Tema 6-Intercambio de datos TCP

El problema que se presenta es que la capa de red, como IP, no ofrece garantía de que los datos enviados se reciban correctamente, se reciban completos o en el orden en que fueron enviados. Por tanto, al implementar un protocolo de transferencia de datos que dependa de IP (como TCP), es necesario añadir mecanismos que permitan ofrecer esos servicios de transferencia fiable de datos.

Por ejemplo, TCP implementa un servicio de transferencia fiable de datos mediante el uso de números de secuencia y confirmación, retransmisiones en caso de pérdida de paquetes y control de flujo para evitar la congestión de la red. De esta manera, aunque IP no garantice la fiabilidad de los datos, TCP se encarga de que la transferencia de datos sea fiable para las aplicaciones que lo usan.

Cuando se dice que el canal es fiable significa que los datos transmitidos no se corrompen ni se pierden durante la transferencia. En este caso, no hay nada que hacer adicionalmente para garantizar la fiabilidad de los datos transferidos.

Sin embargo, si el canal no es completamente fiable y existe la posibilidad de que los datos se corrompan o se pierdan, se necesitan mecanismos adicionales para garantizar la transferencia fiable de datos.

Para la detección de errores, el receptor utiliza un checksum para verificar la integridad de los datos recibidos. Si los datos están corruptos, el receptor enviará una confirmación negativa (NAK) al emisor, solicitando la retransmisión de los datos. El emisor debe esperar una confirmación positiva (ACK) del receptor antes de enviar el siguiente paquete de datos. Si no se recibe un ACK, el emisor retransmite el paquete. Además, se utiliza un número de secuencia para identificar los paquetes enviados y confirmados, y asegurar que se reciben en el orden correcto.

Estos mecanismos adicionales de detección de errores, confirmación y retransmisión se denominan ARQ (Automatic Repeat reQuest) y se utilizan en protocolos como TCP para garantizar la transferencia fiable de datos sobre un canal no fiable.

Cuando el canal también puede perder datos, necesito una solución para garantizar la transferencia fiable de datos. Una solución es esperar y retransmitir los datos en caso de no recibir confirmación del receptor. Para esto, necesito un temporizador (timeout) en el emisor para cada paquete de datos enviado.

El problema con el temporizador es decidir cuánto tiempo esperar antes de retransmitir los datos. Idealmente, el tiempo de espera sería el tiempo de ida y vuelta (RTT - Round

Trip Time) de un paquete en la red, más un tiempo extra de procesamiento. Sin embargo, en muchas redes, como Internet, el RTT no tiene un valor máximo y puede variar significativamente.

Si espero demasiado antes de retransmitir, el proceso de transferencia de datos se ralentiza. Si espero muy poco, envío muchos datos repetidos sin necesidad, lo que aumenta el tráfico en la red y ralentiza el proceso. Por lo tanto, la elección de un valor de temporizador adecuado es crítica para garantizar una transferencia fiable de datos sin comprometer el rendimiento de la red.

Protocolo ARQ de parada y espera

El protocolo de parada y espera es un método utilizado en comunicaciones de redes para asegurar que los datos se envíen y se reciban de forma fiable entre un emisor y un receptor. En este protocolo, el emisor envía un paquete de datos y espera a recibir una confirmación (ACK) del receptor de que los datos han sido recibidos correctamente antes de enviar el siguiente paquete.

En el protocolo de parada y espera de bit alternante, el emisor envía un paquete y espera a recibir un ACK del receptor. Si el ACK no se recibe antes de que expire el temporizador, el emisor reenvía el mismo paquete. Si el receptor recibe correctamente el paquete, envía un ACK al emisor. El emisor entonces envía el siguiente paquete y repite el proceso.

