

CSE 123 : Computer Networks

Project 1

Notes From Anup and Narendran

Project 1 Overview

- Sliding Window Protocol
 - Please read more from the book
- Sliding Window Protocol will handle
 - Drop of packets
 - Delayed packets
 - Corrupted packets
- Need to all the handle the above for the project

Sliding Window Protocol Components

- Framing
 - Breaking up large messages in to smaller pieces called **frames** with additional metadata
- Flow Control
 - Sender and Receiver Buffer to limit in-flight messages (Check in our case what is the value)
 - Frame ordering
- Reliable Communication
 - Detecting missing or corrupted frames
 - CRC computation at sender and receiver to ensure data integrity
 - Acknowledgements from receiver to sender
 - Retransmission in case of loss from sender to receiver

Tools

- Need to get familiarized with these two tools.
Can't stress enough!
 - **Gdb**
 - **Valgrind**

Starter Code

- main.c
- common.h
- communicate.*
- input.*
- sender.*
- receiver.*
- util.*

Starter Code main.c

- Initialization
 - Starts the sender threads
 - Starts the receiver threads
 - Sets the drop and corrupt probability
 - Once finished it frees up memory

Starter Code – common.h

- Contains struct and constant definitions that are shared across different system components
- *Cmd structs*
are populated by input thread
- *Receiver struct*
contains state for the receiver -- Add SWP receiver state
- *Sender struct*
contains state for the sender -- Add SWP sender state

Starter Code – common.h (Continued)

- *Frame struct*

contains data

- modify to contain the header and CRC footer
- Don't forget to modify FRAME_PAYLOAD to be 48!
- You can use at most 16 bytes for the combined header and footer.
- A good place to start is to read P&D chapter 2 up through section 2

Starter Code – communicate.h& communicate.c

- Contains two functions

*void send_msg_to_receivers(char *);*

*void send_msg_to_senders(char *);*

Starter Code – util.h & util.c

- Linked list implementation
 - functions operate on head node
 - don't modify existing functions
- Comparing two struct timeval objects
*long timeval_usecdiff(struct timeval *,
 struct timeval *);*
- Functions to serialize a frame to a char[64] and back to a frame – Need to change this
*char * convert_frame_to_char(Frame *);
Frame * convert_char_to_frame(char *);*
- Suggestion: Write frame printing function and test

Starter code – sender.c & sender.h

- Majority of code base for sender thread
- Initialize
 - *void init_sender(Sender *, int);*
- Perform 4 major steps in loop
 - Receive and process commands from input thread
 - Transmit / buffer messages
 - Process incoming acknowledgements
 - Retransmit timed out frames

Starter code – sender.c & sender.h

- Need to update or fill in the following:
 - *init_sender()*-- fill in with **sender**'s initialization
 - *sender_get_next_expiring_timeval()* -- return timeval* for next buffered message to expire
 - *handle_incoming_acks()* -- update SWP sender state
 - *handle_input_cmds* -- split long input messages in to multiple frames
 - *handle_timeout_frames* -- retransmit timed out frames
- This is not an exhaustive list

Starter code – receiver.c& receiver.h

- Implements receiver thread
- Receive and acknowledge incoming messages from senders
- Need to update the following:
 - *void init_receiver Receiver * receiver, int id)*
 - *void handle_incoming_msgs(Receiver * receiver, LLnode ** outgoing_frames_head_ptr)*
- We recommend following the steps suggested in the TODO

Additional Functions

- Add Additional Functions based on your needs
- Example
 - For Bit Manipulation
 - Linked List Manipulation

General Advice

- Following the steps outlined in the project description in order will help you.
- Trying to implement the whole thing in one go is likely fraught with peril.
- Get started early. This project is often both labor and details-oriented.
- Be aware that the code in P & D is incomplete. Copying it verbatim without understanding SWP is unwise.

Submission Instructions

- Prepare a design document explaining
 - Important data structures
 - Frame structure
 - Algorithms used
 - Anything you want to highlight to the graders
 - Be sure to proofread it before submitting
- Ask yourself, could someone else understand what I did by reading this report?
- To submit, cd to ~/project1 and type **make submit**
- **Test submission today, so we can resolve any issues**

Getting Started

- First form the packets. Fill the frame struct
- Then move to sequence and acknowledgement numbers
- Implement basic SWP. Don't worry about retransmission , drop etc. before this is done.
- Check out Gdb and Valgrind immediately.

Next Time

- Troubleshooting SWP
 - Sequence number wrap around
 - Acks that are not in the sender's window
 - Handling timeouts in the presence of struct timeval overflow in tv_usec
- Other common pitfalls
 - CRC codes and modulo 2 arithmetic
 - SWP and CRC will be covered in lecture