Homework 1 Suggested Solution

1: Why two ISPs peer with each other? (Hint: consider the different types of peering: Regional ISP with Regional ISP, Tier 1 with Tier 1, and Regional with content provider) How does an IXP fit into this ecosystem? (Hint: study some IXP, e.g., https://www.sgix.sg/)

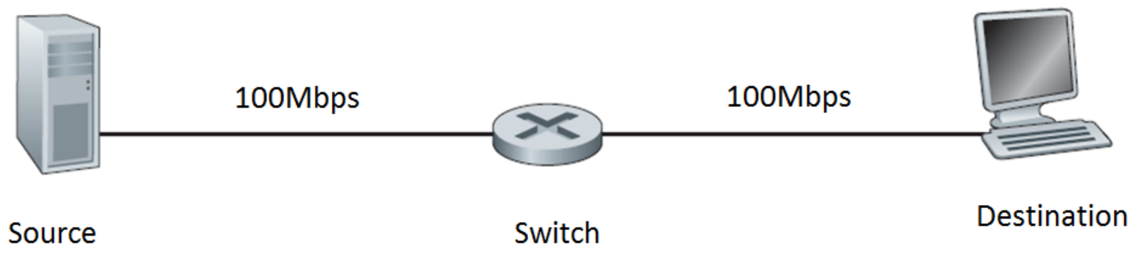
Suggested Solution:

ISPs peer with each other to provide / improve connectivity for their respective customers. The peering between two regional ISPs allow them to route the traffic between their customers directly, without paying their respective provider ISPs. It also provides extra redundancy of available network paths. A content provider ISP peers with a regional or access ISP to have more direct connectivity to the latter's consumers. This peering allows both to not pay for their respective provider ISPs and may also help improve the network performance (bandwidth, delay, etc.) and redundancy for the consumers of the regional /access ISPs to access the content from the content provider.

An Internet Exchange Points (IXP) (typically in a standalone building with its own switches) is a meeting point where multiple ISPs can connect and/or peer together. An IXP's revenue comes from charging each of the ISPs that connect to the IXP a relatively small fee, which may depend on the amount of traffic sent to or received from the IXP.

2. Calculate the end-to-end delay (i.e., the duration from the first bit sent by the source to the last bit received by the destination) for a packet with size of 1500 bytes (12,000 bits) for the following:

2.1 100 Mbps Ethernet (assume 1M = 10^6) with a single store-and-forward switch (that is, with two links, one incoming and one outgoing, as shown in the Figure below) in the path. Assume that each link has a propagation delay of 12 µs (1 µs = 10^{-6}s) and that the switch begins forwarding immediately after it has finished receiving the whole packet. Assume zero processing and queueing delay. Calculate the end-to-end delay.



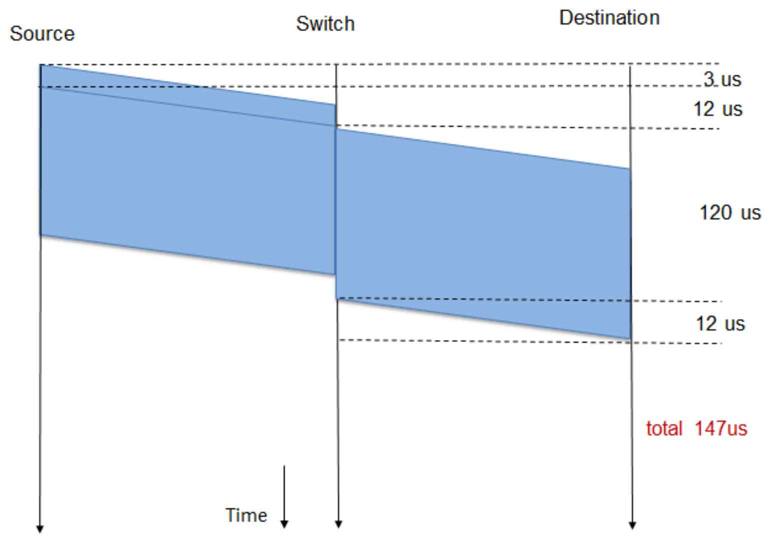
Single switch case

Suggested solution:
Transmission delay: $12000/10^8$ = 120 µs
End-to-end delay: (120 + 12) x 2 = 264 µs

2.2 Same scenario as **2.1** above, calculate the end-to-end delay when there are four switches (each switch has the same setting as in scenario **2.1**) in the path.
Suggested solution:
(120 + 12) x 5 = 660 µs

2.3 Same scenario as **2.1** above, i.e., only a single switch, but assume the switch implements "cut-through" switching, i.e., the switch begins to forward the packet after the first 300 bits of the packet have been received. Calculate the end-to-end delay.
Suggested solution:
$300/10^8$ = 3 µs
End-to-end delay: 12 + 3 + 120 + 12 = 147 µs

2.4 Draw a space-time diagram to depict the transfer in scenario **2.3** above.
Suggested solution:


Source
Switch
Destination
3 us
12 us
120 us
12 us
total 147us
Time

3. (textbook chapter 2, review problem R26): In Section 2.7 of the textbook, the UDP server described needed only one socket, whereas the TCP server needed two sockets. Why? If the TCP server were to support n simultaneous connections, each from a different client host, how many sockets would the TCP server need?

