Redes de Datos

1er parcial

Solución

Esta es una posible solución a las preguntas planteadas. ***Por razones didácticas puede tener bastante más información que la mínima necesaria para responder la pregunta de forma suficiente.***

Pregunta 1 (5 puntos)

a) Explique por qué algunas aplicaciones optan por utilizar el protocolo TCP en capa de transporte. Puede tomar como referencia los protocolos SMTP y POP3 vistos en el laboratorio.

TCP es un protocolo orientado a conexión, lo que significa que se pueden identificar 3 fases durante una comunicación que usa TCP.

- *Fase de inicio de conexión: en esta etapa no se envían datos de usuario, porque se está negociando entre las entidades que dialogan*
- *Fase de datos: en esta etapa se intercambian (de forma bidireccional) los datos de usuario*
- *Fase de finalización o corte: se finaliza la conexión y se liberan los recursos destinados a la misma*

TCP es un protocolo es confiable porque garantiza que el envío de información entre las partes se realiza manteniendo el orden de la información enviada, sin pérdidas, sin duplicados y sin errores. Para lograr esto TCP incluye campos en su encabezado para ordenar la información (números de secuencia), para controlar las pérdidas y duplicados (números de secuencia y reconocimiento), para controlar los errores (suma de comprobación de datos y encabezados).
Cuando las aplicaciones requieren confiabilidad en el intercambio, preferirán usar TCP a pesar que la transferencia global pueda resultar más lenta debido a la fase de inicio de conexión y a las eventuales retransmisiones en caso de pérdidas.

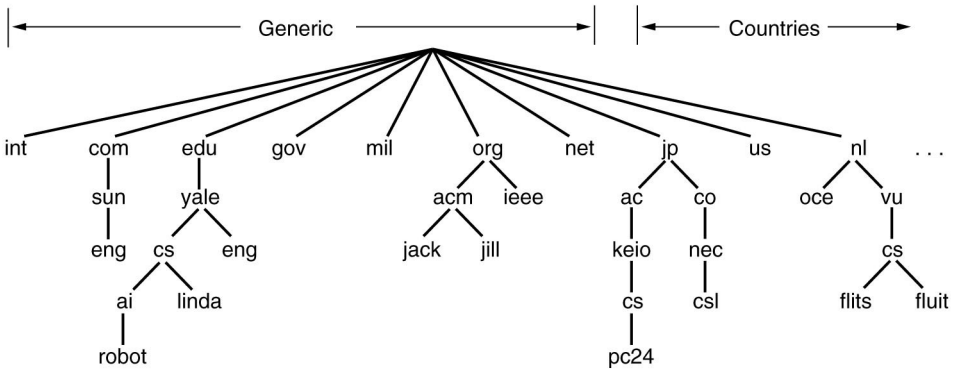
b) Explique por qué algunas aplicaciones optan por utilizar el protocolo UDP en capa de transporte. Puede tomar como referencia el protocolo DNS.

UDP es un protocolo no orientado a conexión y no confiable. Solo se preocupa por la integridad del contenido mediante el campo checksum del encabezado.
Las aplicaciones que necesitan celeridad en el intercambio o que intercambian pocos datos (un segmento para la consulta y una respuesta que puede viajar en un solo segmento, por ejemplo el DNS) preferirán UDP aunque no tengan las garantías de intercambio. UDP al no tener fase de establecimiento y corte como TCP, puede comenzar el intercambio de inmediato, posibilitando una comunicación más ágil con el otro extremo.

Pregunta 2 (12 puntos)

a) Explique cómo es la estructura de nombres en el sistema de nombres de dominio DNS. ¿Qué son las etiquetas o nombres de dominio y cómo se organizan?

