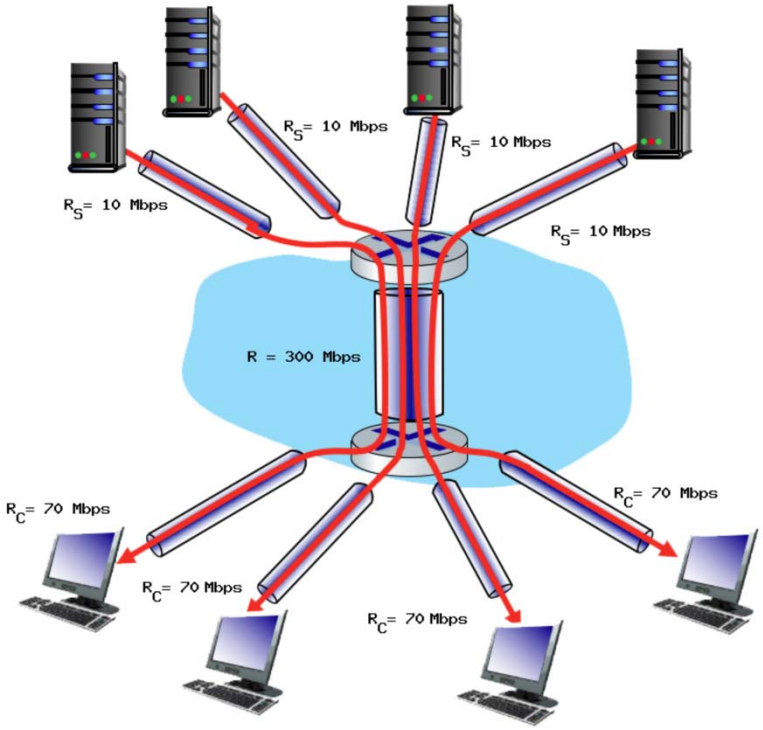
Computer Network Midterm 111-1

1. Consider the scenario shown below, with four different servers connected to four different clients over four three-hop paths. The four pairs share a common middle hop with a transmission capacity of R = 300 Mbps. The four links from the servers to the shared link have a transmission capacity of R_S = 10 Mbps. Each of the four links from the shared middle link to a client has a transmission capacity of R_C = 70 Mbps. (10%)



a. What is the maximum achievable end-end throughput (in Mbps) for each of four client-to-server pairs, assuming that the middle link is fairly shared (divides its transmission rate equally)? (2%)

b. Which link is the bottleneck link? Format as Rc, Rs, or R (2%)

c. Assuming that the servers are sending at the maximum rate possible, what are the link utilizations for the server links (RS)? Answer as a decimal (2%)

d. Assuming that the servers are sending at the maximum rate possible, what are the link utilizations for the client links (RC)? Answer as a decimal (2%)

e. Assuming that the servers are sending at the maximum rate possible, what is the link utilizations for the shared link (R)? Answer as a decimal (2%)

Ans:

a. The maximum achievable end-end throughput is the capacity of the link with the minimum capacity, which is 10 Mbps (2%)

b. The bottleneck link is the link with the smallest capacity between R_S, R_C, and R/4. The bottleneck link is Rs. (2%)

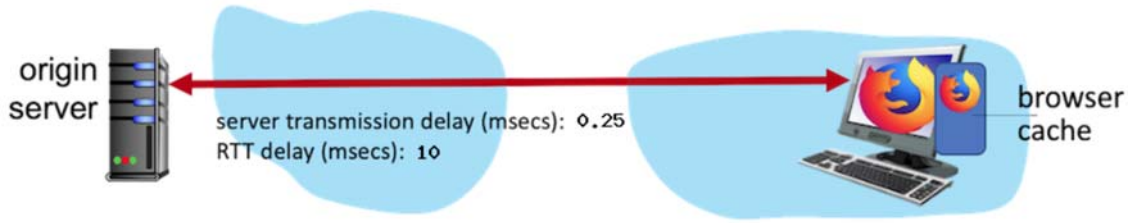
c. The server's utilization = $R_{bottleneck}$ / R_S = 10 / 10 = 1 (2%)

d. The client's utilization = $R_{bottleneck}$ / R_C = 10 / 70 = 0.14 (2%)

e. The shared link's utilization = $R_{bottleneck}$ / (R / 4) = 10 / (300 / 4) = 0.13 (2%)

