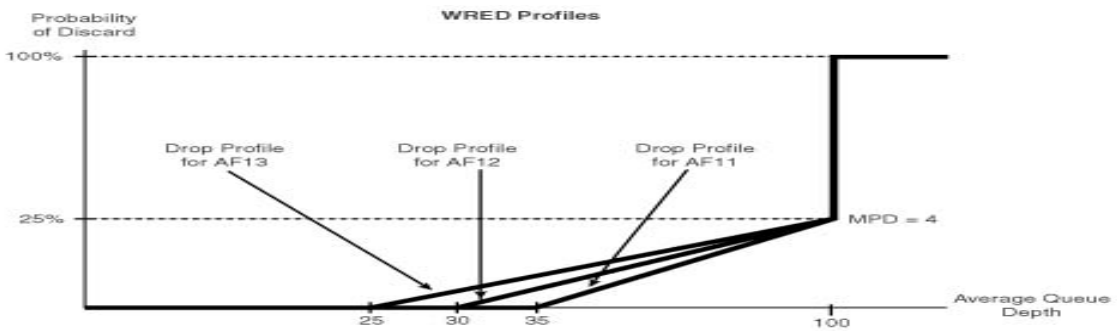
Redes
Extensiones TCP

Luis Marrone

LINTI – UNLP

2 de mayo de 2015



Estamos en:

1. TCP Selective Acknowledge

2. Manejo de Colas
   - Manejo Pasivo de las Colas
   - Manejo Activo de las colas
   - Random Early Detection
   - Explicit Congestion Notification

SACK

- ✓ Produce la retransmisión selectiva
- ✓ RFC 2018
- ✓ Comprende dos opciones de TCP
  - ✓ “Sack-Permitted Option”
  - ✓ “Sack Option”

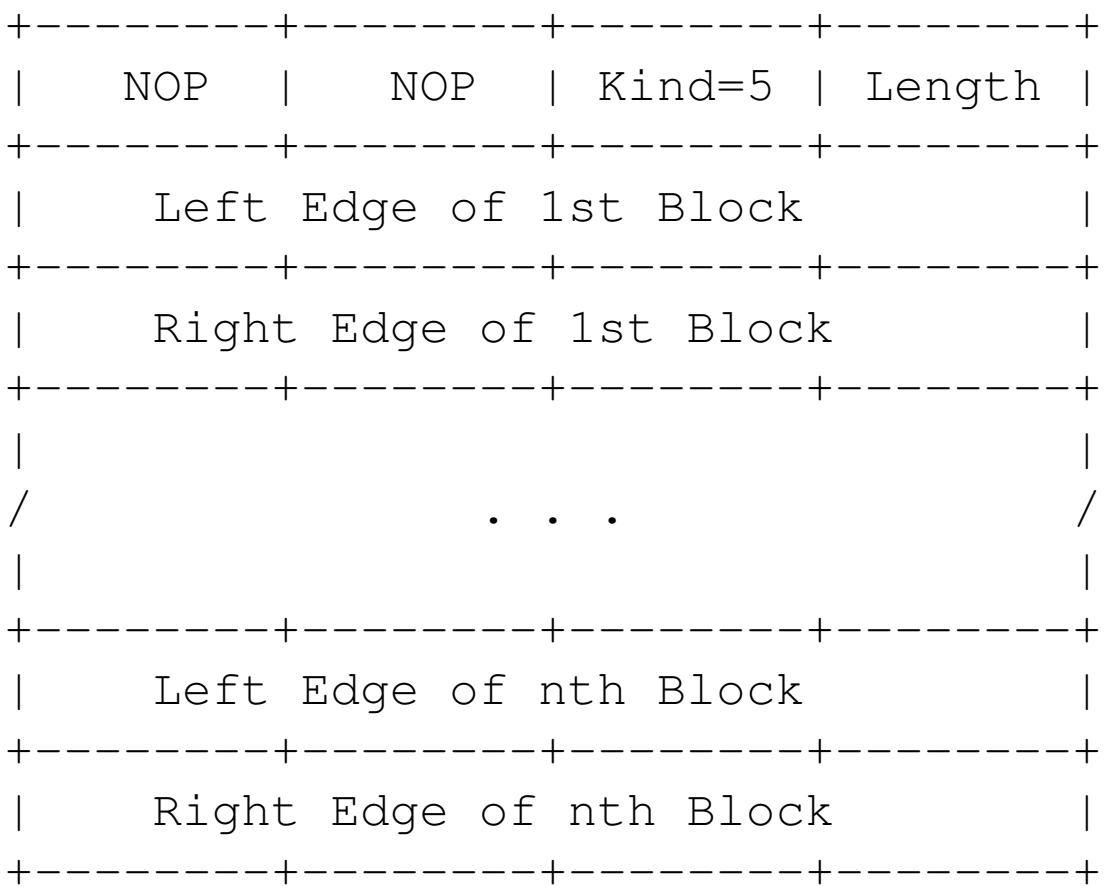
Sack Permitted Option

- ✓ Enviada con el SYN
- ✓ Habilita el uso de SACK

```
+---------+---------+
| Kind=4  | Length=2|
+---------+---------+
```

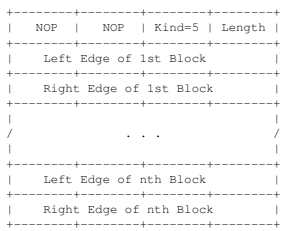
Sack Option – Estructura

Enviada por el receptor



Sack Option – Estructura

Enviada por el receptor



Los NOP se agregan para alinear la opción a palabras de 32 bits (4 bytes).
Left-edge y Right-edge indican el comienzo y fin del bloque de datos correcto que existe en el buffer de recepción de quien envía este segmento
Dado que las opciones tienen una longitud máxima de 40 bytes esto hace que no se puedan indicar más de 4 bloques.

Al recibirse se produce la retransmisión de lo necesario

Ejemplos SACK - rfc 2018

El transmisor envía 8 segmentos de 500 bytes de datos con SN:5000
Caso 1:
Se pierde el primer segmento y llegan OK los 7 restantes:

Triggering Segment	ACK	Left Edge	Right Edge
5000	(lost)		
5500	5000	5500	6000
6000	5000	5500	6500
6500	5000	5500	7000
7000	5000	5500	7500
7500	5000	5500	8000
8000	5000	5500	8500
8500	5000	5500	9000

Ejemplos SACK - rfc 2018...

El transmisor envía 8 segmentos de 500 bytes de datos con SN:5000
Caso 2:
Se pierden los segmentos pares:

Triggering Segment	ACK	1er Bloque		2do Bloque		3er Bloque	
		Left Edge	Right Edge	Left Edge	Right Edge	Left Edge	Right Edge
5000	5500						
5500	(lost)						
6000	5500	6000	6500				
6500	(lost)						
7000	5500	7000	7500	6000	6500		
7500	(lost)						
8000	5500	8000	8500	7000	7500	6000	6500
8500	(lost)						

Ejemplos SACK - rfc 2018...

