

# Report

- Protocol Specification
- User & Operator Guide
- System & Program Design
- Problem and Solution

# Protocol Specification

- Protocol Specification
- User & Operator Guide
- System & Program Design
- Problem & Solution

# Protocol Specification

(1)	(2)	(3)	(4)	(5)	(6)
(7)					
	(8)				
			(9)		
...					
...					

# Protocol Specification

1. Type
2. Error
3. Self Index
4. Sender Index
5. Receiver Index
6. Data Length
7. ID
8. Password
9. Data

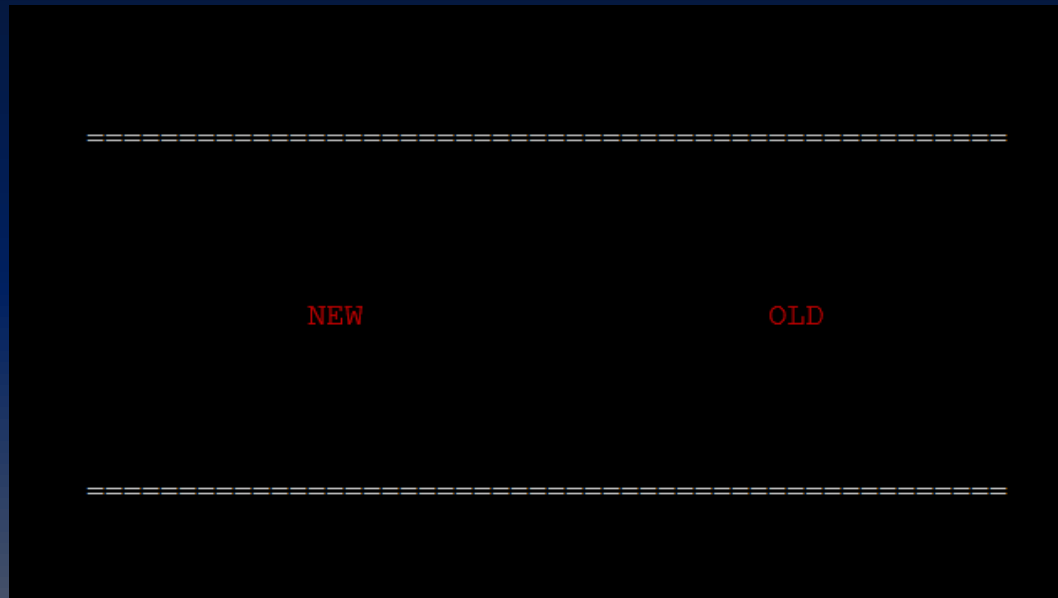
# User & Operator Guide

- Protocol Specification
- User & Operator Guide
- System & Program Design
- Problem & Solution

# Press Left or Right



# Registration



Press Left to choose new

# Registration

```
=====
ID: JJF
=====
```

```
=====
Password: 1234
```

```
Login
list num_user = 11
0: ID = Jeremy, status = 0=====
1: ID = Paul, status = 0
2: ID = Kang, status = 0
3: ID = CCF, status = 0
4: ID = User1, status = 0
5: ID = User2, status = 0
6: ID = User3, status = 0
7: ID = User4, status = 0
8: ID = User5, status = 0
9: ID = User6, status = 0
10: ID = JJF, status = 1
state list, select someone
```

Enter ID and Password, if will auto login if succes



# Login

```
=====
ID: Jeremy
=====
```

&

```
=====
Password:
=====
```

# Messaging

```
=====
                                Password: 1234
Receive 2 type 1448 Byte packet
Login
list num_user = 11=====
0: ID = Jeremy, status = 1
1: ID = Paul, status = 0
2: ID = Kang, status = 0
3: ID = CCF, status = 0
4: ID = User1, status = 0
5: ID = User2, status = 0
6: ID = User3, status = 0
7: ID = User4, status = 0
8: ID = User5, status = 0
9: ID = User6, status = 0
10: ID = P1JJF, status = 0
state list, select someone
1
```

After Login, There will appear a user list and you can choose a user to send message  
Choose a user and press Enter.

# Messaging

```
=====
To: Paul
Text Mode
Hi, How are you? Send me back if you see this message :)
=====
```

Press Enter to send Message.

