UNIX Network Programming

@于龙

目录

[1 简介 2](#_Toc419125659)

[1.1 概述 2](#_Toc419125660)

[1.2 时间获取程序 2](#_Toc419125661)

[2 传输层：TCP、UDP、SCTP 2](#_Toc419125662)

[2.1 TCP 2](#_Toc419125663)

[2.2 TCP连接的建立和终止 3](#_Toc419125664)

[2.3 TIME\_WAIT状态 4](#_Toc419125665)

[2.4 端口号 4](#_Toc419125666)

[2.5 缓冲区大小及限制 5](#_Toc419125667)

[3 套接字编程简介 5](#_Toc419125668)

[3.1 套接字地址结构 5](#_Toc419125669)

[3.2 字节排序函数 5](#_Toc419125670)

[3.3 readn、writen和readline函数 5](#_Toc419125671)

[4 基本TCP套接字编程 5](#_Toc419125672)

[4.1 socket函数 5](#_Toc419125673)

[4.2 connect函数 6](#_Toc419125674)

[4.3 bind函数 6](#_Toc419125675)

[4.4 listen函数 6](#_Toc419125676)

[4.5 accept函数 6](#_Toc419125677)

[4.6 close函数 6](#_Toc419125678)

[5 TCP客户/服务器程序示例 7](#_Toc419125679)

[5.1 正常启动 7](#_Toc419125680)

[5.2 正常终止 7](#_Toc419125681)

[5.3 POSIX信号处理 7](#_Toc419125682)

[5.4 accept返回前连接终止 7](#_Toc419125683)

[5.5 服务器进程终止 8](#_Toc419125684)

[5.6 SIGPIPE信号 8](#_Toc419125685)

[5.7 服务器主机崩溃 8](#_Toc419125686)

[5.8 服务器主机崩溃后重启 8](#_Toc419125687)

[6 I/O复用：select和poll函数 8](#_Toc419125688)

# 简介

## 概述

POSIX 表示可移植操作系统接口（Portable Operating System Interface ，缩写为 POSIX ），POSIX标准定义了操作系统应该为应用程序提供的接口标准.

POSIX标准意在期望获得源代码级别的软件可移植性.换句话说，为一个POSIX兼容的操作系统编写的程序，应该可以在任何其它的POSIX操作系统（即使是来自另一个厂商）上编译执行.

Linux基本上逐步实现了POSIX兼容，但并没有参加正式的POSIX认证.

微软的Windows NT声称部分实现了POSIX标准.

## 时间获取程序

read返回0表明对端关闭连接，返回负值表明发生错误

通常情况下，服务器进程在accept调用中被投入睡眠，等待某个客户连接的到达并被内核接收.TCP使用三次握手建立连接，连接建立后accept返回.

使用snprintf、strncat、strncpy来替代，防止缓冲区溢出.

close关闭与客户的连接，该调用引发每个方向上发送一二FIN，每个FIN又由对端确认.

# 传输层：TCP、UDP、SCTP

SCTP最初设计用于跨因特网传输电话信令.

SCTP与TCP类似之处在于也是一个可靠的传输协议，但它还提供消息边界、传输级别多宿支持以及将头端阻塞减少到最小的一种方法.

## TCP

TCP还提供了可靠性，当TCP向另一端发送数据时，它要求对端返回一个确认.如果没有收到确认，TCP就自动重传数据并等待更长时间.在数次重传失败后，TCP才放弃，如此在尝试发送数据上所花的总时间一般为4-10分钟.

TCP含有动态估算客户和服务器之间的往返时间的算法，以便指导等待一个确认需要多少时间.round-trip time rtt

TCP提供流量控制，总是告知对端在任何时刻一次能够从对端接收多少字节的数据，这称为通告窗口.该窗口指出接收缓冲区中当前可用的空间，从而确保发送端发送的数据不会使接收缓冲区溢出.

TCP是全双工连接，需要时可以转换成一个单工连接.

## TCP连接的建立和终止

三路握手：

SYN J

SYN K + ACK J + 1

ACK K + 1

TCP选项：

MSS最大分节大小

窗口规模选项 65535

时间戳选项 忽略

连接禁止：

FIN M

ACK M + 1

FIN N

ACK N + 1

当一个进程自愿或非自愿关闭时，所有打开的描述符都被关闭，这仍然导致FIN发送.

TCP状态转换图

服务器对客户请求的确认是伴随其应答发送的，这称为捎带piggybacking，它通常在服务器处理请求并产生应答的时间少于200ms时发生.如果服务器耗用更长时间，那么将先确认后应答.



## TIME\_WAIT状态

执行主动关闭的那端经历了这个状态.被动关闭是 CLOSE\_WAIT.

该端点停留在这个状态的持续时间是最长分节生命期MSL的两倍.2MSL

RFC 1122建议MSL为2分钟.源自Berkeley的实现传统上是30秒.MSL是任何IP数据包能在因特网中存活的最长时间.

分组在网络中迷途通常是由路由异常的结果.

TIME\_WAIT状态有两个存在的理由：

可靠地实现TCP全双工连接的终止；

允许老的重复分节在网络中消逝.

## 端口号

Unix系统有保留端口，小于1024的任何端口，使用这些端口的服务器必须以超级用户特权启动.

## 缓冲区大小及限制

IPv4 65535

IPv6 65575

MTU 以太网1500

当一个IP数据包超过相应链路MTU时，IPv4和IPv6都将执行分片，这些分片到达最终目的地之前通常不会被重组.v4路由器会对转发的数据进行分片，v6路由器不会.

IPv4 和 IPv6都定义了最小重组缓冲区大小，它是任何实现都必须保证支持的最小数据包大小.IPv4 576 IPv6 1500

TCP有一个MSS，最大分节大小，用于向对端TCP通告对端在每个分节中能发送的最大TCP数据量.试图避免分片.