Continuando con el caso anterior, supongamos que se recibe el cuarto paquete a continuación. Entonces:

Triggering Segment	ACK	1er Bloque		2do Bloque		3er Bloque	
		Left Edge	Right Edge	Left Edge	Right Edge	Left Edge	Right Edge
6500	5500	6000	7500	8000	8500		

Si luego llega el segundo segmento:

Triggering Segment	ACK	1er Bloque		2do Bloque		3er Bloque	
		Left Edge	Right Edge	Left Edge	Right Edge	Left Edge	Right Edge
5500	7500	8000	8500				

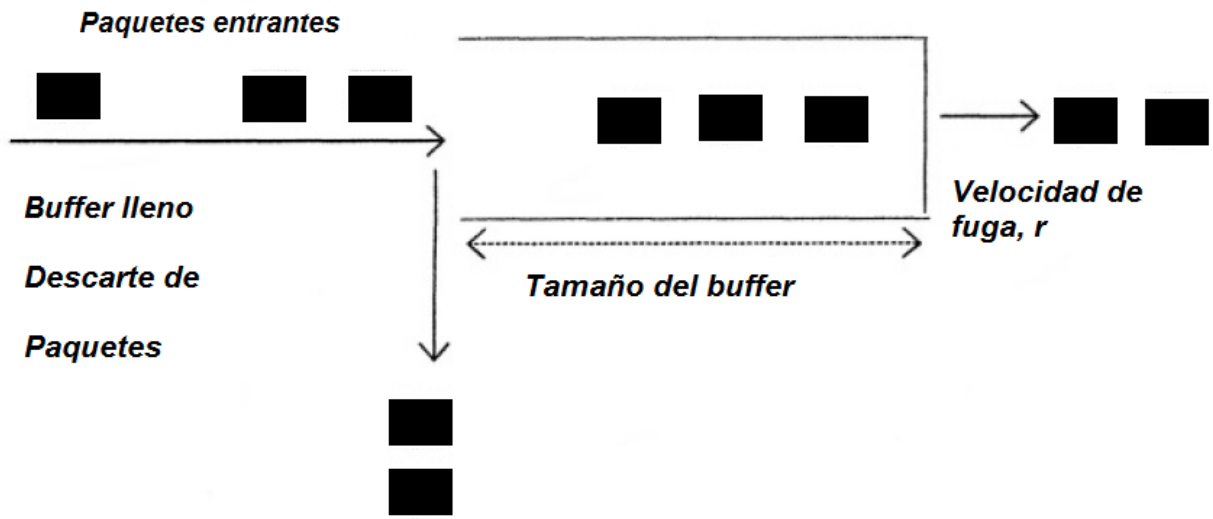
Estamos en:

1. TCP Selective Acknowledge

2. Manejo de Colas
   - Manejo Pasivo de las Colas
   - Manejo Activo de las colas
   - Random Early Detection
   - Explicit Congestion Notification

Descarte de paquetes

- ✓ Mecanismo default: tail drop
- ✓ Paquetes que desbordan se descartan automáticamente
- ✓ Simplicidad
- ✓ Presenta dos estados:
  1. Sin descarte – No hay aviso alguno
  2. 100 % de descarte – Dejan de transmitir



Descarte de paquetes

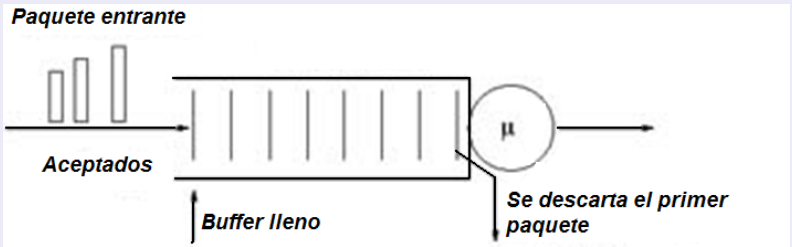
Desventaja – Sincronización global de TCP

- Frente a congestión el emisor disminuye el throughput
- La red descarta paquetes
- Las sesiones TCP reaccionan sincronizadamente
- Los recursos de red se usan ineficientemente
  - capacidad de los enlaces
  - Memoria de los routers

Descarte de paquetes

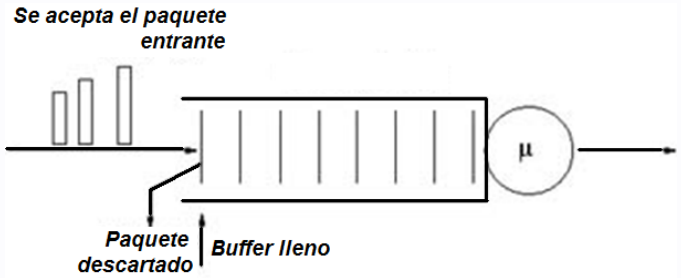
Drop-front

Se descartan los primeros paquetes de la cola



Push-out

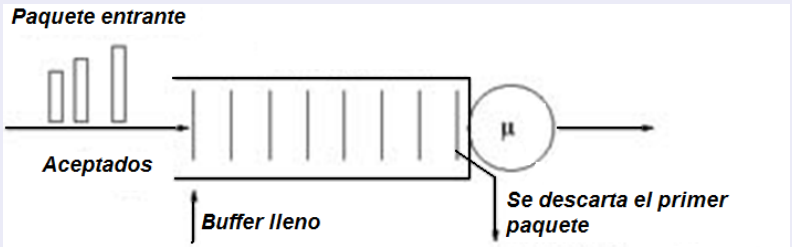
Se descarta el último paquete almacenado



Descarte de paquetes

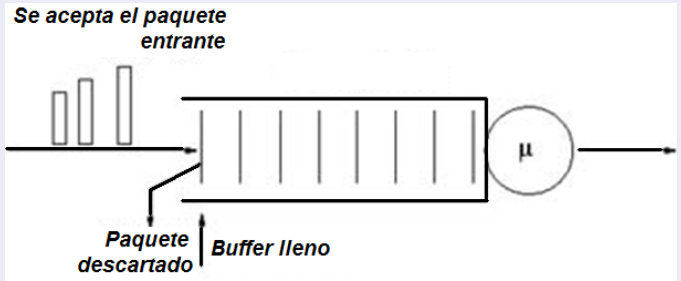
Drop-front

Se descartan los primeros paquetes de la cola



Push-out

Se descarta el último paquete almacenado



Desventajas del Modo Pasivo

- ✓ Un balance entre tamaño de buffer y QoS
- ✓ A mayor throughput mayor retardo
- ✓ Una única conexión puede adueñarse del buffer,(“lock out”)
- ✓ Problemas de “fairness”
- ✓ Buffer lleno por períodos de tiempo largos

AQM – Generalidades

- ✓ Provee acciones preventivas para el manejo del buffer.
  - El descarte de paquetes se realiza antes que el buffer se llene
  - La probabilidad de este descarte crece con el nivel de congestión
- ✓ Reduce la cantidad de paquetes descartados
- ✓ Soporta servicios interactivos de bajo retardo
  Mantiene una longitud promedio de cola baja lo que disminuye el retardo.
- ✓ Evita el “lock out ”
  Asegura que habrá buffer disponible para un paquete entrante.

AQM – Generalidades

- ✓ Provee acciones preventivas para el manejo del buffer.
  - El descarte de paquetes se realiza antes que el buffer se llene
  - La probabilidad de este descarte crece con el nivel de congestión
- ✓ Reduce la cantidad de paquetes descartados
- ✓ Soporta servicios interactivos de bajo retardo
  Mantiene una longitud promedio de cola baja lo que disminuye el retardo.
- ✓ Evita el “lock out ”
  Asegura que habrá buffer disponible para un paquete entrante.

