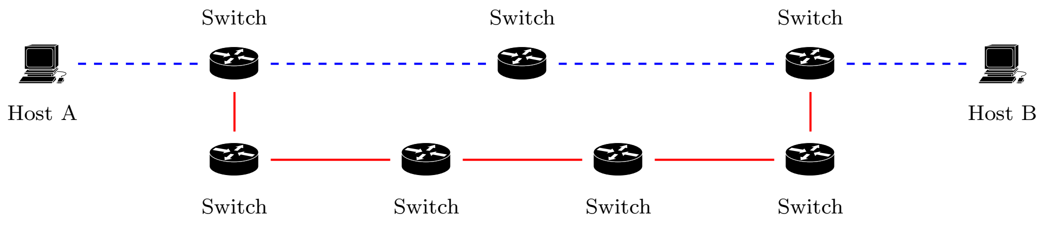
Computer Networks Homework 1

Fall 2019

Due: 3 February 2020

1. (12 points) Imagine that there are two paths a packet could take to travel from Host J to Host K, as illustrated here:

   Minor fix, we're assuming $(HostA = HostJ)$ and $(HostB = HostK)$



   Each link drawn with a dashed blue line has a bandwidth of 250000 bps; each link drawn with a solid red line has a bandwidth of 400000 bps. Packets are always 1000 bytes. You may assume that process, queue, and propegation delays are negligable.

   What is the delay in each path? Which path should the packet take to get to Host K the fastest?

   Solution:

   Since only a single packet is specified and only transmission delays are non negligable calculating the two paths is simply calculating the total transmission delay on both routes.

   Transmission delay is defined as $D_t = N/R$ where:

   1. D_t = the transmission delay in seconds
   2. N = the number of bits
   3. R = the rate of transmission in bps.

   Our total payload is one packet of size 1000 bytes (8000 bits).

The delay from one computer (host or switch) to another is $8000/250000 = 0.032$ seconds on the blue line.

The delay from one computer (host or switch) to another is $8000/400000 = 0.02$ seconds on the red line.

The path along the exclusively blue line contains 4 hops leading us to an easy $0.032 * 4 = 0.128$ seconds

The other path contains two blue hops and 5 red hops. So, $0.032 * 2 = 0.064$ seconds on the blue hops, and $0.02 * 5 = 0.16$ seconds on the red hops. Combining these gives us a total of $0.064 + 0.16 = 0.224$ seconds on the secondary path.

From this, we can state that the packet should take the exclusively blue line to send the packet with the lowest latency.

Solution 1

2. (8 points) Consider again the network in Question 1. Each link drawn with a dashed blue line still has a bandwidth of 250000 bps. What is the minimum bandwidth for solid red links such that the bottom path is faster than the top path?

Solution:

In short, the minimum amount of bandwidth required to ensure that the exclusive blue line has a lower latency than the red line is when the exclusively blue line is over saturated. This happens with an average throughput of 250000 bps, so the volume of traffic where the exclusively blue line is saturated is at least 250001 bps.

3. (12 points) A mountain bike rider is competing in the Tour Divide, a 4418 km race from Banff, Alberta to Antelope Wells, New Mexico. She is carrying a GPS tracker, as do all racers. The tracker sends a signal to a satellite, which relays it back down to a server in New Orleans, LA. A friend of this rider wants to know her current position. What is the delay from when the signal leaves the tracker to when the friends computer in Seattle, Washington?

   Assume the following:

   1. the connection from the tracker to the satellite (and the satellite back to the server) is wireless and the signal travels at the speed of light,
   2. the satellite is orbiting in medium-Earth orbit at approximately 20000 km,
   3. the friend is using a copper connection to retieve the information,
   4. it is approximately 4228 km from New Oreleans to Seattle, and finally,

5. the transmission delay of the packet of the packet is negligible because it is so small.

