

第5章 T/TCP协议的实现：插口层

5.1 概述

从本章开始我们讨论 Net/3 版中 T/TCP 协议的实现。我们沿用卷 2 的编排顺序和表达风格：

- 第5章：插口层
- 第6章：路由表
- 第7章：协议控制块 (PCB)
- 第8章：TCP 概述
- 第9章：TCP 输出
- 第10章：TCP 函数
- 第11章：TCP 输入
- 第12章：TCP 用户请求

在这些章节的介绍中，我们都假设用户已经有了本系列书的卷或者其中源代码的副本。这样我们就只要介绍实现 T/TCP 协议所需的 1 200 行新代码，而不需要重述卷 2 已介绍过的 15 000 行代码。

T/TCP 协议对插口层所作的改动是很小的：`sosend` 函数需要处理 `MSG_EOF` 标志，允许协议调用 `sendto` 函数和 `sendmsg` 函数隐式地打开和关闭连接。

5.2 常量

T/TCP 协议中需要使用三个常量：

- 1) `<sys/socket.h>` 中定义的 `MSG_EOF`。如果在调用 `send`、`sendto` 或 `sendmsg` 函数时设置了该标志，那么利用该连接发送数据就结束了，实际上就是结合了 `write` 和 `shutdown` 两个函数的功能。在卷 2 的图 16-12 中应该加上该标志。
- 2) `<sys/protosw.h>` 中定义了一个新的协议请求 `PRU_SEND_EOF`。在卷 2 的图 15-17 中应该加上该请求。这个请求是由 `sosend` 函数发出的，已在后面的图 5-2 中给出。
- 3) `<sys/protosw.h>` 中定义了一个新的协议标志 `PR_IMPLOPCL` (意指“隐式打开和关闭”)。这个标志有两重含义；(a) 协议允许在调用 `sendto` 或 `sendmsg` 函数时给定对等端的地址，而不必在此之前调用 `connect` 函数 (隐式打开)；(b) 协议能够识别 `MSG_EOF` 标志 (隐式关闭)。注意，只有在面向连接的协议 (如 TCP) 中才需要 (a)，因为在无连接的协议中总是可以直接调用 `sendto` 和 `sendmsg` 函数而不需要事先调用 `connect` 函数。应该在卷 2 的图 7-9 中加上该标志。

协议代码中 `switch` 程序块的 TCP 入口 `inetsw[2]` (卷 2 中图 7-13 的第 51~55 行) 应该在其 `pr_flags` 的标志值中加上 `PR_IMPLOPCL`。

5.3 sosend 函数

`sosend` 函数有两处改动。图 5-1 所示的代码用来替代卷 2 第 397 页中第 314~321 行的程序代码。

```

320         if ((so->so_state & SS_ISCONNECTED) == 0) {
321             /*
322              * sendto and sendmsg are allowed on a connection-
323              * based socket only if it supports implied connect
324              * (e.g., T/TCP).
325              * Return ENOTCONN if not connected and no address is
326              * supplied.
327              */
328             if ((so->so_proto->pr_flags & PR_CONNREQUIRED) &&
329                 (so->so_proto->pr_flags & PR_IMPLOPCL) == 0) {
330                 if ((so->so_state & SS_ISCONFIRMING) == 0 &&
331                     !(resid == 0 && clen != 0))
332                     snderr(ENOTCONN);
333             } else if (addr == 0)
334                 snderr(so->so_proto->pr_flags & PR_CONNREQUIRED ?
335                     ENOTCONN : EDESTADDRREQ);
336         }

```

uipc_socket.c

图5-1 sosend 函数：差错检查

注意，对代码的替换在这里实际上是从第 320行开始的，而不是第 314行。这是因为在这个文件的前面部分还有一些与 T/TCP无关的修改。由于我们要用这里的 17行代码替换卷2中的8行代码以支持T/TCP协议，因而该文件后面部分代码段中的行号也与卷2中对应的代码段的行号不一样。当本书提到卷 2中的代码段时，我们所说的行号通常都是指在卷2中的行号。由于卷3中的代码是卷2中的相应代码经过增删后得到的，因而相同功能代码段的行号会比较接近，但不一定相同。

320-336 修改后代码段的作用是：当设置了协议的 PR_IMPLOPCL标志、并且调用进程给出了目的地址时，允许在面向连接的插口上调用 sendto函数和sendmsg函数。如果调用进程没有给出目的地址，那么对应于 TCP插口将返回 ENOTCONN，对应于 UDP插口则返回 EDESTADDRREQ。

330-331 这个if语句使得当连接处于 SS_ISCONFIRMING状态时，允许只写控制信息而不写任何协议数据。OSI TP4协议采用了这种做法，TCP/IP协议则没有采用。

图5-2所示的是对 sendto函数的修改，图中代码用来替代卷 2第400页的第399~403行。

```

415         s = splnet();          /* XXX */
416         /*
417          * If the user specifies MSG_EOF, and the protocol
418          * understands this flag (e.g., T/TCP), and there's
419          * nothing left to send, then PRU_SEND_EOF instead
420          * of PRU_SEND.  MSG_OOB takes priority, however.
421          */
422         req = (flags & MSG_OOB) ? PRU_SENDOOB :
423             ((flags & MSG_EOF) &&
424              (so->so_proto->pr_flags & PR_IMPLOPCL) &&
425              (resid <= 0)) ? PRU_SEND_EOF : PRU_SEND;
426         error = (*so->so_proto->pr_usrreq) (so, req, top, addr, control);
427         splx(s);

```

uipc_socket.c

图5-2 sosend 函数：协议发送

我们第一次看到内容为xxx的评注。这是为了提醒读者，所注释的代码作用不明确，副作用也不明显，抑或是一个难题的快捷解决方法。本例中，`splnet`函数用于提高处理优先级，以优先执行这段代码。处理优先级用图 5-2底部所示的`splx`恢复。卷2的1.12节叙述了Net/3中各种中断的级别。

416-427 如果指定了MSG_OOB标志，那就发出PRU_SENDOOB请求。否则，如果指定了MSG_EOF标志，协议又支持PR_IMPLOPCL标志，而且再没有数据要交给协议了(`resid`小于或等于0)，那就发出PRU_SEND_EOF请求而不是通常的PRU_SEND请求。

回忆3.6节中的例子。应用程序调用`sendto`函数发送了3300字节数据，并指定了MSG_EOF标志。在`sosend`函数执行的第一次循环中，图5-2所示的代码发出了一个PRU_SEND请求，以发送前2048字节数据(一个mbuf簇)。在第二次循环中，发出PRU_SEND_EOF请求，以发送剩下的1252字节数据(在另一个mbuf簇中)。

5.4 小结

T/TCP给TCP增加了隐式打开和关闭的功能。所谓隐式打开是指应用程序不是通过调用`connect`函数建立连接，而是调用`sendto`函数或`sendmsg`函数并指定目的地址来建立连接。而隐式关闭则是指允许应用程序在调用`send`、`sendto`或`sendmsg`函数时指定MSG_EOF标志，从而把输出和关闭合并起来发布。图1-10中对`sendto`函数的调用就把打开、写数据和关闭合并在一个系统调用中实现。本章所示的程序代码修改给Net/3的插口层加上了隐式打开和关闭功能。