A veces se indica que el hecho de no tener “lock out” consigue “fairness” ; pero ello requiere trabajar a nivel de flujo individual que no está provisto por AQM.
En un router con AQM pero despacho tipo FIFO dos flujos TCP pueden recibir diferentes recursos por tener diferentes RTTs y un flujo que no use control de congestión puede recibir más recursos que uno que lo hace.

Para lograrlo se deben incorporar a AQM algoritmos como FQ (“fair queueing”) o CBQ(Algoritmo basado en clases)

AQM

Previene la sincronización global de TCP

Previene la congestión

Se destacan tres métodos:

- Random Early Detection (RED)
- Weighted Random Early Detection (WRED)
- Explicit Congestion Notification (ECN)

Los métodos no son excluyentes

RED y WRED toman acciones en el router únicamente

ECN requiere participación de los usuarios

RED

- ✓ Descarta paquetes entrantes de acuerdo con una probabilidad
- ✓ La probabilidad aumenta con la longitud promedio de la cola
- ✓ Se complementa con el mecanismo de control de congestión del nivel de transporte
- ✓ No se limita al descarte, marca los paquetes

RED

- ✓ Descarta paquetes entrantes de acuerdo con una probabilidad
- ✓ La probabilidad aumenta con la longitud promedio de la cola
- ✓ Se complementa con el mecanismo de control de congestión del nivel de transporte
- ✓ No se limita al descarte, marca los paquetes

Es importante que colabore con el control de congestión del transporte porque así no requiere que todos los routers lo tengan implementado.

La inserción del mismo en una red como Internet puede hacerse gradual

RED – Algoritmo

Dos algoritmos

1. Longitud promedio de la cola
   - Filtro pasabajos con promedio ponderado exponencial
   - Determina el índice de rafagosidad que se admitirá en el “buffer ”
2. Marcado de paquetes
   - Determina la frecuencia de marcado de paquetes en función del nivel de congestión en la red
   - Se compara la longitud promedio con dos umbrales
     - Si está por debajo del menor no se marcan
     - Si está entre ambos se los marca con una probabilidad p_a
     - Si está por encima del mayor se lo marca
   - El marcado puede ser sólo eso o el descarte.

RED – Algoritmo – Longitud promedio

Initialization:

$avg \longleftarrow 0$

$count \longleftarrow -1$

for each packet arrival
calculate the new average queue size avg:
if the queue is nonempty

$avg \longleftarrow (1 - w_q)\, avg + w_q\, q$

else

$m \longleftarrow f(time - q_{time})$

$avg \longleftarrow (1 - w_q)^m\, avg$

RED – Algoritmo – Marcado de paquetes

if $min_{th} \leq avg < max_{th}$
increment count
calculate probability p_a:

$p_b \longleftarrow max_p\,(avg - min_{th})/(max_{th} - min_{th})$

$p_a \longleftarrow p_b/(1 - count \times p_b)$

with probability p_a:
mark the arriving packet

$count \longleftarrow 0$

else if $max_{th} \leq avg$
mark the arriving packet

$count \longleftarrow 0$

else $count \longleftarrow -1$
when queue becomes empty

$q_{time} \longleftarrow time$

Variaciones del marcado

- Medir la longitud de la cola en bytes en vez de paquetes
- La longitud refleja el retardo promedio en el router
- Se deben hacer cambios:

$$p_b \longleftarrow max_p\,(avg - min_{th})/(max_{th} - min_{th})$$

$$p_b \longleftarrow p_b\,PacketSize/MaximumPacketSize$$

$$p_a \longleftarrow p_b/(1 - count \times p_b)$$

- Es más probable que se marque un paquete FTP que uno de Telenet

RED-Algoritmo-Parámetros

Variables:

avg: longitud promedio de la cola
q_{time}: Comienzo del tiempo ocioso de la cola
$count$: Paquetes desde la última marcación

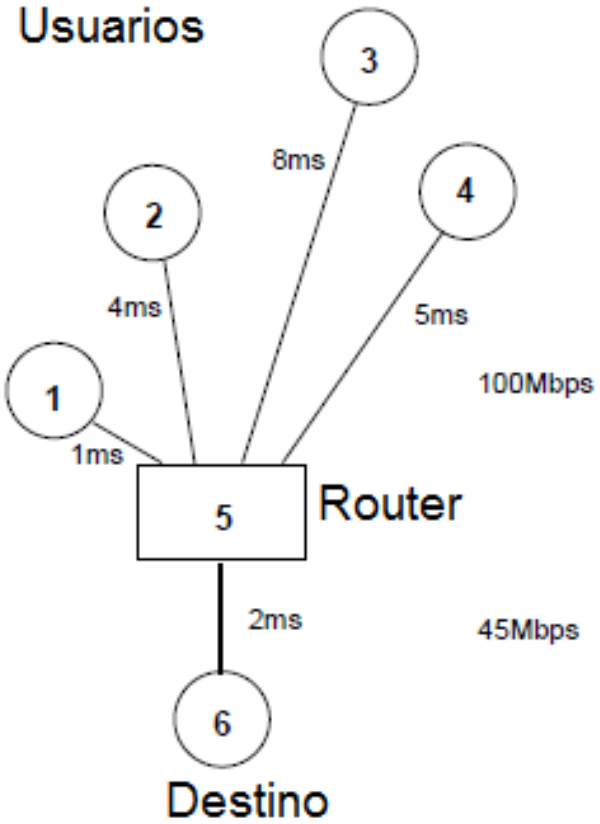
Constantes:

w_q: peso de la cola
min_{th}: Umbral mínimo
max_{th}: Umbral máximo de la cola
max_p: valor máximo para p_b

Otros:

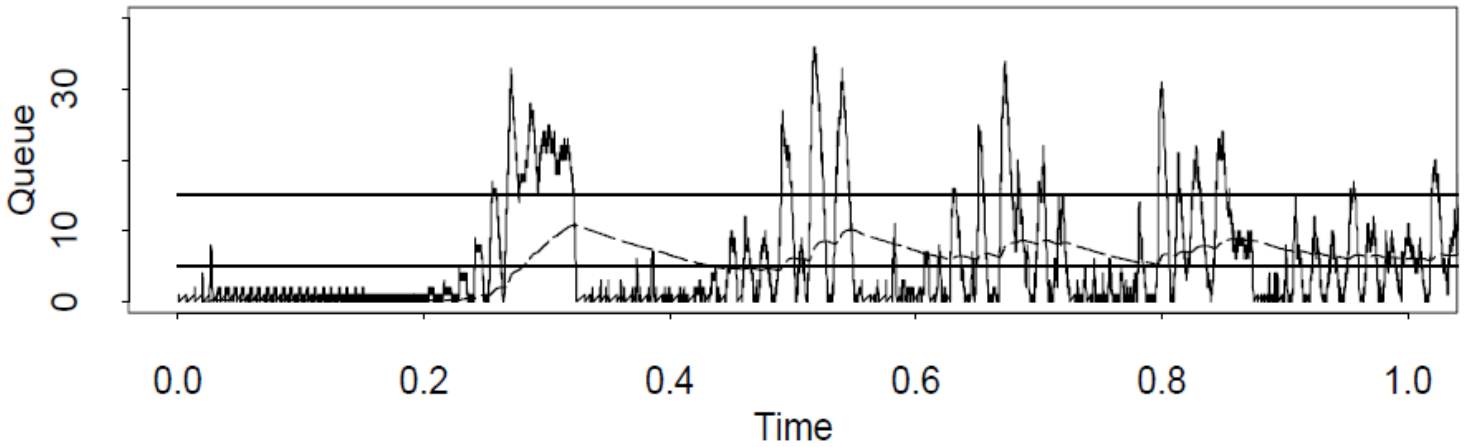
p_a: probabilidad de marcado
q: longitud actual de la cola
$time$: tiempo actual
$f(t)$: función lineal del tiempo t

