**简单 TCP 协议实验**

**一、实验目的**

通过编程实验，学生可以深入理解 TCP 协议原理，掌握连接状态控制、可靠传输等重要机制，还可以进一步探索拥塞控制算法等前沿问题。TCP 协议中的状态控制机制和拥塞控制算法是协议的核心部分。TCP 协议的复杂性主要源于它是一个有状态的协议，需要进行状态的维护和变迁。有限状态机可以很好的从逻辑上表示 TCP 协议的处理过程，理解和实现 TCP 协议状态机是本实验的重点内容。另外， TCP 协议还要向应用层提供编程接口，即网络编程中常用的 Socket 接口。通过实现这些接口函数，可以深入了解网络编程的原理，提高网络程序的设计和调试能力。

**二、实验说明**

从设计实现的角度出发，TCP 协议主要考虑以下几个因素：

1）状态控制

2）滑动窗口机制

3）拥塞控制算法

4）性能问题（RTT 估计）

5）Socket 接口

TCP 协议是非常复杂的，将这些内容完全实现具有很大的难度，很难在一个实验中完成

所有的内容。出于工作量和实验复杂度的考虑，本实验对 TCP 协议进行适当的简化，仅实现客户端角色的、“停—等”模式的 TCP 协议，能够正确的建立和拆除连接，接收和发送 TCP

段，并向应用层提供客户端需要的 Socket 接口。

一般情况下， TCP 协议通过一个数据结构来控制每个 TCP 连接的发送和接收动作，该数据结构被称为传输控制块（TCB） 。TCP 为每个活动的连接维护一个 TCB 结构， 所有 TCP 连接对应的 TCB 结构可以组织成一个 TCB 链表结构进行管理。 TCB 中包含了有关 TCP 连接的所有信息，包括两端的 IP 地址和端口号、连接状态信息、发送和接收窗口信息等。TCB 的数据结构由学生来设计完成，本实验不作具体规定，学生可以根据设计实现的需要来具体定义 TCB 结构。

**三、实验要求**

本实验的要求分为以下几点：(1)理解 TCP 协议的主要原理，针对客户端角色的、“停-等”模式的 TCP 协议，设计接收和发送流程；(2)编程实现 TCP 段的接收流程，重点是段接收的有限状态机；(3)编程实现 TCP 段的发送流程，完成 TCP 段的封装处理；(4)编程实现客户端的 Socket 接口函数。为完成本实验的要求，学生需完成以下几项内容：

(1)设计 TCB 数据结构

该数据结构用于保存 TCP 连接的相关信息。

(2)TCP 协议的接收处理

学生需要实现 stud\_tcp\_input()函数，完成校验和检查、字节序转换功能（对头部中的选项不做处理） ，重点实现客户端角色的 TCP 段接收的有限状态机。不采用捎带确认机制，收到数据后马上回复确认，以满足“停-等”模式的要求。

(3)TCP 协议的封装发送

学生需要实现 stud\_tcp\_output()函数，完成简单的 TCP 协议的封装发送功能。为保证可靠传输，要在收到对上一个段的确认后才能够继续发送。

(4)TCP 协议提供的 Socket 接口函数

实现与客户端角色的 TCP 协议相关的 5 个 Socket 接口函数：stud\_tcp\_socket() 、stud\_tcp\_connect()、stud\_tcp\_recv()、stud\_tcp\_send()和 stud\_tcp\_close()，将接口函数实现的内容与发送和接收流程有机地结合起来。

**四、接口函数说明**

本实验需要学生实现一些接口函数，同时实验系统也为学生提供了一些接口函数。现分别对这两类接口函数作详细说明。

(1) 需要实现的接口函数

本实验中需要学生实现的接口函数较多，包括针对段交互的测试函数和针对 socket API的测试函数两类。

<1> TCP 段输入函数

int stud\_tcp\_input(char\* pBuffer, unsigned shortlen, unsigned int srcAddr, unsigned int

dstAddr)

参数：

char \*pBuffer：指向接收缓冲区的指针，从 TCP 头开始。

unsigned short len：缓冲区数据长度。

unsigned int srcAddr：源 IP 地址。

unsigned int dstAddr：目的 IP 地址。

返回值：

如果成功则返回 0，否则返回-1。

<2> TCP 段输出函数：

void stud\_tcp\_output(char \*pData, unsigned short len, unsigned char flag, unsigned short srcPort, unsigned short dstPort, unsigned int srcAddr, unsigned int dstAddr)

