Resumen #4 - (r5)

Esteban Ignacio Durán Vargas - 2020388144

IC - 7602 - 2023 I Semestre

6.6 ASPECTOS DEL DESEMPEÑO

6.6.1 Problemas de desempeño en las redes de cómputo

El desempeño se degrada con congestión (sobrecarga de recursos) y desequilibrio estructural de de recursos (equipo físico insuficiente pierde paquetes o genera retraso). Las sobrecargas también pueden ser sincrónicas; una tormenta de difusiones (ej. de notificaciones de errores TDPU) paralizan la red. UDP dejó de responder a estos errores en difusión. Otra sobrecarga es cuando se va la luz y vuelve todas las máquinas al mismo tiempo revisan el DHCP y el servidor de archivos para recuperar el SO. Puede también afectar el sistema la falta de afinación de este (bajos recursos como búfer o sin prioridad para procesar TPDUs), de la misma manera los temporizadores para proteger contra perdidas que si tienen un valor muy bajo, van a ocurrir retransmisiones innecesarias o si es muy alto pueden ocurrir retardos. Otro valor afinable es el tiempo de espera para incorporar paquetes antes de enviar una confirmación de recepción. Las redes de gigabit sufren problemas de desempeño al enviar por largas distancias y tener que detenerse hasta recibir la primera confirmación. Para el análisis de desempeño se utiliza el **producto ancho de banda-retardo** (ancho de banda * retardo). Para un buen desempeño, la ventana del receptor debe tener mínimo el tamaño del producto ancho de banda-retardo e incluso más para que tal vez responda instantaneamente. También hay problemas con aplicaciones de tiempo crítico como audio o vídeo es la fluctuación (tiempo medio de transmisión corto no es suficiente y requiere desviación estándar pequeña).

6.6.2 Medición del desempeño de las redes

El ciclo para medirlo es: medir parámetros y desempeño, tratar de entender lo que pasa y cambiar un parámetro. Una medición puede ser con un temporizador cuando inicia la actividad; otros con contadores para el número de eventos que ocurren. Escollos potenciales:

- **Asegúrese que el tamaño de la muestra es lo bastante grande:** Repetir la medición muchas veces y obtener el promedio.
- **Asegúrese de que las muestras son representativas:** Observar el efecto de diferentes cargas sobre la cantidad medida.
- **Tenga cuidado al usar relojes de intervalos grandes:** Requiere cuidado medir eventos de menos de 1 mseg.
- **Asegúrese de que no ocurre nada inesperado durante sus pruebas**: Ejecutar en un sistema inactivo y crear la carga completa.
- **El caché puede arruinar las mediciones**: Se puede desbordar el caché para evitar que las mediciones no comprendan el tráfico de red. Igual con los búferes.
- **Entienda lo que está midiendo:** Entender qué factores se están midiendo (tarjetas de intefaz de hardware, controladores, etc).
- **Tenga cuidado con la extrapolación de los resultados:** Muchos resultados de encolamiento comprenden $1/(1 - \rho)$, donde ρ es la carga, entonces los valores de tiempo de respuesta pueden no ser lineales.

