Tema 1: QoS
Quality of Service

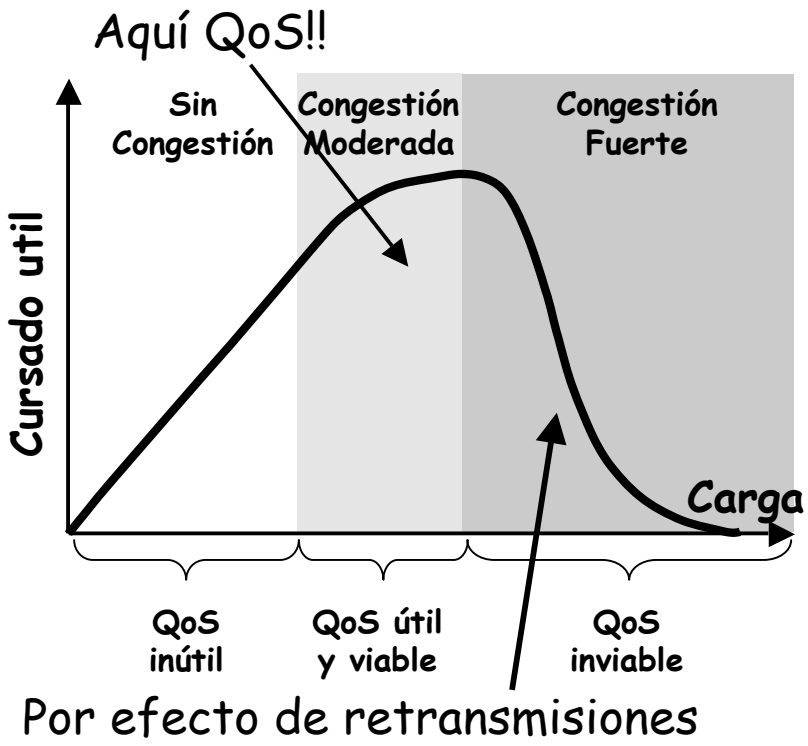
Los principios

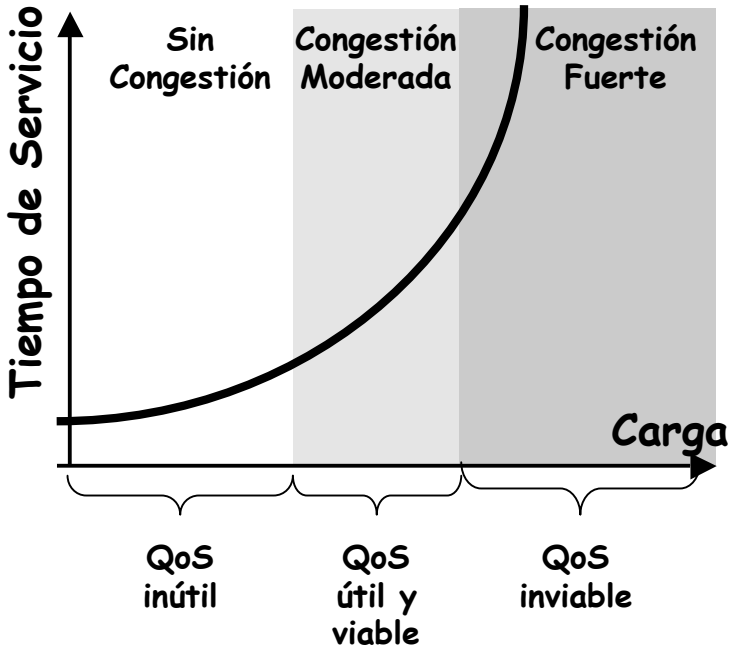
¿Qué es QoS?

- Se refiere a la habilidad de la red de *diferenciar* a unos determinados tipos de tráfico, probablemente de unos servicios concretos
- Ofrecer recursos a clases de alta prioridad *a costa de* las de baja
- Parámetros típicos:
  - Ancho de banda
  - Retardo temporal y variación en el retardo (*jitter*)
  - Probabilidad de error (o pérdida de paquetes o fiabilidad)
- Directamente relacionado con:
  - Tamaño de colas
  - Congestión de la red
  - Velocidad de conmutación
  - Ancho de banda de los enlaces
- QoS provee mejores y más predecibles servicios a la red mediante:
  - Soporte de ancho de banda dedicado
  - Mejorando las características de pérdida de paquetes
  - Evitando y manejando la congestión de la red
  - Organizando el tráfico
  - Introduciendo prioridades de tráfico a lo largo de la red

Congestión y Calidad de Servicio

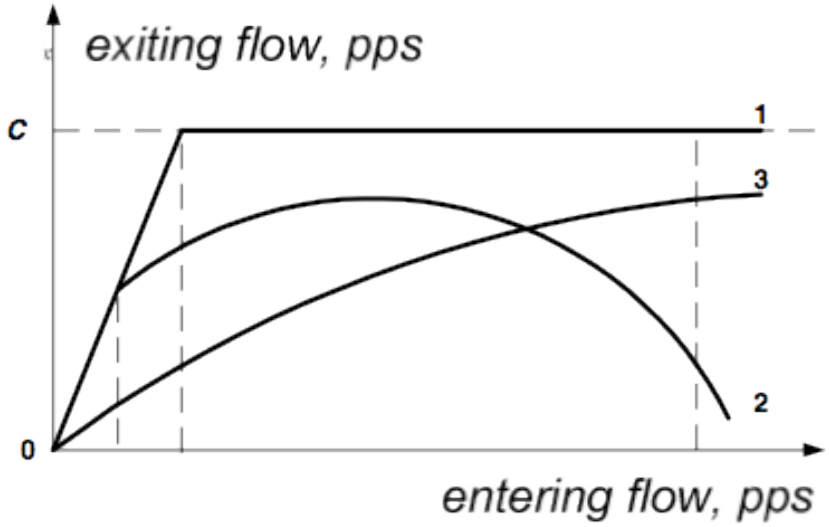
- Fácil dar QoS si nunca hay congestión
- Para ello habría que sobredimensionar todos los enlaces
- Para dar QoS con congestión es preciso tener mecanismos que permitan dar un trato distinto al tráfico preferente

