CSE 333 Section 7 - Client-Side Networking

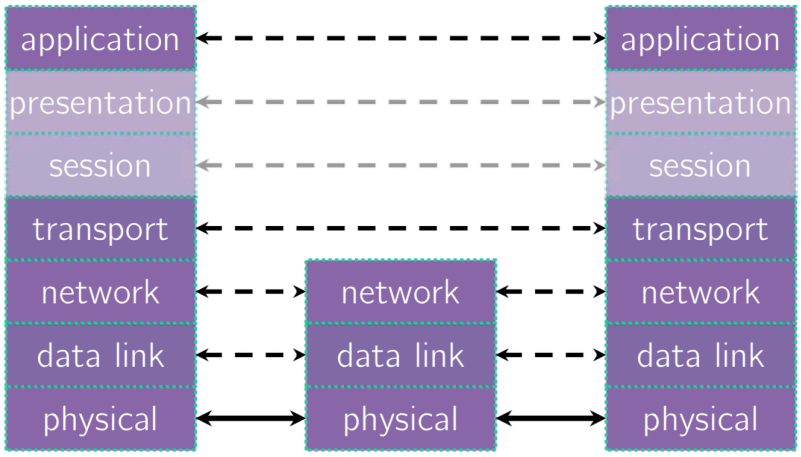
Welcome back to section! We're glad that you're here :)

Networking Quick Review

Exercise 1

a) What are the following protocols used for? (bonus: in which *layer* of the networking stack is it found?)

- DNS: Translating between IP addresses and host names. (Application Layer)
- IP: Routing packets across the Internet. (Network Layer)
- TCP: Reliable, stream-based networking on top of IP. (Transport Layer)
- UDP: Unreliable, packet-based networking on top of IP. (Transport Layer)
- HTTP: Sending websites and data over the Internet. (Application Layer)



b) Why would you want to use TCP over UDP?

TCP is reliable and has simpler semantics than UDP, so it's easier to use for a lot of applications.

c) Why would you want to use UDP over TCP?

Some applications can't tolerate delays from resending lost packets and/or don't mind losing a few packets, so UDP is a better choice for these.

Exercise 2

When reading from a file you can determine the exact amount of data to read based on the file size. In networking, you do not know how much data you may read across the network. Complete the code below that reads data across the network and prints it to stdout until the connection is closed.

```
int main(int argc, char **argv) {
  int socket_fd;
  char readbuf[512];
  int res;
  ...
  // Assume code to set up the network connection is included

  // Read data from the server until the connection is closed
  while ((res = read(socket_fd, readbuf, 512)) != 0) {
    if (res == -1) {
      if(errno == EINTR || errno == EAGAIN) {
        continue;
      }
      close(socket_fd);
      return EXIT_FAILURE;
    }

    // Write the data we read to stdout
    int pos = 0;
    while (res > 0) {
      int write_res = write(STDOUT_FILENO, readbuf + pos, res);
      if (write_res == -1) {
        if (errno == EINTR || errno == EAGAIN) {
          continue;
        } else {
          close(socket_fd);
          return EXIT_FAILURE;
        }
      }
      res -= write_res;
      pos += write_res;
  }
  close(socket_fd)
  return EXIT_SUCCESS;
}
```