Overall flow chart and select()

—— *Final Project*

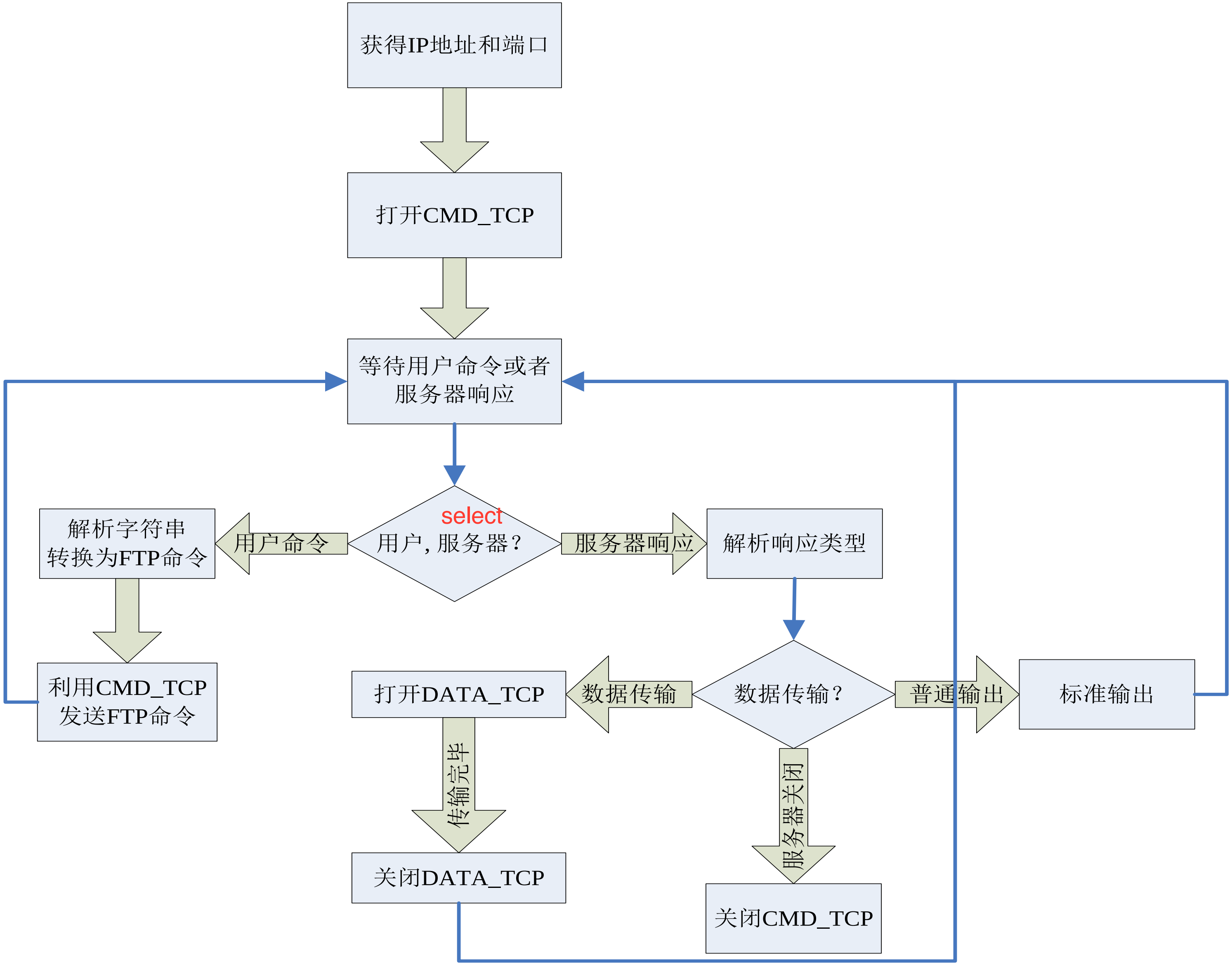
BUPT/QMUL

2019-5-16

Contents

- Overall flow chart
- System call *select()* for I/O multiplexing

Overall flow chart



获得IP地址和端口

打开CMD_TCP

等待用户命令或者服务器响应

select
用户，服务器？

用户命令

解析字符串转换为FTP命令

利用CMD_TCP发送FTP命令

服务器响应

解析响应类型

数据传输？

数据传输

打开DATA_TCP

传输完毕

关闭DATA_TCP

普通输出

标准输出

服务器关闭

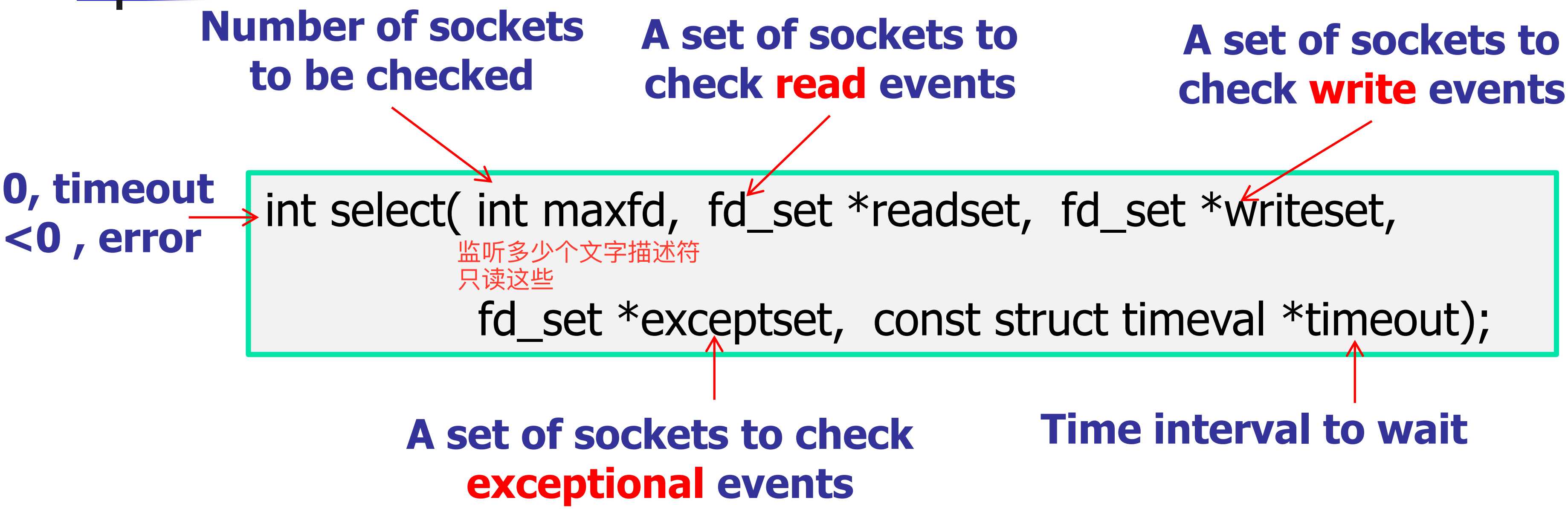
关闭CMD_TCP

About *select()*

- Traditional sockets: blocking I/O mode
- Allowing synchronous I/O (including sockets) multiplexing, enabling a program to monitor multiple file descriptors, thus to handle I/O operations of multiple connections

- You must include
  - sys/select.h and sys/time.h

About *select()*



Number of sockets to be checked

A set of sockets to check **read** events

A set of sockets to check **write** events

0, timeout
<0 , error

```
int select( int maxfd,  fd_set *readset,  fd_set *writeset,
            fd_set *exceptset,  const struct timeval *timeout);
```

监听多少个文字描述符
只读这些

A set of sockets to check **exceptional** events

Time interval to wait

- FD_ZERO(): clearing a set
- FD_SET(): adding a given file descriptor to a set
- FD_ISSET(): testing to see if a file descriptor is part of the set

select() example

```
int fd, next=0;                     /* original socket */
int newfd[10];                /* new socket descriptors */
while(1) {
      fd_set readfds;
      FD_ZERO(&readfds); FD_SET(fd, &readfds);

