Sockets

张海宁

贵州大学

hnzhang1@gzu.edu.cn

May 22, 2018

Overview

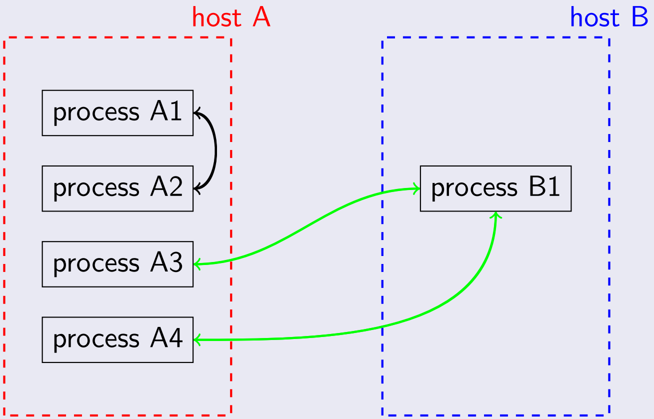
Sockets

What is a Socket

Socket

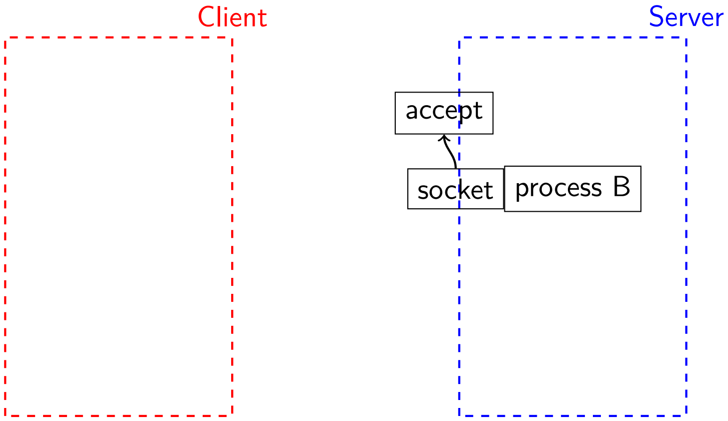
Socket 是一种进程间的通信机制。

进程间通信



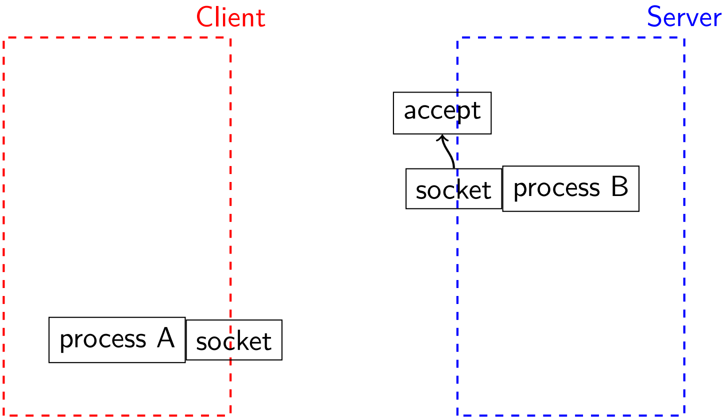
Socket 通信过程 I

服务器端的进程 process B 通过 socket 系统调用创建了一个 socket，这个 socket 对外表现就是一个端口。比如 web 常用端口 80 。该 socket 随即调用 accept 方法，等待客户端的连接。



