Ref：Linux网络编程 7.3节

**服务器：**

/\*-------------------tcp\_server.c--------------------\*/

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <unistd.h>

#define PORT 8888 /\*侦听端口地址\*/

#define BACKLOG 2 /\*侦听队列长度\*/

int main(int argc, char \*argv[])

{

int ss,sc; /\*ss为服务器的socket描述符，sc为客户端的socket描述符\*/

struct sockaddr\_in server\_addr; /\*服务器地址结构\*/

struct sockaddr\_in client\_addr; /\*客户端地址结构\*/

int err; /\*返回值\*/

pid\_t pid; /\*分叉的进行ID\*/

/\*建立一个流式套接字\*/

ss = socket(AF\_INET, SOCK\_STREAM, 0);

if(ss < 0){ /\*出错\*/

printf("socket error\n");

return -1;

}

/\*设置服务器地址\*/

bzero(&server\_addr, sizeof(server\_addr)); /\*清零\*/

server\_addr.sin\_family = AF\_INET; /\*协议族\*/

server\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY); /\*本地地址\*/

server\_addr.sin\_port = htons(PORT); /\*服务器端口\*/

/\*绑定地址结构到套接字描述符\*/

err = bind(ss, (struct sockaddr\*)&server\_addr, sizeof(server\_addr));

if(err < 0){/\*出错\*/

printf("bind error\n");

return -1;

}

/\*设置侦听\*/

err = listen(ss, BACKLOG);

if(err < 0){ /\*出错\*/

printf("listen error\n");

return -1;

}

/\*主循环过程\*/

for(;;) {

socklen\_t addrlen = sizeof(struct sockaddr);

sc = accept(ss, (struct sockaddr\*)&client\_addr, &addrlen);

/\*接收客户端连接\*/

if(sc < 0){ /\*出错\*/

continue; /\*结束本次循环\*/

}

/\*建立一个新的进程处理到来的连接\*/

pid = fork(); /\*分叉进程\*/

if( pid == 0 ){ /\*子进程中\*/

process\_conn\_server(sc); /\*处理连接\*/

close(ss); /\*在子进程中关闭服务器的侦听\*/

}else{

close(sc); /\*在父进程中关闭客户端的连接\*/

}

}

}

**客户端：**

/\*-------------------tcp\_client.c--------------------\*/

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <unistd.h>

#include <arpa/inet.h>

#define PORT 8888 /\*侦听端口地址\*/

int main(int argc, char \*argv[])

{

int s; /\*s为socket描述符\*/

struct sockaddr\_in server\_addr; /\*服务器地址结构\*/

s = socket(AF\_INET, SOCK\_STREAM, 0); /\*建立一个流式套接字 \*/

if(s < 0){ /\*出错\*/

printf("socket error\n");

return -1;

}

/\*设置服务器地址\*/

bzero(&server\_addr, sizeof(server\_addr)); /\*清零\*/

server\_addr.sin\_family = AF\_INET; /\*协议族\*/

server\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY); /\*本地地址\*/

server\_addr.sin\_port = htons(PORT); /\*服务器端口\*/

/\*将用户输入的字符串类型的IP地址转为整型\*/

inet\_pton(AF\_INET, argv[1], &server\_addr.sin\_addr);

/\*连接服务器\*/

connect(s, (struct sockaddr\*)&server\_addr, sizeof(struct sockaddr));

process\_conn\_client(s); /\*客户端处理过程\*/

close(s); /\*关闭连接\*/

return 0;

}

/\*-------------------tcp\_process.c--------------------\*/

**客户端：**

#include <stdio.h>

#include <string.h>

/\*客户端的处理过程\*/

void process\_conn\_client(int s)

{

ssize\_t size = 0;

char buffer[1024]; /\*数据的缓冲区\*/

for(;;){ /\*循环处理过程\*/

/\*从标准输入中读取数据放到缓冲区buffer中\*/

size = read(0, buffer, 1024);

if(size > 0){ /\*读到数据\*/

write(s, buffer, size); /\*发送给服务器\*/

size = read(s, buffer, 1024); /\*从服务器读取数据\*/

write(1, buffer, size); /\*写到标准输出\*/

}

}

}

**服务器端：**

/\*服务器对客户端的处理\*/

void process\_conn\_server(int s)

{

ssize\_t size = 0;

char buffer[1024]; /\*数据的缓冲区\*/

for(;;){ /\*循环处理过程\*/

size = read(s, buffer, 1024); /\*从套接字中读取数据放到 缓冲区buffer中\*/

if(size == 0){ /\*没有数据\*/

return;

}

/\*构建响应字符，为接收到客户端字节的数量\*/

sprintf(buffer, "%d bytes altogether\n", size);

write(s, buffer, strlen(buffer)+1);/\*发给客户端\*/

}

}

Q: 使用fork后，在子进程关闭主进程的监听套接字？

A: 见UNP 4.8节。