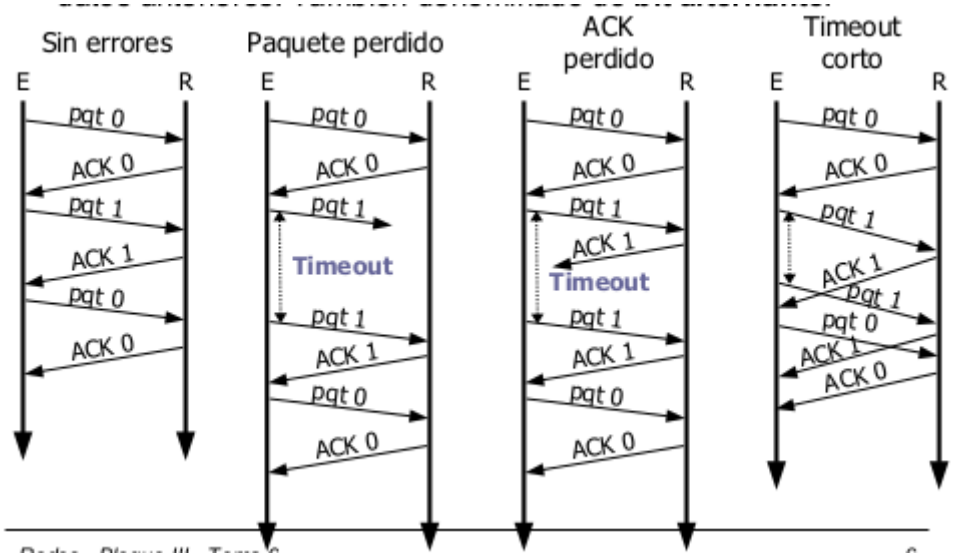
El nombre "bit alternante" se refiere a la forma en que el emisor y el receptor alternan los bits en el campo de control de los paquetes para identificar cada paquete. Por ejemplo, si el primer paquete enviado por el emisor tiene un bit de control "0", el siguiente paquete tendrá un bit de control "1", y así sucesivamente.

Este protocolo es efectivo para garantizar que los datos se envíen de manera confiable y se reciban correctamente, pero puede ser ineficiente en redes con alta latencia o baja capacidad debido al tiempo de espera necesario para recibir una confirmación de cada paquete.

Es un protocolo muy lento, por lo que se necesita aumentar la velocidad de transmisión de datos en un canal no confiable que puede perder datos, se puede utilizar el procesamiento en cadena en los protocolos ARQ. Esto significa que se envían varios paquetes sin esperar a la confirmación de los paquetes anteriores.

Para implementar esto, es necesario aumentar el tamaño de los números de secuencia para que varios paquetes puedan estar en la red simultáneamente sin confirmar. Además, el emisor necesita un buffer para almacenar los paquetes transmitidos pero no confirmados, mientras que el receptor necesita un buffer para almacenar los paquetes recibidos correctamente pero que la capa superior aún no puede procesar.

Existen dos protocolos ARQ de procesamiento en cadena: Retroceder N (Go-Back-N) y Repetición Selectiva (Selective Repeat). En el protocolo GBN, el emisor envía varios paquetes y espera a que el receptor confirme el último paquete recibido. Si se detecta un error en cualquier paquete, el receptor solicita la retransmisión de todos los paquetes desde el último paquete confirmado. Por otro lado, en el protocolo SR, el receptor confirma cada paquete recibido individualmente, lo que permite al emisor retransmitir solo los paquetes que se han perdido o recibido incorrectamente, en lugar de tener que retransmitir todos los paquetes desde el último confirmado.



Protocolo ARQ retroceder N

Este protocolo permite al emisor enviar varios paquetes sin tener que esperar a que cada paquete sea confirmado antes de enviar el siguiente. Sin embargo, para evitar la sobrecarga de la red y garantizar la entrega confiable de los paquetes, se establece un límite en el número de paquetes que se pueden enviar sin confirmación. Este límite se representa como una ventana que se mueve a medida que se reciben ACKs del receptor.

El tamaño de la ventana se establece como máximo N paquetes. Cada vez que el emisor recibe un ACK nuevo del receptor, se puede enviar otro paquete, lo que hace que la ventana se mueva hacia adelante. Este proceso se conoce como el protocolo de ventana deslizante.

El receptor no tiene un buffer y los números de secuencia son finitos, por lo que se utilizan números de secuencia circulares. El protocolo ARQ retroceder N solo utiliza ACKs positivos, lo que significa que el receptor solo envía ACKs para confirmar la recepción exitosa de paquetes. Además, los ACKs son acumulativos, lo que significa que si el receptor recibe varios paquetes en orden, solo enviará un ACK para confirmar todos los paquetes recibidos.

Protocolo ARQ de repetición selectiva

El Protocolo ARQ de repetición selectiva es una variante del Protocolo ARQ retroceder N utilizado en la comunicación de datos entre un emisor y un receptor. Esta variante se utiliza para solucionar el problema de retroceso N, en el que un error en un solo paquete hace que se repitan otros paquetes que se recibieron correctamente.

En lugar de repetir todos los paquetes desde el punto en que se produjo el error, el Protocolo ARQ de repetición selectiva permite que el emisor retransmita solo los paquetes erróneos. Para hacer esto, el receptor envía ACKs individuales o selectivos (SACK) para confirmar la recepción de cada paquete, en lugar de enviar solo un ACK acumulativo.

