哈爾濱工業大學
Harbin Institute of Technology

计算机网络
课程实验报告

实验名称	可靠数据传输协议的设计与实现				
姓名	林燕燕		院系	人工智能	
班级	1903601		学号	1190200501	
任课教师	李全龙		指导教师	李全龙	
实验地点	格物 207		实验时间	2021.11.06	
实验课表现	出勤、表现得分(10)		实验报告得分(40)	实验总分	
	操作结果得分(50)				
教师评语					

计算机科学与技术学院 SINCE 1956...
School of Computer Science and Technology

实验目的：

理解可靠数据传输的基本原理；掌握停等协议的工作原理；掌握基于 UDP 设计并实现一个停等协议的过程与技术。

理解滑动窗口协议的基本原理；掌握 GBN 的工作原理；掌握基于 UDP 设计并实现一个 GBN 协议的过程与技术。

实验内容：

1. 可靠数据传输协议-停等协议的设计与实现
   a) 基于 UDP 设计一个简单的停等协议，实现单向可靠数据传输（服务器到客户的数据传输）。
   b) 模拟引入数据包的丢失，验证所设计协议的有效性。
   c) 改进所设计的停等协议，支持双向数据传输；（选作内容，加分项目，可以当堂完成或课下完成）
   d) 基于所设计的停等协议，实现一个 C/S 结构的文件传输应用。（选作内容，加分项目，可以当堂完成或课下完成）
2. 可靠数据传输协议-GBN 协议的设计与实现
   a) 基于 UDP 设计一个简单的 GBN 协议，实现单向可靠数据传输（服务器到客户的数据传输）。
   b) 模拟引入数据包的丢失，验证所设计协议的有效性。
   c) 改进所设计的 GBN 协议，支持双向数据传输；（选作内容，加分项目，可以当堂完成或课下完成）
   d) 将所设计的 GBN 协议改进为 SR 协议。（选作内容，加分项目，可以当堂完成或课下完成）

实验过程：

停等协议：

1) 基于 UDP 实现的停等协议，可以不进行差错检测，可以利用 UDP协议差错检测；

2) 为了验证所设计协议是否可以处理数据丢失，可以考虑在数据接收端或发送端引入数据丢失。

3) 在开发停等协议之前，需要先设计协议数据分组格式以及确认分组格式。

4) 计时器实现方法：对于阻塞的 socket 可用 int setsockopt(int socket,int level, int option_name, const void* option_value, size_t option_len)函数设置套接字发送与接收超时时间；对于非阻塞 socket 可以使用累加 sleep时间的方法判断 socket 接受数据是否超时(当时间累加量超过一定数值时则认为套接字接受数据超时)。

GBN协议：

1) 基于 UDP 实现的 GBN 协议，可以不进行差错检测，可以利用UDP 协议差错检测；

2) 自行设计数据帧的格式，应至少包含序列号 Seq 和数据两部分；

3) 自行定义发送端序列号 Seq 比特数 L 以及发送窗口大小 W，应满足条件 W+1<=2L。

4) 一种简单的服务器端计时器的实现办法：设置套接字为非阻塞方式，则服务器端在 recvfrom 方法上不会阻塞，若正确接收到 ACK 消息，则计时器清零，若从客户端接收数据长度为-1（表示没有接收到任何数据），则计时器+1，对计时器进行判断，若其超过阈值，则判断为超时，进行超时重传。（当然，如果服务器选择阻塞模式，可以用到 select 或 epoll的阻塞选择函数，详情见 MSDN）

5) 为了模拟 ACK 丢失，一种简单的实现办法：客户端对接收的数据帧进行计数，然

后对总数进行模 N 运算，若规定求模运算结果为零则返回 ACK，则每接收 N 个数据帧才返回 1 个 ACK。当 N 取值大于服务器端的超时阀值时，则会出现服务器端超时现象。

6) 当设置服务器端发送窗口的大小为 1 时，GBN 协议就是停-等协议。

首先，实现 GBN 的单向传输，创建一个套接字，并绑定在指定的端口上。客户端请求数据，读取控制台的请求信息，并解析该命令。根据不同的命令，请求不同的数据。

当执行单向传输的命令时，客户端首先发送请求信息，然后服务器端解析请求，进行一个握手阶段，首先服务器向客户端发送一个 205 大小的状态码（我自己定义的） 表示服务器准备好了，可以发送数据；客户端收到 205 之后回复一个 200 大小的状态码，表示客户端准备好了，可以接收数据了；服务器收到 200 状态码之后，就开始使用 GBN 发送数据了，服务器端读取本地文件，放到缓存中，发送给客户端。