Computer Network Midterm 111-1

2. Consider an HTTP server and client as shown in the figure below. Suppose that the RTT delay between the client and server is 10 msecs; the time a server needs to transmit an object into its outgoing link is 0.25 msecs; and any other HTTP message not containing an object has a negligible (zero) transmission time. Suppose the client again makes 60 requests, one after the other, waiting for a reply to a request before sending the next request. Assume the client is using HTTP 1.1 and the IF-MODIFIED-SINCE header line. Assume 30% of the objects requested have NOT changed since the client downloaded them (before these 60 downloads are performed) How much time elapses (in milliseconds) between the client transmitting the first request, and the completion of the last request? （要寫出兩部分的算式，並說明 6%，答案 4%，10% total）



Ans:

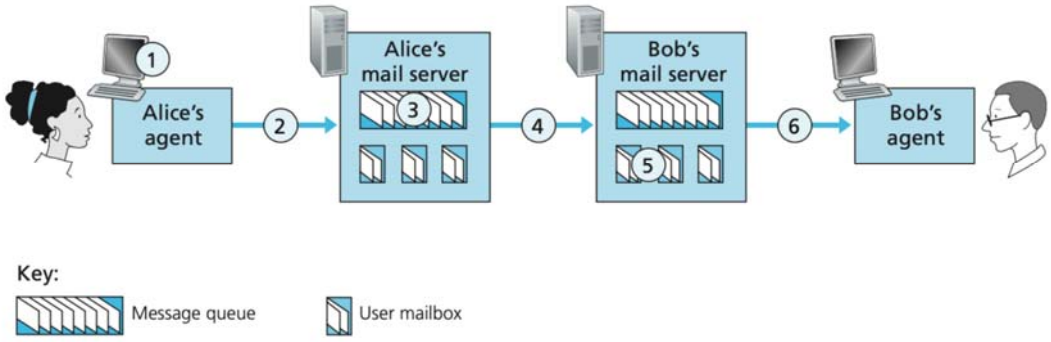
(RTT * NUM_PACKETS) + (NUM_PACKETS * (PERCENT__NOT_CACHED / 100) * TRANS_DELAY) = (10 * 60) + (60 * ((100-30) / 100) * 0.25)

或

(NUM_PACKETS * (PERCENT__NOT_CACHED / 100) * (RTT+ TRANS_DELAY)) (3%)+ (NUM_PACKETS * (PERCENT__CACHED / 100) * RTT) (3%)= (60 * 10 *0.3) + (60 * 0.7* (10+0.25))

= 610.5 ms (4%)

3. Look at the scenario below, where Alice sends an email to Bob. For the questions below, assume both Bob's and Alice's user agents use the IMAP protocol. (2% each, 14% total)



a. At point 2 in the diagram, what protocol is being used? (2%)
b. At point 4 in the diagram, what protocol is being used?
c. At point 6 in the diagram, what protocol is being used?
d. Does SMTP use TCP or UDP?
e. Is SMTP a 'push' or 'pull' protocol?
f. Is IMAP a 'push' or 'pull' protocol?
g. What port does SMTP use?

Ans:

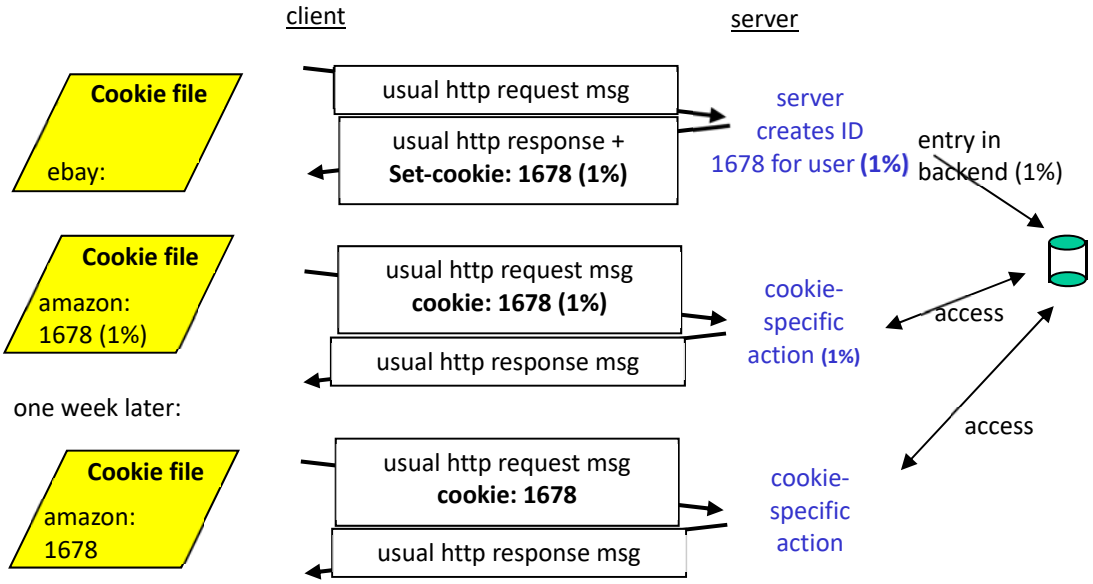
a. At point 2 in the diagram, the SMTP protocol is used. (2%)
b. At point 4 in the diagram, the SMTP protocol is used.
c. At point 6 in the diagram, the IMAP protocol is used.
d. SMTP uses the TCP protocol.
e. SMTP is a 'push' protocol