El sistema de nombres de dominio surgió ante la necesidad de vincular las direcciones IP que entienden las máquinas con los nombres de dominio más amigables para los humanos, aunque desde sus inicios se utilizó para guardar también otros tipos de información. Originalmente, siendo pocos (del orden de cientos) los dominios existentes, se utilizaban tablas que realizaban este vínculo, pero con el paso del tiempo se hizo necesario un sistema escalable, eficiente, descentralizado, que evitara la saturación de los servidores y la superposición de nombres. Así surgió el DNS.
Cada nombre está compuesto por etiquetas alfanuméricas que convencionalmente se representan unidas por puntos. Estas etiquetas se ordenan en un "árbol" de nombres, partiendo de una «raíz», que son los root servers (son 13 direcciones IPv4 y otras tantas IPv6, con copias distribuidas para dar redundancia), coordinados por la IANA. La raíz convencionalmente se representa por un punto.



Las etiquetas en el DNS se organizan en una estructura de árbol con la raíz hacia arriba. En cada nodo y hoja del árbol pueden existir RR (Resource Records) asociando información a esa etiqueta (ver parte b).
Un FQDN (Fully Qualified Domain Name) se conforma concatenando y separando por puntos (".") desde abajo hacia arriba las etiquetas correspondientes. En el diagrama anterior, por ejemplo fluit.cs.vu.nl. Y pc24.cs.keio.ac.jp serían FQDNs.

El primer nivel bajo la raíz del arbol (root) son los TLD (Top Level Domains) donde se clasifican en dominios genéricos (.com, .edu, mil) y los dominios de dos letras asignados a los países (.jp, .us, .uy, .ar, etc).

Bajo cada TLD se pueden definir subdominios.

b) ¿Qué información se puede almacenar en el DNS? Brinde algunos ejemplos de registros de recursos (RR).

En el DNS se puede almacenar información en lo que se llaman Registros de Recursos o Resource Records (RR).
Hay diferentes tipos, dependiendo de la información que se quiere almacenar: A (address IPv4), AAAA (address IPv6), NS (name server), CNAME (Canonical Name o alias), PTR (pointer), MX (mail exchange), etc.
Los RR tienen 5 campos: etiqueta, tiempo de vida (para el caché), familia (siempre IN de internet), tipo de registro (los indicados anteriormente) y valor.

c) ¿Qué significa que un servidor DNS es autoritativo para un dominio? Explique el concepto de delegación de autoridad de dominios o subdominios.

Para que el sistema de nombres sea escalable es necesario que la información pueda ser gestionada de forma descentralizada y que además se encuentre almacenada de forma descentralizada. Para resolver esta descentralización el DNS prevé que la autoridad que gestiona un dominio pueda delegar la autoridad para administrar sus subdominios.

Por ejemplo. Los administradores de la raíz, delegaron al Servicio Central de Informática de la Universidad de la República (SECIU) la gestión del dominio .uy. Para implementar esa delegación SECIU debe disponer un servidor DNS (con respaldo) donde se almacenarán los registros correspondientes al dominio uy. Este será el servidor autoritativo para el dominio .uy, el servidor que es "dueño" de los datos correspondientes a ese dominio.

La delegación se implementa con el registro NS. En los servidores DNS de la raíz (root servers) se configura un registro NS que asocia al dominio uy el nombre del servidor de nombres al que se le delega.

```
uy.     3600    IN      NS      a.nic.uy.
```

En este caso dice que el dominio uy está delegado al servidor cuyo nombre es a.nic.uy, con una validez de 1 hora (3600 segundos).

En general será necesario también un registro de tipo A que asocie una dirección IP al nombre del servidor, por ejemplo:

```
a.nic.uy.       159415          IN      A       164.73.128.5
```

Para los niveles inferiores al TLD, cada administrador de un TLD puede administrar y/o delegar los subdominios que cree bajo su rama.

Por ejemplo para el dominio uy, SECIU decidió delegar la administración del dominio com.uy a ANTEL y administrar él mismo los dominios .edu.uy, mil.uy, gov.uy, org.uy, etc.

d) Explique los pasos que realiza un servidor DNS recursivo cuando desea encontrar la dirección IP asociada a **www.fing.edu.uy**. Asuma que los dominios **uy**, **edu.uy** y **fing.edu.uy** están administrados por servidores diferentes y que ninguno de ellos responde consultas recursivas. Utilice los nombres y direcciones inventados que necesite.
Para cada paso indique qué se consulta y cuál es la respuesta, qué registros están involucrados y sus valores. Se sugiere hacer un diagrama o tabla para apoyar la explicación.