# 套接字编程简介

## 套接字地址结构

结构的名字以sockaddr\_开头，对每个协议族的唯一后缀结尾.

sockaddr\_in IPv4

sockaddr\_in6 IPv6

地址结构中的数据总是以网络字节序来存储.

## 字节排序函数

使用字节排序函数时，并不关心主机字节序和网络字节序的真实值，在使用网络字节序的系统中，这些转换函数被定义为空宏.

## readn、writen和readline函数

字节流套接字如TCP上调用read或write输入或输出的字节数可能比请求的数量少.

这个现象的原因在于内核中用于套接字的缓冲区可能已达到了极限.在read时很常见，在write只有非阻塞时才能出现.

# 基本TCP套接字编程

## socket函数

socket函数在成功时返回一个小的非负整数值，称为套接字描述符.

只指定了协议族和套接字类型，并没有指定地址.

## connect函数

在调用connect前不必非得调用bind函数，如果需要的话内核会确定源IP地址、并选择一个临时端口作为源端口.

如果是TCP，调用connect函数将激发TCP三路握手，在建立成功或出错时才返回.

若客户端没收到SYN的响应，则返回ETIMEOUT，在等待响应中会有超时重发.

对客户的SYN的响应是RST，则表明该服务器主机在指定的端口上没有进程等待与之连接，会返回ECONNREFUSED.

客户发出的SYN在中间的某个路由器上引发ICMP错误，则主机进行超时重发，若在规定的时间内仍未收到响应，则作为EHOSTUNREACH或ENETUNREACH返回.

## bind函数

bind函数把一个本地协议地址赋予一个套接字.

在没有bind一个地址时，connect或listen就要为相应的套接字选择一个临时端口号.

对于客户端，这就为在该套接字上发送的数据指定了源IP；

对于服务端，则限定该套接字只接收那些目的地为这个IP地址的连接.

如果TCP服务器没有bind，内核就把客户发送的SYN的目的IP做为源IP.

## listen函数

listen把一个未连接的套接字转换成一个被动套接字，指示内核应该接收指向该套接字的连接请求.

backlog指定套接字排队的最大连接个数.

内核为任何一个给定的监听套接字维护两个队列：

未完成连接队列：服务器等待完成三路握手；

已完成连接队列：已完成三路握手，处于ESTABLISHED状态.

当进程调用accept时，已完成连接队列中的队头将返回给进程，如果队列为空，那么进程将进入睡眠.

有些系统可能需要修改内核的backlog.

若这些队列是满的，TCP就忽略该分节，也就是不发送RST，已包装客户将重发SYN.

## accept函数

accept用于从已完成连接队列队头返回下一个已完成连接.

## close函数

close把一个套接字标记为已关闭，此套接字不能再作为read或write的第一个参数.

TCP尝试发送已排队等待发送到对端的数据，发送完毕后发送正常的终止序列.

可以用shutdown发送FIN.

注意父子进程引用计数问题.

# TCP客户/服务器程序示例

搞清边界条件

## 正常启动

## 正常终止

客户端进程终止，系统会关闭所有打开的描述符，并进行关闭序列的发送处理，最终客户端进入了TIME\_WAIT状态.

## POSIX信号处理

子进程终止会给父进程发送一个SIGCHLD信号，如果父进程未加处理，子进程于是进入僵死状态.  
信号是告知某个进程发送了某个事件的通知，有时也称为软件中端，信号是异步发生的，进程预先不知道信号发生的时刻.

信号可以：

一个进程发送给另一个进程；

由内核发送给某个进程.

每个信号可以有一个与之关联的处置，sigaction.

SIGKILL和SIGSTOP信号不能捕获.

可以设置SIG\_IGN来忽略. SIGKILL和SIGSTOP信号不能忽略.

可以把信号的处置设置为SIG\_DFL来启动默认处置，默认的通常是收到信号后终止进程.

信号会导致accept返回ENTER.

## accept返回前连接终止

三次握手完成从而连接建立之后，客户TCP发送了一个RST，在服务器端该连接已由TCP排队，等着服务器进程调用accept时RST到达.

有些实现会在内核中处理，有些会返回EPROTO.POSIX指出值必须是ECONNABORTED.

## 服务器进程终止

使用kill命令杀死进程，作为进程终止处理的部分工作，子进程所有打开着的描述符都被关闭.

客户端继续write会收到RST信号.read返回0

## SIGPIPE信号

当一个进程向某个已收到RST的套接字执行写操作时，内核向该进程发送一个SIGPIPE信号，该信号的默认行为时终止进程.

不论该进程是捕获了该信号并从其信号处理函数返回，还是简单的忽略了该信号，写操作都返回EPIPE错误.

也就是第一次写引发RST，第二次写引发SIGPIPE信号.

## 服务器主机崩溃

服务器主机崩溃或直接断网，Berkeley的实现会重传该数据12次，共9分钟才放弃重传.read会返回ETIMEOUT，如果某个中间路由器判定服务器不可达，则发送ICMP，所返回的是EHOSTUNREACH或ENETUNREACH.

## 服务器主机崩溃后重启

服务器会发送RST.

# I/O复用：select和poll函数

## I/O模型

Unix下可用的5种I/O模型：

阻塞式I/O

非阻塞式I/O：轮询内核，往往耗费大量CPU

I/O复用

信号驱动式I/O：让内核在描述符就绪时发送SIGIO信号通知我们.

异步I/O

## select函数

该函数允许进程指示内核等待多个事件中的任何一个发生，并只在有一个或多个事件发生或经历一段指定的时间后才唤醒它.