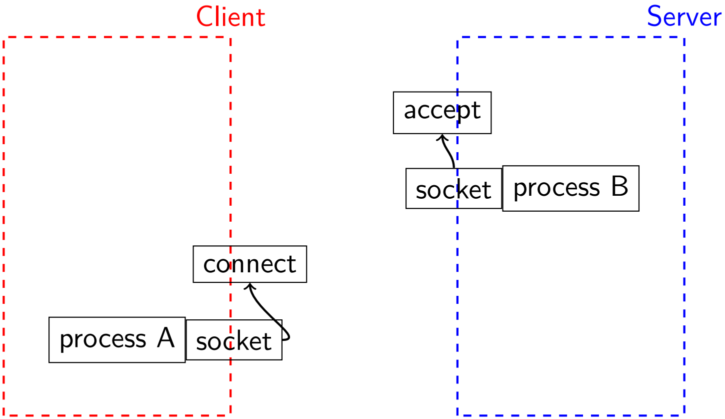
Socket 通信过程 II

客户端的进程 process A 通过 socket 系统调用创建一个 socket。



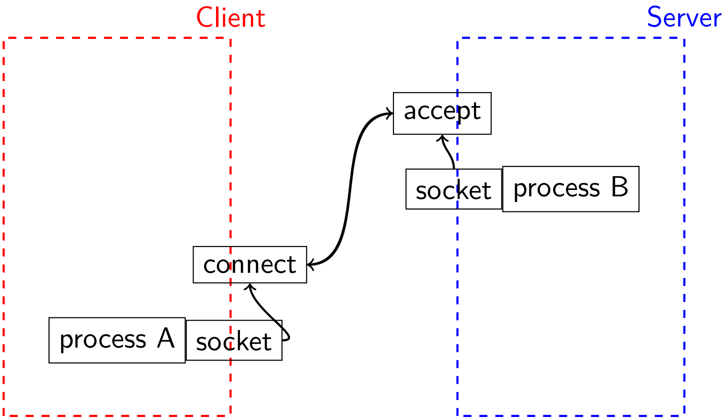
Socket 通信过程 III

客户端的进程 process A 创建的 socket 将服务器的地址和端口传递给 connect 方法。



Socket 通信过程 IV

客户端与服务端建立连接。然后 processA 和 B 就像操作文件描述符一样可以对 socket 进行操作了。



Create a Socket

socket 系统调用

socket() creates an endpoint for communication and returns a descriptor.

原型

```
#include <sys/socket.h>

int
socket(int domain, int type, int protocol);
```

socket-domain

The domain parameter specifies a communications domain within which communication will take place; this selects the protocol family which should be used. These families are defined in the include file <sys/socket.h>. The domains are as table 1.

Domain	Description
PF_LOCAL	Host-internal protocols, formerly called PF_UNIX
PF_UNIX	Host-internal protocols, deprecated, use PF_LOCAL
PF_INET	Internet version 4 protocols
PF_ROUTE	Internal Routing protocol
PF_KEY	Internal key-management function
PF_INET6	Internet version 6 protocols
PF_SYSTEM	System domain
PF_NDRV	Raw access to network device

Table: Domain of socket

socket-type

The socket has the indicated type, which specifies the semantics of communication. Currently defined types are:

1. SOCK_STREAM
2. SOCK_DGRAM
3. SOCK_RAW

A SOCK_STREAM type provides sequenced, reliable, two-way connection based byte streams. An out-of-band data(like urgent mode in tcp) transmission mechanism may be supported.
A SOCK_DGRAM socket supports datagrams (connectionless, unreliable messages of a fixed (typically small) maximum length).
SOCK_RAW sockets provide access to internal network protocols and interfaces. The type SOCK_RAW, which is available only to the super-user.

socket-protocol

The protocol used for communication is usually determined by the socket type and domain. There is normally no choice. The protocol parameter is used where there is a choice. 0 selects the default protocol, which is used in all the examples in our course.