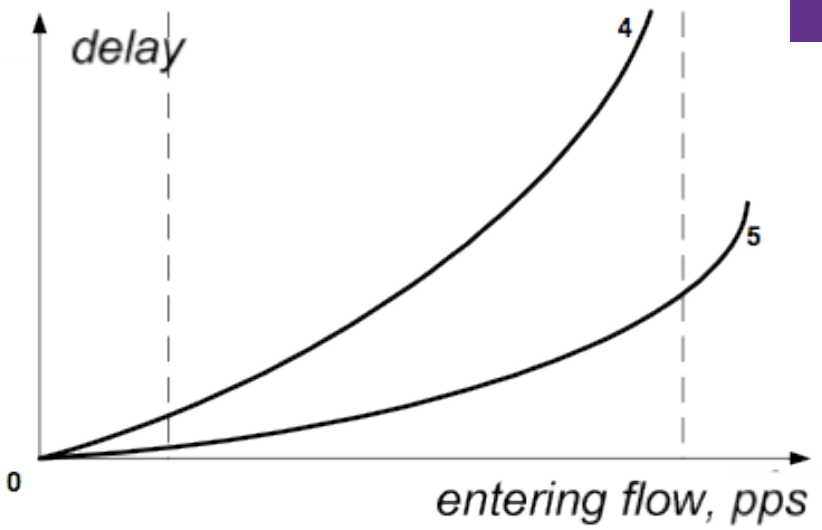




Congestión y Calidad de Servicio

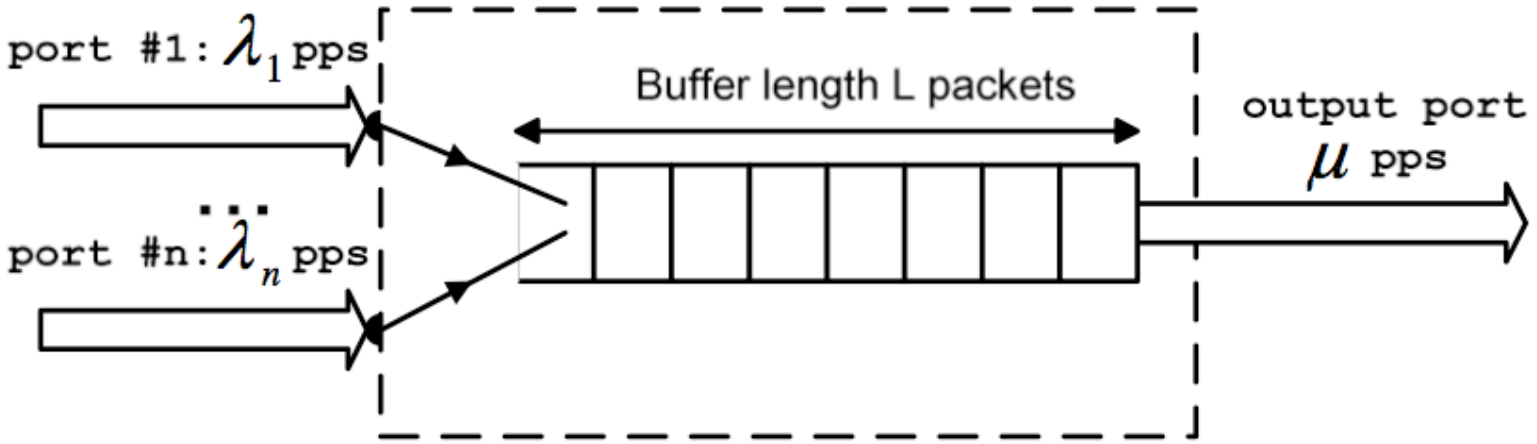
- Con y sin control de congestión
  - Flujo en el enlace y retardo en la cola
  - 1: Funcionamiento ideal
  - 2 y 4: sin control de congestión
  - 3 y 5: con control de congestión
- Control de congestión
  - Por los nodos extremo para tráfico elástico (TCP)
  - Por la red para tráfico no elástico





Servicio *Best Effort* (BE)

- Se trata igual a todo el tráfico
  - Sin separación entre flujos
  - Sin diferenciación entre paquetes
- Ante congestión
  - Crecen los retardos sin control
  - Pérdidas sin control



Ya ya, sí lo de siempre…

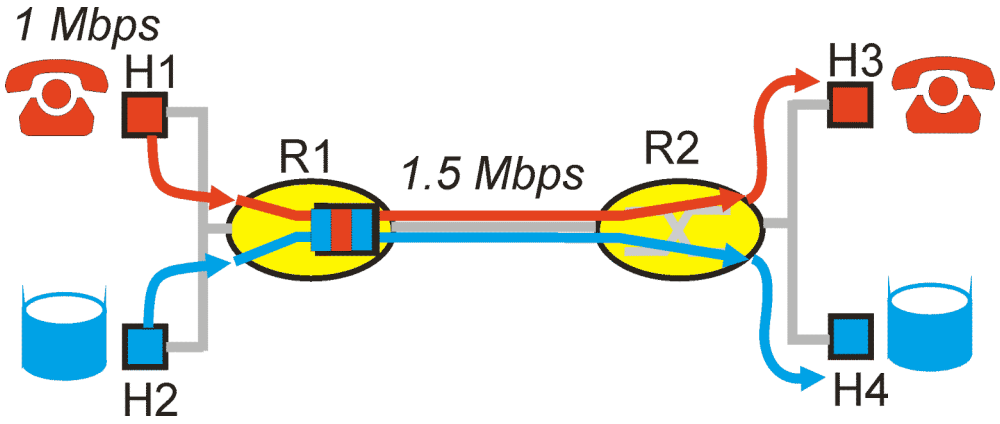
- Solución: ***Oversubscribe*** *!!*
- *Through more bandwidth at the problem !!*
- No siempre es la solución:
  - No siempre es barato aumentar el BW (ej: acceso, *peering*)
  - Si se le da más BW al usuario habrá mayor demanda
  - Un pico en la demanda degradaría la calidad de servicios sensibles
  - ¡¡ El pico puede ser de tráfico *scavenger* !! (DoS, worm… bastan 10 PCs para saturar 1GEth)
- Hoy en día ya no es suficiente para un ISP con ofrecer un servicio *Best Effort*
- La tecnología para ofrecer QoS ya es asequible y fiable
- ¡ Pero la ingeniería de estas redes no es sencilla !

Elementos

- *Connection Admission Control (CAC)*
  - *¿Puede la red cursar el nuevo flujo de tráfico manteniendo los parámetros de QoS ofrecidos a todos los usuarios?*
- *QoS routing / Traffic Engineering*
- Clasificación
  - ¿Cómo distinguir entre flujos?
- Monitorización
  - Analizar la cantidad de tráfico que entra en la red
- *Traffic shaping* y *policing*
  - Marcar, descartar o retrasar el tráfico en exceso
- Gestión de cola
  - ¿Qué paquetes tirar si se llena?
- Planificación de recursos (*scheduling*)
  - El recurso normalmente es el enlace
  - ¿Cómo organizar a los paquete que deben enviarse?
  - ¿Dar prioridades? ¿Repartir la capacidad?

Principio 1: Clasificación

- Ejemplo: Teléfono IP a 1Mbps, comparte enlace de 1.5Mbps con FTP
  - Ráfagas de FTP pueden congestionar el router y causar fallos en el audio
  - Queremos dar prioridad al audio sobre el FTP

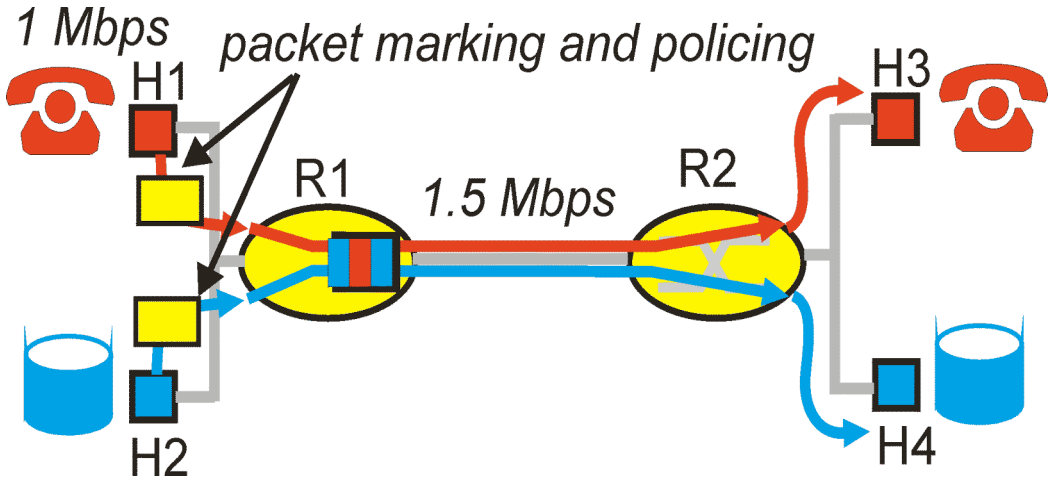


Principio 1

Los routers necesitan distinguir el tráfico de diferentes clases y aplicarles diferentes políticas: *packet marking* (generalmente a la entrada a la red)

Principio 2: Aislamiento

- ¿Qué sucede si las aplicaciones no se comportan como deben?
  - Por ejemplo la aplicación de audio envía más de lo previsto
  - Necesitamos forzar que las fuentes se comporten como se ha acordado