La consulta DNS se inicia desde el servidor recursivo comenzando por la raíz y siguiendo la cadena de delegaciones hasta encontrar el servidor autoritativo que conoce el valor del registro buscado.
Se asume que no hay información previa últil en ningún caché.
Se presenta a continuación una tabla con la explicación de las consultas y respuestas necesarias:

#	Origen	Destino	Consulta/Respuesta	Observaciones
1	IPSrvRecursivo	IPRoot1	¿Registro A asociado a www,fing.edu.uy?	El servidor recursivo no tiene información en el cache y comienza la búsqueda con un root server. Comenzar por la raíz es la forma más óptima de buscar en un árbol. Las IPs de los root servers las tiene que tener pre-configuradas, si no no puede empezar. Siempre se busca el registro final que se necesita, independientemente que se sepa por ejemplo que los root servers no responden recursivamente. La idea es que no sabemos cómo es la cadena de delegaciones y además alguien podría tener información válida en su caché que sea útil para nuestra búsqueda.
2	IPRoot1	IPSrvRecursivo	Registro NS asociado a .uy es ns.uy Registro A asociado a ns.uy es IP_NSUY	El servidor raíz no tiene la información por la que le consultan. Como los servidores raíz no realizan búsquedas recursivas, responde con la mejor información de la que dispone. Devuelve el servidor de nombres autoritativo de .uy, y agrega el glue record para informar también la dirección IP de ese servidor de nombres. Si no agregara la IP, habría que volver a preguntarle y con el glue-record se evita esa doble consulta.
3	IPSrvRecursivo	IP_NSUY	¿Registro A asociado a www,fing.edu.uy?	Las consultas siempre son por el registro final. Se consulta al servidor de nombres autoritativo de .uy.
4	IP_NSUY	IPSrvRecursivo	Registro NS asociado a .edu.uy es ns.edu.uy. Registro A asociado a ns.edu.uy es IP_NSEDUUY	El servidor de nombres autoritativo de .uy no tiene la información por la que le consultan. Según la letra no responde consultas de forma recursiva, por lo que brinda la información para que se pueda seguir la consulta.
5	IPSrvRecursivo	IP_NSEDUUY	¿Registro A asociado a www,fing.edu.uy?	Las consultas siempre son por el registro final. Se consulta al servidor de nombres autoritativo de edu.uy.
6	IP_NSEDUUY	IPSrvRecursivo	Registro NS asociado a fing.edu.uy es ns.fing.edu.uy. Registro A asociado a ns.fing.edu.uy es IP_NSFING	El servidor de nombres autoritativo de edu.uy no tiene la información por la que le consultan. Según la letra no responde consultas de forma recursiva, por lo que brinda la información para que se pueda seguir la consulta.
7	IPSrvRecursivo	IP_NSFING	¿Registro A asociado a www,fing.edu.uy?	Las consultas siempre son por el registro final. Se consulta al servidor de nombres autoritativo de fing.edu.uy.
8	IP_NSFING	IPSrvRecursivo	Registro A asociado a www.fing.edu.uy es IP_NSWWWFING	El servidor es autoritativo para el registro consultado, por lo que devuelve su valor con su tiempo de vida.

Pregunta 3 (10 puntos)

En una conexión TCP entre **A** y **B**, asuma que:

- El tiempo promedio de ida y vuelta entre **A** y **B** son 20 milisegundos.
- No se pierden segmentos en la red.
- **A** inicia la conexión con **B** usando 100 cómo número de secuencia inicial.
- **A** envía un primer segmento con 400 bytes de datos de usuario.

- **A** espera 1 minuto.
- **A** envía un segundo segmento con 500 bytes de datos de usuario.
- **A** espera 1 minuto.
- **A** finaliza finaliza la conexión.
- **B** elige 800 cómo número de secuencia inicial.
- **B** acusa todos los mensajes de **A** inmediatamente
- **B** no necesita enviar datos de usuario hacia **A** durante la conexión y por tanto finaliza la conexión cuando se entera que **A** desea finalizarla.

a) Realice un diagrama de la secuencia de todos los segmentos intercambiados entre **A** y **B**.
Para cada segmento intercambiado indique: los valores de los números de secuencia y reconocimiento, el valor de las banderas (flags) relevantes, el largo del campo de datos del segmento.