If you want to Send multiply lines message you can Press F2 to change into Long Text Mod

# Messaging

```
=====
To: Paul
Long Text Mode
Yo,
What's going on?
█
```

Press F1 to change back to Text Mode and press Enter to send message

# Messaging(Online)

```
=====
To: Jeremy
Text Mode
Jeremy->      Paul
Receive message:
Hi, How are you? Send me back if you see this message :)
█
=====
```

The User you sent message to will receive the message if he or she is online.

# Messaging(Offline)

```
=====
To: Paul
Text Mode
Jeremy->      Kang
Receive message:
Hi, are you there?
█
```

If the user is currently offline, he or she will receive the message when he/she login.

# Messaging(Historical)

```
Log:                Paul->                Jeremy
Hey

Log:                Paul->                Jeremy
How are you
```

The server side will keep a log on sent and received messages for users to query.

```
write messageQ to file..
size = 2
Backup: 1->0                Hey
Backup: 1->0                How are you
shut down in 1 sec...
```

Messages will persist after server restarts, they are stored in a file

```
size = 2
Restore: Paul->Jeremy                Hey
Restore: Paul->Jeremy                How are you
```

# File Transfer

```
=====
To: Paul
File Mode
FILE0█
```

Type in File name and press Enter to transfer



# File Transfer

```
offset = 18466
  Jeremy->      Paul
receive data part:18466
filename = ../download/FILE0
offset = 19785
  Jeremy->      Paul
receive data part:19785
filename = ../download/FILE0
offset = 21104
  Jeremy->      Paul
receive data part:21104
filename = ../download/FILE0
offset = 22423
  Jeremy->      Paul
receive data part:22423
filename = ../download/FILE0
offset = 23742
  Jeremy->      Paul
receive data part:23742
filename = ../download/FILE0
offset = 25061
  Jeremy->      Paul
receive data part:25061
filename = ../download/FILE0
offset = 26380
  Jeremy->      Paul
receive data part:26380
receive file complete: ../download/FILE0
total file size: 26521
```

Receiver side will show these messages

# File Transfer(Multiply Files)

```
=====
To: Paul
File Mode
FILE0 FILE1 FILE2
█
```

You can keep type in the file names and they will all transfer at the same time

# File Transfer(Multiply Files)

```
filename = ../download/FILE0
offset = 25061
Kang-> Paul
receive data part:25061
filename = ../download/FILE1
offset = 25061
Kang-> Paul
receive data part:25061
filename = ../download/FILE2
offset = 25061
Kang-> Paul
receive data part:25061
filename = ../download/FILE0
offset = 26380
Kang-> Paul
receive data part:26380
receive file complete: ../download/FILE0
total file size: 26521
filename = ../download/FILE1
offset = 26380
Kang-> Paul
receive data part:26380
receive file complete: ../download/FILE1
total file size: 26521
filename = ../download/FILE2
offset = 26380
Kang-> Paul
receive data part:26380
receive file complete: ../download/FILE2
total file size: 26521
```

Receiver side will show these messages

# Bonus : Auto Reconnect

```
=====
To: Paul
Text Mode
server is shut down!
Do you want to reconnect? (Y/N)
☐
```

```
=====
NEW                                OLD
=====
```

If Sever is close, client will ask reconnecting or not

# Bonus : Encryption

```
unsigned char AES_key[] = {"DoReMiFaDoReMiFa"};
```

## Using AES-128 Encryption

```
void write_encrypt(){
    memcpy(enc_buffer, &sendpack, sizeof(sendpack));
    aes.Cipher((void *)&enc_buffer, enc_size);
    memcpy(&sendpack, enc_buffer, sizeof(sendpack));
    write(sockfd, &sendpack, sizeof(sendpack));
}

int read_decrypt(){
    int n = read(sockfd, &recvpack, sizeof(struct packet));
    memcpy(enc_buffer, &recvpack, sizeof(recvpack));
    aes.InvCipher((void *)&enc_buffer, enc_size);
    memcpy(&recvpack, enc_buffer, sizeof(recvpack));
    return n;
}
```

