Project 2: Reliable Blast User Datagram Protocol (RBUDP)

Contents

1. Introduction

There are many protocols that can be used to transfer files over a network. In this tutorial you will implement RBUDP, a protocol for data transfer. You will measure its performance in comparison with TCP, which in turn will help you understand the differences and trade-offs between these protocols and how to analyse them.

2. General Information

Please note the following points when implementing your solution:

1. You may code in any programming language, however we recommend Java. The assistants might not know your chosen language and would be unable to help you if you run into trouble.
2. You are to work in allocated groups for this project.
3. This project is to be accompanied by a report.
4. The use of virtual networking tools such as Hamachi for Linux or ZeroTier for Mac must be used when communicating between the clients and the server.
5. All project source code and the report will be checked for plagiarism, including being checked against solutions from previous years. If you are found guilty there will be serious consequences, including receiving a mark of zero for the project.

3. Project Specifications

There are three main aspects of this project. Make sure that you follow all the instructions.

Sender

The following must be implemented in the sender application:

1. Must have a simple GUI.
2. Must be able to specify a file for transfer: text is ok, GUI is better.
3. Files must be transferred to the receiver across the Hamachi/ZeroTier local network using RBUDP and TCP.
4. RBUDP must make use of datagram packets for data transfer and may use a TCP connection for signalling only.
5. RBUDP datagram packets must contain unique sequence numbers. Note that you are allowed to use some form of wrap-around, as long as the time until wrap-around is large enough.

Receiver

The following must be implemented in the receiver application:

1. Must have a simple GUI.
2. Must be able to receive files from sender.
3. Must show the progress of incoming files. There are more marks to be earned if the progress indicator is shown in the GUI rather than simply displaying text.
4. Must be able to handle dropped packets and out of order RBUDP datagram packets appropriately.

Experiments

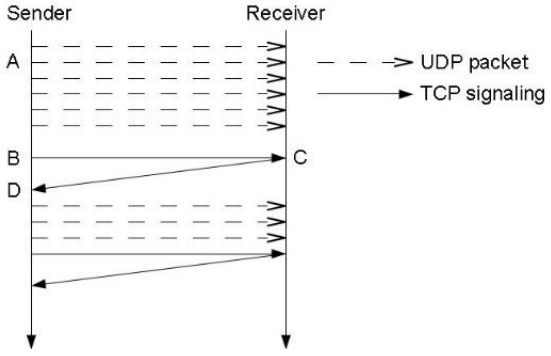
Experiments must include the following:

1. Experiments that compare the throughput of TCP and RBUDP data transfer.
2. Experiments with the transfer rate of RBUDP.
3. Experiments with the packet size used in RBUDP.
4. Experiments using varying packet loss rates. If you are comparing packet loss between RBUDP and TCP, you must artificially cause congestion in the network, which will give you more marks. Alternatively, if you are not comparing the two protocols but rather just independently running experiments for each, then you may use random drop.
5. Any additional experiments you wish to include.

4. RBUDP Protocol

This section gives a brief overview of the RBUDP protocol.

1. The file to be transferred is read into datagram packets.
2. The packets contain unique sequence numbers.
3. (A) These packets are then sent over the network to the receiver.
4. (B) After a set number of packets have been sent, a list of sequence numbers of the sent packets are sent over TCP to the receiver.



5. (C) At the receiver, the list of sent packets is compared to a list of receiver packets.
6. (D) A list of missing packet sequence numbers is sent back over the TCP connection to the sender for retransmission.
7. This process is repeated until no more data needs to be transferred, where steps 3-6 are repeated within a single blast until all packets are received before moving on to the next blast.

For optimal results, try to keep the network pipe as full as possible during transfer, and minimize the amount of signalling that takes place.

5. Repos and submission

Each group shall be given a GitLab repo to work in. The contents of the repo must contain a ***README.md*** to explain how to compile and run your program, and a ***report.pdf*** to be read by your markers. For submission, compress your repo into a single tar or zip file and only one person from the group submits it to SUNLearn.

6. Helpful Hints

1. Start early, do not leave this project to the last few days while thinking that you will finish on time.
2. Look at the marking scheme and at what needs to be done to help you estimate your progress.
3. Note that experiments are emphasised for this project.
4. Get the code working before you start optimizing, but keep in mind that optimal code is essential.

7. Useful Resources

Here follow some resources that may be helpful:

1. Java lesson: All about Datagrams
2. DatagramSocket class description
3. DatagramPacket class description
4. UDP RFC
5. TCP RFC

8. Guidelines

Report

The page limit for the report is strictly 8 pages. Fit the following sections into this page limit:

1. Title

   Your document must have a title which reflects the project you are implementing. Furthermore, you must indicate the names, student email addresses and roles of all group members.

2. Introduction / Overview

   Explain what the purpose of the project was and give an overview of the main sections in the document in a short paragraph, not a table of contents.

3. Features
   Since you will implement most features, indicate which of the required features you did not implement. Also indicate what additional features you implemented.

4. File Descriptions
   Provide concise descriptions of the source files present in your project. You may ignore files that have minimal functionality, for example a file containing only two member fields.

5. Program Description
   Provide a description of your program flow. We suggest that you tailor this according to the design of your implementation. Since most implementations will be event based / reactive, we recommend that you structure your program flow description as follows. Start with a birds-eye view of what components your implementation comprises of, how these components interact with one another, and what purpose of each is. Finally, explain the program flow that takes place for all features as specified in this document. Example of features include client disconnection handling, client interaction, and message routing.

6. Experiments
   This is the major focus of the report.
   The report should have the experiments specified above, where each experiment follows the scientific method which includes a question and/or hypothesis, experimentation, and results/conclusion. Make sure you provide the data that you have gathered, as well as identify the dependent and independent variables for each experiment.

7. Issues Encountered
   Give a brief description of any problems or issues that you encountered.

8. Design
   Mention any interesting design decisions that you made in your implementation.

Source Code Documentation

1. Class level
   You must use Javadoc format to provide a short summary of the program description from the report, i.e. a birds-eye overview of the main functionality for each class.
2. Method/Function level
   Again, use Javadoc format. Generally, methods/functions should be documented with the following:
   - A single line describing the main utility
   - A paragraph describing the algorithm
   - Inputs, outputs and return value
   - Caveats and limitations
3. In-line level
   Use comments sparingly inside method bodies; for intricate code only. Don't state the obvious. Also, provide comments that explain why you are doing something rather than what you are doing, since what you are doing should be obvious from the method name.

9. Marking Scheme

Sender	**10**
- GUI	2
- Selecting file to transfer	2
- TCP is implemented	1
- RBUDP is implemented	1
- For RBUDP, only use TCP for signalling	2
- Sequential number allocation	2
Receiver	**18**
- GUI	2
- TCP file correctly received	3
- RBUDP file correctly received	3
- Handle dropped packets	3
- Handle late packets	3
- Progress indicator	3
- Progress indicator (GUI)	1
Experiments (Code must be functional)	**17**
- Data transfer, TCP vs RBUDP	5
- Transfer rate	4
- Packet size	4
- Packet loss	4
Report	**14**
- Language, grammar and spelling	3
- Project description	3
- Include all sections specified in report guidelines	8
Documentation	**8**
- Class level	2
- Method level	2
- In-line level	2
- Readme present	2
General	**8**
- Presentation quality during demo	2
- Simulated artificial congestion	3
- Hamachi/ZeroTier	3
TOTAL	**75**