Question 3

1. s

Throughput = (0.75 \* W) / RTT = (0.75 \* 128 \* 1024 \* 8) / 2 \* 100 \* 10^-3

= 3932160 bps = 3.75 Mbps

b)

client sends recursive request to the DNS resolver

DNS resolver sends request to the root DNS server to get the .edu TLD server

DNS resolver sends request to the TLD server to get the server that is responsible for pitt

DNS resolver sends request to the server that is responsible for the pitt to get the server that is responsible for the cs subdomain

DNS resolver sends request to the server that is responsible for the cs subdomain to get the IP address of [www.cs.pitt.edu](http://www.cs.pitt.edu)

DNS resolver sends back the IP to the client.

c)

The advantage of using UDP is that it has less overhead since it doesn’t need to establish a connection, nor does it have any error recovery mechanism. UDP is typically used when you only want to send small amount of data, for example when you want to request an IP address from DNS server.

It wouldn’t be appropriate to use the UDP if you want to send large file, since it is very likely that the UDP packet is going to be dropped during transmission and UDP doesn’t have any retransmission mechanism.

d)

Pros: Increased throughput?

Cons: Greater chance congesting the network

e)

HTTP 1.0 doesn’t have persistent connection, so it will needs to establish and close a connection when sending each resources.

So if you’re using HTTP 1.0, you will need 3 connections meaning 3 \* 7 (3 packets for TCP connection establishment + 4 packets for TCP connection termination) plus 3 packets of actual payload, 2 of which is images and 1 is text. In total you will need 24 packets. While if you’re using HTTP 1.1, you can reuse the existing connection, so you will only need to transfer 7 + 3 = 10 packets.

Question 4

a)

i)

2000 / (980 - 20) = 2 … 80

Assuming that we use the minimum IP header length 20 bytes, thus we minus 20 in the above calculation. So that means we need 3 fragmented packet in total to transfer down the link.

ii)

Identifier: Group together fragments of the same datagram.

Datagram length: The length of the original datagram.

More fragments: Indicates whether or not there are more fragments of the datagram to follow

Fragmentation offset: The offset from the start of the original datagram in chunks of 8B.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Packet #1 | Packet #2 | Packet #3 |
| identifier | 227 | 227 | 227 |
| datagram length | 980 | 980 | 100 |
| “more fragments” flag | 1 | 1 | 0 |
| fragmentation offset | 0 | 120 | 240 |

iii)

*Advantage of assembling the packets in the intermediate hosts:*

When intermediate hosts try to assemble the packet, it might realize that some packet is dropped during the transmission, thus it could trigger re transmission. And by enabling assembly we can trigger the retransmission as early as possible.

*Advantage of not assembling the packets in the intermediate hosts:*

Reduce the overhead in the network. We can potential have simpler implementation of the network.

b)

i) look up the forwarding table to see where to send the packet using longest prefix match

update the TTL and checksum, queue the packet if necessary

ii)

broken network see [wikipedia](https://en.wikipedia.org/wiki/Routing_loop_problem) for an example or incorrect/corrupted forwarding table

iii)

some ip routing protocol has built-in cycle prevention scheme

other will resolve the loop routing problem once the new network topology is known by all the routers within a network. TTL is a hop counter -> discard packet after N hops.

iv) not feasible. that might potentially limit the speed of the network.