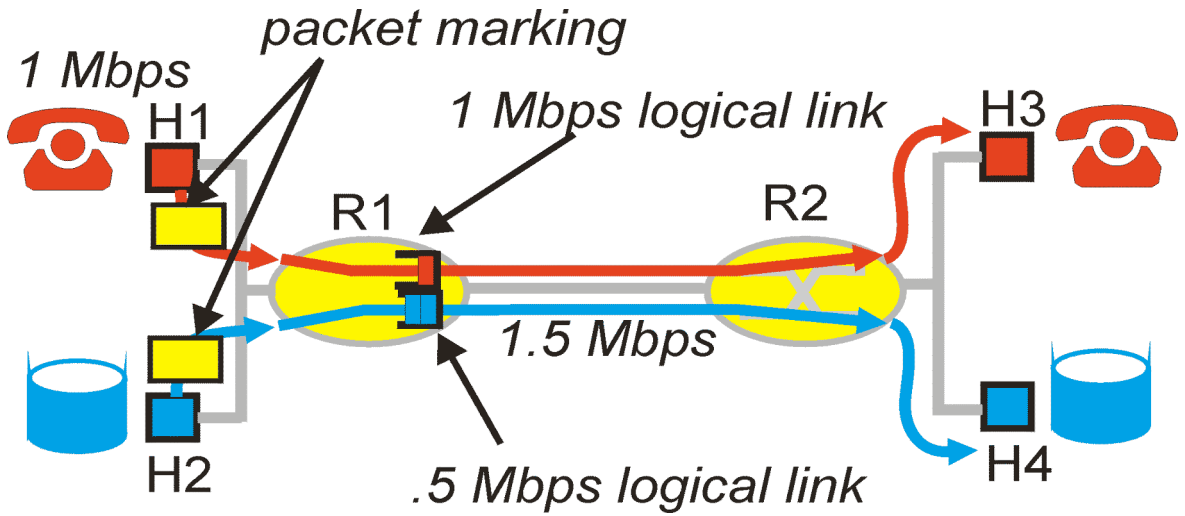


Principio 2

Forzar que una clase de tráfico se comporte dentro de lo contratado: *policing* (tipicamente a la entrada)

Principio 3: Eficiencia

- Reservar BW fijo (no compartido) a un flujo es ineficiente si ese flujo no lo emplea todo

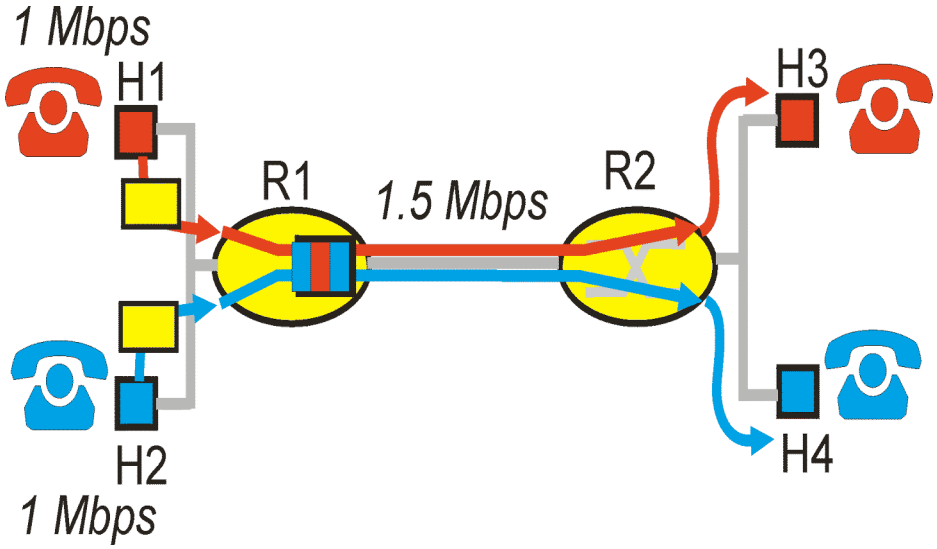


Principio 3

Mientras se ofrece aislamiento es deseable emplear los recursos de forma eficiente (*work conserving*): *scheduling* (en todos los routers del camino)

Principio 4: Límite de recursos

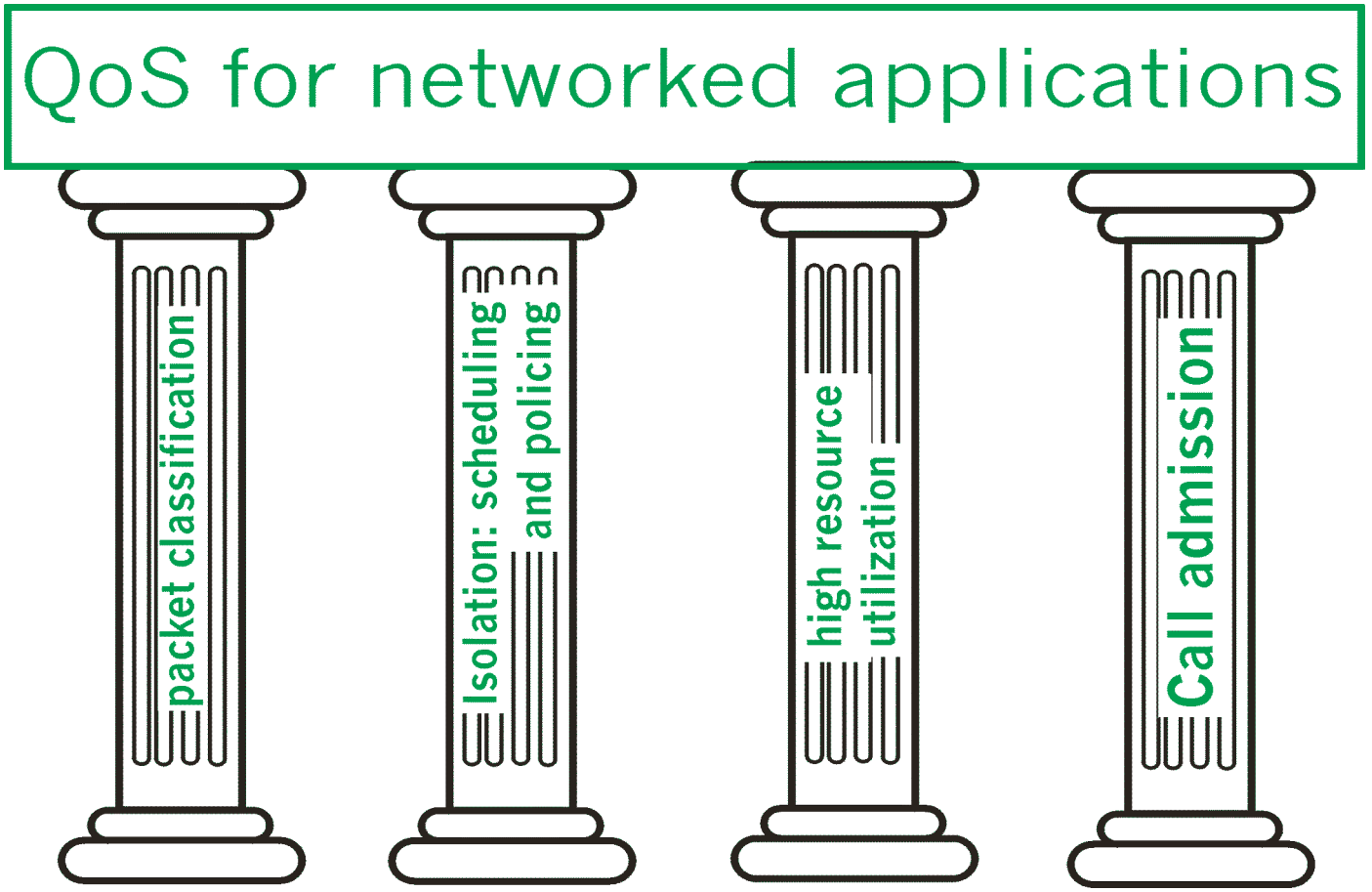
- No se pueden satisfacer las demandas más allá de la capacidad del enlace



