Nombre: Kevin Hurtas

CMP-4005 -- Homework 1

Answer the following questions.

1. Calculate the total time required to transfer a 1000-KB file in the following cases, assuming an RTT of 100 ms, a packet size of 1 KB data, and an initial 2×RTT of "handshaking" before data is sent:

    1. The bandwidth is 1.5 Mbps, and data packets can be sent continuously.
    2. The bandwidth is 1.5 Mbps, but after we finish sending each data packet we must wait one RTT before sending the next.
    3. The bandwidth is "infinite," meaning that we take transmit time to be zero, and up to 20 packets can be sent per RTT.
    4. The bandwidth is infinite, and during the first RTT we can send one packet, during the second RTT we can send two packets, during the third we can send four, and so on.

Handshaking Inicial: $2 \cdot RTT = 2 \cdot 100 = 200$ ms

a) Data 1000 KB, Bandwidth 1.5 Mbps

$= 2\,RTT + \frac{1000\ KB}{1.5\ Mbps} + \frac{RTT}{2}$

$= 2(100) + \left(\frac{1000\ KB}{1.5\ Mbps} + \frac{100}{2}\right) \approx \frac{8\ Mbit}{1.5\ Mbit}$

$= 0.25\ sec + \left(\frac{8\ Mbits}{1.5\ Mbps}\right)$

$= 5.58\ sec$

Conversion de los megabits:

$\frac{8\,192\,000\ bits}{1\,500\,000} = 5.46$

↓ Por lo tanto

$5.46 + 0.25 = 5.71$ sec //

b) Utilizamos el valor de arriba y anadimos los 99 paquetes (99.9 sec) → $5.71 + 99.9$ sec

$= 105.61$ sec //

c) Paquete 1K

solo 20 por R.T.T

$\frac{1000}{20} = 50\ RTT$

El ultimo 0.5 RTT

$49.5(R.T.T) = 49500$ ms

Tiempo Total = Hand...Inicial + paquetes T.T

$= 200 + 49500$

$= 49700$ ms //

d) 1st R.T.T → 1 paquete

2do RTT → 2 paquetes

3ro R.T.T → 4 paquetes

⋮

Hasta 10 R.T.T.

El ultimo 0.5 R.T.T = 9.5 R.T.T

9 R.T.T. = 950 ms

Tiempo Total = Hand...Inicial + paquete RTT

$= 200 + 950$ ms

$= 1150$ ms //

2. One property of addresses is that they are unique; if two nodes had the same address it would be impossible to distinguish between them. What other properties might be useful for network addresses to have? Can you think of any situations in which network addresses might not be unique?

Pregunta 1

Lo que puede ayudar a que sean unicos:

- Una (entidad / autoridad) que haga una asignacion centralizado
- Una forma para controlar duplicados
- Cambio en la longitud de las direcciones

Pregunta 2

Direcciones que no pueden ser unicas:

- Numeros de telefono
- Nombres de Usuarios: Un usuario puede ser de "Juan" en una pagina, pero el mismo nombre de usuario puede ser de otra persona en otra página

- Nombres de dominio: Algunas páginas puede tener el mismo nombre de dominio, que tienen que distiguirse con la extension: .com / .ec / .org

3. For each of the following operations on a remote file server, discuss whether they are more likely to be delay sensitive or bandwidth sensitive:

    1. Open a file
    2. Read the contents of a file
    3. List the contents of a directory
    4. Display the attributes of a file

(1) Delay sensitive: No requiere de mucha Data

(2) Bandwidth: Algunos archivos pueden ser grandes

(3) Delay sensitive: Los directorios no suelen tener mucha Data de gran tamaño

(4) Delay sensitive: Los atributos no tienen un gran tamaño, incluso pueden ser menor al tamaño del archivo.

4. Suppose that a certain communications protocol involves a per-packet overhead of 100 bytes for headers. We send 1 million bytes of data using this protocol; however, when one data byte is corrupted, the entire packet containing it is lost. Give the total number of overhead + loss bytes for packet data sizes of 1000, 5000, 10000, and 20000 bytes. Which of these sizes is optimal?

