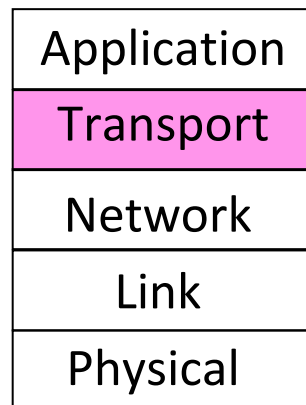


# Capítulo 3

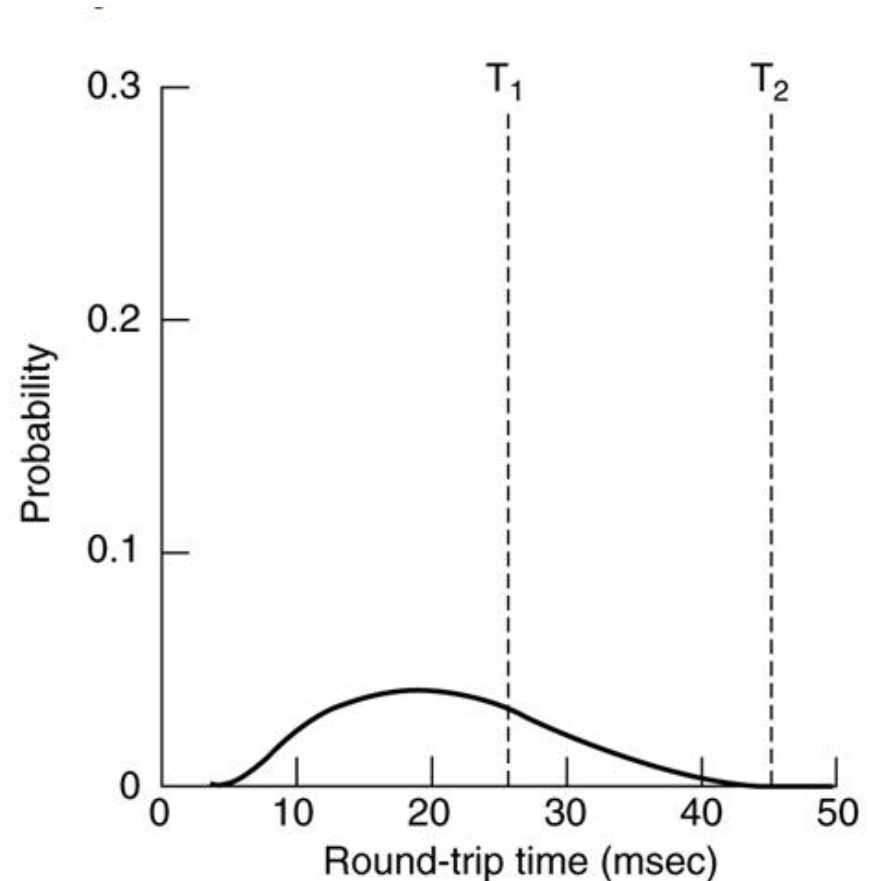
## Capa de Transporte

### Administración del temporizador de retransmisiones en TCP



# Administración del temporizador del TCP

- **Problema: ¿qué tan grande debe ser el intervalo de expiración del temporizador de retransmisión?**
  - Si se hace demasiado corto - digamos  $T_1$  en la Figura:
    - Ocurrirán retransmisiones innecesarias.
  - Si se hace demasiado largo? - digamos  $T_2$ :
    - Sufrirá el desempeño por el gran retardo de retransmisión de cada paquete perdido



# Administración del temporizador del TCP

- **Situación:**
  - La función de densidad de probabilidad del tiempo que tarda en regresar un ack TCP se parece a la Fig. anterior.
  - La varianza y la media de la distribución de llegada de las ack pueden variar a medida que se generan y se resuelven congestionamientos.
- **Idea:** Ajustar constantemente el intervalo de expiración del temporizador, con base en mediciones continuas del desempeño de la red.

# Administración del temporizador del TCP

- **Solución: Algoritmo de Jacobson** (1988) usado por TCP
  - Por cada conexión el TCP mantiene una variable, **RTT (round trip time)**,
    - **significa** estimación actual del tiempo de ida y vuelta al destino.
  - Al enviarse un segmento se inicia un temporizador,
    - para saber el **tiempo que tarda el ack**,
    - y para habilitar una retransmisión si se tarda demasiado.
  - Si llega el ack antes de expirar el temporizador:
    - TCP **mide** el tiempo que tardó el ack , digamos  $M$ ,
    - entonces actualiza el RTT así:
$$\text{RTT} = \alpha \text{ RTT} + (1-\alpha) M,$$
    - $\alpha$  es el peso que se le da al valor anterior. Por lo común  $\alpha = 7/8$ .