Principio 4

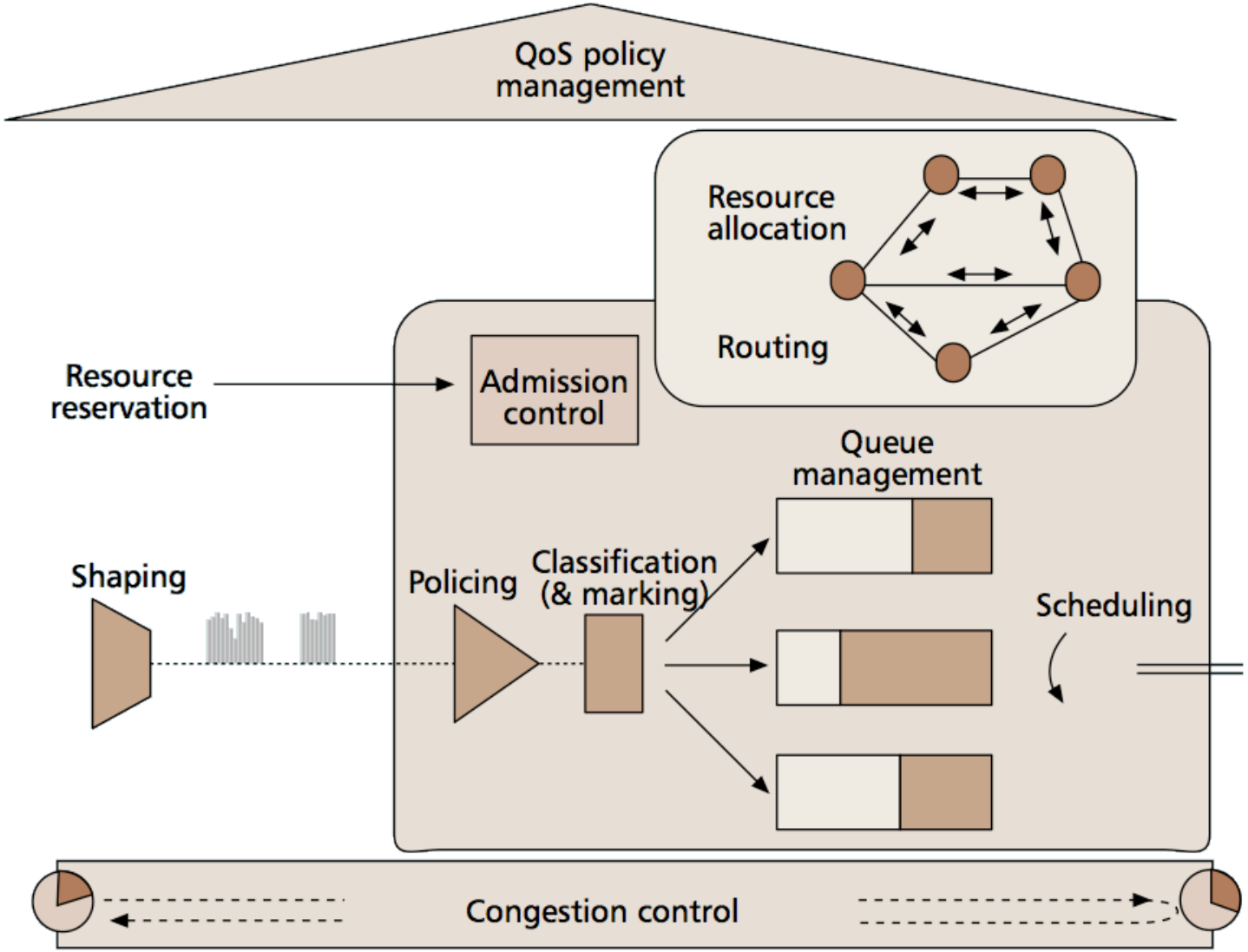
El flujo declara sus necesidades pero la red puede *bloquear* al flujo si no puede satisfacerlas: *call admission*

Summary of QoS Principles



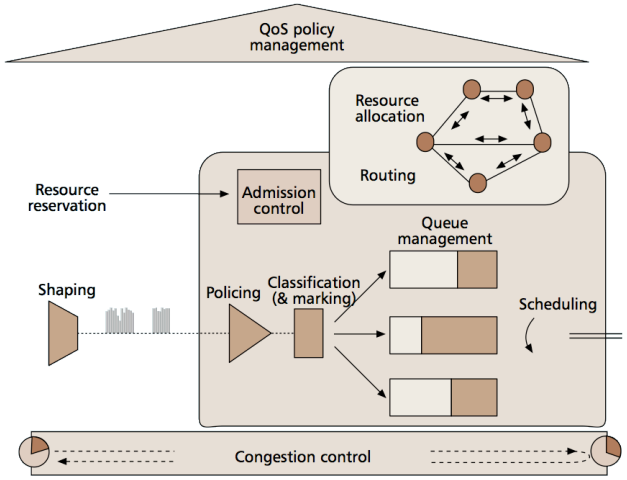
Localización de los elementos

- Router, switch o similar

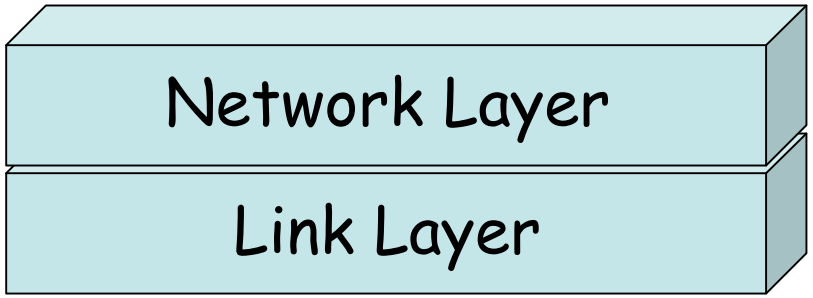


Localización de los elementos

- Router, switch o similar



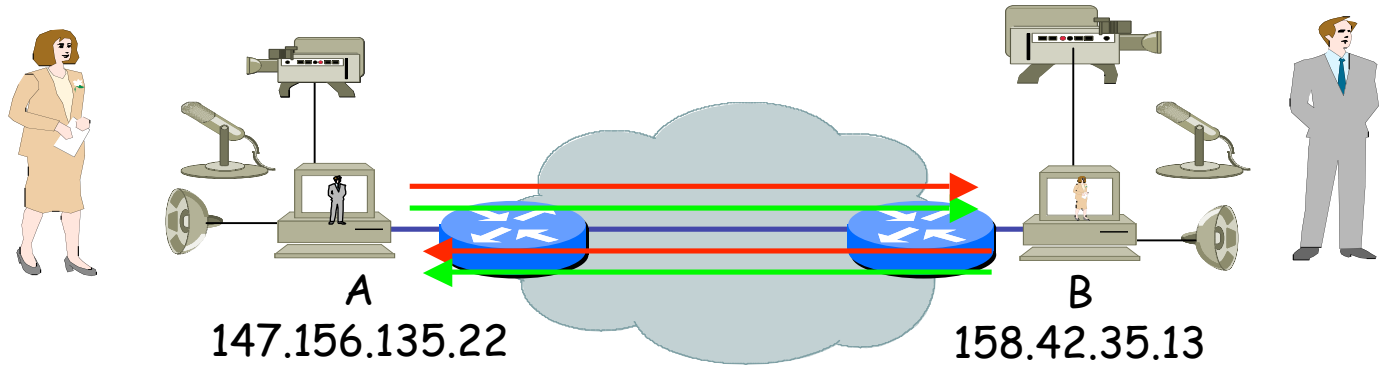
- Van a tener sentido en más de una capa