6.6.3 Diseño de sistemas para un mejor desempeño

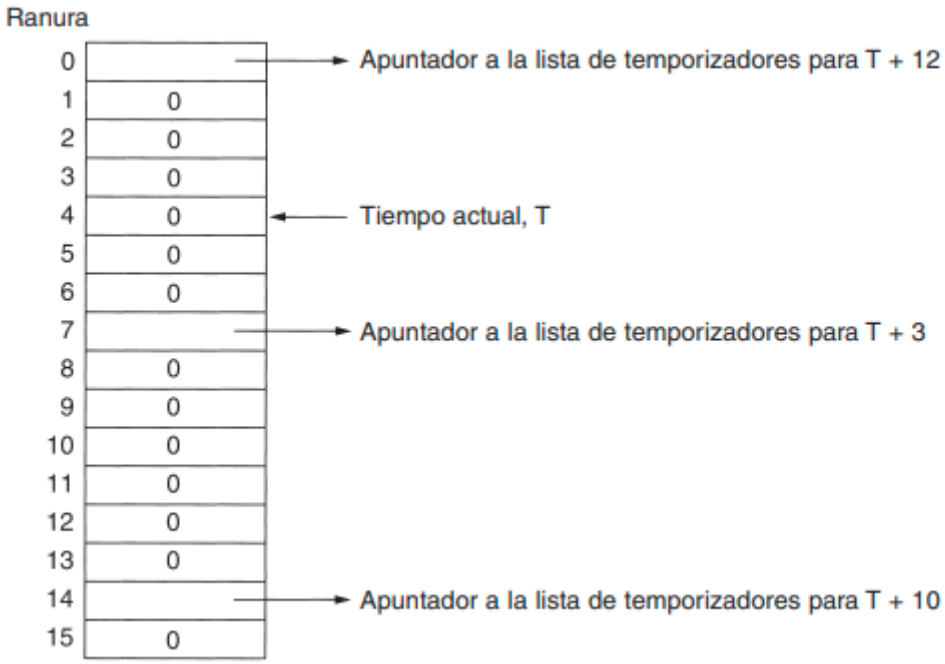
Un buen diseño puede ser mejor que ajustes, aquí las reglas:

- **Regla #1 - La velocidad de la CPU es más importante que la velocidad de la red:** Si se duplica la velocidad de la CPU, se puede duplicar la velocidad del transporte. Los cuellos de botella están en los hosts en procesar datos y ponerlos en la primera fibra.
- **Regla #2 - Reducir el número de paquetes para reducir la sobrecarga de software:** Reducir el tamaño de las TPDUs reduce la sobrecarga de la interrupción y de los paquetes en un factor de n una vez que llega el paquete al host. Enviar paquetes más pequeños hace más sobrecarga.
- **Regla #3 - Reducir al mínimo las conmutaciones de contexto:** Pueden generar series de fallas de caché. Pueden recucirse enviando los datos un búfer interno hasta tener una buena cantidad. El receptor de TPDUs hace lo mismo. En algunos sistemas operativos ocurren otras adicionales que desperdician tiempo en el CPU.
- **Regla #4 - Reducir al mínimo las copias:** Si se recibe un TPDU en un búfer de una tarjeta, se copia al kernel, luego a la capa de red, transporte y en la aplicación. Por lo general el Sistema Operativo es el problema y no permite manejar toda la velocidad.
- **Regla #6 - Evitar la congestión es mejor que recuperarse de ella:** Si se congestiona se pierden paquetes, desperdicia ancho de banda, hay retardos; la recuperación lleva mucho tiempo.
- **Regla #7 - Evitar expiraciones del temporizador:** Deben usarse con cuidado y con el mínimo de expiraciones; y no usarlo para repetir acciones innecesarias. Si expira tarde agrega retardo extra en caso de una pérdida de TPDU. Si expira muy antes desperdicia tiempo de CPU e impone sobrecarga.

6.6.4 Procesamiento rápido de las TPDUs

La sobrecarga de procesamiento de TPDUs tiene 2 componentes: TPDU y sobrecarga por byte. Se debe separar el caso normal de transferencia de datos de un solo sentido y manejarlo como especial. Una vez se entra en *ESTABLISHED*, el procesamiento es directo hasta que un lado cierra la conexión. La entidad revisa que es el caso normal: en *ESTABLISHED*, ningún lado cierra conexión, se envía un TPDU normal y hay espacio de ventana en el recptor. Con esto se sigue la trayectoria rápida. Los encabezados son casi iguales y se almacena el encabezado prototipo en la entidad de transporte. Los campos que cambian se sobreescriben en el buffer. Luego pasan a la capa de red a un apuntador del encabezado más otro para los datos de usuario. Luego la capa de red entrega el paquete a la de datalink para transmitir. Un ejemplo es TCP/IP. El proceso de trayectoria rápida, el tcp se almacena el registro de conexión, ambos puertos y direcciones se comparan. También se puede mantener un apuntador al último registro usado como una optimización. Luego se revisa el TPDU que sea normal y se invoca un TCP para trayectoria rápida. Esta actualiza el registro de conexión y copia los datos de usuario y va calculando el checksum y eliminando datos extra y se devuelve la confirmación de recepción. A este proceso se le llama **predicción de encabezado**. Se pueden encontrar mejoras en el manejo de buffers, evitando copiado innecesario, y en la administración de temporizadores (se ajustan para protegerse contra pérdidas, entonces se debe optimizar que nunca expiren). Para esto último se da una lista enlazada de eventos del temporizador ordenada por hora de expiración, cada entrada tiene la cantidad de pulsos que faltan para expirar después de la entrada previa. Hay uno más eficiente donde el intervalo máximo del temporizador está limitado y se conoce por adelantado. Para esto está la **rueda de temporización** donde