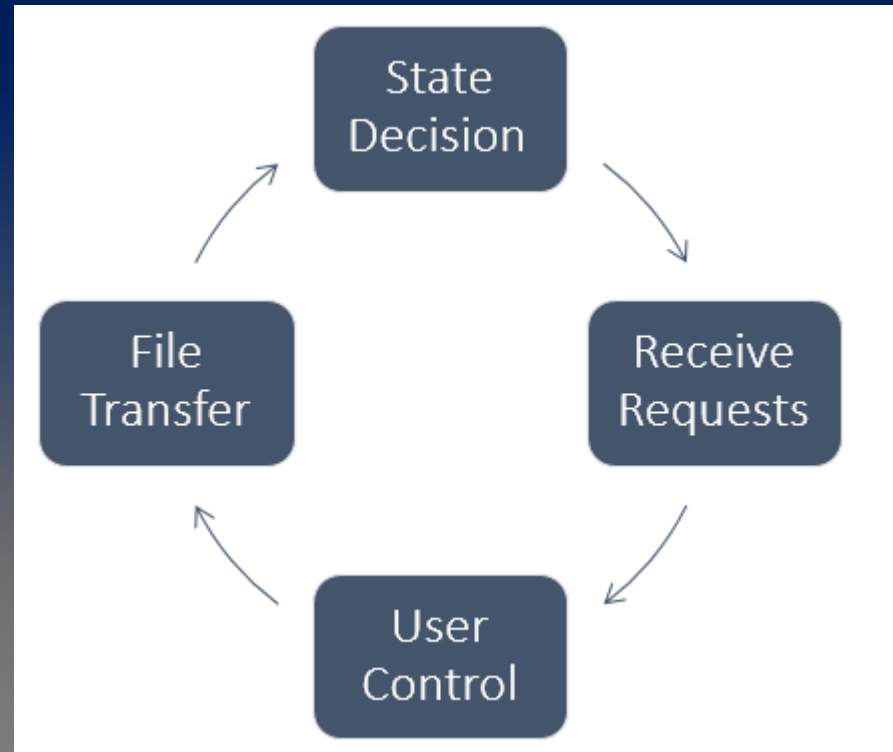
When sending or receiving pack, encryption or decryption it

# System & Program Design

- Protocol Specification
- User & Operator Guide
- System & Program Design
- Problem & Solution