f. IMAP is a 'pull' protocol
g. SMTP uses port 25

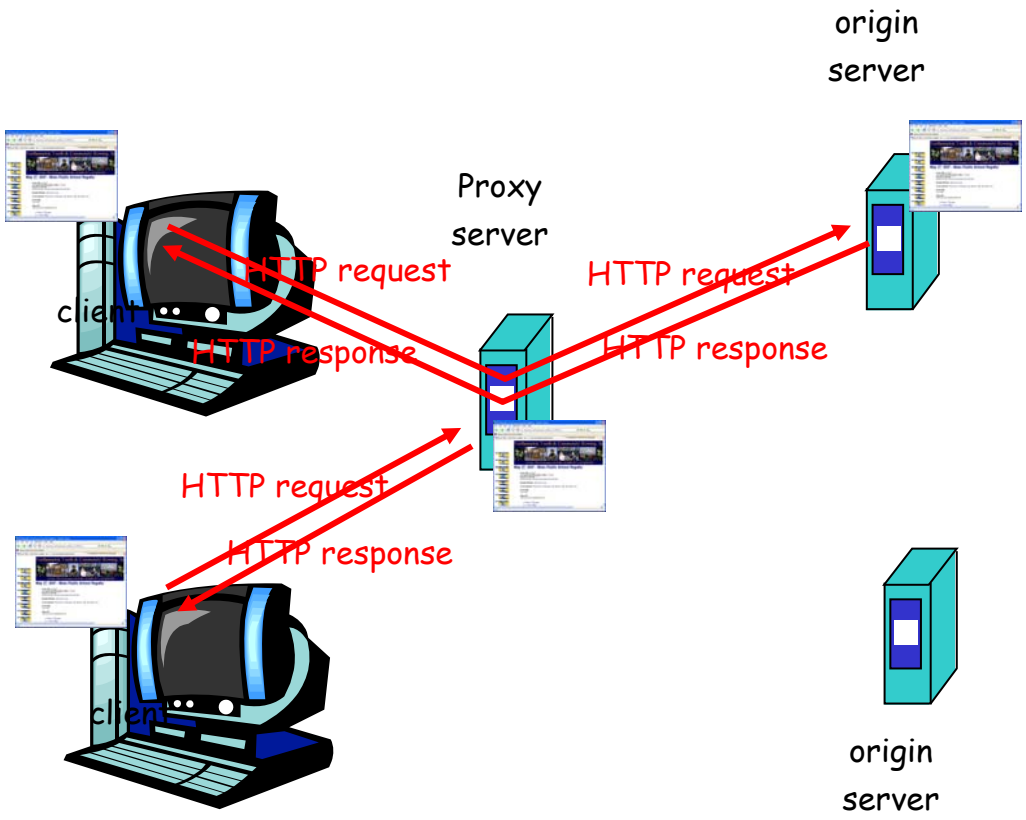
4. Describe detailed operations of HTTP cookie, web caching and conditional GET. (6*3=18%)（說明其用處，並畫圖加解釋每步驟）

Ans:

❑ cookie:
when a user visits a specific web site for first time and initial HTTP requests arrives at site, site creates a unique ID and creates an entry in backend database for recording user states of this ID.
=> keep client's states (cookie-specific action)!