Clasificación y marcado

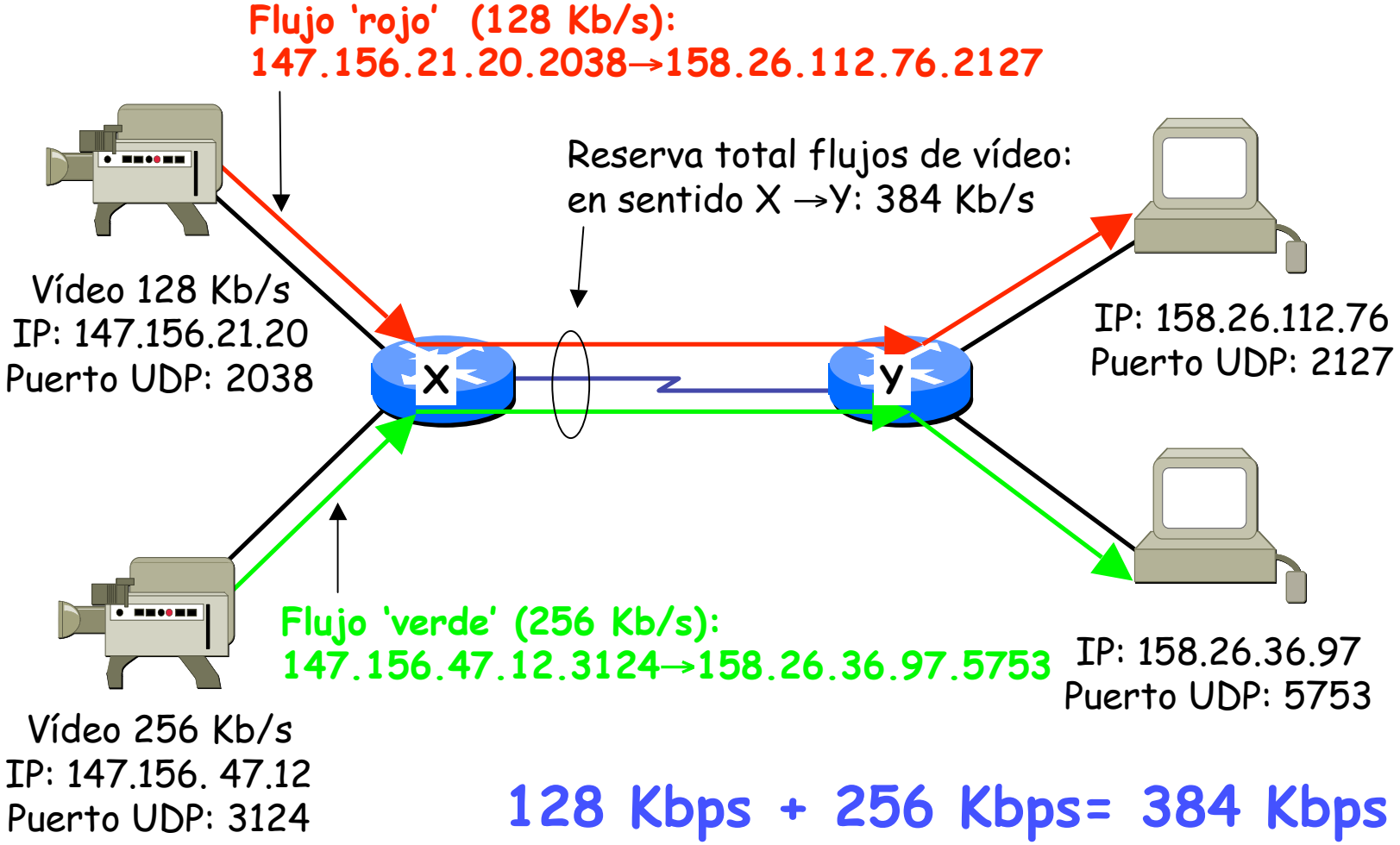
Concepto de flujo en QoS

- Secuencia de datagramas que se produce como resultado de una acción del usuario y requiere la misma QoS
- Normalmente es simplex (unidireccional)
- Es la entidad más pequeña a la que los routers pueden aplicar una determinada QoS
- Ejemplo: una videoconferencia estaría formada por cuatro flujos, dos en cada sentido, uno para el audio y otro para el vídeo.
- Los flujos pueden agruparse en clases; todos los flujos dentro de una misma clase reciben la misma QoS.



Flujo vídeo A->B: 147.156.135.22:2056 -> 158.42.35.13:4065
Flujo audio A->B: 147.156.135.22:3567 -> 158.42.35.13:2843
Flujo vídeo B->A: 158.42.35.13:1734 -> 147.156.135.22:6846
Flujo vídeo B->A: 158.42.35.13:2492 -> 147.156.135.22:5387

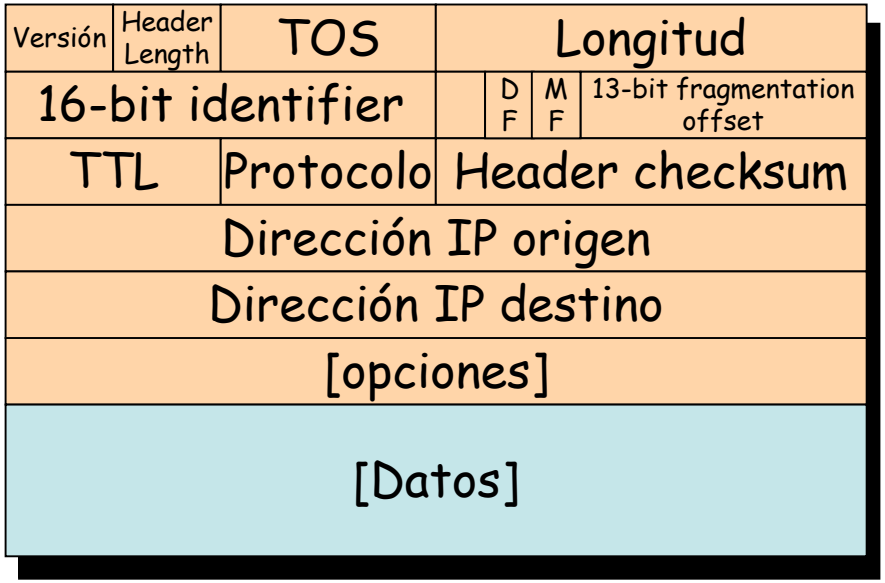
Agrupación de flujos o clases en vídeo



128 Kbps + 256 Kbps= 384 Kbps

Identificación/clasificación de flujos

- Marcar al paquete como perteneciente a un flujo
- O marcarlo como perteneciente a una clase (prioridad)
- En IPv4 (layer 3) la clasificación se hace por:
  - Dirección IP de origen
  - Dirección IP de destino
  - Protocolo de transporte utilizado (TCP o UDP)
  - Next-hop address
  - TOS

