25 de setiembre de 2023

Redes de Datos 1
1er parcial – 2023
Solución

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

Ud se encuentra de viaje en Italia y su abuela lo llama desde Montevideo:
- ***¡No puedo leer el correo!*** – le suplica con voz angustiada.
- ***¿Qué mensaje de error te da?*** – contesta Ud. recordándole que está estudiando "Redes 1" como forma de tranquilizarla.
- ***Sólo me dice: "no se puede acceder a su solicitud"*** – le dice su abuela, logrando desconcertarlo.
- ***No se preocupe abuelita, ¡yo lo solucionaré!*** – responde con confianza, mientras verifica que tiene descargados los materiales del curso y se apresta a encontrar soluciones.

A continuación abre una terminal e intenta descubrir qué está pasando.

Pregunta 1 (7 puntos)

a) Explique por qué un modelo basado en capas podría ayudarlo a encarar el problema. ¿Qué ventajas tiene usar un modelo de este tipo, particularmente en el contexto de las redes de comunicaciones?

En las redes de comunicaciones están involucrados problemas de distinta índole: diferentes tecnologías de capa física y capa de enlace, distancias entre dispositivos, implementaciones de las aplicaciones. El modelo basado en capas permite dividir un problema complejo en problemas más acotados y más abordables, simplificando el diseño y la implementación de las redes de datos.

Dicha simplificación se logra mediante la división en distintos niveles de abstracción y la independencia entre dichos niveles. Cada capa o nivel, realiza un conjunto de funciones que ofrecen servicios a la capa superior. La implementación de cada una de esas funciones es transparente para las otras capas por lo que mediante una correcta definición de las interfaces entre las capas se logra una verdadera independencia. Dicho modelo permite diseñar e implementar considerando que la comunicación se da horizontalmente entre entidades pares de los dispositivos involucrados, aunque el flujo real de datos se da en sentido vertical y no horizontal.

b) Explique cuál es el cometido principal de cada capa del modelo analizado en el curso. Destine no más de un párrafo para cada capa.

La capa física se encarga de la transmisión de bits por un canal de comunicaciones. Abarca la especificación de conectores y cables, representación de bits con señales, reconocimiento de bits, etc.

La capa de enlace se encarga de controlar los errores siempre presentes en cualquier medio físico. Agrupa los bits en tramas y hace control de errores sobre las tramas (corrección o detección), para lo que necesita agregar bits de redundancia. Si se necesita una comunicación confiable puede agregar números de secuencia, reconocimientos, puede implementar ventanas deslizantes y puede implementar control de flujo. En las redes que lo necesitan, realiza el control de acceso a medios compartidos (sub-capa MAC).

La capa de red se encarga de encontrar un camino en la red para llegar desde un origen a un destino, utilizando los recursos de la red (enrutadores y enlaces disponibles o sea la topología de la red). En caso de existir múltiples caminos debe elegir el mejor camino en función de alguna métrica de interés. Se identifican la función de ruteo (encontrar el camino) y la función de forwarding (lo que tiene que hacer cada enrutador para que los paquetes sigan el camino elegido).

El usuario final o los equipos finales no tienen control de los problemas que pueda presentar la capa de red, porque puede haber muchos proveedores de servicio involucrados en una comunicación entre máquinas distantes. Cada proveedor opera sus equipos y resuelve sus problemas. La capa de transporte brinda una comunicación de extremo a extremo bajo control de los equipos finales. Puede incorporar funciones de control de errores, control de secuenciamiento, control de flujo y control de congestión. Puede brindar servicios orientados a conexión o no orientados a conexión. Puede brindar servicios confiables y/o no confiables.

La capa de aplicación alberga las aplicaciones con las que el usuario finalmente interactúa y utiliza los servicios brindados por la capa de transporte de acuerdo a sus requerimientos. Ejemplos de aplicaciones: navegador, agente de correo, juegos en línea, video conferencias, etc, etc.

Pregunta 2 (10 puntos)

¿Será un problema con el DNS?

a) ¿Qué es un registro en el DNS y cómo se compone? La dirección de correo de su abuela es **tata@adinet.com.uy** ¿qué registro(s) del DNS necesita averiguar para identificar cuál es el servidor de correo de su abuela y poder verificar el posible problema? ¿Qué información contienen en particular los registros que necesita?

