COMP6461 Assignment 2

Current Version: 0.1
Date: Mar. 06, 2012

Author:
Yuan Tao (ID: 5977363)
Xiaodong Zhang (ID: 6263879)

Course Instructor: Amin Ranj Bar
Lab Instructor: Steve Morse
Lab number: Friday

Concordia University

Table of Contents

1. Protocol between clients and server ---------- 2
1.1. Data structure ---------- 2
1.1.1. UDP frame ---------- 2
1.1.2. Data of Three-way Handshake ---------- 2
1.2. Problems under high packets loss ---------- 2
2. Performance evaluation ---------- 4
2.1. Testing environment ---------- 4
2.2. 40554 bytes with 5% drop rate ---------- 4
2.3. 40554 bytes with 10% drop rate ---------- 4
2.4. 40554 bytes with 15% drop rate ---------- 5
2.5. 15272 bytes with 20% drop rate ---------- 5
2.6. 15272 bytes with 25% drop rate ---------- 6
2.7. 15272 bytes with 30% drop rate ---------- 6
2.8. 15272 bytes with 35% drop rate ---------- 6
2.9. 2009 bytes with 40% drop rate ---------- 7
2.10. 2009 bytes with 45% drop rate ---------- 7
2.11. 2009 bytes with 50% drop rate ---------- 8
3. Program files list ---------- 9
Appendix A: Revision History ---------- 11

Table of Figures

Figure 2-1 Ratio for 40554 bytes with 5% drop rate 4
Figure 2-2 Ratio for 40554 bytes with 10% drop rate 5
Figure 2-3 Ratio for 40554 bytes with 15% drop rate 5
Figure 2-4 Ratio for 15272 bytes with 20% drop rate 6
Figure 2-5 Ratio for 15272 bytes with 25% drop rate 6
Figure 2-6 Ratio for 15272 bytes with 30% drop rate 6
Figure 2-7 Ratio for 15272 bytes with 35% drop rate 7
Figure 2-8 Ratio for 2009 bytes with 40% drop rate 7
Figure 2-9 Ratio for 2009 bytes with 45% drop rate 7
Figure 2-10 Ratio for 2009 bytes with 50% drop rate 8
Figure 3-1 Project File List 10

1. Protocol between clients and server

It is designed based on the previous protocol of assignment 1. **So please read the design document of assignment 1 first.**
The general idea is that the TCP data stream is divided into small pieces, and each piece is added a UDP header. These two parts together consist of the UDP frame. In this way, we can maximum reuse the source code of assignment 1.

1.1. Data structure

1.1.1. UDP frame

```
// UDP packet
typedef struct {
    unsigned int seq:1;            // sequence number
    char data[BUFFER_LENGTH];   // stores TCP packets, in order to reuse code of assignment1
} UDPPACKET, *PUDPPACKET;
```

The original TCP packets are stored in data variable of this structure.

1.1.2. Data of Three-way Handshake

```
// three-way hand shake
typedef struct {
    int clientSeq;
    int serverSeq;
} HANDSHAKE, *PHANDSHAKE;
```

It defines the data exchanged during the handshake procedure.

1.2. Problems under high packets loss

During the development of this assignment, most of the time is spent dealing with following problems that may occur with very high packets loss:

- The three-way handshake procedure may fail
- The ACK for the last packet may fail

2. Performance evaluation

2.1. Testing environment

The client, server and router run on the same machine.
If the sender fails to get the ACK, it will send again at a maximum of 25 times. Each time the sender waits 300ms * 3 for the ACK.

Because the "low performance" of the router, here the largest file we choose for testing is just 40544 bytes. And we decrease the size of file when increase the drop rate.

All of the original log files please see $PROGRAM/logs/logs.zip

2.2. 40554 bytes with 5% drop rate

Sender: number of effective bytes sent: 42672 (508 * 84)
Sender: number of packets sent: 554
Sender: number of bytes sent: 46536 (554 * 84)

Here is the ratio (blue color) of the number of effective packets to the total number of packets. And the ratio (red color) of the number of resent packets to the total number of packets.

