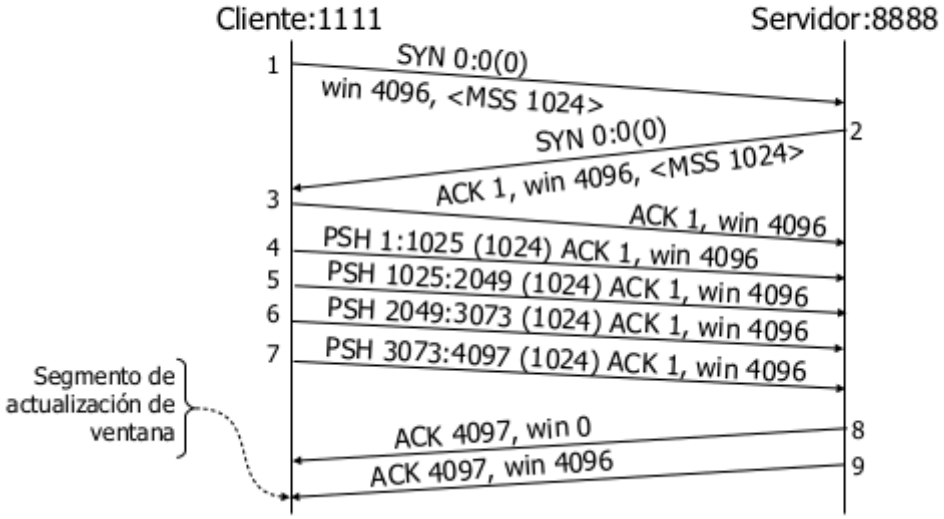
## Tema 6-Intercambio de datos TCP

El flujo de datos no interactivo, también conocido como flujo de datos en masa o bulk data flow, es aquel que genera pocos segmentos pero de gran tamaño. En este tipo de tráfico, el principal problema a resolver es el control de flujo, es decir, evitar que un emisor rápido sature a un receptor lento. Para solucionar esto, TCP utiliza una ventana deslizante, que permite al emisor enviar múltiples paquetes antes de detenerse y esperar por el ACK. De esta manera, se aumenta la rapidez del tráfico.

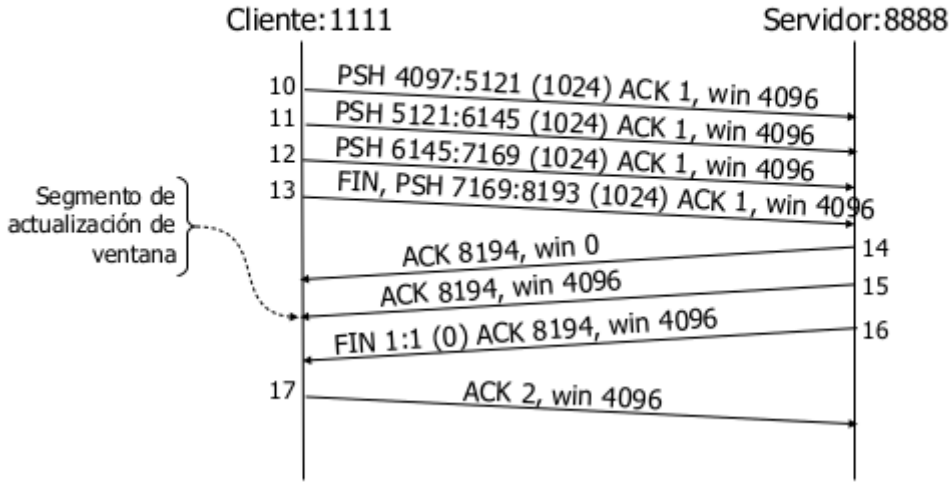
La ventana deslizante es un mecanismo que permite al emisor enviar varios paquetes seguidos, sin necesidad de esperar por el ACK de cada uno. El receptor, por su parte, envía el ACK confirmando la recepción de varios paquetes en lugar de cada uno por separado. Esto permite que el emisor envíe más paquetes antes de tener que detenerse y esperar por el ACK.