Un registro en el DNS asocia un tipo y un valor a un nombre. Típicamente contiene 5 campos: etiqueta (nombre por el que se consulta), tipo (tipo de registro, que indica qué información se agrega en "valor"), ttl (tiempo de vida del registro), clase (clase de registro, en la práctica actual siempre es IN) y valor (contenido del registro).
El registro MX (mail exchanger) indica cuál es el nombre del servidor que recibe correo para un cierto dominio, por lo tanto para saber cuál es el servidor de correo para una dirección tata@adinet.com.uy, necesitamos el registro MX asociado al dominio adinet.com.uy. Supongamos que es mail.adinet.com.uy, entonces el registro será de la forma (el valor de los registros de tipo MX tiene un valor de prioridad, ya que puede haber más de un servidor de correo para un dominio, y un nombre):

adinet.com.uy IN MX TTL 10 mail.adinet.com.uy

Además para realizar cualquier verificación de conectividad con ese servidor vamos a necesitar la IP asociada a ese nombre, por lo que necesitaremos el registro A (Address) asociado a ese nombre, lo que se escribiría en el siguiente formato (los valores son inventados):

mail.adinet.com.uy IN A TTL 200.40.0.10

b) Asumiendo que los dominios de primer nivel (TLDs) no aceptan consultas recursivas y que los servidores autoritativos del domino "com.uy" también son autoritativos de "adinet.com.uy", explique detalladamente cómo su servidor recursivo local realiza la búsqueda de los registros identificados en la parte a). Detalle la secuencia de consultas y respuestas en las que participa el servidor recursivo local, incluyendo los servidores involucrados, así como los datos consultados y recabados con esas consultas. Utilice nombres y direcciones de fantasía para los servidores requeridos en el ejemplo.

Se presenta a continuación una posibilidad de respuesta para esta parte del ejercicio. Existen otras variantes que también pueden ser consideradas correctas.

Sec	IP origen	IP destino	Consulta/Respuesta	Comentarios
1	IP pc1	IP dnslocal	¿Registro MX de adinet.com.uy?	pc1 inicia la búsqueda consultando a su servidor DNS local.
2	IP dnslocal	IP root	¿Registro MX de adinet.com.uy?	dnslocal es recursivo, por lo cual se encarga de la búsqueda. La búsqueda comienza por root porque no se tiene nada guardado en el caché.
3	IP root	IP dnslocal	Registro NS de .uy (ns.uy) Registro A de ns.uy	El servidor raíz root no es recursivo. Devuelve el registro NS, y el registro A de ns.uy como glue record.
4	IP dnslocal	IP nsuy	¿Registro MX de adinet.com.uy?	
5	IP nsuy	IP dnslocal	Registro NS de .com.uy (ns.com.uy) Registro A de ns.com.uy	Según la letra los TLDs (top level domains) no son recursivos. ns.uy no es autoritativo de adinet.com.uy.
6	IP dnslocal	IP nscomuy	¿Registro MX de adinet.com.uy?	
7	IP nscomuy	IP dnslocal	Registro MX de adinet.com.uy (correo.adinet.com.uy)	ns.com.uy es autoritativo de adinet.com.uy, por lo que nos devuelve la información solicitada.
8	IP dnslocal	IP pc1	Registro MX de adinet.com.uy	Termina la búsqueda inicial.
9	IP pc1	IP dnslocal	¿Registro A de correo.adinet.com.uy?	pc1 necesita la dirección IP asociada al servidor de correo para comunicarse con él.
10	IP dnslocal	IP nscomuy	¿Registro A de correo.adinet.com.uy?	Se tiene guardada en caché el registro NS de adinet.com.uy, y la dirección de ns.com.uy.
11	IP nscomuy	IP dnslocal	Registro A de correo.adinet.com.uy (IP de correo.adinet.com.uy)	Como ns.com.uy es autoritativo de adinet.com.uy, nuevamente nos devuelve la información solicitada.
12	IP dnslocal	IP pc1	Registro A de correo.adinet.com.uy	Termina la búsqueda.

c) El valor de TTL (time to live) de los registros obtenidos en a) y b), es de 300 segundos. ¿Para qué sirve y cómo se usa el TTL en los registros DNS?

El TTL (time to live) es el tiempo de validez de un registro DNS que se utiliza para saber por cuánto tiempo puede almacenarse ese valor en el caché de los equipos que consulten por ese registro. El caché sirve para evitar consultas innecesarias (si ya se consultó ese registro y aún es válido en su caché, puede usarse sin necesidad de hacer una nueva consulta) y también limita el tiempo durante el cual es válida cierta información, lo que permite evitar errores ante cambios en el valor del registro.

d) Si ud. tuviera que configurar un servidor de DNS, ¿en qué escenarios o situaciones se inclinaría por valores de TTL bajos (de 300 segundos como en el ejemplo) y en qué casos usaría valores altos (24 horas, por ejemplo)? Explique en cada caso las implicancias desde el punto de vista de los servidores de DNS y de los clientes o servidores que hacen consultas.