# Administración del temporizador del TCP

- Un RTT inicial de 1 sec se aconseja en RFC 6298.
- **Problema:** Dado RTT, hay que elegir una **expiración adecuada** del temporizador de retransmisión.
- **Solución 1:** En las implementaciones iniciales:  
$$\text{Expiración del temporizador} = 2 \times \text{RTT}.$$
- **Evaluación:** Este valor es inflexible, pues falla en responder a la suba de la varianza de la función de densidad de probabilidad del tiempo de llegada de los ack.

# Administración del temporizador del TCP

- **Solución 2:** (Jacobson 1988) hacer que el valor de timeout sea sensible tanto a la variación de RTT como a la varianza de la función de densidad de probabilidad del tiempo de llegada de los ack.
  - Se mantiene una variable amortiguada  $D$  (la desviación media).
  - Al llegar un ack, se calcula  $|RTT - M|$ .
  - Se mantiene en  $D$  mediante:
$$D = \beta D + (1 - \beta) |RTT - M|,$$
  - donde  $\beta$  típicamente es  $\frac{3}{4}$ .
  - $D$  es una aproximación bastante cercana a la desviación estándar.
- ¿Cómo estimar la expiración del temporizador? ¿De qué parámetros depende?

# Administración del temporizador del TCP

- La mayoría de las implementaciones TCP usan ahora este algoritmo y establecen:

**Expiración del temporizador =  $RTT + 4 \times D$ .**

- Con esto menos del 1% de los ack vienen en más de 4 desviaciones estándares tarde.

# Administración del temporizador del TCP

- **Problema: ¿qué se hace al recolectar muestras  $M$  cuando expira el temporizador de un segmento y se envía de nuevo?**
  - Cuando llega el ack no es claro si éste se refiere a la primera transmisión o a una posterior.
  - Si se adivina mal, se puede contaminar seriamente la estimación del RTT.
- **¿Cómo se puede estimar el temporizador de retransmisiones en ese caso?**



# Administración del temporizador del TCP

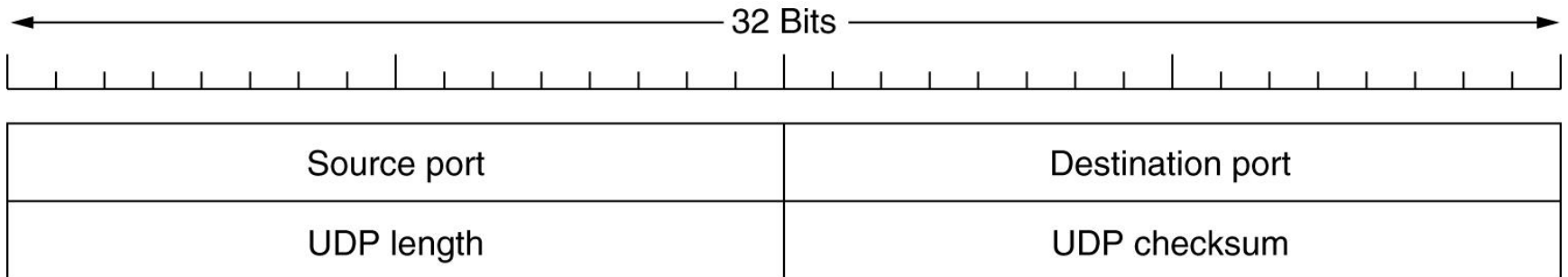
- **Solución: (algoritmo de Karn)**
  - No actualizar el RTT (cuando llega ack) de ninguno de los segmentos retransmitidos.
  - Cuando ocurre un timeout se **duplica la expiración del temporizador**.
  - Tan pronto se recibe un ack de segmento no retransmitido, el RTT estimado es actualizado y la expiración del temporizador se computa nuevamente usando la fórmula anterior.
- El algoritmo de Karn lo usan la mayoría de las implementaciones TCP.

# UDP

- UDP (**protocolo de datagramas de usuario**)
  - Es no orientado a la conexión.
- segmentos = encabezado de 8 B + carga útil.
  - 2 puertos de 16b.
  - El campo **longitud UDP** incluye el encabezado de 8 bytes y los datos.

# UDP

The UDP header.



# UDP

- **UDP no realiza:**
  - control de flujo, control de congestión, o retransmisión cuando se recibe un segmento erróneo.
  - Todo lo anterior le corresponde a los procesos de usuario.

# UDP

- UDP es especialmente útil en las situaciones cliente-servidor.
  - El cliente envía una solicitud corta al servidor y espera una respuesta corta.
  - **¿Qué pasa si se pierde la solicitud o la respuesta?**
  - El cliente puede probar nuevamente.