Project 2: Reliable Blast User Datagram Protocol (RBUDP)

Department of Mathematical Sciences
Computer Science Division
University of Stellenbosch
7600 Stellenbosch

July 2020

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 Project Specifications

You can code this program in any language you like, although we recommend that you code it in Java. Keep in mind that the assistants might not know your chosen language and therefore not be able to help you with all coding problems you might have. The practical has been successfully implemented in Java and the marking scheme has been set up with Java socket programming and multi-threading difficulties in mind.

The project should be handed in together with a report. All projects and reports will be checked for plagiarism. If plagiarism is detected there will be serious consequences. You are allowed to work in groups of two.

2.1 Sender

The following must be implemented in your sender application:

1. Must have a simple GUI.
2. Must be able to select a file for transfer.
3. Files must be transferred to the receiver across the local network using RBUDP.
4. Files must be transferred to the receiver across the local network using TCP.
5. RBUDP must make use of datagram packets for data transfer and may use a TCP connection for signalling only.
6. 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 till wrap-around is large enough.

2.2 Receiver

The following must be implemented in your receiver application:

1. Must have a simple GUI.
2. Must be able to receive files from the sender.
3. Must show progress of incoming files.
4. Must be able to handle dropped packets and out of order RBUDP datagram packets appropriately.

2.3 Report

Your report must adhere to the structure given in the report guideline document. In addition, remember to 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 not comparing RBUDP and TCP you may use random drop. Alternatively if you are comparing TCP and RBUDP, you must artificially cause congestion in the network.
5. Any additional experiments you wish to include

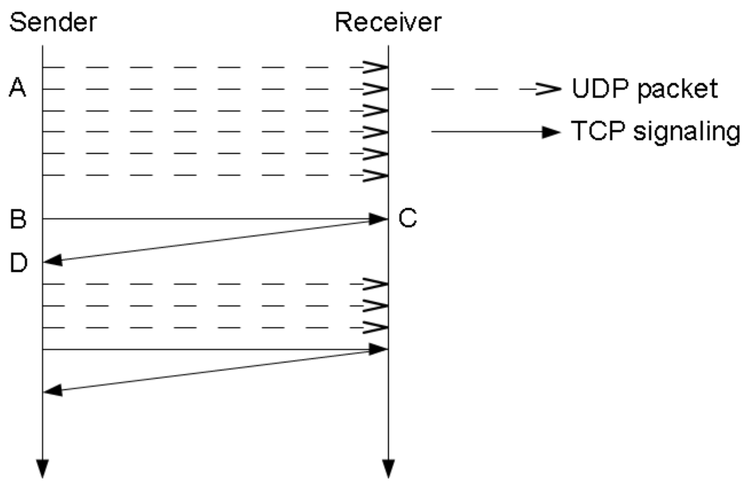

Figure 1: Transmission model

3 RBUDP Protocol

1. The file to be transferred is read into datagram packets.
2. The datagram packets contain unique sequence numbers.
3. These packets are then sent over the network to the receiver. (A)
4. 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. (B)

5. At the receiver the list of sent packets are compared to a list of received packets. (C)
6. a list of missing packet sequence numbers is sent back over the TCP connection to the sender for retransmission. (D)
7. This process is repeated until no more data needs to be transmitted.

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

4 References

Here follow some resources that may be helpful:

- Java lesson: All about Datagrams:
  `http://docs.oracle.com/javase/tutorial/networking/datagrams/`
- DatagramSocket class description:
  `http://download.java.net/jdk7/archive/b123/docs/api/java/net/DatagramSocket.html`
- DatagramPacket class description:
  `http://download.java.net/jdk8/docs/api/java/net/DatagramPacket.html`

5 Dates

- **Project starts:** 12 August 2020.
- **Project deadline:** 26 August 2020 (13:00).

The report, Makefile and source codes must be submitted as a single tar or zip file whose name has the following form: `GROUP_NUMBER.tar.gz` where `GROUP` is your group name and `NUMBER` is the project number. Your project must be submitted to the GitHub repository `Computer-Science/rw354/2020/Project 2`.

6 Helpful Hints

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

7 Marking Scheme

Sender	**10**
- *GUI*	3
- *Sequence number allocation*	4
- *Retransmission protocol*	3
Receiver	**22**
- *GUI*	3
- *Handles dropped packets*	3
- *Handles late packets*	3
- *RBUDP File uploads correctly*	3
- *TCP File uploads correctly*	3
- *Progress indicator*	3
- *Data capture*	4
Housekeeping and style	**2**
Data-structures and optimality	**3**
Report	**19 (+4)**
- *Language, spelling and grammar*	3
- *Description of the experiments*	6 (+2)
- *Conclusions drawn from the results*	6 (+2)
TOTAL	**52 (+4)**

1. Marks in brackets indicate additional marks that are available
2. Data capture refers to the method used to capture and interpret your throughput data. The more accurate, the higher the marks obtained. It is recommended that you use a program like `WireShark` to analyse network performance.