Los TTL bajos se suelen utilizar para información que cambia muy seguido. Algunas consideraciones que resultan de TTL bajos son el hecho de que casi nunca se va a tener información en el caché, ya que la misma es válida por muy poco tiempo (lo que aumenta la cantidad de búsquedas realizadas) y que se tienen mayores retardos al realizar las consultas, pues se debe realizar la búsqueda. Aumenta la cantidad de consultas que llegan a los servidores autoritativos de la zona.
Los TTL altos se suelen utilizar para información que cambia poco. Permiten aprovechar el caché, acelerando el funcionamiento de las consultas y disminuyendo la cantidad de consultas necesarias. Sin embargo, en caso de cambios en la red, podría quedar información incorrecta o desactualizada en el caché durante un tiempo significativo.
Los valores de TTL de un registro se pueden modificar cuando se sabe que será necesario cambiarlo en el futuro. Por ejemplo si un nombre de dominio tiene un registro A asociando la IP1 con un TTL de 24 horas y se sabe que va a ser necesario cambiar IP1 por IP2, al menos 24 horas antes de hacer el cambio hay que bajar el TTL a un valor más bajo (5 minutos por ejemplo), de modo que una vez realizado el cambio del valor del registro A, no queden copias del valor anterior en los cachés de otros servidores.

Pregunta 3 (10 puntos)

¿Será un problema de capa de transporte?

a) Como resultado de la pregunta 2, Ud logró identificar el servidor de correo que utiliza su abuela y recuerda que Ud mismo le había configurado el cliente de correo con el protocolo POP3. ¿Qué información de direccionamiento necesita para poder comunicarse con el servidor POP3 de su abuela? Justifique su respuesta.

En las redes TCP/IP (como internet), para poder comunicarse con cualquier servicio requerimos conocer la dirección de capa de red, lo que nos permite encaminar paquetes hacia el servidor correspondiente, el protocolo de capa de transporte (por ejemplo TCP o UDP), y la dirección o identificador en la capa de transporte (el puerto correspondiente al servicio, en este caso POP3).

b) Ud recuerda que el protocolo POP3 utiliza TCP en capa de transporte y que, según lo visto en el curso, TCP es orientado a conexión y confiable. ¿Qué implica que TCP sea orientado a conexión? ¿Por qué se considera TCP un protocolo confiable? ¿Qué elementos incorpora TCP para lograr esa confiabilidad? Justifique su respuesta.

Un protocolo es orientado a conexión cuando se identifican 3 fases para el intercambio de información: inicio de conexión, intercambio de datos y finalización de 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 como la escala de ventana (window-scale) y el tamaño máximo de segmento (MSS).
Al establecer la conexión el transmisor y el receptor reservan recursos (por ejemplo buffers de memoria) para la conexión. Una vez establecida la conexión, se puede pasar a la fase de datos donde se envían y reciben los datos del usuario. Una vez que el intercambio finalizó, se debe finalizar la conexión para que ambos extremos liberen los recursos asignados (buffers de memoria). En el caso de TCP esta finalización puede realizarse de forma independiente por ambos extremos, pudiendo finalizar el envío de datos en uno de los sentidos y mantenerse en el otro.

Decimos que un protocolo es confiable cuando 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).
TCP incorpora también temporizadores, en particular 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)
También implementa políticas de control de flujo (para evitar pérdidas por incapacidad del receptor) (mediante el campo window size del encabezado) y políticas de control de congestión (para evitar pérdidas por congestión de la red).

c) Explique detalladamente cómo se realiza el inicio de una conexión en TCP, indicando los campos del encabezado que se utilizan, su función y cómo se usan.

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. Imaginemos que A quiere establecer una conexión con B.

El primer mensaje enviado por A se caracteriza por llevar la bandera SYN=1, y en el campo de número de secuencia un valor de secuencia inicial X (elegido a partir del valor del reloj del equipo y una componente aleatoria por seguridad). 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, al ser este el primer mensaje de la conexión. En el campo de opciones de ese primer segmento, pueden agregarse las opciones y valores que A soporta y está dispuesta a negociar (como MSS, soporte de Timestamp, Window Scale, etc).

Como respuesta a ese primer mensaje, en caso que B esté disponible para establecer esta conexión, este enviará un segmento con la bandera SYN=1 y un número de secuencia Y 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 X+1, indicando que el próximo byte se espera tenga X+1 como número de secuencia. De este modo B reconocerá que el número de secuencia inicial de A es X. 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), 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 X+1 (el que espera B) y el campo de reconocimiento será Y+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.

d) Recordando lo realizado en el laboratorio y para refrescar sus habilidades con el wireshark, decide realizar una captura del tráfico mientras utiliza el comando **telnet** o **netcat** (nc), ¿cómo probaría si el servidor de correo está respondiendo? ¿Cómo concluiría que pudo establecer una conexión exitosa con el servidor de correo? Especifique qué información de los segmentos intercambiados le permitiría llegar a esa conclusión. Justifique.

Para probar que el servidor de correo está atendiendo conexiones se puede hacer:

telnet IP_servidor Puerto_POP3

o

nc IP_servidor Puerto_POP3

utilizando la dirección IP (o el nombre del servidor de correo) y el número de puerto del protocolo POP3.