Además, con un protocolo de ventana deslizante, no es necesario confirmar todos los paquetes recibidos, sino que se pueden confirmar varios paquetes simultáneamente en un único ACK, lo que se conoce como ACK acumulativo. Esto reduce la cantidad de paquetes que se envían y se confirman en la red, lo que mejora el rendimiento y reduce la sobrecarga en la red.

- Emisor rápido, receptor lento




- Emisor rápido, receptor lento (continuación)

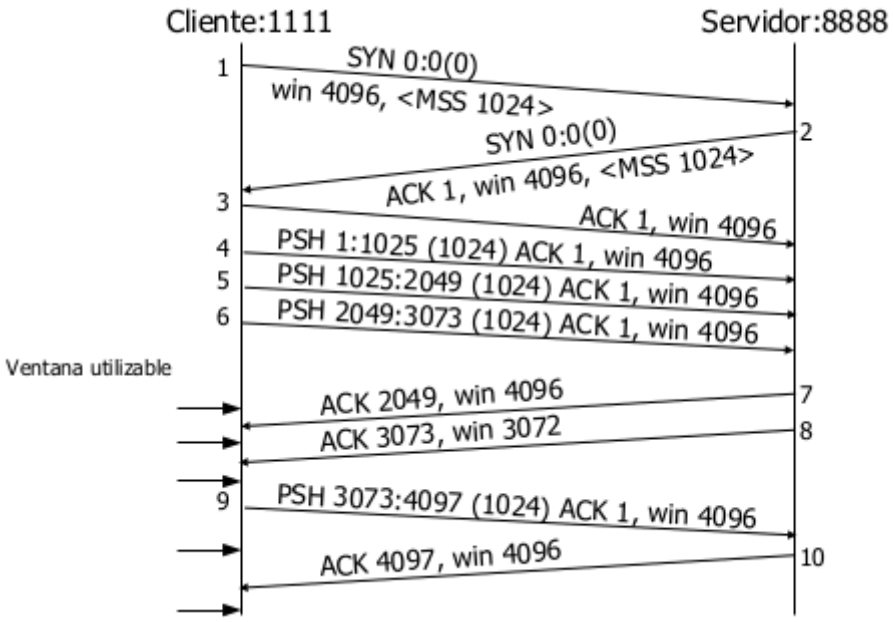



## Ventana deslizante

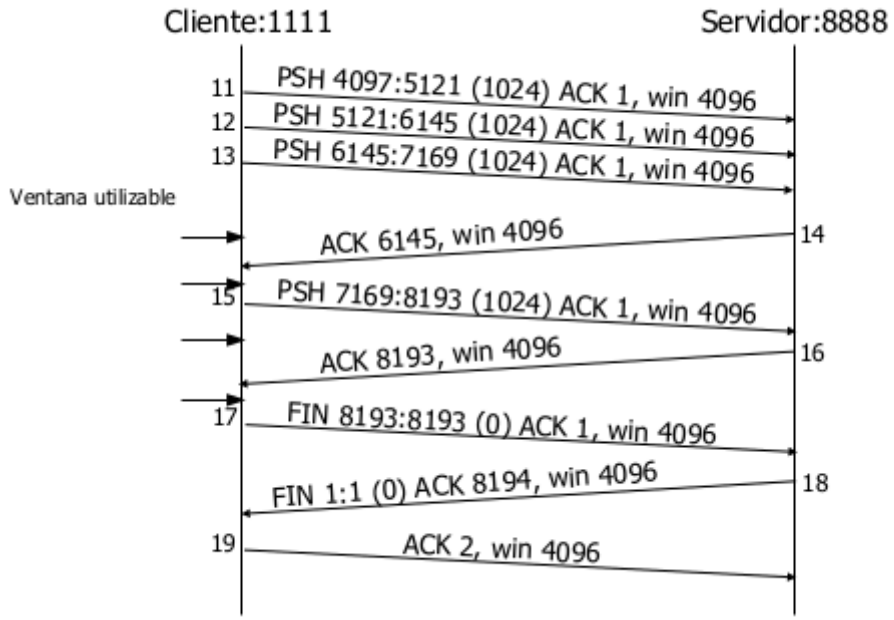
El funcionamiento de la ventana deslizante se basa en el uso de dos ventanas:

