Departamento de
Automática y Computación
Automatika eta
Konputazio Saila

Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es

Práctica 4 – Control de congestión en TCP

1. Introducción y objetivos

En esta práctica veremos los mecanismos de TCP en funcionamiento en la Internet, tanto para una conexión sola como compartiendo un cuello de botella con otras.

Para acortar el tiempo dedicado a la práctica se han recogido previamente las trazas que se van a analizar. Las traza a analizar corresponden a la subida de ficheros por parte de un usuario ADSL mediante una conexión TCP.

2. Análisis de una transferencia sobre TCP (70%)

Nos interesa evaluar el comportamiento del protocolo TCP durante una transferencia de subida empleando un acceso ADSL. Nos centraremos en el sentido en el que se ha producido la transferencia.

La captura se ha hecho directamente en el PC del usuario que ha subido el fichero, que mediante una LAN accedía al router ADSL. En la traza está solo la conexión indicada. Es improbable que hubiera más tráfico en el enlace aunque no se puede descartar.

Para cada resultado explique cómo lo ha obtenido.

2.1. Throughput (30%)

A efectos de un usuario que esté transfiriendo un fichero en un servicio elástico parece que lo que más interesa es alcanzar el mayor throughput posible. Veamos a continuación el throughput que alcanza el usuario y qué conclusiones podemos sacar de ello. ¿Le estará sacando provecho por completo a su enlace ADSL?

2.1.1. Throughput en el enlace

¿Cuál es el throughput medio presente en el enlace de acceso?

Puede hacer una primera aproximación con los tamaños de las tramas Ethernet. A continuación tenga en cuenta que el enlace ADSL en cuestión emplea ATM con LLC/SNAP (tiene este encapsulado explicado en los materiales de la asignatura FTPR) y estime el throughput real a nivel ATM (contando hasta las cabeceras de las celdas).

Puede calcular un valor promedio total de la conexión, otro valor entre el primer paquete de datos y el último paquete de datos y finalmente uno promediado en ventanas más pequeñas de tiempo y representarlo frente al tiempo.

Comente los resultados.

2.1.2. Throughput visto por el usuario

¿Cuál es el throughput útil que ve el usuario?

Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es

Esto tendría en cuenta solo los datos que han llegado al destino. Contaremos como datos útiles los que transporta TCP. Tenga en cuenta que en caso de retransmisiones solo debe contar una vez esos bytes y no una vez por paquete retransmitido pues a efectos del usuario final esos bytes solo están una vez en el documento transferido.

Puede hacer los mismos tres promedios que en el caso del throughput en el enlace de acceso y compararlos

2.1.3. Bytes en vuelo

TCP es un protocolo de ventana deslizante. Evalúe la situación de dicha ventana. Puede representar el número de bytes en vuelo (sin confirmar) frente al tiempo (wireshark permite hacerlo pues lo calcula en tcp.analysis.bytes_in_flight).

¿Qué valor alcanza? ¿Es estable? ¿Si cambia, por qué? ¿Si no cambia por qué? Comente la situación.

2.1.4. Predicción para otros ficheros

A partir de lo que acaba de calcular, ¿cuánto estima que tardaría el usuario en las mismas condiciones en transferir un fichero de 5MBytes? ¿Y un fichero de 1 MByte?

¿Cuál es el tamaño inicial de la ventana (IW) que emplea esta implementación de TCP?

¿Cuánto tardaría el usuario en las mismas condiciones en transferir un fichero de 10KBytes? (tenga en cuenta el efecto del slow start).

2.2. Retardo (40%)

Sabemos que el RTT juega un papel fundamental en los protocolos basados en ventana deslizante.

Debemos tener suficientes paquetes "en vuelo" para poder sacar el máximo rendimiento en estos protocolos. Si no, por ejemplo con una ventana de solo 1 paquete el protocolo se convierte en un STOP-and-WAIT y el máximo throughput alcanzable es de 1 paquete por RTT, lo cual con RTTs altos resulta un cociente bastante bajo (con un RTT europeo de 60ms y un MSS de 1460 bytes hablaríamos de un máximo de 200Kbps).

Es decir, es inevitable que el protocolo mantenga varios paquetes en vuelo sin confirmar y saber cuántos debería tener requiere conocer el RTT y el BW del cuello de botella. El RTT que ve la conexión suele depender principalmente de la situación de la red aunque también puede verse afectado por el tráfico de la propia conexión si produce encolado, como veremos a continuación.

2.2.1. Retardo entre SYN y SYN-ACK

Vamos a intentar hacer una estimación del RTT que hay en el trayecto sin tener en cuenta el efecto del propio flujo del usuario. Llamémoslo una medida "en vacío".