Si la conexión es exitosa, en el wireshark deberían verse los 3 segmentos de inicio de conexión de acuerdo a lo comentado en la parte anterior (banderas de SYN, ACK, números de secuencia y reconocimiento). Adicionalmente en este protocolo, seguramente tengamos un mensaje del servidor esperando por el ingreso de los datos del usuario (USER, PASS, etc).

Pregunta 4 (9 puntos)

Ante la desilusión de no poder encontrar aún la solución al problema de su abuela, Ud. realiza la consulta a un colega que le sugiere que podría haber un problema de congestión en la red.

Observación: *A lo largo de la solución encontraran referencias a varias notas aclaratorias que pretenden reforzar algunos conceptos que suelen ser mal interpretados y que exceden el alcance de las respuestas esperadas.*

a) ¿Qué se entiende por congestión en una red? ¿Cómo se detecta la congestión? ¿Puede haber diferentes niveles o grados de congestión? Justifique su respuesta.

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. La noción de demora acumulada de los paquetes es más extensa, involucra por ejemplo demora en dar ACK del receptor, demora de análisis en Firewall u otros fenómenos que puede darse en las capas inferiores.

De forma genérica es difícil detectar la congestión, porque de alguna forma debemos tener capacidad de compararnos contra el escenario normal. En capa 3 podemos detectar la congestión observando en nivel de utilización de los enlaces y de las colas de atención de paquetes. En capa 4, el protocolo podría implementar estrategias de detección de congestión, que en el fondo son básicamente detecciones de la variación de la tasa de intercambio de datos (por ejemplo en TCP no recibir ACK antes de que expire el temporizador se asume como un indicador de congestión). En capa 5 las aplicaciones detectarán la congestión percibiendo variaciones en la tasa de transferencia de datos, llegando en caso extremo a la ausencia de transferencia de datos.
Por completitud del tema, y si bien aún no fue visto en el curso, también pueden haber problemas en capa 2 que podrían percibirse como congestión, no suelen ser relevantes salvo en casos particulares, por lo general donde la capa física es inalámbrica.

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. 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 al destino (o reconocimientos al transmisor). (ver Nota 1 y Nota 2 en anexo).

b) ¿Sería posible que un problema de congestión afectara a su abuela en Montevideo pero no a Ud. en Italia? Justifique su respuesta.

Es posible que uno perciba congestión y el otro no, la congestión o el efecto percibido por congestión, es el acumulado por el trayecto que están siguiendo los paquetes de ida y vuelta, y la respuestas del servidor de correos. Dependiendo de en qué parte de la red ocurre la congestión, puede afectarse uno de los caminos y no el otro. (ver Nota 3 en anexo).

c) ¿Cómo implementa TCP el control de congestión? Explique cómo TCP detecta la congestión y cómo actúa para controlarla.

TCP asume que hay congestión cuando no le llegan los reconocimientos de los segmentos a tiempo (vence el temporizador de retransmisión), ya que asume 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 (por ejemplo redes inalámbricas). 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 (que puede ser en el segmento enviado o en su reconocimiento, o solo una demora de más de lo esperado en las colas de atención en los enrutadores) debido a congestión.

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 (fase de crecimiento lento) o cerca del punto donde puede ocurrir la congestión (fase de evitar la congestión). Para esto utiliza otra variable llamada **ssthresh**.*

*¿Por qué una variable y porqué le llamamos ventana?. Recordar que un transmisor solo esta autorizado a enviar W bytes y luego debe esperar a los ACKs que llegaran un tiempo RTT más tarde. Si despreciamos el tiempo de serialización frente al RTT, la tasa (media) 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) y del estimado de **cnwd**, para contemplar ambos efectos.*

*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 **cwnd[n]**, en un determinado instante, el valor se duplica **cwnd[n+1] = 2 cwnd[n]** (n y n+1 es una simplificación de la complejidad que corresponde a la transmisión de todos los bytes dentro de una ventana en un determinado instante n y luego cuando llegan todos los ACK, un RTT más tarde, evoluciona a cwnd[n+1]). Esto ocurre hasta alcanzar el **ssthresh** 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 **MSS/cwnd[n]** (**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[n], en un determinado instante, el valor se de **cwnd** crece en **1 MSS**, **cwnd[n+1] = cwnd[n] + 1 MSS**. Esto ocurre hasta que se detecte la congestión o se alcance la ventana del receptor "window size".*

*Cuando se detecta la congestión, TCP disminuye el valor de **cwnd[n]** y el valor de **ssthresh** a la mitad del valor de **cwnd** de donde ocurrió la congestión. Hay variantes (ver Nota 4 en anexo) de a qué valor disminuir, pero consideraremos la variante original de TCP, donde **cwnd[n] = 1 MSS** y retoma en 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.*

Falta mencionar el punto de comienzo del algoritmo, normalmente se comienza desde valores fijos, por ejemplo cwnd[0] = 1 MSS y ssthresh[0] = 32 MSS (observar que es fase de crecimiento lento), aunque hay variantes.

