Computer Networks

CS 4514

Term B04

THE BS/MS PROJECT

DUE DATES

Project Proposal: Friday, November 12, 2004 at 2 p.m.

Final Project: Friday, December 17, 2004 at 4 p.m.

Each project team must make a one hour appointment for a LIVE demo of their project. At that time, the project team needs to turn in: a hardcopy of the source program, the original graded design report, and a short addendum to the design report relating changes and improvements to the design.

1. Introduction

   This project is a team assignment to design and implement a three layer communication model which permits two or more application processes to communicate over a network. The expectation is that the program will be done in Cor C++ on CCC hosts which use TCP/IP to exchange messages. The project is intended to expose students to at least three aspects of computer networks - the client/server paradigm, issues in the design of the data link layer, and working with a real network protocol (TCP/IP).

   The following description outlines the project but does not specify all the details. This is done to give the student choices in the functionality of the model and to force the student to make some of the design decisions.

   Throughout the description it is important to separate the abstraction of a three level model, referred to henceforth as MININET, from the details of interfacing to TCP/IP.

   The MININET protocol consists of three layers - application, data link, and physical. The application layer consists of a protocol between a client/server pair of processes. Communication is in the form of request and response messages. The client sends requests to the server and receives responses to those requests. The data link layer provides frame-level communication between client and server hosts. The data link layer operates over an unreliable channel subject to lost and garbled data. The physical layer provides a full-duplex, communication channel on which the data link layer can send an arbitrary byte stream. The physical layer is emulated using TCP/IP.

2. MININET APPLICATIONS LAYER

   The application layer implements an interactive request/response protocol. The client reads commands from standard input, converts them into server requests, sends them to the server, and prints the response returned by the server. The server fields client requests and sends back appropriate responses.

   The project team MUST propose, design and implement their own application layer. Your proposal includes an explanation the functionality of both the client and the server and the specification of the protocol. You must specify valid commands that the client

accepts from input, the meaning of the command request to the server, and the server responses. Your protocol should have at least five distinct commands and at least one command must involve a file transfer over the network. Your commands should include at least one large transfer in both directions. The transfer should be of sufficient size to test your sliding window mechanism.

Messages, the data units between the application and data link layer on MININET, are limited in size to 300 bytes. This includes application layer protocol bytes. Your design must be able to handle multi-message application level peer communication such as file transfers.

If the server encounters an application-level error while processing a request, it returns an appropriate error message. The client in turn prints out appropriate error explanations to the user.

The application layer interfaces with the data link layer ONLY via two routines: dl_send and dl_recv. Both the client and server processes can issue dl_send and dl_recv.

Note: choices made on how to control the concurrency of the layers is the KEY design decision for this project. It is highly recommended that each project team discuss their approach to this problem PRIOR TO writing the design report.

3. MININET DATA LINK LAYER

   The data link layer accepts data passed via dl_send, places the data into a data link frame, and sends the frame to the physical layer for transmission. The data link layer communicates with its peer server process over a full duplex communications channel accessed through a UNIX file description (see physical layer).

   The design of your data link layer must include the ability to handle arbitrary data streams and an error-prone channel. Thus, the data link layer must include considerations for data stuffing, buffering, retransmissions, checksums, and some form of flow control.

   You must implement a sliding window protocol (see Tanenbaum – for details). You must use a send window size of four. Conceptually, the data link layer buffers the data internally and allows the application to continue processing. If the caller attempts to transmit messages beyond the window size, the data link layer will block until space becomes available.

   Retransmissions will require a timer mechanism to detect lack of acknowledgments.

   On the dl_recv side, note that a new data frame may arrive before the application layer actually calls dl_recv. Because frames must be processed when they arrive (otherwise you might ignore an acknowledgement), received data frames will have to queued. When dl_recv is called, it must check for the presence of data in the received queue. If none is present, it will have to suspend itself until an appropriate event occurs. As in Tanenbaum's discussion, the data link layer processes must respond to timeout, frame_arrival and frame_error events.

4. MININET PHYSICAL LAYER

   The actual communication with the server takes place using Unix sockets and TCP/IP. The physical layer of MININET deals with the details of establishing a TCP/IP connection and sending messages over the WPI campus network. Assume the maximum size of the frame payload is 30 bytes. That is, the data link layer hands off frames to the

physical layer for transmission. To simplify matters, you may make reasonable assumptions on the maximum amount of data stuffing needed. The design of the full frame is the students choice but must include framing bytes at each end.

The physical layer is responsible for ARTIFICALLY generating errors in the transmission. Your design must adjust the error rate as a command line argument.

5. Input

   Since the students are specifying the application layer, it is also the group's responsibility to provide interesting and meaningful test data to show that your MININET works. If the final project turned in does not work completely, then you should provide ways to demonstrate which modules in fact are working. All routines MUST have a single, primary author.

6. Output

   In order to observe the performance of the sliding window protocol and to check whether your implementation is working properly, you need to collect statistics. For debugging reasons, you should design a statistics gathering process on **both** the client and server machines. These processes record ongoing and final statistics. When you demonstrate your project, it is advantageous to have both monitoring processes' outputting information in separate windows. The following are suggested statistics (You should add your own which are appropriate to your specific proposal):

   (a) the total number of data frames transmitted

   (b) the total number of retransmissions

   (c) the total number of acknowledgments sent

   (d) the total number of acknowledgments received

   (e) the total number of rejected data frames

   (f) the total number of duplicate frames received

   (g) the number of times the client blocks due to a full window.

   For your debugging purposes you may wish to show the size of frames sent and received. It is wise to have a Debug enable/disable switch to provide easily switching the volume of output to go to the screen. For performance analysis purposes you should consider measuring the time required to satsify a client request.

7. Comments

   There are three aspects of this project which require specialized knowledge - the timer mechanism, the suspending of processes dl_send and dl_recv, and the TCP/IP interface. Since this project was originally designed for a 14 week graduate course, you MUST read ahead to complete the design report on time. When making design decisions, it is valid to propose a less-than-elegant solution due to your time constraints. However, your design paper needs to explain and defend such choices.

   NOTE WELL - This document outlines the project without taking up the issues of suggested ways to build the MININET. However, it is a good idea to think about adding complexity to the problem gradually. For example, build and test an errorless data link layer first. Furthermore, do not get hung up on the TCP/IP details right away.

8. Design Report

   Your design report will be graded based on basic technical writing standards including: grammar, formal presentation style, typos and clarity of writing.

   Your report must include: specification of the application protocol, a diagram that explains the components and interfaces of your design, an allocation of work between team members, a milestones schedule, and a bibliography. The design paper should 10-20 pages not including pictures.