RELIABLE UDP PROGRAM

«EXECUTION ENVIRONMENT»

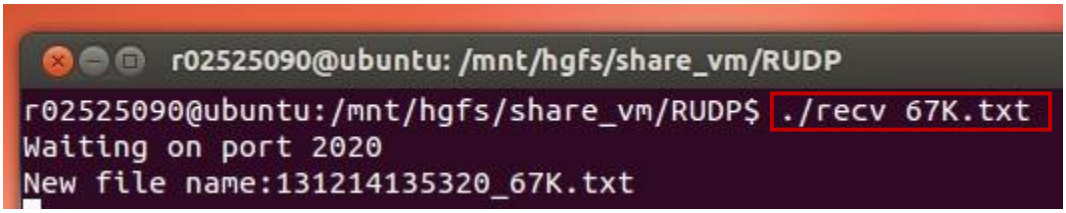
Operating system: Linux(Ubuntu-12.04.3-desktop-i386)
Programming language: C
File size: 67KB
Packet size: 512bytes
Timeout threshold: 3 seconds

«FUNCTIONS»

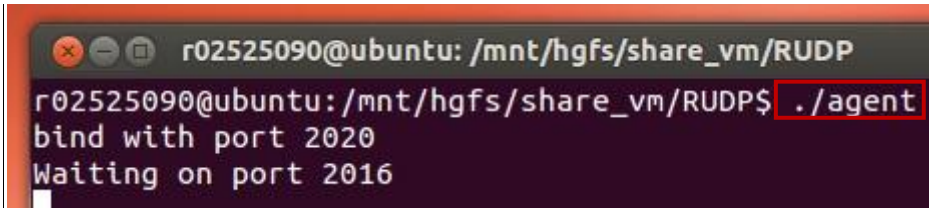
■INITIALIZATION

【Step 1】Input a new file name and start the receiver.
【Step 2】Start the receiver
【Step 3】Input the file name which you want to transmit and start the sender.

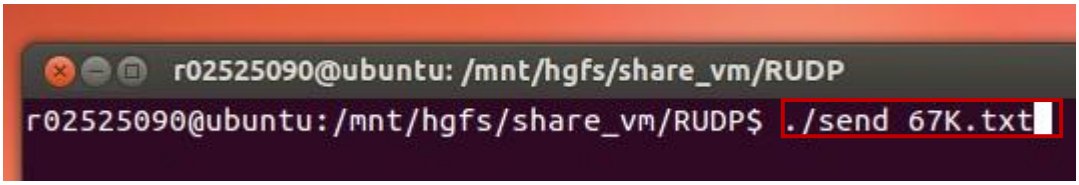
Receiver



Agent



Sender



■RETRANSMISSION

If receiving no corresponding acknowledgement during the time interval, sender will retransmit the packet until receives the acknowledgement.

Sender

```
Recv Ack #37
Send Data #38
Time out! Data #38
Resend Data #38
Recv Ack #38
```

■RANDOMLY PACKET DROP

Agent randomly drop packet and show the current loss rate.

Sender

```
Recv Ack #37
Send Data #38
Time out! Data #38
Resend Data #38
Recv Ack #38
Send Data #39
Time out! Data #39
Resend Data #39
Recv Ack #39
Send Data #40
Recv Ack #40
Send Data #41
Recv Ack #41
Send Data #42
Recv Ack #42
Send Data #43
Time out! Data #43
Resend Data #43
```

Agent

```
Get Data #38
Fwd Data #38
Drop Data #38.  loss rate: 0.210526
Get Data #38
Fwd Data #38
Get ACK #38
Fwd ACK #38
Get Data #39
Fwd Data #39
Drop Data #39.  loss rate: 0.230769
Get Data #39
Fwd Data #39
Get ACK #39
Fwd ACK #39
```

Receiver

```
Recv Data #37
Send Ack #37
Recv Data #38
Send Ack #38
Recv Data #39
Send Ack #39
Recv Data #40
Send Ack #40
Recv Data #41
Send Ack #41
Recv Data #42
Send Ack #42
Recv Data #43
Send Ack #43
```

■FIN ACKNOWLEDGEMENT

After receiving the final acknowledgement from the receiver, the sender will be terminated.

Sender

```
Recv Ack #134
Send Data #135
Time out! Data #135
Resend Data #135
Recv Ack #135
Last packet!Send Data #136
Recv FIN ACK #136
r02525090@ubuntu:/mnt/hgfs/share_vm/RUDP$
```

Agent

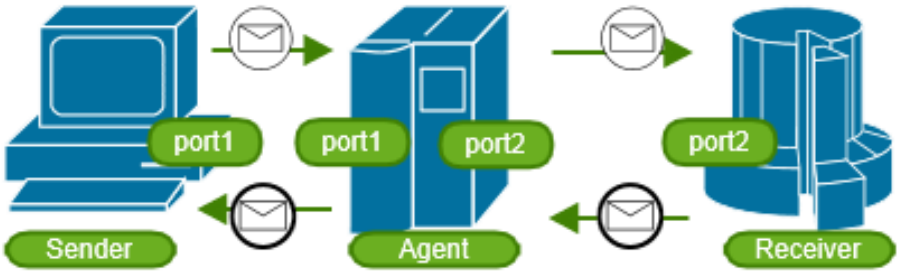
```
Get Data #135
Fwd Data #135
Drop Data #135.  loss rate: 0.237037
Get Data #135
Fwd Data #135
Get ACK #135
Fwd ACK #135
Get Data #136
Fwd Data #136
Get ACK #136
Fwd ACK #136
```

Receiver

```
Recv Data #133
Send Ack #133
Recv Data #134
Send Ack #134
Recv Data #135
Send Ack #135
Recv Data #136
Send FIN ACK #136
```

«ABOUT DESIGN»

■ ARCHITECTURE



■ DIFFICULTY AND SOLUTION

- Difficulty: Efficient time out implementation
  - ✓ Solution: “select” architecture defined in <sys/select.h>

```
int check_timeout(int sock){
    int result;
    fd_set fds, read_fds;
    struct timeval timeout = {TIME_THRESHOLD, 0};

    FD_ZERO(&read_fds);
    FD_SET(sock, &read_fds);
    if(select(sock+1, &read_fds, NULL, NULL, &timeout) <0){
        perror("[ERROR]timeout error ¥n");
        exit(1);
    }

    //waiting for result
    if(FD_ISSET(sock, &read_fds)){
        result = 0;
    }else{
        result = 1;  //time out!
    }
```