Como se mencionó antes, TCP es orientado a conexión, por lo que antes de enviar datos se requiere realizar la fase de inicio de conexión, luego se pueden intercambiar datos en ambos sentidos (de ser necesario) y luego es necesario finalizar la conexión. En la fase de inicio de conexión se intercambian mensajes de control (no llevan datos de usuario) para garantizar que el otro extremo está disponible y dispuesto a intercambiar información, pero además se pueden negociar parámetros que se utilizarán durante la conexión. En TCP por ejemplo, se acuerdan los números de secuencia que utilizará cada extremo en esa conexión y se pueden negociar opciones.
TCP utiliza un mecanismo de establecimiento conocido como "establecimiento en 3 vías" o "three way handshake" que utiliza 3 mensajes para asegurar el correcto establecimiento de una conexión.

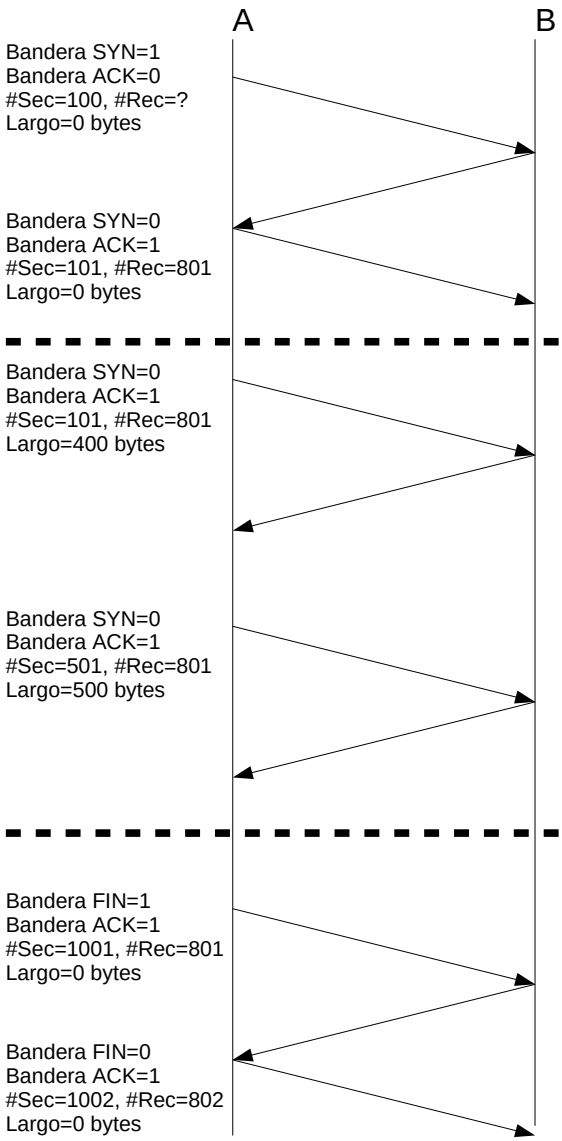
Cuando A quiere establecer una conexión con B envía un primer mensaje que se caracteriza por llevar la bandera SYN=1, y en el campo de número de secuencia un valor de secuencia inicial (aleatorio) de 100 en el problema planteado. La bandera de ACK estará en 0 (indicando que el campo de número de reconocimiento no es válido) ya que no hay nada previo para reconocer, por ser este el primer mensaje de la conexión.
En caso que B esté disponible para establecer esta conexión, B enviará un segmento con la bandera SYN=1 y un número de secuencia 800 a partir del cual se numerarán los bytes enviados por B durante la conexión. Asimismo el segmento llevará la bandera de ACK en 1 y el campo número de reconocimiento contendrá el valor 100+1=101, indicando que el próximo byte que se espera de A tendrá que tener el número de secuencia 101 como número de secuencia. De este modo B reconocerá que el número de secuencia inicial de A es 100. Cuando A reciba este segmento tendrá la confirmación que B aceptó su conexión.