La detección de congestión está basada en si recibo o no el ACK antes de que expire el temporizador de retransmisión y por tanto 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). 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. Formalmente esto ocurre de forma general en ausencia de ningún ACK. En el laboratorio vimos que duplica el último valor de timeout con los re-envíos hasta alcanzar un valor máximo de 120 segundos. Luego re-intenta cada 120 segundos.
Ver Nota 5 en anexo.

d) Si efectivamente estuviéramos ante un problema de congestión de la red, ¿cómo afectaría la misma a un protocolo que utilizara UDP como transporte en lugar de TCP?

La pregunta asume que el tráfico TCP y UDP comparten el trayecto que está ocasionando la congestión. Si nos concentramos en los enrutadores que tienen colas de atención de paquetes, los efectos de demora por no estar en el primer lugar o de descartes siguen ocurriendo, por lo que el tráfico UDP también percibirá el efecto de la congestión. Los routers encolan y descartan paquetes, sin preocuparse (lo podrían hacer) si es tráfico UDP o TCP el que llevan esos paquetes.

Lo que resulta más relevante es que UDP carece de la capacidad de detección de congestión o de control de congestión, por lo cual puede no reaccionar frente a la misma. Esto se traduce en que no disminuye su tasa de transmisión por congestión. Recuerde que al no haber ACK o alguna realimentación, es posible que transmisor (tanto la aplicación como la capa de transporte del transmisor) no tenga consciencia de que está ocurriendo la congestión.

Cuando ocurre congestión TCP decrementará su tasa de transmisión pero el tráfico UDP no lo hará, pudiendo incluso aprovecharse de la capacidad liberada por las conexiones TCP que si bajan su tasa. Parece un escenario "poco justo" y de hecho lo es.
Ver Notas 6 a 9 en el anexo.

Pregunta 5 (7 puntos)

Ud. consulta a otro colega que le asegura que el problema está en capa de red, posiblemente en las tablas de forwarding o en el mecanismo de NAT/PAT (Network Address Translation/Port Address Translation), habitual en las conexiones hogareñas.
Ud no coincide con ese diagnóstico, ya que si eso fuera cierto, habría problemas para conectarse también a otros sitios de internet y no solamente al servidor de correo.
Intente convencer a su colega, explicándole cómo están configuradas las tablas de forwarding en la red de su abuela y cómo funciona el NAT/PAT.

a) Explique qué información contienen y cómo se usan las tablas de forwarding. Escriba las tablas de forwarding mínimas que necesitan la computadora de su abuela y el Router NAT/PAT de la figura para que su abuela pueda acceder a sitios de internet. En la figura aparecen las direcciones IP de las interfaces de los equipos y los rangos de direcciones utilizados.

Las tablas de forwarding se utilizan en cada equipo para saber cómo llegar a direcciones IP con las que necesite intercambiar información. Estas tablas son necesarias en los equipos finales cuando envían paquetes y en los enrutadores cuando tienen que re-enviar paquetes hacia otros destinos. Para cada paquete que el equipo tenga que enviar, consulta la tabla de forwarding utilizando el algoritmo Longest-prefix-match.
Entonces, las entradas de las tablas de forwarding tienen: Destino (rango de direcciones especificado con número de red y máscara de la red) y próximo salto para ese destino.
Para cada paquete que sea necesario encaminar, el equipo utilizará el algoritmo Longest-prefix-match para buscar en su tabla de forwarding la mejor entrada para la IP de destino de ese paquete y de ahí obtener el próximo salto a usar para encaminarlo. La mejor entrada será el menor rango de direcciones destino que contenga la IP buscada. Para esto los rangos de direcciones destino de la tabla se ordenan desde los rangos más pequeños (con menos direcciones o con máscara más larga) hacia los rangos más generales (con más direcciones o con máscaras más cortas). De este modo, si comenzamos la búsqueda por las primeras entradas, encontraremos primero (y preferiremos usar) los rangos más pequeños (más específicos) que contengan la IP buscada pudiendo más adelante en la tabla haber otros rangos que también contengan la IP buscada. Para verificar si una dirección IP pertenece a un rango, se realiza el AND bit a bit entre la dirección buscada y la máscara de la entrada con la que se está comparando. Si el resultado es igual al número de red de ese rango en esa entrada, entonces la IP buscada pertenece a ese rango y se utilizará el próximos salto asociado a esa entrada para encaminar el paquete. Una vez encontrada una coincidencia, no se continúa recorriendo la tabla. Si no se encuentra coincidencia en una entrada, se prueba con la siguiente (que será posiblemente más genérica por tener máscara meor o igual). Si se llega al final de la tabla sin haber encontrado una coincidencia, se descarta el paquete y se envía al originador del mismo (dirección origen del paquete) un mensaje ICMP del tipo "network unreacheable". Pueden existir rutas por defecto especificadas con el destino 0.0.0.0/0 (que matchea con

todas las IPs) para asociar un próximo salto a todos los paquetes que no hayan encontrado coincidencias anteriormente. Esta ruta por defecto, al tener máscara de largo 0 será la última de la tabla.
Los equipos tendrán entradas en su tabla de forwarding para las redes directamente conectadas a él (que se crean cuando se configuran las interfaces) y en general tendrán otras entradas para redes no directamente conectadas y/o una ruta por defecto.
Cuando la red es sencilla como en el caso del ejemplo, conviene usar rutas por defecto y la configuración mínima necesaria sería la siguiente:

