My Phase 3 is almost complete, it is able to send and receive icmp message. It sends a initial icmp message, udp train, and several following icmp messages. When it a echo reply is received the time is recorded and after they all arrive the RTT is displayed on the screen. I was not able to read from /dev/random and used urandom instead.

I opted to use threads instead of fork() because it made it easier to pass variable between functions. Also because threads are initiated faster and require less resources.

If I could do the program again I would of used a different method of storing the time stamps.

Challenges:

- Sending icmp / udp messages on the same socket
- Race condition between threads
- Reading from the /dev/random file

To resolve the first issue I looked at other examples of programs that send udp packets I found that they used a different protocol when opening the socket. After trying IPPROTO RAW I was able to send and receive messages on the same socket.This caused a problem later on because I used the same protocol to open the receiving socket, there where no issues after changing it to IPPRO ICMP.

In order to record a time stamp for each message I used a global array so that each function could access it. Since they where running on different threads it created a race condition. The first thread would write the sending time and it would get overridden by the second. I was getting negative RTT vales. To fix the problem I used pthread mutex so that it could only be access by a single thread at a time.

On my system the time it took to read from /dev/random was inconsistent, at time it would take minutes. In order to avoid the long delay I used urandom instead, since its nonblocking.