Solution:

1. First is calculating the latency from the tracker to the satellite (hop 1 to 2). The speed of light is 299792 km/s, over a distance of 20000 km. This gives us $20000km/299792(km/s)$ = 0.066712921 seconds. This is then doubled from 0.066712921 seconds to 0.13342584 to account for bouncing back to the server in New Orleans. for the transmission delay.
2. Second is calculating the hop from the server to the friends computer in Seattle. The propagation speed of copper cable is $2.3 * 10^8$ m/s from the given slides. Converting 4228 km to meters gives us 4228000. From here, we can calculate the latency with $4228000/(2.3 * 10^8) = 0.018382609$ seconds.
3. Combining these together we get a transfer speed of $0.13342584 + 0.018382609 =$ 0.15180845 seconds, or more commonly:151.80845 ms.

4. (12 points) A security camera captures images that are each 100000 bytes in size. It takes one image every .25 s. If there is a 500 m point-to-point (cable, $2.3 \cdot 10^8$ bps propegation rate) connection to the monitor where the security guard inspects these images, what is the minimum acceptable bandwidth (bandwidth required to transmit one image before the next is taken) between the camera and monitor?

Solution:

1. First we convert 100000 bytes to bits -¿ 800000 bits.
2. Second figure out the potential throughput by dividing 500m by $2.3 * 10^8$ equaling 0.000002173913 capability of a moment on the line.
3. The minimum bandwidth needs to be capable of dealing with 800000 bits per image four times a second, meaning 3200000 bits.
4. 3200000 bit of images per second across a line of 0.000002173913, multiplying together getting a required bandwidth of 6.9565216 bps.

5. (8 points) A standard Ethernet frame (or packet) is 1500 bytes. The most common version of Ethernet found on consumer devices is Gigabit Ethernet, which operates with a bandwidth of 1 Gbps. If two hosts are placed 2500 m away from each other, how many frames should be sent out to “peek the pipe full” for the full Round Trip Time?

Solution:

1. Consumer devices usually run on copper cabling with a propagation rate of $2.3*10^8$ m/s. Across a range of 2500m, doubled to 5000m for round trips is 0.00002173913 seconds for a round trip.

2. Since we're dealing with round trip traffic, we can essentially slice the bandwidth in half for our calculations (half up half down). Giving us 500 Mbp/s (500000 bp/s) of traffic to saturate, given 12000 bits per packet. It will take $41\frac{2}{3}$ packets to statically saturate the line, which we'll round up to 42.

3. Since traffic is constantly going on and off the line the transmission delay is $\frac{data}{bps}$ is $\frac{42}{0.00002173913}$ which is 1932000 packets to saturate the line.

6. Let's think a little about security in my analogy of sending a multi-part present to Samantha. Suggest mechanisms to accomplish each of the following tasks below. Remember that we are protecting physical packages, not packets!

   (a) (4 points) I don't want the post office to know who the source of the presents is, but I want Samantha and her mom to know.

   (b) (4 points) I don't want anybody except Samantha to know who the source of the presents is.

   (c) (4 points) Even if the postal worker opens up the packages, I don't wnat him to know that the three packages are related.

Solution:

1. To solve (a) the answer is to not provide a return address on the box, but leave a note saying who it was from inside the box.

2. To solve (b) Put a box inside of a box, where the outside box has only a mailing address to Samantha's mom. The inside box should have a label on it saying for Samantha, and then a note inside the inner box that says who it is from.

3. Assuming the postal worker only opens up the outer package, the method from solution (b) would solve this, with the addition of putting the package number inside a slip in the inner box.

7. (8 points) A standard Ethernet frame has a payload of 1500 octets. I have designed an application-layer protocol with a fixed 16 octets of header data. Use any reliable resource to find the standare header sizes for TCP and IP. What is the maximum actual data that can fit into the payload of that Ethernet frame after all the headers? What is the percentage overhead?

Solution:

The maximum size of a TCP header is 60 bytes (minumum of 20 bytes) and the maximum size of the packet is 65535 bytes. The 16 octet header conveniently falls at the required 32 bit boundary defined, however is not at the required minimum header size of 20 bytes so we have 4 bytes of zero padding.

The maximum actual data that may fit in is $1500 - 20 = 1480$ bytes of data.