Tabla de Abuela	
Destino	***Próximo salto***
10.10.10.0/24	*Directamente conectada*
0.0.0.0/0 (default)	*10.10.10.1 (al Router)*

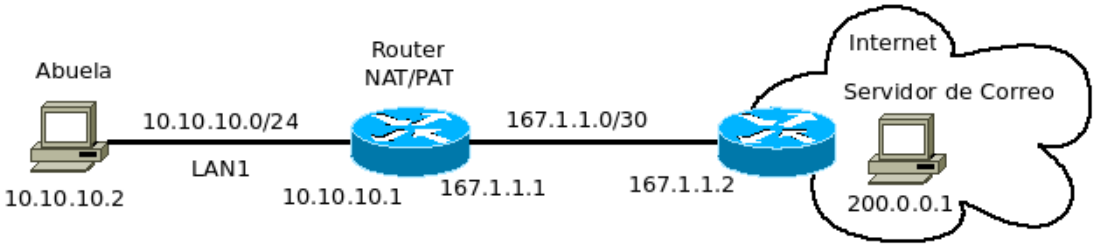
Tabla de Router NAT/PAT	
Destino	***Próximo salto***
10.10.10.0/24	*Directamente conectada (interfaz interna)*
167.1.1.0/30	*Directamente conectada (interfaz externa)*
0.0.0.0/0 (default)	*167.1.1.2 (al Router del Proveedor)*

En este caso, alcanza con que los equipos conozcan sus redes directamente conectadas y tengan una ruta por defecto para el resto del tráfico.

b) ¿Cuál es el problema principal que intenta resolver el mecanismo de NAT/PAT? ¿Cómo trata de solucionarlo?

El problema principal que motiva la creación del sistema de NAT/PAT es la escasez de direcciones IPv4. Por una mala asignación inicial (sin usar máscaras y usando las clases A, B y C predefinidas) y por el crecimiento de los dispositivos que necesitan conexión a la red, se comenzaron a agotar las direcciones IPv4. La solución de fondo al problema fue la creación de un nuevo protocolo IPv6 que aumenta considerablemente el espacio de direcciones, pasando de direcciones de 32 bits (IPv4) a direcciones de 128 bits (IPv6). Pero hubo necesidad de implementar un mecanismo como NAT/PAT para que la red pudiera seguir incorporando nuevos dispositivos hasta que IPv6 estuviera disponible.
El mecanismo propone la definición de rangos de direcciones exclusivos para uso interno de las redes (direcciones privadas). Estas direcciones privadas (a diferencia de las direcciones públicas) se pueden repetir en diferentes redes ya que solo tienen alcance local, no siendo posible su uso en la internet pública. Esto permite que las redes internas de una institución u hogar tengan direcciones para todos los dispositivos que necesitan y los equipos puedan usar el protocolo IPv4 con estas direcciones privadas para comunicarse internamente.
Cuando un dispositivo de una red interna necesita comunicarse con un dispositivo de la red pública (con una IP pública), necesitará realizarse una traducción o mapeo entre las IPs privadas y las públicas.
Para poder resolver el problema de forma general para cualquier tipo de comunicación TCP o UDP y para evitar posibles situaciones puntuales donde dos equipos internos se están comunicando simultáneamente con la misma IP pública destino y con el mismo puerto destino, se utiliza una traducción tanto de direcciones IPs como de puertos.

c) Explique detalladamente cómo funciona el mecanismo de NAT/PAT. Tomando los valores de la figura, detalle los valores de los campos relevantes de los encabezados de TCP e IP, para los dos primeros segmentos/paquetes de una conexión exitosa desde la máquina de su abuela al servidor de correo, tanto en LAN1 como en el enlace a Internet de su abuela.



