**Northeastern University**

CS 5700 Fundamentals of Networking

**FINAL PROJECT REPORT**

**Team members:** Sanjivani Jethwaney, Pranjal Nimse, Paavan Gopala Reddy

**Title:** File Transfer Using TCP/IP Protocol Implementation with Raw Sockets

**Project description:**

The objective of the project is to implement a TCP/IP protocol using raw sockets to transfer files from one host to another. Normally, the headers for network transfer requests are provided by the kernel/OS network stack. In this project, we have used raw sockets to provide custom headers for the transfer protocol. Here, we can provide customization for various header fields as opposed to the headers provided by the underlying operating system. Raw socket programming is often used in applications related to network security.

In the project, we have implemented the following customized fields for headers:

* IP Header:
  + ip\_ihl: header length
  + ip\_ver: version of protocol
  + ip\_tos: type of service
  + ip\_tot\_len: total length
  + ip\_id: ID of the packet
  + ip\_frag\_off: fragmented offset
  + ip\_ttl: time to live
  + ip\_proto = socket.IPPROTO\_TCP
  + ip\_check: header checksum
  + ip\_saddr: source IP address
  + ip\_daddr: destination IP address
* TCP Header
  + tcp\_source: source port
  + tcp\_dest: destination port
  + tcp\_seq: sequence number
  + tcp\_ack\_seq: acknowledgement sequence number
  + tcp\_doff: size of tcp header
  + TCP Flags
    - tcp\_fin
    - tcp\_syn
    - tcp\_rst
    - tcp\_psh
    - tcp\_ack
    - tcp\_urg
    - tcp\_window: maximum allowed window size
    - tcp\_check
    - tcp\_urg\_ptr

**Execution:**

* Sender sends the file mentioned in cmd args to the Receiver whose ip address is specified. The application wraps the data packet to be sent with TCP and IP headers.
* The application uses raw sockets to communicate with the Sender and waits for the Sender to establish connection using TCP handshake.
* After connection establishment, the Receiver receives the data packet, extracts the TCP and IP headers and verifies its integrity using checksum in TCP header.
* If all goes well, the Receiver sends ACK to the Sender and awaits for the next packets to be received.
* When a FIN packet is received, the Receiver closes the output file and closed the connection.

**Coding language:** Python

**Libraries**:

* sys
* socket
* signal
* os
* time