cada ranura es un pulso de reloj. Si la entrada a la que ahora se apunta es diferente de cero, todos sus temporizadores se procesan.



6.6.5 Protocolos para redes de gigabits

El primer problema es que muchos protocolos usan secuencia de 32 bits. Antes tardaban una semana en devolver el número de secuencia. Con 1 Gbps, se dura 34 segundos, menos que el tiempo de vida de un paquete. Otro problema es que las velocidades de comunicación mejoraron. Ahora una computadora de 1000 MIPS intercambia paquetes de 1500 bytes por una línea de gigabits y su procesamiento debe de ser en 6.25 µseg y esto es mucho más rápido que los hosts de ARPANET y con RISC se usan menos instrucciones. O sea, menos tiempo para procesar protocolos, por lo que deben volverse más sencillos. Otro problema es que el protocolo de retroceso n se desempeña mal con un producto ancho de banda-retardo grande. Se desperdician los recursos mientras sen envían paquetes de error que tienen que esperar el emisor. El cuarto error es que las líneas de gigabit son diferentes a megabits porque gigabit está limitada por el retardo en lugar de ancho de banda. El retardo de ida y vuelta domina lo que se tarda en insertar bits en fibra. Protocolos como RPC tienen un límite superior (velocidad de la luz) por esto. Un quinto problema es con las aplicaciones donde por multimedia la variación de los paqutes es tan importante como el retardo medio mismo.

Los protocolos deberían diseñarse para reducir el procesamiento de protocolos. Se pueden construir interfaces más rápidas pero implica otro CPU dedicado. Esto provoca que la CPU principal quede a la espera de la segunda. Para la retroalimentación en protocolos de alta velocidad, que debido al ciclo de retardo se evitan. Es mejor usar un protocolo basado en tasa donde el emisor pueda enviar todo lo que quiera mientras sea a mayor velocidad que cierta tasa acordada. Luego está el algoritmo de arranque lento de Jacobson donde se hacen sondeos para saber qué tanto maneja la red. Es mejor hacer que el emisor y receptor reserven recursos cuando se conectan; esto reduce flucturación. La disposición de paquetes es importante ya que debe contener la menor cantidad de datos para su procesamiento y estalineados con los límites de palabras. No deben existir problemas de números de secuencia que den vuelta mientras estén los viejos. El encabezado y datos deben tener checksums separados porque debe ser posible obtener el checksum del encabezado y no los datos y que se debe comprobar que el encabezado esté correcto antes de ver los datos.

Se prefiere calcularlo al mismo tiempo que se copian los datos en el espacio de usuario. Para evitar hacer un copiado incorrecto si falla el encabezado, se debn tener separados. El tamaño máximo de los datos debe ser lo bastante grande para permitir una operación eficiente, inclusive en retardos grandes. Mientras más grande, menor la fracción de ancho de banda para encabezados. También se puede enviar una cantidad normal de datos junto con la solicitud de conexión.

Los protocolos se centran en casos exitosos. Procesar errores es secundario. Para minimizar tiempo de copiado, el hardware debe poner en memoria cada paquete continuo. Dependiendo del funcionamiento del caché, puede ser deseable evitar un ciclo de copiado.