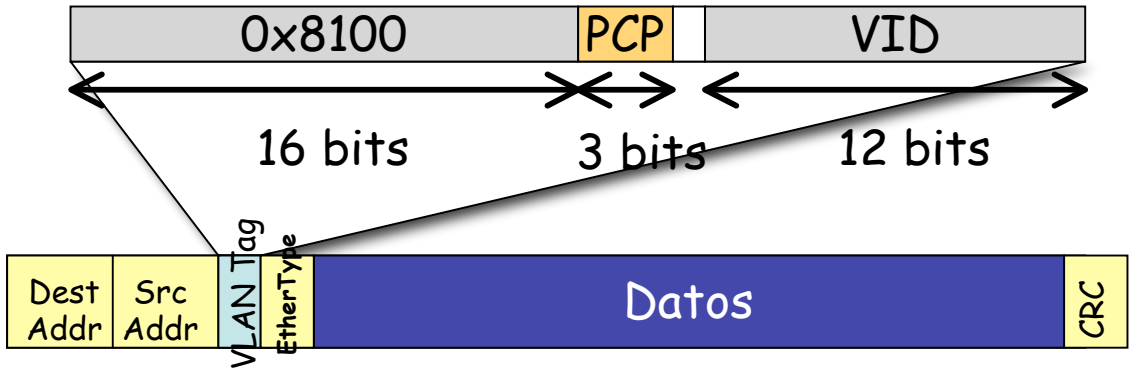


Identificación/clasificación de flujos

- En layer 4
  - Puerto origen
  - Puerto destino
- En layer 1
  - Interfaz físico por el que llega el paquete
  - PVC, subinterfaz, etc.
- En layer 7
  - Reconocimiento de la aplicación
  - URLs

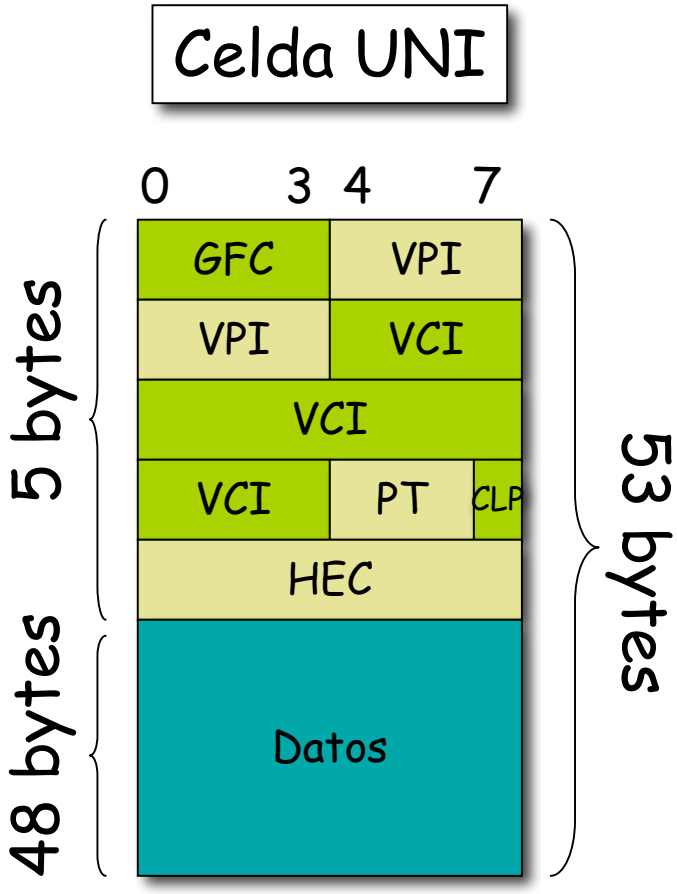
Identificación/clasificación de flujos

- En Ethernet (layer 2) se hace por:
  - Dirección MAC de origen
  - Dirección MAC de destino
  - Ethertype
  - VLAN (802.1Q)
  - Bits de prioridad (802.1p) (0=low, 7=high) (marcado)



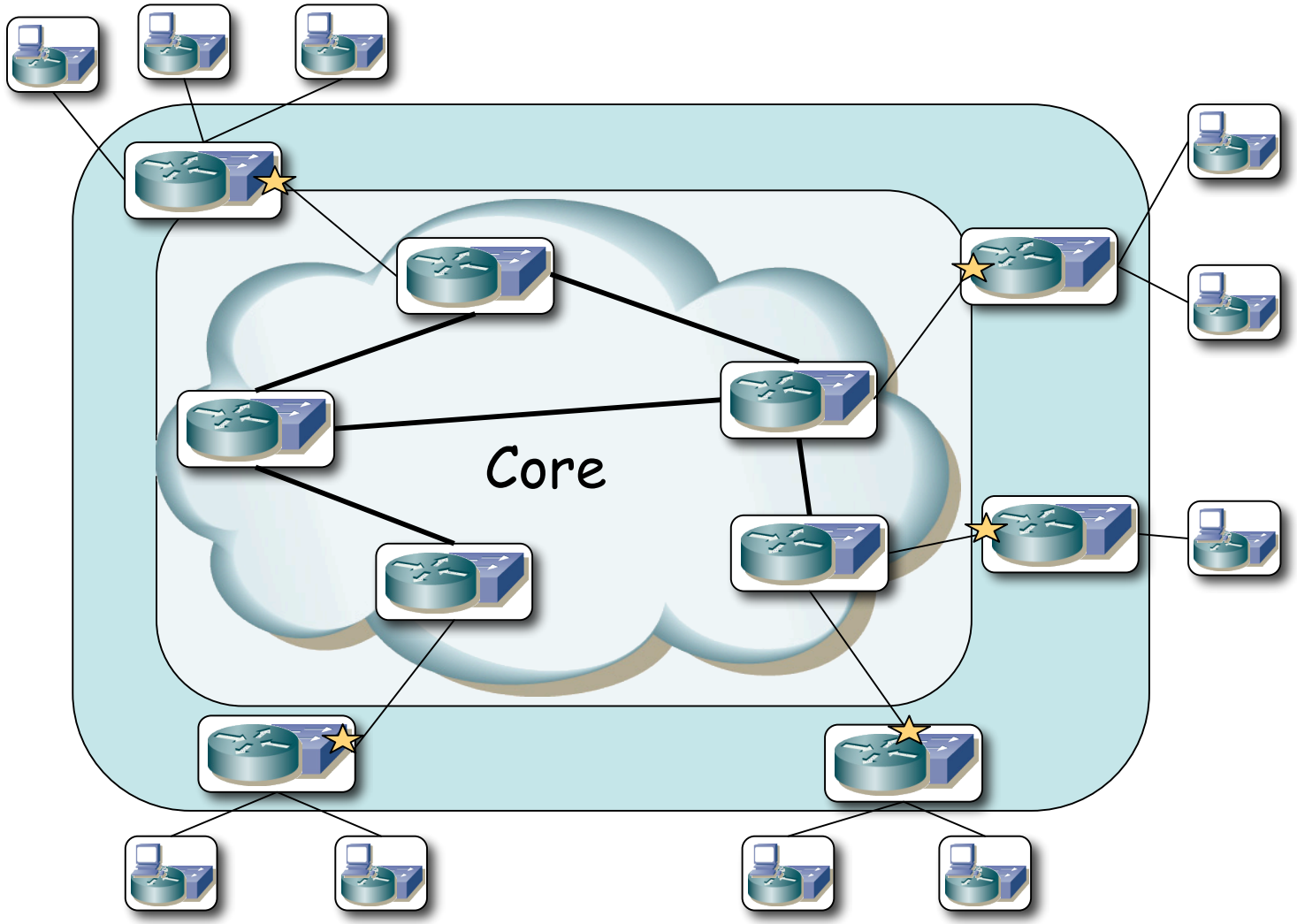
Ejemplo: Marcado en ATM

- CLP bit
  - 0 = high priority
  - 1 = low priority



¿ Dónde = Quién ?