El mecanismo de NAT/PAT implica que el enrutador de una red que usa direcciones privadas deba traducir las direcciones privadas en direcciones públicas cuando los paquetes pasan por él.
Cuando la aplicación de correo electrónico instalada en el equipo de la Abuela intenta leer el correo almacenado en su servidor de Correo, necesitará establecer una conexión TCP al puerto de POP3 de dicho servidor.
Para eso, realizará un inicio de conexión TCP con los siguientes campos:
- IP de origen: 10.10.10.2
- IP de destino: 200.0.0.1
- puerto de origen: efímero mayor a 1024 elegido por el sistema operativo del equipo de la abuela, ej 1025
- puerto de destino: puerto del servidor POP3 (por ejemplo 110 como se vio en el laboratorio)
Ese segmento TCP será encaminado por el equipo de la Abuela, consultando su tabla de forwarding, hacia el enrutador NAT/PAT de su red.
Cuando ese paquete llegue al router NAT/PAT, deberá encaminarlo hacia el servidor de correo (usando su tabla de forwarding sabrá que el próximo salto es la IP 167.1.1.2) y además deberá traducir sus direcciones para evitar que las direcciones privadas salgan a internet (donde no tienen sentido porque están repetidas en otras redes).
En principio se podría traducir solamente la IP de origen, saliendo el paquete por la interfaz hacia internet con los siguientes campos:
- IP de origen: 167.1.1.1 (la IP pública del router NAT/PAT)
- IP de destino: 200.0.0.1
- puerto de origen: 1025
- puerto de destino: 110
Si existiera otro equipo interno en la red de la Abuela que quisiera simultáneamente conectarse al mismo servidor de correo con el mismo puerto y por casualidad ambos equipos usaran el mismo puerto efímero como puerto de origen, las respuestas del servidor a esos dos segmentos serían indistinguibles cuando lleguen al router NAT/PAT ya que los campos serían idénticos en ambos casos:
- IP de origen: 200.0.0.1
- IP de destino: 167.1.1.1
- puerto de origen: 110
- puerto de destino: 1025
Por este motivo, el router NAT/PAT cambia también los puertos de origen por valores que él mismo elige y que no repite (2001 en el ejemplo).
El router NAT/PAT para poder saber a qué equipo interno tiene que hacer llegar las respuestas necesita crear una tabla dinámica con el mapeo de los valores de los campos traducidos para las conexiones establecidas.
Entonces los paquetes que se verían para los dos primeros intercambios tendrían los siguientes encabezados:

#	**Red privada**				**Enlace a internet**			
	IP		**TCP**		**IP**		**TCP**	
	Origen	**Destino**	**Origen**	**Destino**	**Origen**	**Destino**	**Origen**	**Destino**
1	10.10.10.2	200.0.0.1	1025	110	167.1.1.1	200.0.0.1	2001	110
2	200.0.0.1	10.10.10.2	110	1025	200.0.0.1	167.1.1.1	110	2001

El router NAT/PAT deberá guardar en su tabla de NAT/PAT la información que utilizó para traducir el segmento #1, de modo de poder realizar consistentemente la traducción de los campos del segmento #2 en el sentido opuesto.

Es importante aclarar que para cada comunicación TCP o UDP de cada máquina interna, se utilizan diferentes puertos de origen en el enrutador NAT/PAT. A continuación se muestra con una tabla ejemplo cómo se traducirían diferentes segmentos originados “simultáneamente” por la máquina de la abuela y por otra ficticia dentro de la misma red (IP 10.10.10.3) hacia diferentes destinos. Para simplificar solo se muestran los segmentos salientes de la red y no las respuestas.

	Red privada				***Enlace a internet***			
Proto	***IP***		***Puertos***		***IP***		***Puertos***	
	Origen	***Destino***	***Origen***	***Destino***	***Origen***	***Destino***	***Origen***	***Destino***
TCP	*10.10.10.2*	*200.0.0.1*	*1025*	*110*	*167.1.1.1*	*200.0.0.1*	*2001*	*110*
UDP	*10.10.10.2*	*200.0.0.1*	*1026*	*53*	*167.1.1.1*	*200.0.0.1*	*2010*	*53*
UDP	*10.10.10.3*	*8.8.8.8*	*1026*	*53*	*167.1.1.1*	*8.8.8.8*	*2011*	*53*
TCP	*10.10.10.3*	*8.8.8.8*	*1025*	*110*	*167.1.1.1*	*8.8.8.8*	*2012*	*110*
TCP	*10.10.10.3*	*8.8.8.8*	*1026*	*110*	*167.1.1.1*	*8.8.8.8*	*2013*	*110*

Anexo Pregunta 4 (Información complementaria)

Nota 1: *La identificación de niveles de congestión solo busca hacer notar los diferentes factores que contribuyen a lo que percibe el transmisor y poder explicarlo con sus respectivas causas más comunes: incremento de RTT, retransmisiones que generan duplicados, algunos descartes (retransmisiones que no generan duplicados), muchos descartes.*

Nota2: *El encolamiento de paquetes lo que busca es resolver una congestión puntual, asume que unos instantes mas tarde se resuelve y puede despachar los paquetes que acumuló. Si no existiera, entonces cada vez que dos paquetes quisieran utilizar la misma interfaz de salida se descartaría uno de ellos y no es algo deseable, más aun si el tráfico es bajo respecto a las capacidades de los enlaces. No es algo negativo usar colas de atención, todo lo contrario, solo hay que entender que no resuelven todo el problema. Lo resuelven muy bien en eventos puntuales pero no pueden ser infinitos porque expiran los temporizadores de retransmisión y porque incrementan mucho el RTT.*