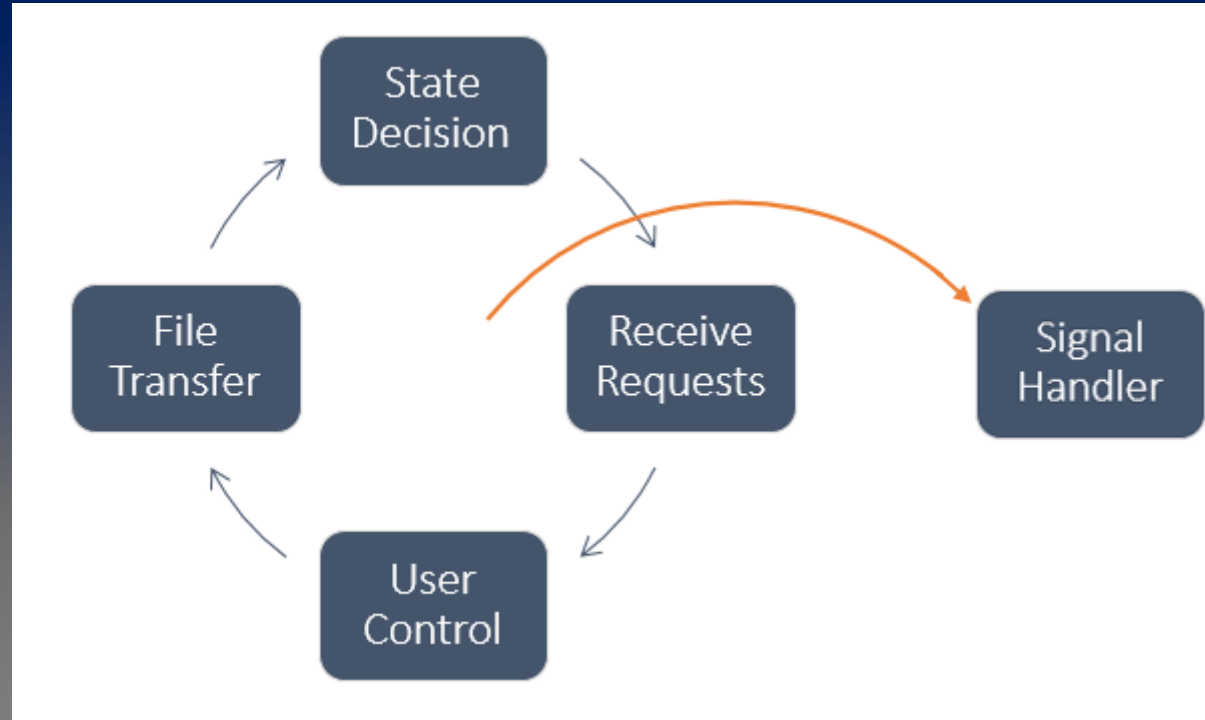
# FSM Client

- while (1) {  
    State Decision;  
    Receive Requests;  
    User Control;  
    File Transfer;  
• }



# FSM Client (signal handler)

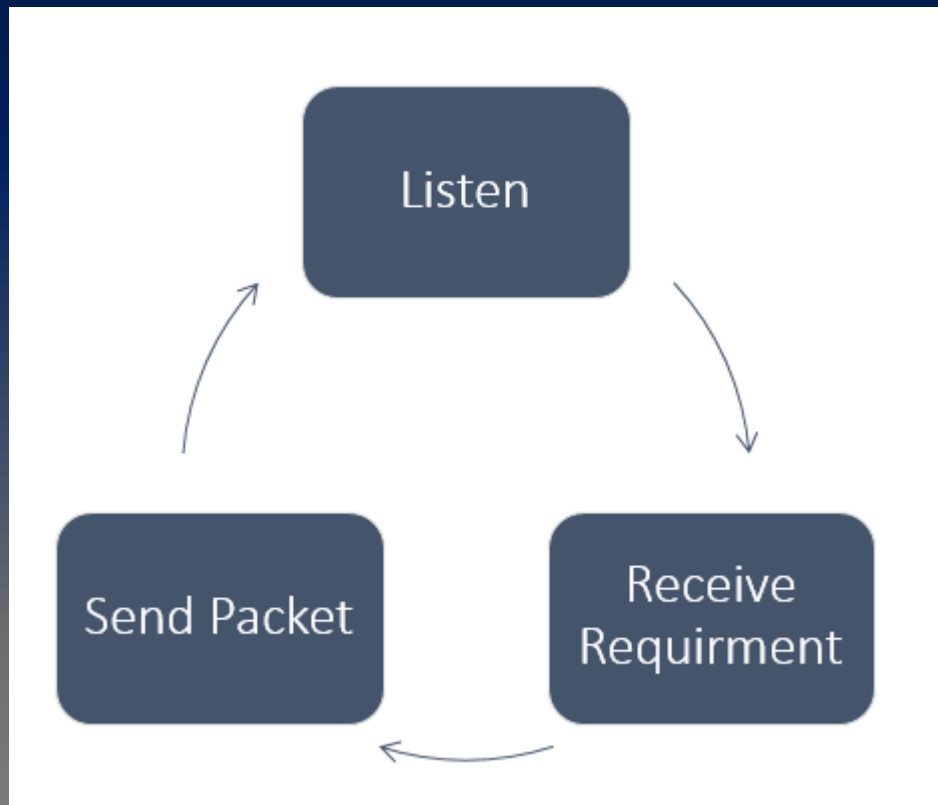
- void sigint\_shut\_down
- void sigpipe\_shut\_down