Escenario de pruebas



- Long. paquete = 1000 bytes
- $w_q = 0.002$
- $min_{th} = 5\ paquetes$
- $max_{th} = 15\ paquetes$
- $max_p = 1/50$

Resultados



- Línea punteada: Longitud promedio
- Línea llena: Longitud real
- Líneas horizontales: Umbrales

RED - Probabilidad de marcado

$$p_b \longleftarrow max_p(avg - min_{th})/(max_{th} - min_{th})$$

Consideramos la *VA* , X el número de paquetes que llegan entre dos marcas sucesivas
Si se marcan los paquetes con prob p_b entonces:

$$Prob[X = n] = (1 - p_b)^{n-1} p_b$$

Resulta una distribución geométrica, generándose marcas en forma de ráfagas y con intervalos largos entre marcas.
Mejor alternativa es que X sea uniformemente aleatoria en el intervalo $\{1, 2, \ldots, 1/p_b\}$
Entonces:

$$p_a = \frac{p_b}{1 - count \times p_b}$$

RED - Implementación

- ✓ *avg* se calcula en cada arribo
- ✓ Se debe recalcular cuando llega a una cola vacía
- ✓ Se estima un nro de paquetes que debería haber llegado en el intervalo en que la cola estuvo vacía.

$$m = \frac{time - q_{time}}{s}$$

donde s es el tiempo de inserción de un paquete pequeño

$$avg \longleftarrow (1 - w_q)^m avg$$

RED - Implementación

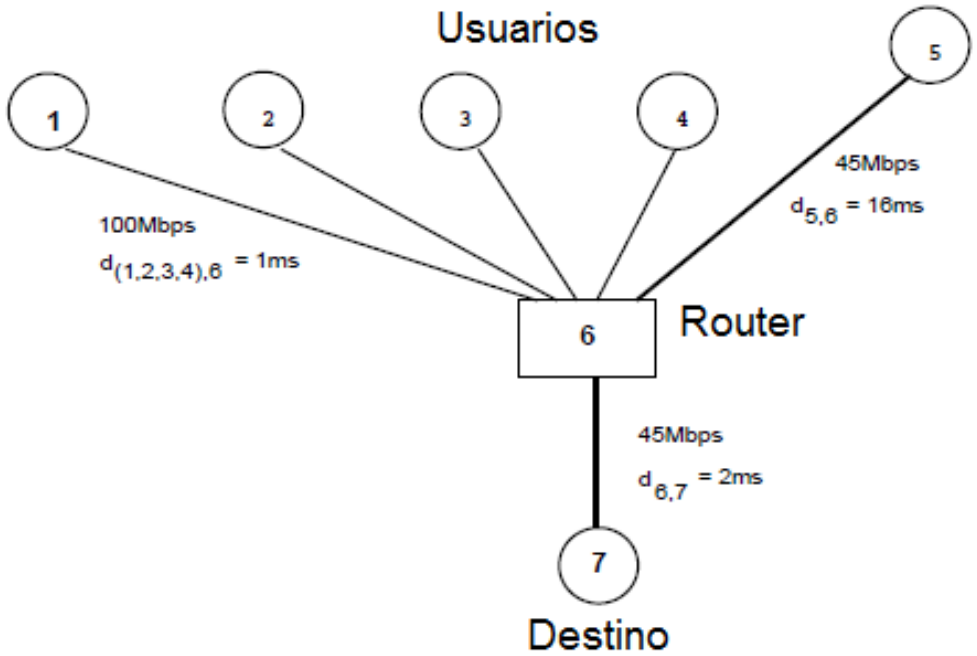
- ✓ Si *avg* está dentro de las cotas se debe marcar con probabilidad
- ✓ ¿Qué paquete se marca?
- ✓ $R = Random[0, 1]$
- ✓ Se marca si:

$$R < \frac{p_b}{1 - count \times p_b}$$

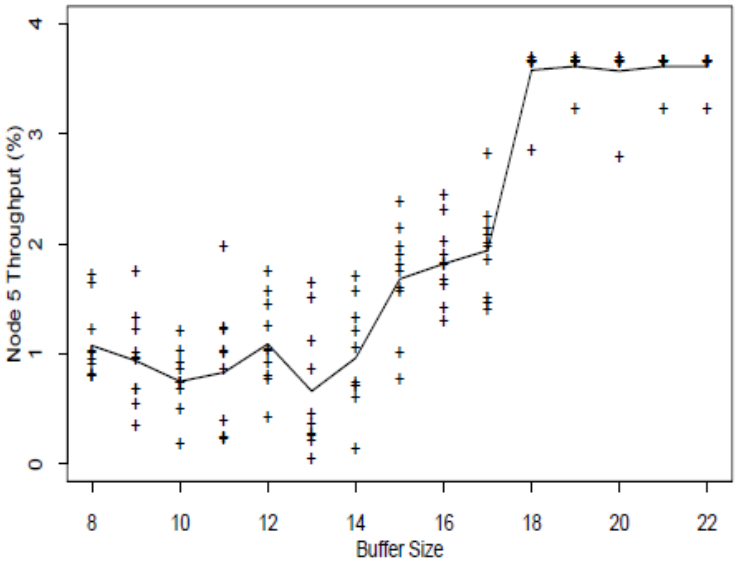
RED - Algoritmo optimizado

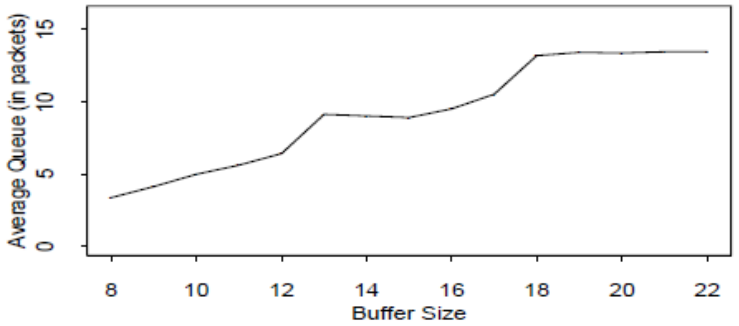
```
Initialization:
    avg ← 0
    count ← −1
for each packet arrival:
    calculate the new average queue size avg:
        if the queue is nonempty
            avg ← avg + w_q (q − avg)
        else using a table lookup:
            avg ← (1 − w_q)^((time−q_time)/s) avg
    if min_th ≤ avg < max_th
        increment count
        p_b ← C_1 · avg − C_2
        if count > 0 and count ≥ Approx[R/p_b]
            mark the arriving packet
            count ← 0
        if count = 0 (choosing random number)
            R ← Random[0,1]
    else if max_th ≤ avg
        mark the arriving packet
        count ← −1
    else count ← −1
when queue becomes empty
    q_time ← time
```

