Project

This page describes the project for Computer Networks for fall 2017.

The goal of this project is to design a UDP-based file transfer program and then implement and measure its performance. You will try to answer the question, "Can a UDP-based file transfer protocol be faster than a TCP file transfer?". You may also explore possible issues with fairness to TCP flows.

Prizes will be offered for the first, second, and third fastest implementations. The logistics of this file transfer contest will be discussed in class.

Service to be provided:

The service to be provided is reliable and fast file transfer between two Internet hosts that uses UDP only.

Assumptions about the environment:

The environment of interest is one of both short and long RTT and across a range of connection speeds and packet loss rates. That is, the environment is any two hosts on the Internet.

Application Interface:

The sender of the file has an interface defined as:

```
retcode = sendFile(char *fileName, char *destIpAddr, int destPortNum, int options);
```

where `retcode` is an integer valued return code, -1 for failed and 0 for successful transfer, `filename` is the name of the file to be sent, `destIpAddr` and `destPortNum` are the IP address and port number of the destination host running the `recvFile` command below, and `options` is a to be defined input for providing (for example) a verbose output mode, debugging mode, and so on.

The receiver of the file has an interface defined as:

```
retcode = recvFile(char *fileName, int PortNum, int maxSize, int options);
```

where `retcode` is an integer valued return code, -1 for failed and 0 for successful transfer, `filename` is the name of the file to be created and written with the contents of the sent file, `portNum` is the port number on which this command is listening, `maxSize` is a defined maximum size for the received file, and `options` is a to be defined input for providing (for example) a verbose output mode, debugging mode, and so on.

Project requirements

Students may work in teams of two. There will be one deliverable per team. The team is responsible for designing, implementing, and measuring the performance of a UDP-based file transfer protocol that meet the above application interface. The target OS should be Windows (preferred), but Linux is acceptable. The

programming language used to implement the protocol is up to the students to choose - "C" is recommended and preferred. A key component of the project is a test on a test bed to be set-up by the TA. Students will be able to pre-test their project on this test bed.

Project deliverables

1. A document describing the protocol including list of messages, message format, and message rules and timing. Message rules and timing can best be described using a finite state machine.
2. A discussion on the design choices made including trade-offs.
3. Source code for the protocol and a test program that sends and receives a file specified via the command line using the new protocol.
4. Successful execution of the test program on a test bed set-up by the TA.

Project grading

Project grading is described below. The base grade of 100 is as follows:

- 40 points - a complete and correct protocol design. Points will be deducted if the design is not described to a level where it can be implemented by someone "skilled in the art".
- 10 points - a coherent and technically correct explanation of the design choices made. Points will be deducted if the students do not understand the design choices inherent in their design.
- 50 points - successful operation of the protocol including both successful file transfer and a performance (transfer time) that is reasonably close to that of a TCP file transfer. First priority is successful file transfer, second priority is performance. The test bed for this evaluation will be described in class.

Additional grade points can be earned as follows:

- 5 points extra - Protocol design for fairness to TCP flows.
- 10 points extra - A clear demonstration of fairness to TCP flows. The students and TA will design this demonstration together.

Notes on project grading and submission

1. You will demo your implementation to the TA at the end of the semester for a grade. The exact test procedure and test bed will be described in class. The test bed will very likely consist of three types of connections:
    - Point-to-point (PC to PC) on a 1 Gb/s Ethernet link
    - One PC on the campus wireless, the second PC connected to the Department 1 Gb/s Ethernet.
    - A high error rate (1% packet loss) connection - this may be emulated in software (how to do this will be described in class).
2. Text files of size 10 MB, 100 MB, and (possibly) 1 GB will be used for the evaluation.
3. The TA will use wireshark to verify that the implemented protocol matches the design.
4. The TA will compare your implementation against a TCP file transfer for transfer time.
5. You will be allowed three attempts for your demo in a **single** maximum 30 minute period. It is highly recommended that you pre-test on the TAs test bed well before your assigned time slot.
6. Early ship bonus is 110% multiplier for delivery on 11/27/17 or 11/28/17, 105% multiplier for 11/29/17 or 11/30/17, and 100% multiplier for 12/01/17. The last time of delivery on each date is 7pm. **No late**

submissions will be accepted - the TA doing the grading will "go home" at 7pm on my instructions. A sign-up sheet for grading slots will be made available in class.

Remarks

Some remarks for the project are:

1. You wrote a simple TCP-based file transfer implementation for assignment #3 (the assignment solutions includes the source code for a reference implementation). This reference implementation in the solution for assignment #3 will be used as the speed benchmark for TCP-based file transfer.
2. It is encouraged (even expected) that you will look in the literature and open source community for previous work in non-TCP file transfer. Your findings should be properly cited in your report and if you build on any previous work, it must be clear how you do so.
3. You may not just download UDT and call it a day. But I assume you will study UDT (see comment (2)). If you get UDT running, I would like to see it (and see how fast it is for file transfers).
4. You may choose to work in any language that you think is best suited for this project. The use of "C" is recommended.
5. You may work together within your team of two students. Submitted code (between teams) will be compared for copied code. Copying code is cheating - cheaters will earn (at minimum) a zero in the project and (at most) an FF in the class.
6. If and when in doubt, ask a question.
7. Have fun! As with all things, this project is what you make of it  .

Last update on September 24, 2017