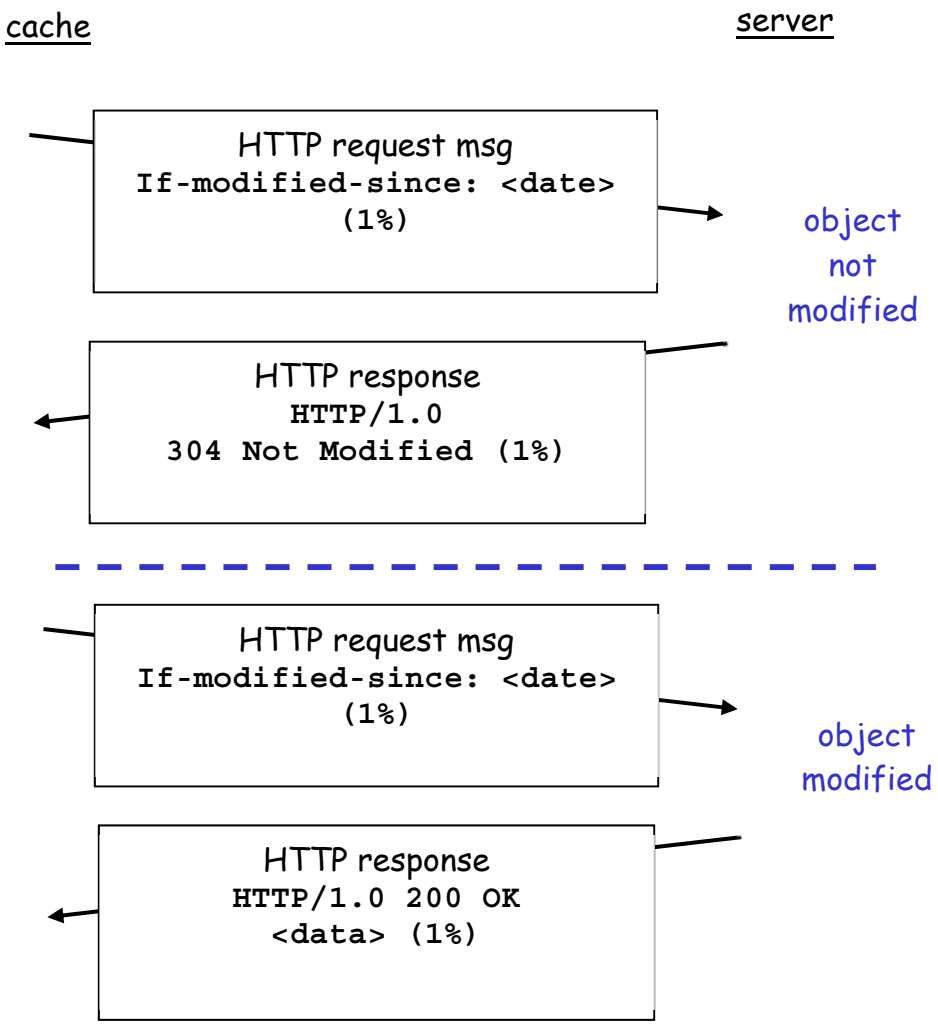


❑ web caching:
- user sets browser: Web accesses via cache
- browser sends all HTTP requests to cache (2%)
  if object in cache
    cache returns object (2%)
  else
    cache requests object from origin server, then returns object to client (2%)



conditional GET (6%)

- Conditional GET: don't send object if cache has up-to-date cached version (1%) => reduce traffic loads (delays) on network links! (1%)
  cache: specify date of cached copy in HTTP request (1%)
  If-modified-since: <date> (1%)
  server: response contains no object if cached copy is up-to-date: (1%)
  HTTP/1.0 304 Not Modified (1%)



5. (a) Explain Internet protocol stack (1% each layer's name, 1% each layer's function, 10% total) (b) Besides, you have to write the name of data unit of upper four layer. (寫出最上面四層資料單位的專有名稱，如 xx 層：yy) (4%) (14% total)

Ans:

(a)

application: supporting network applications
transport: host-host data transfer
network: routing of datagrams from source to destination
link: data transfer between neighboring network elements
physical: bits "on the wire" (1% each layer's name, function 1%, 10% total)

application
transport
network
link
physical

(b)

application layer: message
transport layer: segment
network layer: datagram
link layer: frame
各 1%，共 4%

6. Assume you request a webpage consisting of one document and 6 images. The document size is 1kbyte, all images have the same size of 10 kbytes, the download rate is 2Mbps, and the RTT is 300ms. How

long in second (秒為單位) does it take to obtain the whole webpage under the following conditions? Why? （要說明(a) (b)(c)的動作與計算過程每個部分在算什麼值）(Assume no DNS name query is needed and the impact of the request line and the headers in the HTTP messages is negligible). (18% total)

a. Nonpersistent HTTP with serial connections. (說明 2%, 計算過程與結果 4%)
b. Persistent HTTP with one connection.
c. Persistent HTTP with pipelining