Además, el Protocolo ARQ de repetición selectiva utiliza una ventana que contiene algunos paquetes confirmados. El receptor necesita un buffer para almacenar los paquetes confirmados hasta que se reciba la confirmación de todos los paquetes dentro de la ventana. Los paquetes que no están confirmados dentro de la ventana se consideran perdidos y se retransmiten.

Finalmente, el Protocolo ARQ de repetición selectiva utiliza un temporizador para cada paquete enviado. Si no se recibe una confirmación antes de que expire el temporizador, se considera que el paquete se perdió y se retransmite. Este proceso ayuda a garantizar la entrega confiable de los paquetes y reduce la cantidad de paquetes retransmitidos innecesariamente.

Intercambio de datos TCP

En TCP (Protocolo de Control de Transmisión), se consideran dos tipos de tráfico de datos:

- **Interactivo**: transmisión de un gran número de segmentos de pequeño tamaño, generalmente de menos de 10 bytes. Este tipo de tráfico se utiliza para aplicaciones que requieren una comunicación en tiempo real, como telnet o SSH. Estas aplicaciones permiten a los usuarios interactuar directamente con el sistema remoto y requieren una transmisión rápida y confiable de pequeñas cantidades de datos.
- **No interactivo**: transmisión de segmentos de gran tamaño, generalmente el máximo permitido por las limitaciones de la red. Ejemplos de aplicaciones que utilizan este tipo de tráfico incluyen HTTP, FTP y correo electrónico. Estas aplicaciones no requieren una comunicación en tiempo real y, por lo tanto, pueden tolerar cierto grado de retraso en la transmisión de datos.

Para implementar la fiabilidad en la transmisión de datos, TCP se basa en el modelo ARQ (Automatic Repeat Request) retroceder N. Este es un protocolo de ventana deslizante, lo que significa que utiliza una ventana de transmisión para controlar el flujo de datos. Los

ACKs que se reciben son acumulativos y positivos, lo que significa que el receptor informa al emisor sobre el número de paquetes que ha recibido correctamente.

Cuando el receptor recibe un paquete fuera de orden, en lugar de descartarlo, lo almacena en un buffer y envía un ACK del último paquete recibido correctamente. Si el emisor recibe tres ACKs repetidos, entonces se activa la retransmisión rápida, lo que significa que el emisor retransmite solo el paquete siguiente al número de ACK recibido.

El emisor mantiene un temporizador para cada grupo de paquetes enviado, y si no recibe un ACK dentro de un tiempo determinado, se retransmiten los paquetes correspondientes.

En el RFC 2018, se propone una mejora para el receptor TCP con confirmación selectiva (SACK). Esto significa que el receptor puede informar al emisor sobre los paquetes que han sido recibidos correctamente y aquellos que no lo han sido, lo que permite al emisor retransmitir solo los paquetes perdidos en lugar de toda la ventana. Esto aumenta la eficiencia y la velocidad de la transmisión de datos.

El Retransmission Timeout (RTO) es el tiempo que debe transcurrir antes de que un emisor retransmita un paquete que no ha recibido confirmación de su recepción por parte del receptor. Este tiempo es importante en TCP para garantizar la fiabilidad de la transmisión de datos.

En TCP, el RTO se calcula a partir del Round-Trip Time (RTT), que es el tiempo que tarda un paquete en viajar desde el emisor hasta el receptor y luego de regreso. El RTT se estima continuamente durante toda una conexión TCP para determinar el valor del RTO. Cuando un segmento se envía, se mide el tiempo que tarda en recibir su ACK. Luego, se utiliza esta información para estimar el valor del RTT y, por ende, el valor del RTO.

Además, existe la opción de Timestamp (TSOPT) en TCP que se utiliza para mejorar la precisión de la estimación del RTT. El emisor indica el valor de su reloj en el momento de la transmisión en el campo Timestamp Value (TSval), y el receptor copia este valor en el campo Timestamp Echo Reply (TSecr) de su segmento de respuesta.

Con la ayuda de TSOPT, el emisor puede estimar el RTT de manera más precisa. El emisor indica el TSval en los segmentos que envía, y al preparar la respuesta (ACK), el receptor copia el TSval en el campo TSecr. Luego, el emisor al recibir el ACK, comprueba el reloj del sistema, le resta el valor de TSecr y de esta forma tiene la estimación del RTT.

En resumen, el RTO es el tiempo que debe transcurrir antes de que un emisor retransmita un paquete que no ha sido confirmado, y se calcula a partir del valor del RTT. La opción de Timestamp (TSOPT) en TCP se utiliza para mejorar la precisión de la estimación del RTT y, por ende, del valor del RTO.