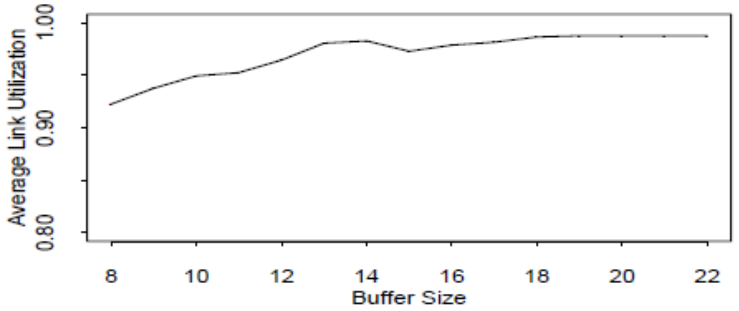
RED – Simulaciones



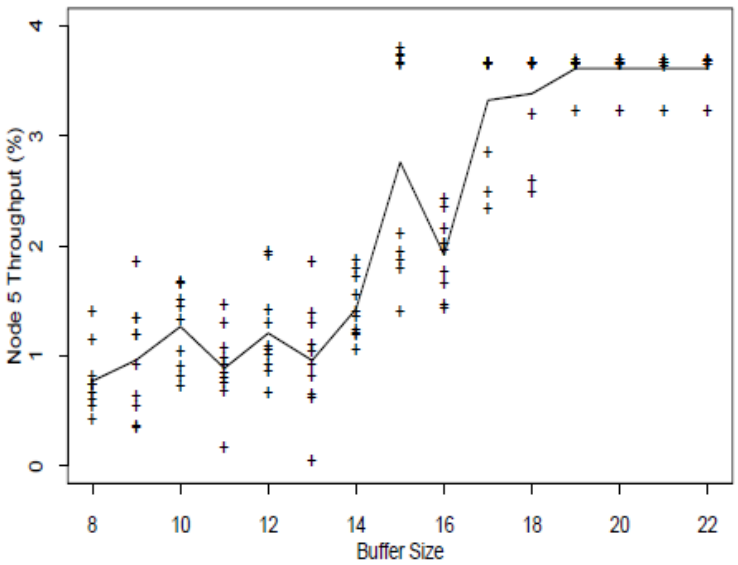
RED – Simulaciones – Drop Tail

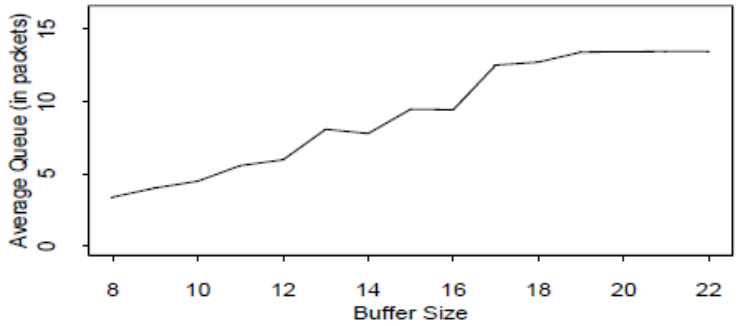


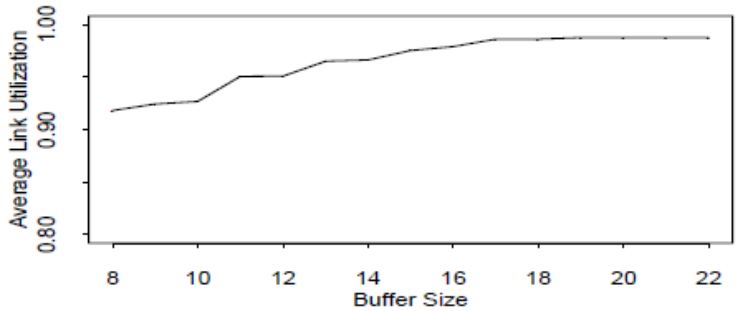




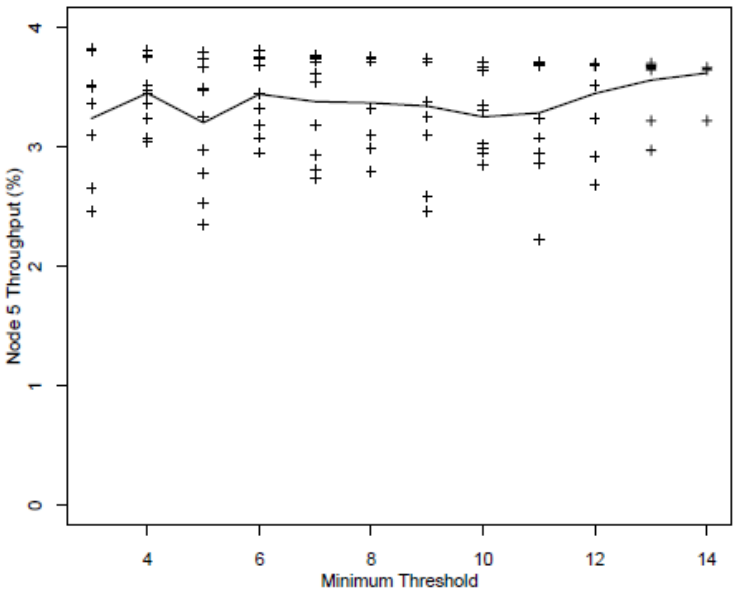
RED – Simulaciones – Drop Tail Random

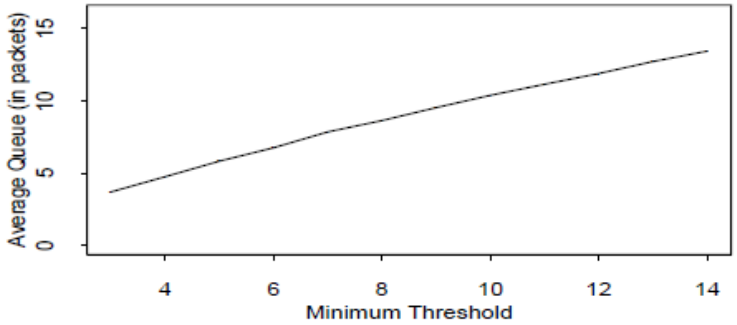


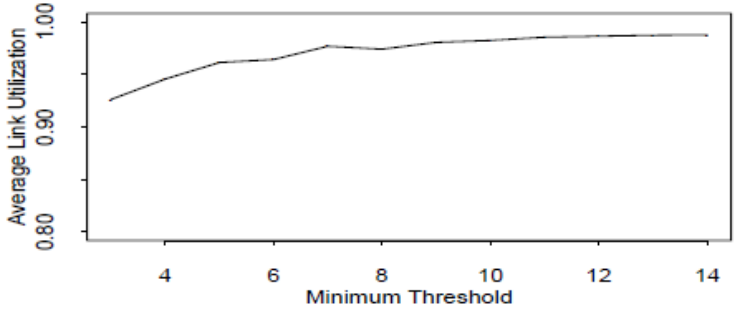




RED – Simulaciones – RED







RED y el tamaño del paquete

RED_1	*RED_2*
$count \leftarrow count + 1$	$count \leftarrow count + 1$
$p_b \leftarrow \max_p \cdot \frac{avg - \min_{th}}{\max_{th} - \min_{th}}$	$p_b \leftarrow \max_p \cdot \frac{avg - \min_{th}}{\max_{th} - \min_{th}}$
-	$p_b \leftarrow p_b \cdot L / M$
$p_a \leftarrow p_b / (1 - count \cdot p_b)$	$p_a \leftarrow p_b / (1 - count \cdot p_b)$