Name a socket

To make a socket (as created by a call to socket) available for use by other processes, a server program needs to give the socket a name.
Thus, PF_LOCAL sockets are associated with a file system pathname.
PF_INET sockets are associated with an IP port number.

bind

bind 原型

```
#include <sys/socket.h>
int bind(int socket, const struct sockaddr *address,
  size_t address_len);
```

The bind system call assigns the address specified in the parameter, address, to the unnamed socket associated with the file descriptor socket. The length of the address structure is passed as address_len. The length and format of the address depend on the address family. A particular address structure pointer will need to be cast to the generic address type(struct sockaddr *) in the call to bind. On successful completion, bind returns 0. If it fails, it returns -1 and sets errno .

create a socket queue

To accept incoming connections on a socket, a server program must create a queue to store pending requests. It does this using the listen system call.

listen 原型

```
#include <sys/socket.h>
int listen(int socket, int backlog);
```

The listen function will return 0 on success or -1 on error.

accept connection

Once a server program has created and named a socket, it can wait for connections to be made to the socket by using the accept system call.

accept 原型

```
#include <sys/socket.h>
int accept(int socket, struct sockaddr *address,
  size_t *address_len);
```

The accept system call returns when a client program attempts to connect to the socket specified by the parameter socket.
The client is the first pending connection from that socket’ s queue.
The accept function creates a new socket to communicate with the client and returns its descriptor.
The new socket will have the same type as the server listen socket.

accept 补充

If there are no connections pending on the socket’ s queue, accept will block (so that the program won’ t continue) until a client does make connection.
(We can change this behavior by using the O_NONBLOCK flag on the socket file descriptor, using the fcntl function.)

Requesting Connections

Client programs connect to servers by establishing a connection between an unnamed socket and the server listen socket. We can do this by calling connect.

connect 原型

```
#include <sys/socket.h>
int connect(int socket, const struct sockaddr *address,
  size_t address_len);
```

The socket specified by the parameter socket is connected to the server socket specified by the parameter address, which is of length address_len. The socket must be a valid file descriptor obtained by a call to socket. If it succeeds, connect returns 0, and -1 is returned on error.

close a socket

We can terminate a socket connection at the server and client by calling close, just as we would for low-level file descriptors.

connect 原型

```
#include <unistd.h>


     int
     close(int fildes);
```

Socket Communications

作业

编写一个程序，要求：

1. 创建两个线程：一个生产者线程，一个消费者线程
2. 当缓冲区为空时，生产者线程提示用户输入一串字符，然后将用户输入的字符写入缓冲区
3. 当缓冲区非空时，消费者线程从缓冲出取出所有字符
4. 当用户输入 end 时，程序结束

The End

Appendix

本课程相关资源下载

1. ppt
   https://github.com/gmsft/ppt/tree/master/linux
2. 实验指导书
   https://github.com/gmsft/ppt/tree/master/book/linux

about man page

The manual is generally split into eight numbered sections, organized as follows (on Research Unix, BSD, macOS and Linux):

section	**description**
1	General commands
2	System calls
3	Library function(C standard library)
4	Special files(devices) and drivers
5	File formats and conventions
6	Games and screensavers
7	Miscellanea
8	System administration commands and daemons

Table: man page

在终端中运行 man read **与** man 2 read **，观察其输出的区别。**

detached threads

前面的多线程程序中，主线程都调用了 pthread_join 方法来等待线程结束。在主线程需要子线程返回结果的时候，这种方法是必要的。但是在主线程不需要子线程的返回结果时，就没必要这样做了，只需要让子线程自已去运行，然后自行结束就可以了，这样的线程就被称为 *detached threads*。
在任何一个时间点上，线程是可结合的（joinable）或者是分离的 (detached)。一个可结合的线程能够被其他线程收回其资源和杀死。在被其他线程回收之前，它的存储器资源（例如栈）是不释放的。相反，一个分离的线程是不能被其他线程回收或杀死的，它的存储器资源在它终止时由系统自动释放。

detached threads 示例

```
pthread_attr_t att;
pthread_attr_init(&att);
pthread_attr_setdetachstate(&att,PTHREAD_CREATE_DETACHED);
char  ch='a';
int ra=pthread_create(&pth[0],&att,fmt,&ch);
```