El tercer mensaje originado por A llevará ahora la bandera de SYN en 0 (SYN=1 se utiliza solamente en los dos primeros mensajes de una conexión y va en 0 en todos los restantes segmentos de una conexión), la bandera de ACK=1 (todos los mensajes de una conexión excepto el primero llevan ACK en 1, ya que siempre se enviará un reconocimiento de lo próximo que se espera recibir por si algún reconocimiento previo se perdió). El campo de número de secuencia será 101 (el que espera B) y el campo de reconocimiento será 800+1 (para reconocer a B el número de secuencia que envió en el mensaje anterior).

Se observa que si bien en TCP los números de secuencia numeran los bytes enviados, en el caso de inicio de conexión (y también del fin de conexión) se utiliza un número de secuencia (la numeración avanza 1 pero no se envían aún bytes de datos de usuario). Cuando B reciba este segmento tendrá la confirmación que A aceptó su conexión. En la fase de datos, cuando un segmento no lleva datos de usuario (largo=0 bytes), el número de secuencia no se incrementa.

En la fase de finalización de la conexión se usa la bandera de FIN en 1. Cada sentido de la conexión (A hacia B y B hacia A) se corta de forma independiente. En este caso, B no envía nunca datos de usuario hacia A, por lo que cuando A decide cortar enviando su bandera de FIN=1, B también corta su sentido de la comunicación, respondiendo con FIN=1.

El diagrama completo del intercambio sería el que se muestra en la siguiente figura:



Fase de establecimiento de conexión, no se intercambian datos de usuario y se debe esperar la respuesta de B.
Se consume un número de secuencia aunque no haya datos enviados.
SYN=1 sólo en los dos primeros mensajes.
En el resto de la conexión SYN=0 ACK=0 en el primer mensaje, en el resto va en 1.

Fase de datos, los números de secuencia se incrementan según los datos enviados.
Si no se envían datos, no se incrementan

Fase de corte, usa la bandera FIN.
Al igual que en el inicio, se consume un número de secuencia para acusar la bandera de FIN

Pregunta 4 (5 puntos)

En el protocolo TCP:

a) ¿Por qué es necesario utilizar un el temporizador o “timeout” de retransmisión? ¿Cómo se usa?

Para asegurarse que los datos llegan al destino, TCP incorpora un temporizador de retransmisión. Este se utiliza para determinar cuándo considerar que un segmento no fue reconocido (indicando la pérdida del segmento o su reconocimiento).
Si ese temporizador expira antes de la llegada del segmento de reconocimiento, TCP asumirá que hay una pérdida (que puede ser en el segmento enviado o en su reconocimiento) y decidirá reenviar el segmento. Reenviará todas las veces necesarias hasta tener el reconocimiento del otro extremo.

b) ¿Por qué no se elige un valor fijo para el temporizador de retransmisión?

Como la red puede estas más o menos congestionada y puede haber diferentes caminos con más o menos equipos intermedios entre los extremos, es poco razonable tomar un valor fijo del temporizador. Por este motivo TCP utiliza un temporizador de retransmisión dinámico que se va ajustando al comportamiento de la red en cada momento. La idea es medir el tiempo de ida y vuelta de los segmentos (RTT) y construir un estimador estadístico tanto de la media como de la varianza de ese tiempo. En base a esos estimadores, que se van ajustando de acuerdo al estado de la red, se determina el valor del temporizador. Si hay un escenario de congestión leve, habrá un suave incremento de RTT que llevará a que el transmisor incremente sus temporizadores de retransmisión siendo más paciente en la espera de los ACKs. En los escenarios de congestión severa, no podemos estimar el RTT por la ausencia de ACKs y los temporizadores se ajustan con otra estrategia.

Pregunta 5 (10 puntos)

a) Describa qué es la congestión en una red y cuáles pueden ser sus causas. Explique por qué es necesario realizar control de congestión.

