秒数。
- P *n* 指明在第一次对网络进行读或写之前暂停的秒数。这个选项可以和接收器服务器（-is）一起使用，完成在接受了客户的连接请求之后但在执行从网络中第一次读之前的延迟。和接收源（-i）一起使用时，完成连接建立之后但第一次向网络写之前的延迟。参看-p选项，实现在接下来的每一次读或写之间进行暂停。
- Q *n* 指明当一个TCP客户或服务器收到了另一端发来的一个文件结束符，在它关闭自己这一端的连接之前需要暂停的秒数。
- R *n* 把插口的接收缓存（SO_RCVBUF插口选项）设置为*n*。这可以直接影响TCP通告的接收窗口的大小。对于UDP，这个选项指明了可以接收的最大的UDP数据报。
- S *n* 把插口的发送缓存（SO_SNDBUF插口选项）设置为*n*。对于UDP，这个选项指明了可以发送的最大的UDP数据报。
- U *n* 在向网络写了数字*n*后进入TCP的紧急模式。写一个字节的数据以启动紧急模式（20.8节）。