在发送端设置分组丢失率和ACK丢失率，ACK采用累积确认，当收到一个字段的序列号时，在其之前的所有分组全都确认被收到。当发生超时情况时，发送端重新发送整个窗户中的所有数据分组。

GBN 双向传输也是相同的原理。只不过发送端变成了客户端。

实验结果：

数据分组格式：

Seq 为 1 个字节，取值为 0~255，（故序列号最多为 256 个）；

Data≤1024 个字节，为传输的数据；

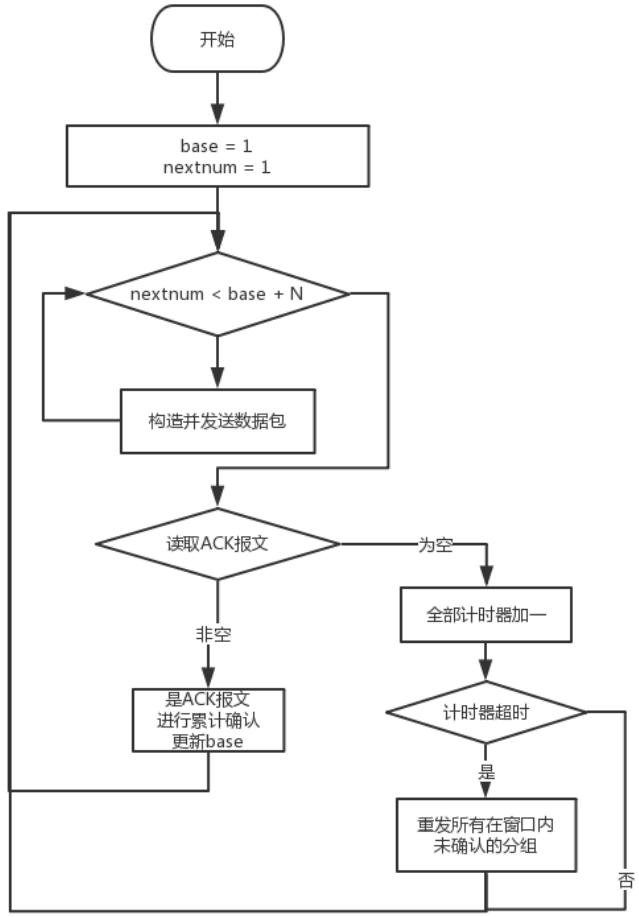
最后一个字节放入 EOF0，表示结尾。

确认分组格式：

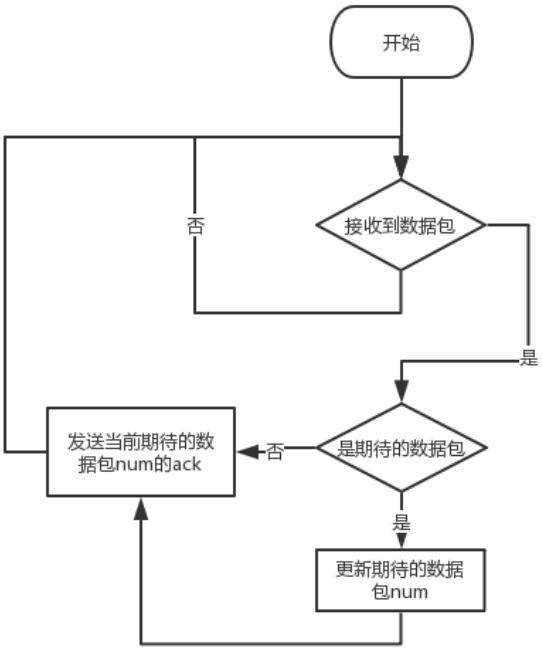
ACK 字段为一个字节，表示序列号数值；

末尾放入 0，表示数据结束。

服务器端：



客户端：



数据分组丢失：

设置一个分组丢失率，将丢失率乘上100，随机生成一个0到100的数，检查该数是否在0到丢失率的范围内。如果在，则分组丢失；否则，没有丢失。

主要函数及其作用：

void getCurTime(char *ptime)：获取当前时间

BOOL lossInLossRatio(float lossRatio)：根据丢失率随机生成一个数字，判断是否丢失，丢失则返回 TRUE，否则返回 FALSE

void printTips()：打印提示信息

bool seqIsAvailable()：当前序列号 curSeq 是否可用

void timeoutHandler()：超时重传处理函数

void ackHandler(char c)：收到 ack，累积确认，取数据帧的第一个字节

运行结果：

```
PS E:\Program\Network\Lab2\Code> g++ .\client.cpp -o .\client.exe -l Ws2_32 ; .\client.exe
                                                                                        E
The Winsock 2.2 dll was found okay
****************************************
| -time to get current time |
| -quit to exit client |
| -testgbn [X] [Y] to test the gbn |
| -testgbn_Send [X] [Y] to test the gbn |
****************************************
recv from client: -time

-time
2021/10/19 19:3:19
```

单向传输：

客户端：

```
****************************************
| -time to get current time |
| -quit to exit client |
| -testgbn [X] [Y] to test the gbn |
| -testgbn_Send [X] [Y] to test the gbn |
****************************************
-testgbn
Begin to test GBN protocol, please don't abort the process
The loss ratio of packet is 0.10,the loss ratio of ack is 0.10
Ready for file transmission
recv a packet with a seq of 1
```

服务器端：

```
send a packet with a seq of 12
send a packet with a seq of 13
Recv a ack of 11
                curAck > index , totalAck += 0
send a packet with a seq of 14
send a packet with a seq of 15
Recv a ack of 11
                curAck > index , totalAck += 0