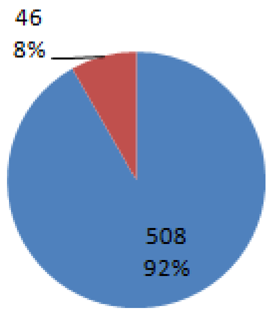


Figure 2-1 Ratio for 40554 bytes with 5% drop rate

2.3. 40554 bytes with 10% drop rate

Sender: number of effective bytes sent: 42672 (508 * 84)
Sender: number of packets sent: 643

Sender: number of bytes sent: 54012 (643 * 84)

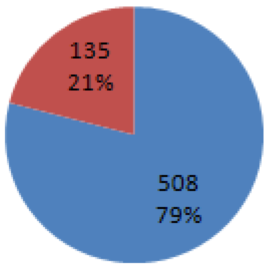


Figure 2-2 Ratio for 40554 bytes with 10% drop rate

2.4. 40554 bytes with 15% drop rate

Sender: number of effective bytes sent: 42672 (508 * 84)
Sender: number of packets sent: 739
Sender: number of bytes sent: 62076 (739 * 84)

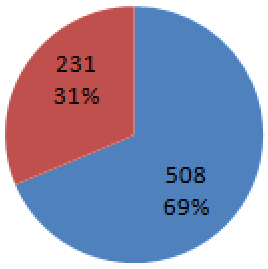


Figure 2-3 Ratio for 40554 bytes with 15% drop rate

2.5. 15272 bytes with 20% drop rate

Sender: number of effective bytes sent: 16128 (192 * 84)
Sender: number of packets sent: 284
Sender: number of bytes sent: 23856 (284 * 84)

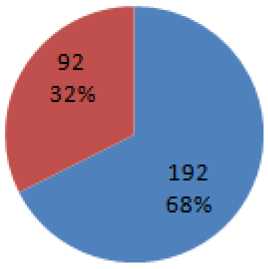


Figure 2-4 Ratio for 15272 bytes with 20% drop rate

2.6. 15272 bytes with 25% drop rate

Sender: number of effective bytes sent: 16128 (192 * 84)

Sender: number of packets sent: 335

Sender: number of bytes sent: 28140 (335 * 84)

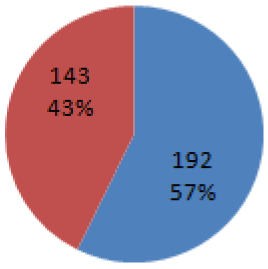


Figure 2-5 Ratio for 15272 bytes with 25% drop rate

2.7. 15272 bytes with 30% drop rate

Sender: number of effective bytes sent: 16128 (192 * 84)

Sender: number of packets sent: 374

Sender: number of bytes sent: 31416 (374 * 84)

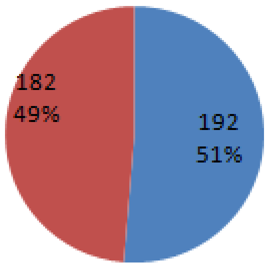


Figure 2-6 Ratio for 15272 bytes with 30% drop rate

2.8. 15272 bytes with 35% drop rate

Sender: number of effective bytes sent: 16128 (192 * 84)

Sender: number of packets sent: 423

Sender: number of bytes sent: 35532 (423 * 84)

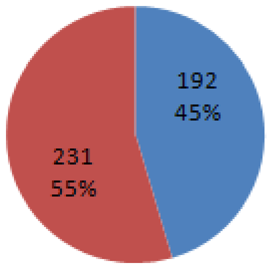


Figure 2-7 Ratio for 15272 bytes with 35% drop rate

2.9. 2009 bytes with 40% drop rate

Sender: number of effective bytes sent: 2268 (27 * 84)
Sender: number of packets sent: 87
Sender: number of bytes sent: 7308 (87 * 84)