      /* Now use FD_SET to initialize other newfd's
         that have already been returned by accept() */

      select(maxfd+1, &readfds, 0, 0, 0);
      if(FD_ISSET(fd, &readfds)) {
            newfd[next++] = accept(fd, ...);
      }
      /* do the following for each descriptor newfd[n] */
      if(FD_ISSET(newfd[n], &readfds)) {
            read(newfd[n], buf, sizeof(buf));
            /* process data */
      }
}
```

Now the server can support multiple connections...

Select() in main.c

```
                fd_set      rset;
                FD_ZERO(&rset);
                 maxfdp1 = sockfd + 1;      /* check descriptors [0..sockfd] */
                 for ( ; ; )
                 {
                            FD_SET(STDIN_FILENO, &rset);
                            FD_SET(sockfd, &rset);
                            if (select(maxfdp1, &rset, NULL, NULL, NULL) < 0)
                                      printf("select error\n");
                            /* data to read on stdin */
                            if (FD_ISSET(STDIN_FILENO, &rset)) {
                             if ( (nread = read(STDIN_FILENO, rbuf, MAXBUF)) < 0)
                                                printf("read error from stdin\n");
    …                       /* data to read on cmd_tcp */
                           if (FD_ISSET(sockfd, &rset)) {
                                      if ( (nread = recv(sockfd, rbuf, MAXBUF, 0)) < 0)
                                                printf("recv error\n");

    …
```