RED_3	*RED_4*	*RED_5*
$count \leftarrow count + 1$	-	-
$p_b \leftarrow \max_p \cdot \frac{avg - \min_{th}}{\max_{th} - \min_{th}}$	$p_b \leftarrow \max_p \cdot \frac{avg - \min_{th}}{\max_{th} - \min_{th}}$	$p_b \leftarrow \max_p \cdot \frac{avg - \min_{th}}{\max_{th} - \min_{th}}$
-	-	-
$p_a \leftarrow \frac{p_b \cdot L}{(1 - count \cdot p_b) \cdot M}$	$p_a \leftarrow \frac{p_b \cdot L}{(1 - count \cdot p_b) \cdot M}$	$p_a \leftarrow \frac{p_b}{(1 - count \cdot p_b)} \cdot (\frac{L}{M})^2$
-	$count \leftarrow count + \frac{L}{M}$	$count \leftarrow count + (\frac{L}{M})^2$

- *L* es el tamaño del paquete
- *M* es el MSS

RED – Más variantes

- ✓ RED 3 Condiciona la probabilidad de descarte final en función del tamaño de paquete
- ✓ RED 4 Linealiza la probabilidad de descarte
- ✓ RED 5 Optimiza el “goodput”(throughput a nivel de aplicación)

$$\text{goodput} \leq \frac{MSS \times C}{RTT\sqrt{p}}$$

Escenario de pruebas

- ✓ 3 grupos
- ✓ 20 sesiones por grupo
- ✓ Enlace de 30 Mbps
- ✓ Retardo del enlace: 15 y 80 mseg
- ✓ MTU: 1500,750 y 375 bytes

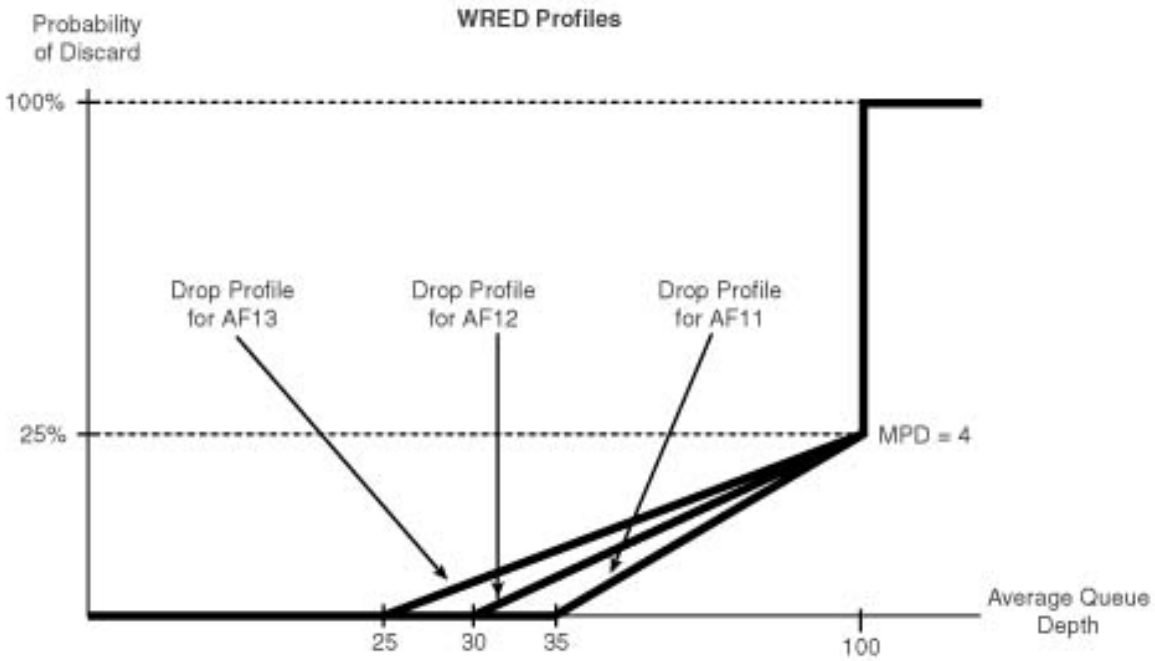
Resultados

		PLR (%)					Goodput (Mbits/s)				
Low propagation delay	*MTU*	*RED_1*	*RED_2*	*RED_3*	*RED_4*	*RED_5*	*RED_1*	*RED_2*	*RED_3*	*RED_4*	*RED_5*
	small	2.40	0.33	1.03	0.84	0.34	3.85	13.16	6.75	7.17	10.54
	medium	2.37	1.36	1.85	1.75	1.36	8.25	12.43	9.81	9.70	10.32
	large	2.41	13.65	3.57	3.49	5.22	16.07	1.29	11.20	10.85	6.42
Sum							28.17	26.88	27.76	27.71	27.27
Large propagation delay	*MTU*	*RED_1*	*RED_2*	*RED_3*	*RED_4*	*RED_5*	*RED_1*	*RED_2*	*RED_3*	*RED_4*	*RED_5*
	small	0.80	0.12	0.34	0.27	0.10	3.75	9.03	5.69	5.90	9.06
	medium	0.78	0.41	0.63	0.55	0.47	8.06	11.15	8.85	9.09	9.28
	large	0.74	2.72	1.07	1.06	1.73	15.84	6.84	12.92	12.47	8.73
Sum							27.65	27.01	27.46	27.46	27.08

Weighted Random Early Detection(WRED)

- ✓ Equilibrar las probabilidades de descarte para los diferentes tráficos
- ✓ Proveer QoS tipo Diffs
- ✓ Actuar RED a nivel de clase de tráfico

WRED – Probabilidades



1 TCP Selective Acknowledge

2 Manejo de Colas

- Manejo Pasivo de las Colas
- Manejo Activo de las colas
- Random Early Detection
- Explicit Congestion Notification

Generalidades

- ✓ Retomar el modelo basado en la red para control de congestión (FR, ATM)
- ✓ Participan IP y TCP
- ✓ Colabora con RED
- ✓ RED procede al marcado de IP
- ✓ TCP interpreta y completa la acción de control

ECN – IP

IP Header

```
0     1     2     3     4     5     6     7
+-----+-----+-----+-----+-----+-----+-----+-----+
|          DS FIELD, DSCP                 | ECN FIELD |
+-----+-----+-----+-----+-----+-----+-----+-----+
        DSCP: differentiated services codepoint
        ECN:  Explicit Congestion Notification

      +-----+-----+
      | ECN FIELD |
      +-----+-----+
        ECT   CE
         0     0          Not-ECT
         0     1          ECT(1)
         1     0          ECT(0)
         1     1          CE
```

