111062108 許凱棋

編譯結果

```
canlab@ubuntu:~/template (1)$ ./server 7777
═══════ Server ═══════
Server IP is 127.0.0.1
Listening on port 7777
═══════════════════════

Server is waiting...
```

```
canlab@ubuntu:~/template (1)$ make
gcc client.c -o client
gcc server.c -o server
canlab@ubuntu:~/template (1)$ ./client
═══ Enter Server Info ═══
Server IP: 127.0.0.1
Server port: 7777
═══════════════════════

Please enter a command:
download video.mp4
```

```
Oops! Packet loss!
Received SEQ = 255
Received SEQ = 256
Oops! Packet loss!
Received SEQ = 257
Received SEQ = 258
Oops! Packet loss!
Oops! Packet loss!
Received SEQ = 259
Received SEQ = 260
Oops! Packet loss!
Oops! Packet loss!
Oops! Packet loss!
Received SEQ = 261
Received SEQ = 262
Received SEQ = 263
Oops! Packet loss!
Received SEQ = 264
Oops! Packet loss!
Received SEQ = 265
Received SEQ = 266
Oops! Packet loss!
Received SEQ = 267
Received SEQ = 268
Received SEQ = 269
Elapsed: 11 sec
═══════════════════════

Saving download_video.mp4
File has been written
═══════════════════════

Please enter a command:
```

```
Send SEQ = 261
Timeout! Resend!
Send SEQ = 261
Timeout! Resend!
Send SEQ = 261
Received ACK = 261
Send SEQ = 262
Received ACK = 262
Send SEQ = 263
Received ACK = 263
Send SEQ = 264
Timeout! Resend!
Send SEQ = 264
Received ACK = 264
Send SEQ = 265
Timeout! Resend!
Send SEQ = 265
Received ACK = 265
Send SEQ = 266
Received ACK = 266
Send SEQ = 267
Timeout! Resend!
Send SEQ = 267
Received ACK = 267
Send SEQ = 268
Received ACK = 268
Send SEQ = 269
Received ACK = 269
═══════════════════════

Server is waiting...
```

```
canlab@ubuntu:~/template (1)$ cmp -s video.mp4 download_video.mp4 && echo "Same!" || echo "Different!"
Same!
```

程式碼解釋:

Server:

此段是傳送封包的程式，一次只會船一個封包，current 就當前傳送的封包最後一個 byte，當 current+1024<=filesize 時就是還沒抵達最最後一個封包了，反之最後一個封包要將 islast 設為 1。也就是結束時，再來是 fseek 適用於將指標指向 fd 的第 current 個 byte，fread 是從 fseek 指到的那個位置開始往後讀 1024 個 bytes，再利用 sendto 傳到 client 端。接下來是 poll 的運用，其意義為監視 timeout 期間內是否收到任何東西，若在 timeout 期間沒有收到任何 ack 就會重新傳，反之將 current+1024，seq+1。

```
void sendFile(FILE *fd) {
    Packet send, recv;
    memset(&send, 0, sizeof(send));
    memset(&recv, 0, sizeof(recv));
    size_t filesize = getFileSize(fd);
    size_t current = 0;
    int seq=0;
    while (current < filesize) {
        fseek(fd,current,SEEK_SET);
        fread(send.data, sizeof(char), 1024, fd);
        if(current+1024<=filesize){
            send.header.isLast=false;
            send.header.size=1024;
            send.header.seq=seq;
            sendto(sockfd, &send, sizeof(send), 0, (struct sockaddr *)&clientInfo, sizeof(struct sockaddr_in));
        }
        else{
            send.header.isLast=true;
            send.header.size=filesize-current;
            send.header.seq=seq;
            sendto(sockfd, &send, sizeof(send), 0, (struct sockaddr *)&clientInfo, sizeof(struct sockaddr_in));
        }
        printf("Send SEQ = %u\n", send.header.seq);

        struct pollfd pd;
        pd.fd=sockfd;
        pd.events=POLLIN;
        if (poll(&pd,1,TIMEOUT)==0){
            printf("Timeout! Resend!\n");
            continue;
        }
        recvfrom(sockfd, &recv, sizeof(recv), 0, (struct sockaddr *)&clientInfo, (socklen_t *)&addrlen);
        printf("Received ACK = %u\n", recv.header.ack);
        current+=1024;
        seq++;
    }
}
```

Client:

以下為接收封包的程式，用於收封包，定隨機丟失，若無丟失就會回傳 ack，每收到一個封包就會將 seq++, 並當收到最後一個封包內 islast 為 1 時，就停止收封包。

```
void recvFile(char *buffer) {
    Packet packet;
    unsigned int seq = 0;
    time_t start, end;
    start = time(NULL);
    while (true) {
    memset(&packet, 0, sizeof(packet));
        recvfrom(sockfd, &packet, sizeof(packet), 0, (struct sockaddr *)&serverInfo, (socklen_t *)&addrlen);
        if (isLoss(LOSS_RATE)) {
            printf("Oops! Packet loss!\n");
            continue;
        }

        printf("Received SEQ = %u\n", packet.header.seq);
        sendAck(packet.header.seq);
        memcpy(buffer+(seq)*1024,packet.data,packet.header.size);
        seq++;
        if(packet.header.isLast) break;
    end = time(NULL);
    printf("Elapsed: %ld sec\n", end - start);
}
```

以下是撰寫檔案的部分，用 wb 寫出二進位檔案。最後要 fclose。

```
void writeFile(char *buffer, unsigned int filesize, char *filename) {
    char newFilename[strlen("download_") + 64];  // filename[64]
    memset(newFilename, '\0', sizeof(newFilename));
    snprintf(newFilename, sizeof(newFilename) - 1, "download_%s", basename(filename));
    printf("Saving %s\n", newFilename);
    FILE *filedescriptor=NULL;
    filedescriptor = fopen(newFilename,"wb");
    if(filedescriptor==NULL){
        perror("Error opening the file");
        return ;
    }
    fwrite(buffer, sizeof(char), filesize, filedescriptor);
    fclose(filedescriptor);
    printf("File has been written\n");
}
```

學到的東西:

1. Poll 用法。
2. Stop and wait 的運作方式。
3. 撰寫二進制檔案。
4. Fread, fseek 用法。