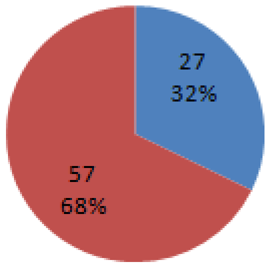


Figure 2-8 Ratio for 2009 bytes with 40% drop rate

2.10. 2009 bytes with 45% drop rate

Sender: number of effective bytes sent: 2268 (27 * 84)
Sender: number of packets sent: 99
Sender: number of bytes sent: 8316 (99 * 84)

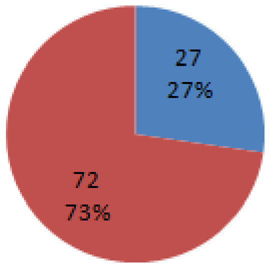


Figure 2-9 Ratio for 2009 bytes with 45% drop rate

2.11. 2009 bytes with 50% drop rate

Sender: number of effective bytes sent: 2268 (27 * 84)

Sender: number of packets sent: 128

Sender: number of bytes sent: 10752 (128 * 84)

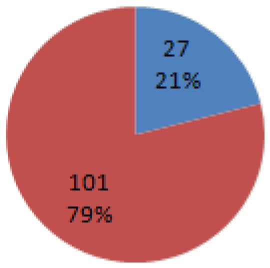
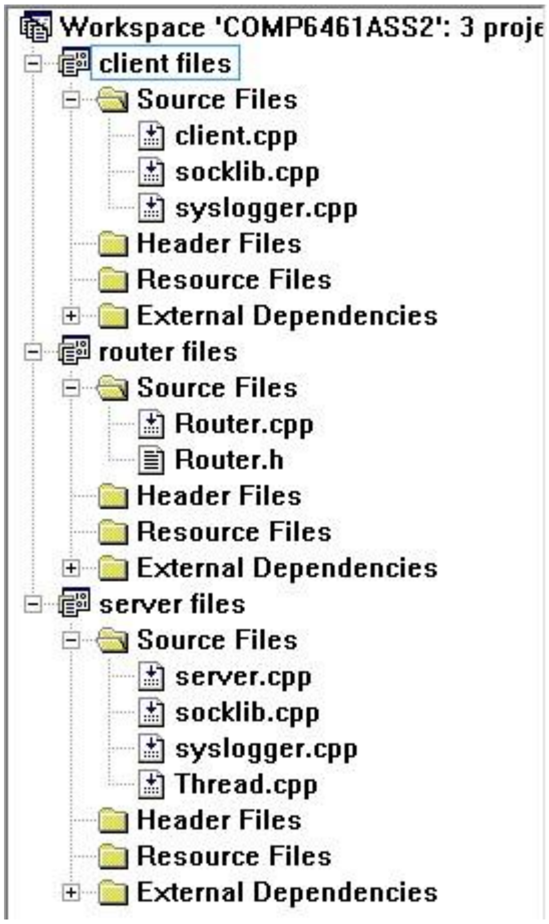


Figure 2-10 Ratio for 2009 bytes with 50% drop rate

3. Program files list

- bin
  output directories
- client
  source code of client side
- client_files_root
  test files for the client side
- common
  source code for both client and server sides
- router
  source code for router
- server
  source code of server side
- server_files_root
  test files for the server side
- logs
  log files for both client and server
- Readme.pdf
  This file.



```
Workspace 'COMP6461ASS2': 3 proje
  client files
    Source Files
      client.cpp
      socklib.cpp
      syslogger.cpp
    Header Files
    Resource Files
    External Dependencies
  router files
    Source Files
      Router.cpp
      Router.h
    Header Files
    Resource Files
    External Dependencies
  server files
    Source Files
      server.cpp
      socklib.cpp
      syslogger.cpp
      Thread.cpp
    Header Files
    Resource Files
    External Dependencies
```

Figure 3-1 Project File List

Appendix A: Revision History

Version	Date	Author	Remark
V0.1	Mar. 06, 2012	Yuan Tao, Xiaodong Zhang	Draft