- Preferiblemente en los extremos (edge) de la red
- O en los propios generadores de los paquetes (ej. Teléfono IP)



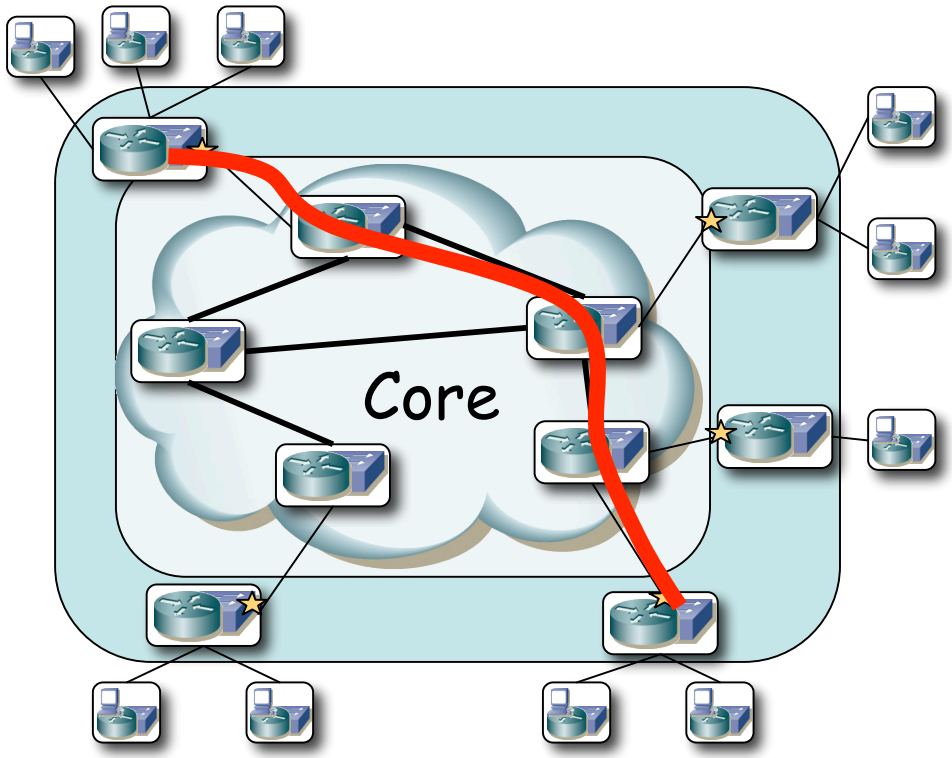
Connection Admission Control

CAC

Connection Admission Control

- Durante el establecimiento de la conexión
- Acciones para determinar si se permite o no
- Es lo correcto para flujos RT en vez de control de congestión

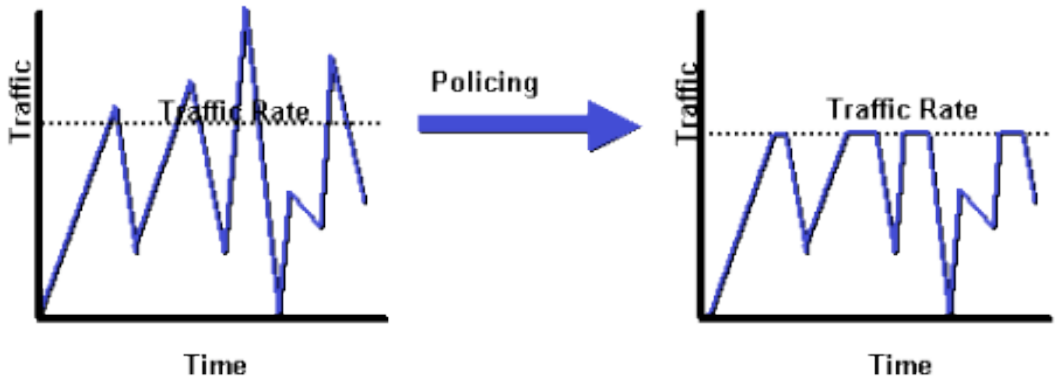
- Implica estudio de:
  - *Call routing*
  - *Preemption capabilities*
  - *Call redirection*
- ¡Proteger tráfico RT del tráfico RT!



Policing and Shaping

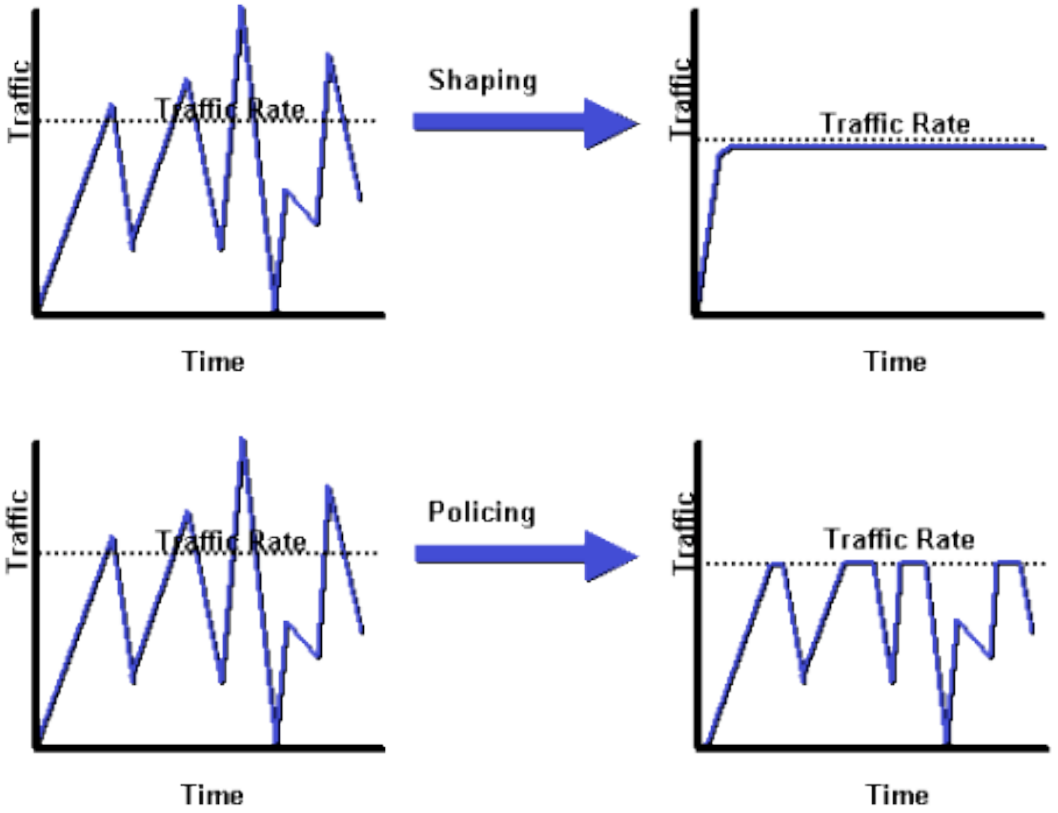
Policing

- **Objetivo**: Limitar el tráfico a la entrada a la red para que no exceda el declarado
- Características del tráfico
  - Tasa media (media a largo plazo)
  - Tasa de pico
  - Tamaño máximo de ráfaga: máx nº paquetes a tasa de pico
- Los que excedan se descartan o marcan (*conditional marker*)
- Policer as Marker:
  - Puede que el tráfico ya venga marcado según la aplicación a la que pertenezca: Separación vertical del tráfico
  - Un *policer* no distingue la aplicación, marca en función del tráfico: Separación horizontal del tráfico