Nota 3: *Haciendo la analogía con el trayecto en auto dentro de una ciudad, y cronometrando el viaje de ida y vuelta por el mismo camino. Solo controlo el tiempo insumido, pero no sabemos en que esquina o en que semáforo demoramos más en cruzar. Que esté congestionado (se perciba más demora), en el viaje desde los accesos desde el centro de Montevideo a X, no implica que se perciba congestión en trayecto desde el puente de las Américas a X. Observar que implícitamente está la noción de comparar, porque tanto la capa 4 como la capa 5, solo pueden medir la demora total. En una ciudad, la referencia es con una hora de bajo tránsito (lo que suelen hacer google map o waze), en redes se suele compara con lo que estuvo sucediendo en los últimos segundos.*

Nota 4: *Las variantes de cuánto disminuir la ventana en caso de congestión básicamente intentan que no se baje tan bruscamente, una noción intuitiva es si estaba en fase de evitar la congestión y detecto congestión, disminuir a la mitad la ventana* ***cwnd*** *(observar que coincidiría con* ***ssthresh****) y seguir en fase de evitar la congestión.*

Nota 5: Observación (conceptos congestión, cwnd, RTT y timer de retransmisión): *Asumiendo que el receptor tiene capacidad suficiente de buffer, el valor de Window Size no limita el envío del transmisor (o la ventana de congestión, cwnd, actúa antes). La velocidad media máxima que podemos obtener es W_{TX}/RTT que en este caso (por la simplificación de que no restringe el control de flujo) es cwnd/RTT. Es máxima porque el transmisor podría enviar menos, media porque la promedio sobre el intervalo RTT, y porque la instantánea es a velocidad del medio físico que se utilice. La congestión es causada por más paquetes, que incrementan las interrupciones en CPU de los routers, que quieren utilizar un mismo link y se agregan a las colas para procesarlas cuando se resuelva la causa, o que el destino debe de enviar ACK y entregar a la Aplicación.*
Ajustar el valor de RTT no es la solución para disminuir la congestión, la "solución" es disminuir cwnd. El RTT solo buscar estimar la realidad, el timer de retransmisión se ajusta para no dar falsas pérdidas (siguiendo la analogía, hora pico, entonces los viajes demoran un poco más, pero para evitar la hora pico, tengo que disminuir los autos no incrementar lo que les va a insumir los viajes). El aumento de RTT es consecuencia de que existan muchos paquetes en la red, no es la causa. Al disminuir cwnd estoy enviando menos datos, recordando que TCP no supera MSS y asumiendo que TCP busca optimizar el envío completando los bytes hasta el MSS, menos datos implica menos paquetes.
La variable de ajuste es cwnd, RTT y el timer de retransmisión solo buscan calibrar el escenario actual, al instante actual, de forma de no dar timeout y retransmitir innecesariamente.

Nota 6*: Es usual es el tráfico UDP sea bajo en comparación con el tráfico TCP (por ejemplo 10%, 90% respectivamente de la capacidad de enlace). Como el efecto nocivo de demoras y descartes lo sufren todos, lo usual es que las aplicaciones "bien comportadas" sobre UDP incorporen alguna limitación del ancho de banda máximo, o solo realicen transferencias de pequeños volúmenes de datos o implementen algún mecanismo de detección y control de congestión (QUIC u otra alternativa donde exista realimentación en capa de aplicación).*

Nota 7: Conceptos sobre UDP no confiable: *Las aplicaciones que deciden utilizar UDP son conscientes de que es no confiable, y por ende siempre implementan estrategias para detectar pérdidas. Conceptualmente aceptar que pueden haber perdidas y no ser consciente de ellas es muy diferente. La elección de utilizar UDP o TCP normalmente es para evitar el handshake del inicio y fin de conexión, el control de congestión, o evitar las retransmisiones o porque hay otras capas que resuelven las pérdidas.*

***Nota 8**: En general no es posible tener una comunicación efectiva sin ser conscientes de que hubo pérdida de información y re-ordenación, dependiendo de la sensibilidad de la aplicación, la misma será responsable de implementar (en capa de aplicación) las estrategias que requiera para resolver estas pérdidas y reordenamiento.*
En el curso vimos dos simples: utilizar un temporizador en las consultas DNS y si expira sin respuesta, repetir la consulta y utilizar un esquema de secuenciamiento de la información en RTP para detectar que se perdieron muestras de una conversación de voz.

***Nota 9**: En un escenario de congestión, se puede tornar insostenible la comunicación, pero esto ocurre para todos, no se distingue entre UDP o TCP, pero el gran punto de diferencia es que UDP (la capa de transporte) no es quien reacciona a la congestión, eventualmente será la capa de aplicación o el usuario. Algunas aplicaciones de P2P solían utilizar UDP como estrategia para que no lo forzaran a disminuir el ancho de banda por control de congestión.*