Suggested solution: With the UDP server, there is no welcoming socket, and all data from different clients enters the server through this one socket. With the TCP server, there is a welcoming socket, and each time a client initiates a connection to the server, a new socket is created. Thus, to support n simultaneous connections, the server would need n+1 sockets.

4. (textbook problem chapter 2, problem 4) Consider the following string of ASCII characters that were captured by Wireshark when the browser sent an HTTP GET message (i.e., this is the actual content of an HTTP GET message). The characters <cr><lf> are carriage return and line-feed characters (that is, the italized character string <cr> in the text below represents the single carriage-return character that was contained at that point in the HTTP header). Answer the following questions, indicating where in the HTTP GET message below you find the answer.

GET /cs453/index.html HTTP/1.1<cr><lf>Host: gaia.cs.umass.edu<cr><lf>User-Agent: Mozilla/5.0 (Windows;U; Windows NT 5.1; en-US; rv:1.7.2) Gecko/20040804 Netscape/7.2 (ax) <cr><lf>Accept:ext/xml, application/xml, application/xhtml+xml, text/html;q=0.9, text/plain;q=0.8, image/png,*/*;q=0.5<cr><lf>Accept-Language: en-us,en;q=0.5<cr><lf>Accept-Encoding: zip,deflate<cr><lf>Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7<cr><lf>Keep-Alive: 300<cr><lf>Connection:keep-alive<cr><lf><cr><lf>

a. What is the URL of the document requested by the browser?

b. What version of HTTP is the browser running?

c. Does the browser request a non-persistent or a persistent connection?

d. What is the IP address of the host on which the browser is running?

e. What type of browser initiates this message? Why is the browser type needed in an HTTP request message?

Suggested solution:

a) The document request was http://gaia.cs.umass.edu/cs453/index.html. The Host: field indicates the server's name and /cs453/index.html indicates the file name.

b) The browser is running HTTP version 1.1, as indicated just before the first <cr><lf> pair.

c) The browser is requesting a persistent connection, as indicated by the Connection: keep-alive.

d) This is a trick question. This information is not contained in an HTTP message anywhere. So there is no way to tell this from looking at the exchange of HTTP messages alone. One would need information from the IP datagrams (that carried the TCP segment that carried the HTTP GET request) to answer this question.

e) It is from a Mozilla compatible browser, likely the Netscape version 7.2. See https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent.

   The browser type information is needed by the server to send different versions of the same object to different types of browsers.

   But be aware about "User agent spoofing" and the long & interesting history behind it
   https://webaim.org/blog/user-agent-string-history/

5. (textbook problem chapter 2, problem 5) The text below shows the reply sent from the server in response to the HTTP GET message in the question above. Answer the following questions, indicating where in the message below you find the answer.

HTTP/1.1 200 OK<cr><lf>Date: Tue, 07 Mar 2008 12:39:45GMT<cr><lf>Server: Apache/2.0.52 (Fedora)<cr><lf>Last-Modified: Sat, 10 Dec2005 18:27:46 GMT<cr><lf>ETag: “526c3-f22-a88a4c80”<cr><lf>Accept-Ranges: bytes<cr><lf>Content-Length: 3874<cr><lf>Keep-Alive: timeout=max=100<cr><lf>Connection:Keep-Alive<cr><lf>Content-Type: text/html; charset=ISO-8859-1<cr><lf><cr><lf><!doctype html public “//w3c//dtd html 4.0 transitional//en”><lf><html><lf><head><lf> <meta http-equiv=”Content-Type”content=”text/html; charset=iso-8859-1”><lf> <meta name=”GENERATOR” content=”Mozilla/4.79 [en] (Windows NT 5.0; U) Netscape]”><lf> <title>CMPSCI 453 / 591 / NTU-ST550A Spring 2005 homepage</title><lf></head><lf> <much more document text following here (not shown)>

a. Was the server able to successfully find the document or not? What time was the document reply provided?

b. When was the document last modified?

c. How many bytes are there in the document being returned?

d. What are the first 5 bytes of the document being returned? Did the server agree to a persistent connection?

Suggested solution:

a) The status code of 200 and the phrase OK indicate that the server was able to locate the document successfully. The reply was provided on Tuesday, 07 Mar 2008 12:39:45 Greenwich Mean Time.

b) The document index.html was last modified on Saturday 10 Dec 2005 18:27:46 GMT.

c) There are 3874 bytes in the document being returned.

d) The first five bytes of the returned document are: <!doc. The server agreed to a persistent connection, as indicated by the Connection: Keep-Alive field