Ans:

The total download time is:

a. <u>At most one object is sent over a TCP connection</u>. *(2%)*

$2 \cdot 300\ \text{ms}\left(\text{TCP handshaking} + \text{HTTP}\frac{\text{request}}{\text{response}}\text{for web page}\right) + \frac{8*1*10^3 bits}{\frac{2*10^6 bits}{s}}\ (document\ transmission\ delay) + 6 \cdot \left\{2 \cdot 300ms\left(\text{TCP handshaking} + \text{HTTP}\frac{\text{request}}{\text{response}}\text{for 1 object}\right) + \frac{8*10*10^3 bits}{\frac{2*10^6 bits}{s}}(image\ transmission\ delay)\right\} = 600ms + 4ms + 6 * (600ms + 40ms)(2\%) = 4.444\,\text{sec}\ \ (2\%)$ (4% total)

b. <u>Multiple objects can be sent over single TCP connection between client and server. The browser first waits to receive a HTTP response from the server before issuing a new HTTP request</u>. *(2%)*

$2 \cdot 300\ \text{ms}\left(\text{TCP handshaking} + \text{HTTP}\frac{\text{request}}{\text{response}}\text{for web page}\right) + \frac{8*1*10^3 bits}{\frac{2*10^6 bits}{s}}\ (document\ transmission\ delay) + 6 \cdot \left\{300ms\left(\text{HTTP}\frac{\text{request}}{\text{response}}\text{for 1 object}\right) + \frac{8*10*10^3 bits}{\frac{2*10^6 bits}{s}}(image\ transmission\ delay)\right\} = 600ms + 4ms + 6 * (300ms + 40ms)(2\%) = 2.644\ sec\ \ (2\%)$. (4% total)

c. persistent HTTP with pipelining:
<u>The browser issues requests as soon as it has a need to do so, without waiting for response messages from the server.</u> *(2%)*

$2 \cdot 300\ \text{ms}\left(\text{TCP handshaking} + \text{HTTP}\frac{\text{request}}{\text{response}}\text{for web page}\right) + \frac{8*1*10^3 bits}{\frac{2*10^6 bits}{s}}\ (document\ transmission\ delay) + 300ms\left(\text{HTTP}\frac{\text{request}}{\text{response}}\text{for 6 objects}\right) + 6 \cdot \left\{\frac{8*10*10^3 bits}{\frac{2*10^6 bits}{s}}(image\ transmission\ delay)\right\} = 600ms + 4ms + 300ms + 6 * 40ms\ \ (2\%) = 1.144\,\text{sec}\ \ (2\%)$ (4% total)

7. Consider the figure below, where a client is sending an HPPT GET message to a web server, gaia.cs.umass.edu. Suppose the client-to-server HTTP GET message is the following: (8%)

GET /kurose_ross/interactive/hello.htm HTTP/1.1
Host: gaia.cs.umass.edu
Accept: text/plain, text/html, text/xml, image/png, image/jpeg, audio/vnf.wave, audio/basic, video/mpeg, video/wmv, application/, */**
Accept-Language: en-us, en-gb;q=0.6, en;q=0.1, fr, fr-ch, de, ar
If-Modified-Since: Mon, 13 Nov 2013 09:00:58 EST
User Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:10.0.2) Gecko/20100101 Firefox/10.0.2

Answer the following questions:

1. What tool can capture ("sniff") messages being sent/received from/by your computer? (2%)
2. What header is used to show the native platform the browser is running on? (2%)

3. What is the name of the file that is being retrieved in this GET message? (2%)
4. What version of HTTP is the client running? (2%)

Ans:

1. The packet sniffer (Wireshark) (2%)
2. *User Agent:*. (2%)
3. The file being fetched is /kurose_ross/interactive/hello.htm (2%)
4. The client is running HTTP version 1.1. (2%)

8. Describe four sources of packet delays (1% each delay name, 1% reason of each delay, 8% total)

Ans:

(a) nodel processing delay: check bit errors, determine output link

(b) queueing delay: time waiting at output link for transmission, depends on congestion level of router

(c) transmission delay: R=link bandwidth (bps) L=packet length (bits), time to send bits into link = L/R

(d) propagation delay: d = length of physical link, s = propagation speed in medium, propagation delay = d/s (1% each delay name, 1% reason of each delay, 8% total)