ECN

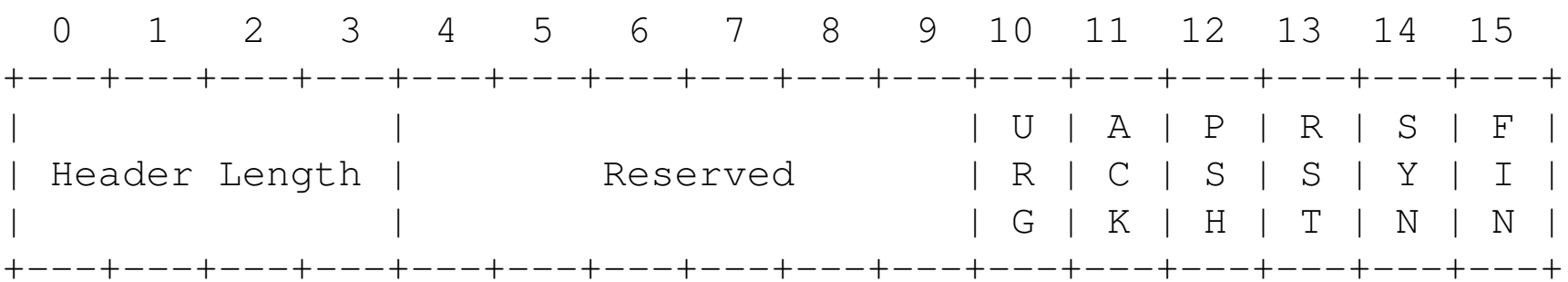
- ✓ Not-ECT indica que no se soporta ECN
- ✓ ECT(1)(01) y ECT(0)(10) son activados por los extremos
- ✓ ECT : ECN Capable Transport
- ✓ CE (11) es activado por el router para indicar congestión a los extremos
- ✓ CE :Congestion Experienced

ECN – Funcionalidad de TCP

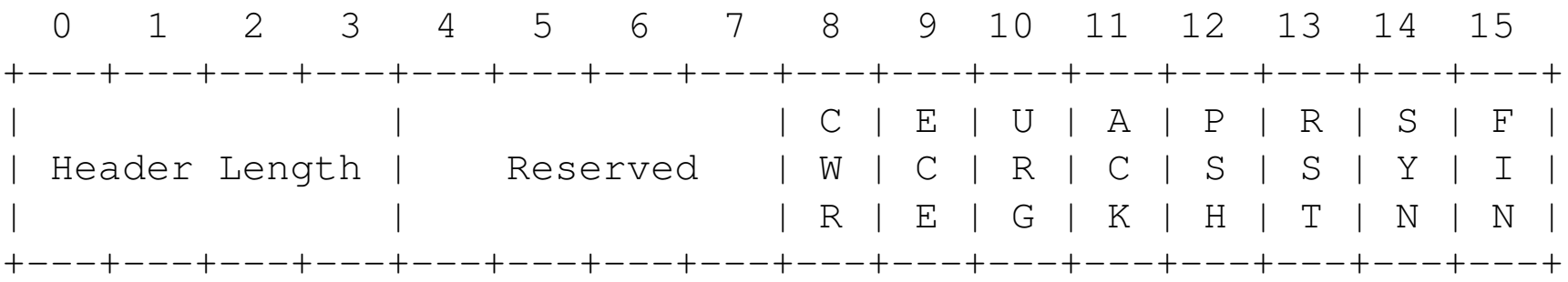
- ✓ Negociación en el set-up para acordar soporte de ECN
- ✓ Activar el bit ECN-Echo (ECE) por parte del receptor. Avisa al emisor la recepción de un CE.
- ✓ Activar el bit CWR (Congestion Window Reduced) para informar al receptor que la ventana de congestión se redujo a la mitad

Header TCP

Antes



Ahora



ECN – Secuencia de Eventos

- ✓ Se activa ECT “codepoint”en paquetes transmitidos por el emisor indicando que se soporta ECN.
- ✓ Un router ECN detecta congestión y detecta que el ECT “codepoint”está activo en el paquete que está pronto a marcar (antes lo descartaba).
- ✓ El router activa el CE y forwardea el paquete.

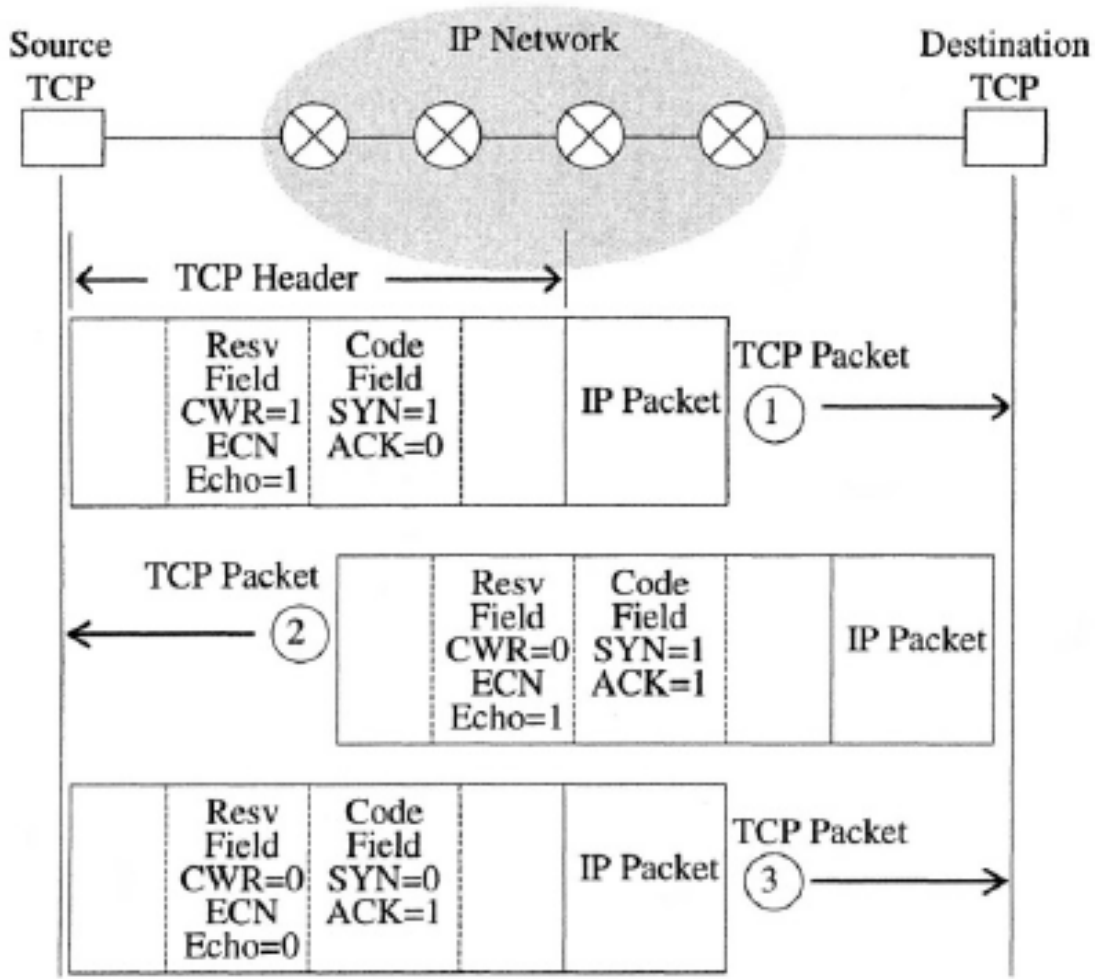
ECN – Secuencia de Eventos

- ✓ El receptor recibe el paquete con CE activo y activa el flag ECN-E en el header del próximo paquete TCP que enviará al emisor.
- ✓ El emisor recibe el paquete TCP con el bit de Flag ECN-Echo activo y actúa como si hubiera detectado un paquete perdido.
  Acorde con su mecanismo de control de congestión.
- ✓ El emisor activa el Flag de CWR en el header TCP del próximo paquete que envía al receptor.