La congestión ocurre cuando se sobrepasa la capacidad de algún elemento en la red, típicamente de algún enlace o la CPU/memoria de algún enrutador de la red. En esos casos, existen más paquetes de los que se puede procesar o que quieren utilizar un mismo enlace, por lo que se utiliza una cola donde almacenarlos temporalmente hasta poder procesarlos. Cuando los paquetes tienen que esperar en esas colas, tardarán más tiempo en llegar a destino e incluso podrían ser descartados en algún enrutador de la red si se supera la capacidad de la cola de atención.
Los paquetes terminan acumulando las demoras ocurridas en cada salto de red a lo largo se su trayecto y los extremos no conocen exactamente dónde ocurre la congestión, solo se percibe el efecto acumulado.

Es importante controlar la congestión porque las demoras mayores a lo esperado en las respuestas a los segmentos TCP o las pérdidas de paquetes (con los segmentos que transportan), generan retransmisiones. Estas se producen porque TCP un protocolo con garantía de entrega y el transmisor retransmitirá cuando considere que la respuesta ya no va a llegar. Si se generan retransmisiones, se estarán generando aún más paquetes en la red, pudiendo ocurrir que en las colas existan paquetes duplicados o que directamente no haya mas lugar en estas colas de atención y los paquetes se descarten. Esto contribuirá a aumentar la congestión en un proceso de realimentación positiva que podría llevar a que la red deje de funcionar casi por completo, debido a la sobrecarga de muchos nodos. Si analizamos la información que envía el transmisor y la que recibe el receptor, veremos que muchas retransmisiones bajan el ancho de banda efectivo (información nueva por unidad de tiempo), y peor aún si debido a los descartes nunca llegan los reconocimientos. En pocas palabras la congestión no permite una utilización eficiente de los recursos (idealmente las retransmisiones deberían ser pocas), y degrada la percepción de los usuarios.

b) ¿Cómo se perciben por parte de los usuarios o de los equipos los diferentes niveles de congestión que pueden existir en la red?

La congestión puede ser “leve” cuando en un período de tiempo aumentan un poco los tiempos de encolamiento en algún nodo, y por tanto el RTT (tiempo de ida y vuelta de los mensajes), pero sin llegar a generarse pérdidas. Este aumento de los retardos será visto por los usuarios finales o por sus aplicaciones como enlentecimiento de transacciones, lentitud en el refresco de pantallas, demoras en culminar las descargas, interrupción de las conexiones en caso de pérdidas excesivas.

Si los transmisores incrementan la tasa de transmisión, la congestión aumenta, y podría suceder que los tiempos de ida y vuelta excedan el temporizador previsto (timeout de retransmisión en TCP) y entonces comiencen a generarse retransmisiones. En este escenario estamos alcanzando congestión “intermedia”

donde tendremos RTT altos, varias retransmisiones y algunos pocos descartes porque no hay lugar en las colas de atención.

Si los transmisores persisten en intentar enviar más información, la congestión aumenta aún más y en algún enrutador de la red podría alcanzarse la capacidad máxima de la cola de procesamiento, ocurriendo descartes de paquetes. En este escenario es muy probable que los nuevos envíos no lleguen al receptor y estaríamos en un estado de congestión "severa". Cuanto mas esfuerzo realizo en transmitir (y retransmitir), llega cada vez menos información nueva al destino.

c) Explique detalladamente cómo se realiza el control de congestión en TCP. Explique las hipótesis de trabajo, qué variable se controla, cómo se controla y cuál es la medida del estado de la red que se utiliza para tomar las decisiones.

TCP asume que hay congestión cuando no le llegan los reconocimientos de los segmentos a tiempo, ya que se considera que los medios físicos actuales son confiables (la tasa de errores en baja) y que no hay pérdidas por otras causas. Esta hipótesis puede no cumplirse en redes donde puede haber mucha interferencia (redes inalámbricas o enlaces distantes con deformación de onda). Para determinar si una respuesta llega o no a tiempo, TCP implementa un temporizador de retransmisión que inicializa con cada segmento que envía. Si ese temporizador expira antes de la llegada del segmento de reconocimiento, TCP asumirá que hay una pérdida debido a congestión. La pérdida puede ser en el segmento enviado o en su reconocimiento, eso no lo puede saber y por eso aparecen los duplicados. Como la decisión de si hay o no congestión queda determinada por el valor del temporizador, es importante que se use un valor acorde al tiempo razonable de ida y vuelta en la red, más algún tiempo de guarda o de tolerancia (ver pregunta 4).