参数：

char \*pData：数据指针。

unsigned short len：数据长度。

unsigned char flag：段类型。

unsigned short srcPort：源端口。

unsigned short dstPort：目的端口。

unsigned int srcAddr：源 IP 地址。

unsigned int dstAddr：目的 IP 地址。

其中，段类型 flag 为以下几种可能：

1> PACKET\_TYPE\_DATA：数据。

2> PACKET\_TYPE\_SYN：SYN 标志位打开。

3> PACKET\_TYPE\_SYN\_ACK：SYN、ACK 标志位打开。

4> PACKET\_TYPE\_ACK：ACK 标志位打开。

5> PACKET\_TYPE\_FIN：FIN 标志位打开。

6> PACKET\_TYPE\_FIN\_ACK：FIN、ACK 标志位打开。

<3> 获得 socket 描述符函数：

int stud\_tcp\_socket(int domain, int type, int protocol)

参数：

int domain：套接字标识符，默认为 INET。

int type：类型，默认为 SOCK\_STREAM。

int protocol：协议，默认为 IPPROTO\_TCP。

返回值：

如果正确建立连接则返回 socket 描述符，否则返回-1。

<4> TCP 建立连接函数：

int stud\_tcp\_connect(int sockfd, struct sockaddr\_in\* addr, int addrlen)

参数：

int sockfd：套接字标识符。

struct sockaddr\_in\* addr：socket 地址结构指针。

int addrlen socket：地址结构的大小。

返回值：

如果正确发送则返回 0，否则返回-1。

<5> TCP 段发送函数：

int sutd\_tcp\_send(int sockfd, const unsigned char\* pData, unsigned short datalen, int

flags)

参数：

int sockfd：套接字标识符。

const unsigned char\* pData：数据缓冲区指针。

unsigned short datalen：数据长度。

int flags：标志。

返回值：

如果正确接收则返回 0，否则返回-1。

<6> TCP 段接收函数：

int stud\_tcp\_recv(int sockfd, const unsigned char\* pData, unsigned short datalen, int

flags)

参数：

int sockfd：套接字标识符。

const unsigned char\* pData：数据缓冲区指针。

unsigned short datalen：数据长度。

int flags：标志。

返回值：

如果正确接收则返回 0，否则返回-1。

<7> TCP 关闭连接函数：

int stud\_tcp\_close(int sockfd)

参数：

int sockfd：连接描述符。

返回值：

如果正常关闭则返回 0，否则返回-1。

(2) 系统提供的接口函数说明

<1> TCP 丢弃段函数：

void tcp\_DiscardPkt(char\* pBuffer, int type)

参数：

char\* pBuffer：指向被丢弃的分组。

int type：分组丢弃的原因，有以下 3 种可能：

1> STUD\_TCP\_TEST\_SEQNO\_ERROR：序号错误。

2> STUD\_TCP\_TEST\_SRCPORT\_ERROR：源端口错误。

3> STUD\_TCP\_TEST\_DSTPORT\_ERRPR：目的端口错误。

<2> IP 分组发送函数：

void tcp\_sendIpPkt(unsigned char\* pData, uint16 len, unsigned int srcAddr, unsigned int dstAddr, uint8 ttl)

参数：

pData：IP 上层协议数据。

len：IP 上层协议数据长度。

srcAddr：源 IP 地址。

dstAddr：目的 IP 地址。

ttl：最大跳数。

<3> IP 数据分组主动接收函数：

int waitIpPacket(char \*pBuffer, int timeout)

参数：

char \*pBuffer：接收缓冲区的指针。

int timeout：超时时间。

返回值：

如果正确接收则返回接收到的数据长度，否则返回-1。

<4> 客户端获得本机 IPv4 地址：

UINT32 getIpv4Address()

<5> 客户端获得服务器 IPv4 地址：

UINT32 getServerIpv4Address()

(3) 全局变量

本实验中涉及的全局变量包括源端口号、目的端口号、TCP 序号和确认号。这些变量的取值可由学生自己定义，比如可以做如下定义：

int gSrcPort = 2008

int gDstPort = 2009

int gSeqNum = 1234

int gAckNum = 0

需要注意的是，序号不能从 0 开始定义，否则系统调用会出现错误。