- **Ventana ofrecida**: también conocida como ventana de recepción, indica al emisor la cantidad de datos que el receptor está dispuesto a recibir en un momento dado. El receptor envía esta información en los paquetes de ACK que envía al emisor.
- **Ventana utilizable del emisor**: indica la cantidad de datos que el emisor puede enviar en un momento dado. Esta ventana se calcula:
  $Ventana\ utilizable = Ventana\ ofrecida - bytes\ no\ confirmados$

## Ejercicio




1. SYN→establece conexión TCP, Tamaño ventana envío=4096, MSS=1024
2. ACK 1→acepta conexión, Tamaño ventana recepción=4096, MSS=1024
3. ACK 1→El emisor confirma que está todo correcto.
4. El emisor envía 1024, Tamaño ventana envío=4096-1024=3072
5. El emisor envía 1024, Tamaño ventana envío=3072-1024=2048
6. El emisor envía 1024, Tamaño ventana envío=2048-1024=1024
   →LA VENTANA UTILIZABLE ES DE 1 SEGMENTO
7. El receptor confirma 1:2049
   →LA VENTANA UTILIZABLE ES DE 3 SEGMENTOS
8. El receptor confirma 1:3073, pero disminuye el tamaño de la ventana a 3072
   →LA VENTANA UTILIZABLE ES DE 3 SEGMENTOS
9. El emisor envía 1024, Tamaño ventana envío=3072-1024=2048
   →LA VENTANA UTILIZABLE ES DE 2 SEGMENTOS
10. El receptor confirma 1:4097 y aumenta el tamaño de la ventana a 4096.
    →LA VENTANA UTILIZABLE ES DE 4 SEGMENTOS




11. El emisor envía 1024, Tamaño ventana envío=4096-1024=3072
12. El emisor envía 1024, Tamaño ventana envío=3072-1024=2048
13. El emisor envía 1024, Tamaño ventana envío=2048-1024=1024
→LA VENTANA UTILIZABLE ES DE 1 SEGMENTO
14. El receptor confirma 1:6045 sin modificar el tamaño de la ventana
→LA VENTANA UTILIZABLE ES DE 3 SEGMENTOS
15. El emisor envía 1024, Tamaño ventana envío=3072-1024=2048
→LA VENTANA UTILIZABLE ES DE 2 SEGMENTOS
16. El receptor confirma 1:8193 sin modificar el tamaño de la ventana
→LA VENTANA UTILIZABLE ES DE 4 SEGMENTOS

## Factores de control de flujo

El envío de los mensajes TCP no es determinista, lo que significa que no se puede predecir con precisión cuánto tiempo tardará un paquete en llegar a su destino final debido a múltiples factores que pueden afectar la red, como la carga del tráfico, la congestión en la red, la velocidad del emisor y receptor, etc.

Para manejar esta incertidumbre, TCP utiliza ACKs acumulativos, lo que significa que un ACK confirma la recepción de todos los bytes recibidos correctamente hasta ese momento. Por ejemplo, en el caso de ACK 2049, el receptor está confirmando que ha recibido correctamente todos los bytes hasta el 2048 y que el siguiente byte que se espera recibir es el 2049.

El uso de ACKs acumulativos significa que el emisor no tiene que enviar una ventana completa de datos y que el receptor no tiene que esperar a que se llene la ventana para enviar un ACK. En lugar de eso, el receptor puede enviar ACKs acumulativos tan pronto

como se reciba correctamente un segmento de datos, lo que permite una mayor eficiencia en el flujo de datos.