send a packet with a seq of 16
Recv a ack of 11
                curAck > index , totalAck += 0
Timer out error.
send a packet with a seq of 12
send a packet with a seq of 13
Recv a ack of 13
                curAck <= index , totalAck += 2
send a packet with a seq of 14
Recv a ack of 14
                curAck <= index , totalAck += 1
send a packet with a seq of 15
Recv a ack of 15
                curAck <= index , totalAck += 1
send a packet with a seq of 16
Timer out error.
send a packet with a seq of 16
Recv a ack of 16
                curAck <= index , totalAck += 1
Finished
```

双向传输：

客户端：

```
****************************************
| -time to get current time |
| -quit to exit client |
| -testgbn [X] [Y] to test the gbn |
| -testgbn_Send [X] [Y] to test the gbn |
****************************************
-testgbn_Send
Begain to test GBN protocol,please don't abort the process
Shake hands stage
Begin a file transfer
File size is 115712B, each packet is 1024B and packet total num is 17
send a packet with a seq of 0
Recv a ack of 0
                curAck <= index , totalAck += 1
send a packet with a seq of 1
Recv a ack of 1
                curAck <= index , totalAck += 1
send a packet with a seq of 2
Recv a ack of 1
send a packet with a seq of 3
```

服务器端：

```
recv from client: -testgbn_Send
Begin to test GBN protocol, please don't abort the process
The loss ratio of packet is 0.20,the loss ratio of ack is 0.20
Ready for file transmission
The packet with a seq of 1 loss
recv a packet with a seq of 1
```

问题讨论：

SR协议和GBN协议的区别主要在哪里？

GBN特点：

在GBN协议中，发送方可以在窗口大小N的限制内发送足够多的分组，接收方接收到分组后就发送ACK给发送方，当发送方接收到连续的ACK时，该窗口就向前滑动，传输新的分组。在接收方分组丢失时，就从丢失的分组起重新传输丢失的分组之后的所有的分组，这样无需接收方准备缓存空间来储存分组。

SR特点：

SR协议相比GBN协议而言，其在接收方增加了接收窗口，对于接收窗口内乱序到达的分组进行缓存，当有一定数量的分组确认后将接收窗口向前滑动；在发送方，增加针对于每个数据包的计时器，不采取累计确认机制，对于每个数据包超时单独进行重传。

心得体会：

通过实现GBN协议与SR协议的通信，对于可靠数据通信的认识有了提高。在实践过程中，对于协议的理解更加深刻。