Una vez detectada la congestión, TCP busca disminuir la tasa de transmisión.

TCP intenta ajustar la tasa de transmisión a la capacidad del canal, para esto utiliza una variable ***cwnd*** *en la cual estima la capacidad de la red para que no ocurra congestión. La ventana de congestión es el crédito del transmisor para enviar bytes sin afectar la red, el transmisor lo usará en su totalidad en la medida que tenga suficientes datos para enviar, si la aplicación no envía suficientes datos, usará parte de ese crédito.*
TCP tiene dos modalidades de ajuste del valor ***cwnd*** *dependiendo si considera que está lejos de la congestión (fase de crecimiento lento) o cerca del punto donde puede comenzar a ocurrir congestión (fase de evitar la congestión). Para esto utiliza otra variable llamada* ***ssthresh (umbral).***

¿Por qué una variable y porqué le llamamos ventana?. Es necesario recordar que un transmisor solo esta autorizado a enviar la cantidad de bytes limitada por el tamaño de la ventana de transmisión (W) y luego debe esperar a los reconocimientos que llegaran un tiempo RTT más tarde. Si despreciamos el tiempo de serialización frente al RTT, la tasa (promedio) máxima de transmisión es W/RTT. Si disminuyo W, disminuyo la tasa de transferencia, por lo que disminuyendo el valor de ***cwnd****, se disminuye la tasa de transferencia. Recordar que el transmisor elige el menor valor entre el "Window Size" de control de flujo (anunciado por el receptor en el encabezado) y del estimado de* ***cnwd****, para contemplar ambos efectos. En lo que sigue consideramos que la ventana del receptor es infinita, para no mezclar ambos conceptos.*

En fase de crecimiento lento: la ventana (el valor de ***cwnd****) se incrementa en un byte por cada byte reconocido (recibido el ACK antes de que expire el temporizador). Esto implica que al recibir todos los ACK de los bytes autorizados a transmitir dentro de una ventana* ***cwnd*** *en el paso n podremos duplicarla para el siguiente paso n+1. Este comportamiento exponencial de duplicar la ventana siempre que se reciban todos los reconocimientos, continúa así hasta alcanzar el* ***ssthresh*** *(umbral) donde cambiamos de fase de crecimiento lento a fase de evitar la congestión. El termino "crecimiento lento" no es muy consistente con lo que efectivamente está sucediendo, tiene una razón histórica, existieron alternativas previas con crecimiento más pronunciado.*

En fase de evitar la congestión: el valor de ***cwnd*** *se incrementa en 1 MSS (****MSS*** *Maximum Segment Size) por cada byte transmitido y recibido el ACK antes de que expire el temporizador. Esto implica que al recibir todos los ACK de los bytes autorizados a transmitir dentro de* ***cwnd****, el valor se de* ***cwnd*** *crece en* ***1 MSS*** *para el siguiente paso. Esto continúa así hasta que se detecte la congestión (expire el timeout) o se alcance la ventana del receptor "window size".*

Cuando se detecta la congestión, TCP disminuye el valor de ***cwnd*** *y el valor de* ***ssthresh*** *a la mitad del valor de* ***cwnd*** *de donde ocurrió la congestión. Hay variantes de a qué valor disminuir, pero la variante original de TCP (Tahoe) el* ***cwnd*** *baja a 1 MSS (como al inicio) y retoma una fase de crecimiento lento. Con esto se logra que el transmisor disminuya su tasa de transmisión a la espera que la red pueda liberar espacio en las colas de atención en los routers, disminuir el valor de la variable* ***ssthresh****, fuerza a que entremos antes en la fase de evitar la congestión.*

Como se muestra en la siguiente figura, la evolución del tamaño de ventana del transmisor, va comportándose como un diente de sierra tratando de ajustar la ventana al valor óptimo que le permita usar la red al máximo de su capacidad, sin llegar a congestionarla (o apenas tocando la congestión cada tanto).