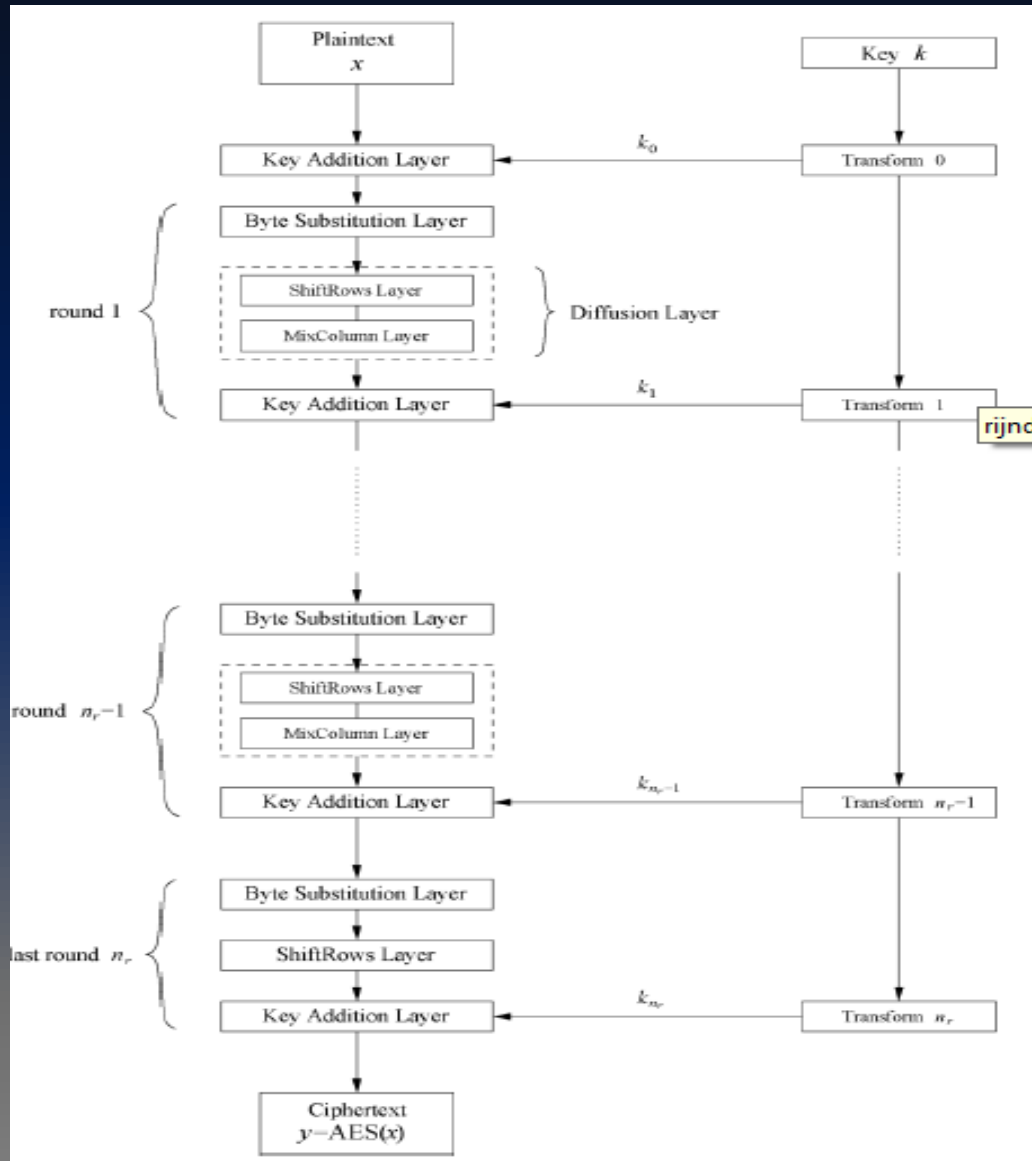


# FSM Server

- Receive a packet
- According to packet type do different things



# AES Encryption



# Problem & Solution

- Protocol Specification
- User & Operator Guide
- System & Program Design
- Problem & Solution

# Problem and Solution

- Problem: 用Command Line做檔案傳輸需要輸入許多複雜指令，不夠人性化。
- Solution: 自行設計simple interface，使用直覺化，除了測試時比較方便，新使用者也容易上手。

```

To: Kang
Text Mode
Kang->      Jeremy
Receive message:
Hello Kang
Hi Jeremy

Kang->      Jeremy
Receive message:
Today is CN demo time
Oh yes!

Kang->      Jeremy
Receive message:
The interface is soo cool!
Thankyou~!
```

# Problem and Solution

- Problem: 封包傳送過程會遺失或不齊全, 傳送出去跟接收端收到的大小不同, 因為我們封包大小太大(2048 Byte), 每次傳出去後半段的封包都會遺失。
- Solution: 將封包大小壓到1488Byte以下, 封包傳送就不會遺失了。

# Problem and Solution

- Problem: 封包中的data陣列如果被資料全填滿，將會因為沒有終止字元使得讀取發生錯誤。
- Solution: 在char陣列最後面留幾個byte來存放終止字元，問題即解決。