Departamento de
Automática y Computación
Automatika eta
Konputazio Saila

Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es

Una posibilidad es tomar el tiempo entre el SYN que envía el cliente y su confirmación (dado que la medida está tomada en el cliente). Normalmente el sistema operativo contesta al SYN sin esperar ninguna acción de la aplicación final así que la misma no debería estar alterando la medida.

Calcule este valor del RTT que llamaremos RTT del establecimiento.

2.2.2. Throughput máximo

Dado el retardo en vacío que acaba de calcular y el tamaño máximo de ventana que anuncia el receptor calcule cuál sería el máximo throughput que podría alcanzar una conexión TCP.

2.2.3. Variación del retardo

Para una conexión larga, el retardo que sufran los paquetes en la red para un trayecto largo va a variar.

Estos cambios pueden deberse a cambios en las condiciones de la red debido a otros flujos (es lo que vemos si hacemos un ping sostenido durante un tiempo contra un servidor internacional por ejemplo).

Si el flujo del usuario es de muy baja capacidad frente a los enlaces por los que pasa esperamos que no afecte al retardo. Sin embargo, si el flujo del usuario consigue saturar algún enlace entonces podríamos estar en un caso en que el propio flujo del usuario altera el retardo (en general aumentándolo, pero esto puede variar según la interacción con otros flujos).

Una vez que empieza el flujo va a ser más difícil calcular el retardo "en vacío". Podríamos haberlo hecho enviando ICMPs echo_request sin tener el flujo pero entonces no nos serviría de mucho pues no sería el retardo que hay cuando hacemos la transferencia sino en otro intervalo de tiempo (salvo que confiemos en que sea muy estable).

Una posible estimación del retardo "en vacío" durante la conexión se puede hacer cuando la misma vacía la *pipe*, es decir, no tiene paquetes en vuelo, lo cual suele darse cuando caduca un timer de retransmisión.

Con un poco de suerte esto no va a suceder (se ha refinado TCP en parte para evitar que caduque este timer). Sin embargo, si se producen pérdidas sí que bajará la velocidad a la que se envía y eso permitirá que se vacíe un poco la red de los paquetes de esta conexión. Un paquete que se envíe cuando hay pocos "en vuelo" sufrirá el menor retardo debido a interferencia entre los paquetes de la propia conexión.

Haga una representación gráfica del número de bytes (o paquetes) en vuelo frente al tiempo, así como del RTT frente al tiempo (medido entre el envío de un paquete de datos y la recepción del ACK que los confirma, lo cual da wireshark en tcp.analysis.ack_rtt).

Departamento de
Automática y Computación
Automatika eta
Konputazio Saila

Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es

¿Se vacía en algún momento el canal? ¿Cómo varía la estimación del retardo medido como el tiempo entre datos y ack a medida que evoluciona la conexión?

Calcule el RTT medio que ven los paquetes.

El retardo adicional es en gran medida debido a encolado en el cuello de botella. ¿Cuántos bytes/paquetes calcula que deben estar encolándose en el buffer del enlace de acceso?

2.2.4. Throughput con retardo mayor

Suponga que mientras se hace el *upload* de un fichero como se ha visto en las secciones anteriores otro usuario intenta descargar un fichero de 10Kbytes (el tráfico de subida que genera es solo el de control de establecimiento de la conexión y confirmaciones). ¿Cuánto tardará en esta descarga de 10Kbytes si el *upload* se extiende más allá de lo que tarde en esta descarga? Suponga que no se producen pérdidas para los paquetes de esta nueva conexión y compare el resultado con el tiempo que tardaría en hacer esa misma descarga de 10Kbytes si no estuviera el *upload* simultáneo. Suponga que a descarga se hace del mismo servidor al que se está subiendo el fichero.

3. Análisis de dos transferencias simultáneas sobre TCP (30%)

A continuación tenemos una traza en un escenario similar, solo que ahora el usuario intenta dos transferencias de subida, que deben compartir el canal.

En este caso es muy probable que haya retransmisiones. Localícelas. Represente frente al tiempo los instantes en que se producen. Comente cómo está relacionado con el RTT medido y con el número de bytes en vuelo. ¿En general son retransmisiones por timer o fast-retransmits? ¿Cómo lo reconoce?

A partir de lo que se ha analizado con anterioridad estudie este caso y evalúe en qué medida se están repartiendo el uso del canal las dos conexiones, cómo y por qué, qué efectos tienen los diferentes mecanismos de TCP, los bytes en vuelo que hay en cada momento de cada conexión, la evolución del retardo, la localización de las pérdidas, etc. No debe repetir el análisis sino centrarse en lo que vea interesante en este caso.

4. Conclusiones

Comente a modo de resumen los resultados sobre la capacidad de TCP de obtener el máximo throughput en el cuello de botella y de repartir la capacidad del mismo.

¿Qué opina del reparto que hace TCP del canal?