La situación de "abrazo mortal" se produce cuando el emisor no recibe ACKs del receptor y, por lo tanto, no puede enviar más datos debido a que la ventana del receptor sigue cerrada. Para evitar esto, se utiliza un temporizador de persistencia que envía periódicamente unos segmentos especiales llamados window probes para comprobar si la ventana del receptor se ha actualizado.

Los window probes son segmentos de un byte que se envían para comprobar si realmente la ventana del receptor se ha modificado o no. El receptor puede confirmar o no el window probe en función del espacio libre en su buffer. Si el temporizador de persistencia se agota antes de recibir una respuesta del receptor, se activa un exponential backoff (un algoritmo utilizado en la comunicación en red que se utiliza para manejar la retransmisión de paquetes de datos. Cuando un paquete de datos se pierde o se daña durante la transmisión, el emisor debe reenviar el paquete. Para evitar el congestionamiento de la red, el algoritmo de backoff exponencial espera un tiempo aleatorio antes de reenviar el paquete. Si el paquete se vuelve a perder, el algoritmo de backoff exponencial espera un tiempo aún mayor antes de reenviarlo. El tiempo de espera aumenta exponencialmente cada vez que el paquete se pierde, lo que ayuda a evitar la congestión de la red) y se continúa retransmitiendo hasta que la ventana del receptor se abra o se cierre la conexión TCP por las aplicaciones.

## Control de congestión

El control de flujo en TCP se refiere a la regulación de la cantidad de datos que se transmiten entre el emisor y el receptor para evitar que el receptor se sature y no pueda procesar más datos. Sin embargo, en un entorno de red con routers intermedios, el control de flujo no es suficiente para garantizar una transmisión eficiente de datos porque los routers no tienen un nivel de transporte para regular el flujo de datos.

Para solucionar este problema, se utiliza el control de congestión. El control de congestión en TCP es un mecanismo que permite evitar que una red se sature debido a la sobrecarga de tráfico. El objetivo del control de congestión es evitar que los routers y los enlaces de la red se saturen y, por lo tanto, se produzcan retrasos y pérdidas de paquetes.

El algoritmo para evitar la congestión es una técnica que evita que la red se sature y se produzcan pérdidas de paquetes o colisiones. En resumen, el algoritmo establece un umbral de inicio lento, que se refiere a un límite en la cantidad de datos que el emisor puede enviar a la red. Cuando el emisor envía paquetes por debajo del umbral, se aplica

el algoritmo de inicio lento, que consiste en que la velocidad a la que se envían nuevos paquetes es la misma que la velocidad a la que se reciben los ACK del otro extremo.

Cuando el emisor alcanza el umbral, se aplica un crecimiento suavizado, lo que significa que el emisor aumentará gradualmente la cantidad de datos que envía a la red. En lugar de enviar grandes cantidades de datos de una sola vez, el emisor aumenta lentamente la cantidad de datos que envía, monitoreando constantemente la respuesta de la red y ajustando su velocidad de transmisión para evitar la congestión.

Cuando algo va mal en la transmisión de paquetes a través de la red, se produce una situación de congestión que puede causar la pérdida de paquetes. Para detectar esta situación, se utilizan dos indicadores: el tiempo de espera de retransmisión y los ACK duplicados.

Si un paquete se pierde y no se recibe un ACK dentro del tiempo de espera de retransmisión, se asume que se ha producido una congestión en la red y se inicia un proceso de retransmisión del paquete. Este proceso se llama Fast Retransmit.

Además, si el emisor recibe varios ACKs duplicados de un mismo paquete, asume que este paquete se ha perdido y procede a retransmitirlo. Este proceso se llama Fast Recovery.

Para evitar la congestión, el algoritmo de control de congestión establece un umbral de inicio lento, que es la velocidad de transmisión a la que se inyectan paquetes nuevos en la red. Por debajo del umbral, se aplica el algoritmo de inicio lento, que es la velocidad a la que se reciben ACKs del otro extremo. Por encima del umbral, se aplica un crecimiento suavizado. Cuando se detecta una situación de congestión, se reduce la velocidad de transmisión para evitar una mayor congestión.