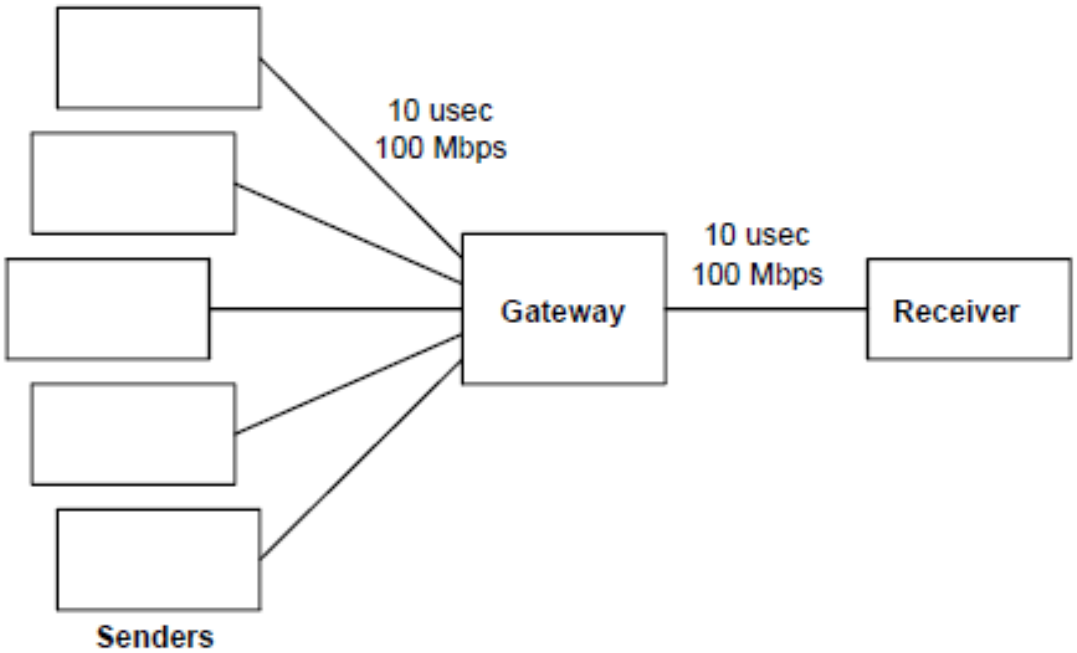
ECN – Set-up

- ✓ Suponemos un set-up de A a B
- ✓ Un paquete de SYN con los flags ECE y CWR activos se lo llama paquete de SYN ECN set-up
- ✓ Un paquete de SYN con a lo sumo uno de los dos bits activos (ECE o CWR) lo llamamos de NO-SYN ECN set-up
- ✓ Un paquete de SYN-ACK con el bit de ECE activo pero no el de CWR se lo considera un paquete de SYN-ACK ECN set-up.
- ✓ Cualquier otra combinación será NO-SYN-ACK ECN set-up

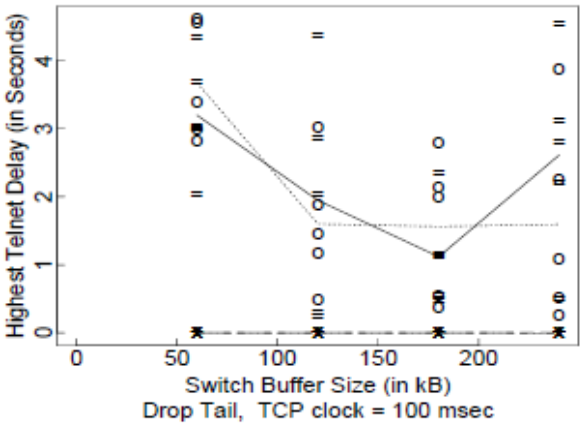
ECN – Set-up

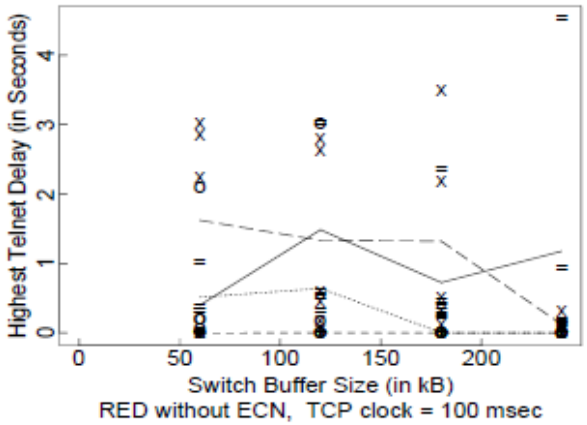

Source
TCP
IP Network
Destination
TCP
TCP Header
Resv
Field
CWR=1
ECN
Echo=1
Code
Field
SYN=1
ACK=0
IP Packet
TCP Packet
1
TCP Packet
2
Resv
Field
CWR=0
ECN
Echo=1
Code
Field
SYN=1
ACK=1
IP Packet
Resv
Field
CWR=0
ECN
Echo=0
Code
Field
SYN=0
ACK=1
IP Packet
TCP Packet
3

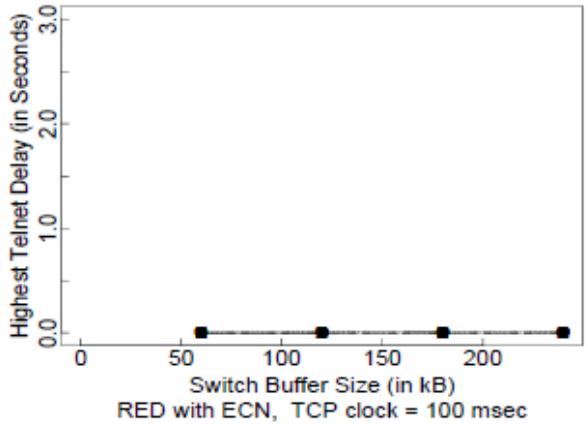
Escenario de pruebas



Resultados


Highest Telnet Delay (in Seconds)
0
1
2
3
4
0
50
100
150
200
Switch Buffer Size (in kB)
Drop Tail, TCP clock = 100 msec


Highest Telnet Delay (in Seconds)
0
1
2
3
4
0
50
100
150
200
Switch Buffer Size (in kB)
RED without ECN, TCP clock = 100 msec


Highest Telnet Delay (in Seconds)
0.0
1.0
2.0
3.0
0
50
100
150
200
Switch Buffer Size (in kB)
RED with ECN, TCP clock = 100 msec