Overhead can be calculated with the following:

$$\frac{PacketSize - PayloadSize}{PacketSize}$$

For us this is $\frac{1500-1480}{1500}$, which gives us a protocol overhead of 0.013333333%.

8. (8 points) Shown below is a simple HTTP GET request with some cookie values. What is the ultimate address that is requested by the client? (You may have to make some assumptions about how the server interprets the cookies. State these assumptions and make them logical!)

```
GET /pages/home.html HTTP/1.1
Host: www.mtu.edu
Connection: close
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6)
   AppleWebKit/537.36 (KHTML, like Gecko) Chrome/76.0.3809.132
   Safari/527.36
Cookie: user=newton
Cookie: dept=ECE
Cookie: userprefixchar=%7E
```

Solution:

The web URL being accessed is $https://www.mtu.edu/pages/home.html$.

MTU enforces a redirect policy on non SSL/TLS traffic to SSL/TLS (https), since the header doesn't make any note of HSTS it's likely a normal Apache rewrite rule.

Appending the GET location to Host is rather obvious, I think the only really tricky thing in this problem is the Cookie sections. However, they are meant to stay in the header, and not be added to the URL. While you technically could pass and parse cookies along the URL as part of a $?key = value$; (similar to that PHP uses) it's considered rather insane to do so. Cookies tend to store private information and should be handled over encrypted traffic (in the header). The URL sent out in plain text so it knows where to go.

9. (8 points) Show a full HTTP response message for a simple HTML document that just has the text, "Hello, World!" as its sole content. Send two cookies to the client: a

random session key called PersonID; and a cookie called Planet with the value “Earth” that expires on January 1, 2099 at 11:59 pm.

Solution:

```
HTTP/1.1 200 OK
Date: Mon, 3 Feb 2020 12:28:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Mon, 22 Feb 2020
   19:15:56 GMT
Set-Cookie: PersonID="
   randomSessionKey"
Set-Cookie: Planet="Earth";  Expires
   =1 Jan 2099 11:59:00 GMT
Content-Length: 31
Content-Type: text/html
Content:

<html>
Hello, World!
</html><cr><lf>
```

10. (12 points) A somewhat smart browser makes an initial request for a webpage an closes the connection. After parsing the requested page, it will make several consecutive requests for the images without closing the connection. If the webpage below shows the HTML page requested by the original request, show the subsequent request and response messages. You do not need to make up data for the images, but get the Content Type correct!

```
<HTML>
  <HEAD>
    <TITLE>My Vacation Pictures</TITLE>
  </HEAD>
  <BODY>
    <H1>Pictures from my vacation to Japan!</H1>
    <IMG src=GoldenPavillion.png>
    <br>
    <IMG src=food.jpg>
    <br>
    <IMG src=Geisha.png>
    <br>
  <BODY>
</HTML>
```

Solution:

Golden Pavillion GET

```
GET /GoldenPavillion.png HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01;
   Windows NT)
Host: www.tutorialspoint.com
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

Golden Pavillion Response

```
HTTP/1.1 200 OK
Date: Mon, 3 Feb 2020 19:15:56 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Mon, 22 Feb 2020 19:15:56 GMT
Content-Length: someNumberInBytes
Content-Type: text/html
Content: image/png

<image data><cr><lf>
```

food GET

```
GET /food.png HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01;
   Windows NT)
Host: www.tutorialspoint.com
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

food Response

```
HTTP/1.1 200 OK
Date: Mon, 3 Feb 2020 19:15:56 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Mon, 22 Feb 2020 19:15:56 GMT
Content-Length: someNumberInBytes
Content-Type: image/jpeg
Content:

<image data><cr><lf>
```

Geisha GET

```
GET /Geisha.png HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01;
   Windows NT)
Host: www.tutorialspoint.com
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

Geisha Response

```
HTTP/1.1 200 OK
Date: Mon, 3 Feb 2020 19:15:56 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Mon, 22 Feb 2020 19:15:56 GMT
Content-Length: someNumberInBytes
Content-Type: image/png
Content:

<image data><cr><lf>
```