S = size

Utilizaremos

$100 \cdot \left(\frac{10^6}{1000}\right) + 1000$

$100 \text{ bits} \times \left(\frac{10^6}{S}\right) + S$

Size	overhead + loss
1000 →	101 000
5000 →	25 000
10 000 →	20 000
20 000 →	25 000

→ 10 000 es lo mas optimo

5. Suppose we want to transmit the message 11001001 and protect it from errors using the CRC polynomial $x^3 + 1$

    1. Use polynomial long division to determine the message that should be transmitted.
    2. Suppose the leftmost bit gets inverted in transit. What is the result of the receiver's CRC calculation?

(1) 11001001

↳ Agregamos 000 y Dividimos por 1001.

1001 / 11001000

↓

Remanente es 11

por lo tanto el mensaje para enviar : 11001001011

(2) Señal recibida: 01001001011

↳ dividimos por 1001

↳ Nos da un

Remanente de 10

↓

Ya que el remanente no es 10

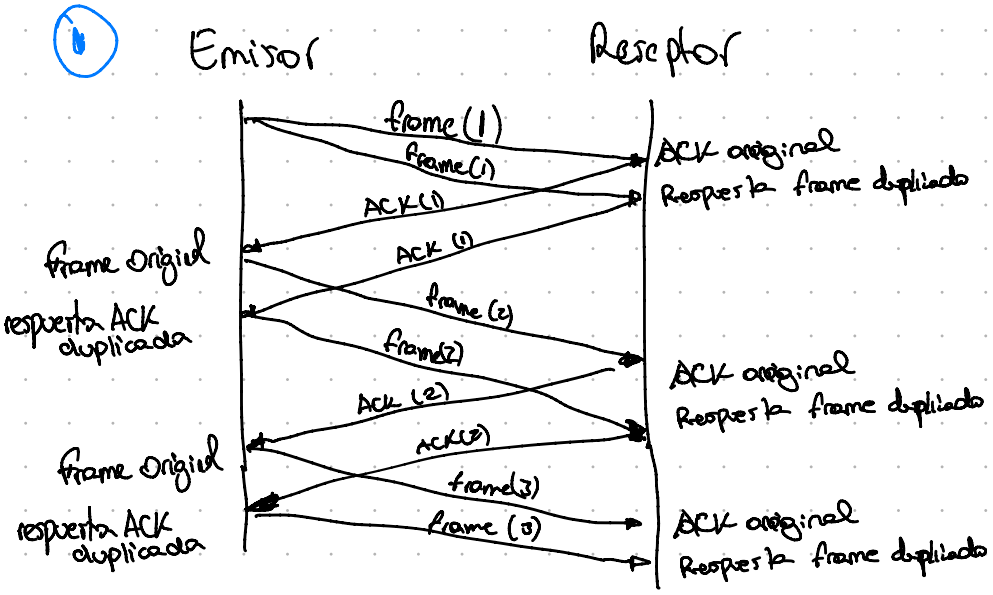
nos dice que hubo un error de bit

6. In stop-and-wait transmission, suppose that both sender and receiver retransmit their last frame immediately on receipt of aduplicate ACK or data frame; such a strategy is superficially reasonable because receipt of such a duplicate is most likely tomean the other side has experienced a timeout.

   1. Draw a timeline showing what will happen if the first data frame is somehow duplicated, but no frame is lost. How long will theduplications continue? This situation is known as theSorcerer's Apprentice bug.

   2. Suppose that, like data, ACKs are retransmitted if there is no response within the timeout period. Suppose also that bothsides use the same timeout interval. Identify a reasonably likely scenario for triggering the Sorcerer's Apprentice bug.

1



2

Para que suceda eso el data duplicado tiene que cruzarse en la red con el ACK previo.

El emisor y receptor tienen que volver a retransmitir con alguna forma de reenvio en el tiempo de espera.

7. Draw a timeline diagram for the sliding window algorithm with SWS = RWS = 4 frames for the following two situations. Assume the receiver sends a duplicate acknowledgement if it does not receive the expected frame. For example, it sends DUPACK[2] when it expects to see FRAME[2] but receives FRAME[3] instead. Also, the receiver sends a cumulative acknowledgment after it receives all the outstanding frames. For example, it sends ACK[5] when it receives the lost frame FRAME[2] after it already received FRAME[3], FRAME[4], and FRAME[5]. Use a timeout interval of about 2 × RTT.

    1. Frame 2 is lost. Retransmission takes place upon timeout (as usual).
    2. Frame 2 is lost. Retransmission takes place either upon receipt of the first DUPACK or upon timeout. Does this scheme reduce the transaction time? Note that some end-to-end protocols (e.g., variants of TCP) use a similar scheme for fast retransmission.