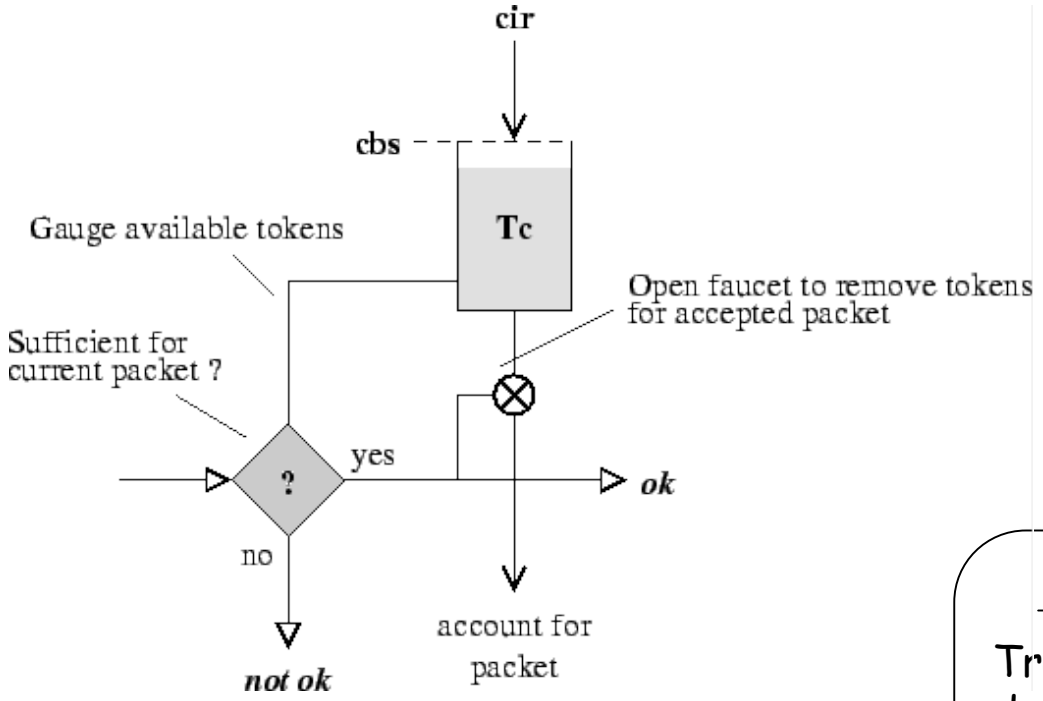
Shaping

- Los que excedan no se descartan sino que se encolan

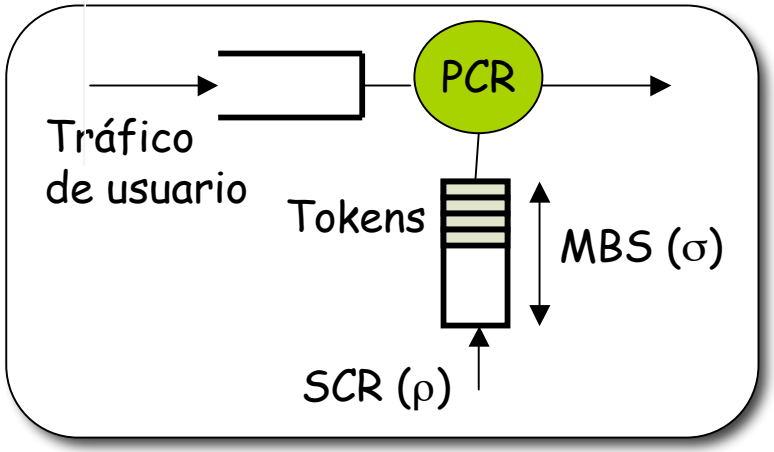


Ejemplo: *Single Leaky Bucket*

- Parámetros:
  - CIR = *Commited Information Rate* (bytes de paquetes IP por seg.)
  - CBS = *Commited Burst Size* (bytes)

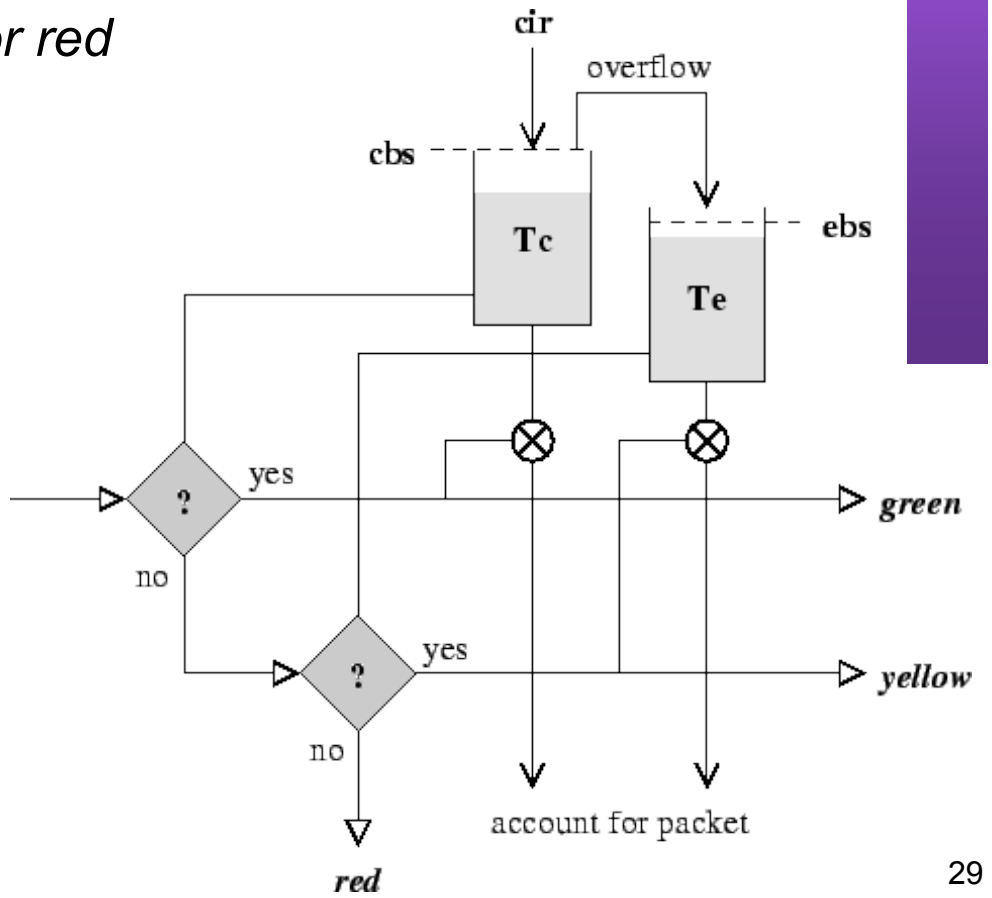


- A(0,t) = tráfico cursado en intervalo (0,t)
- $A(0,t) \leq \rho t + \sigma$
- *"Restricción (σ, ρ)"* a la salida (LBAP)



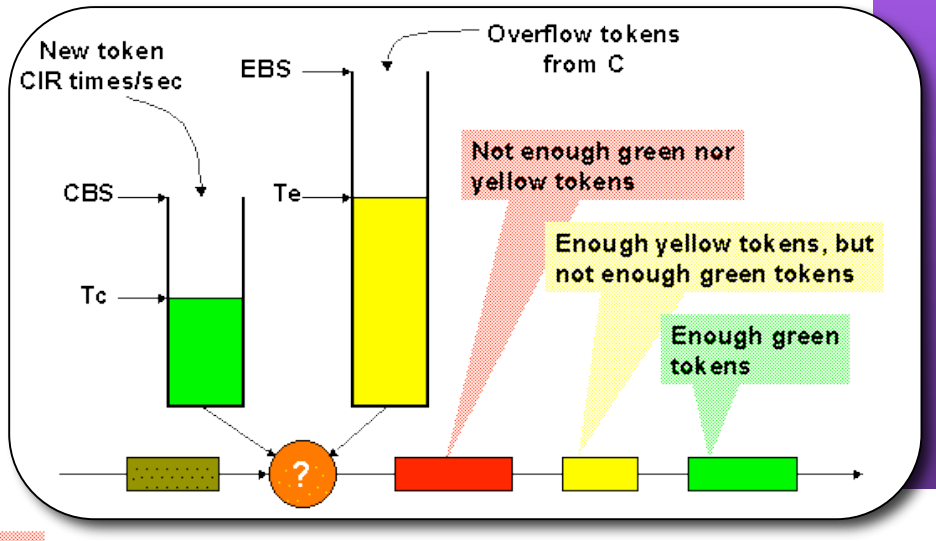
Ejemplo: srTCM

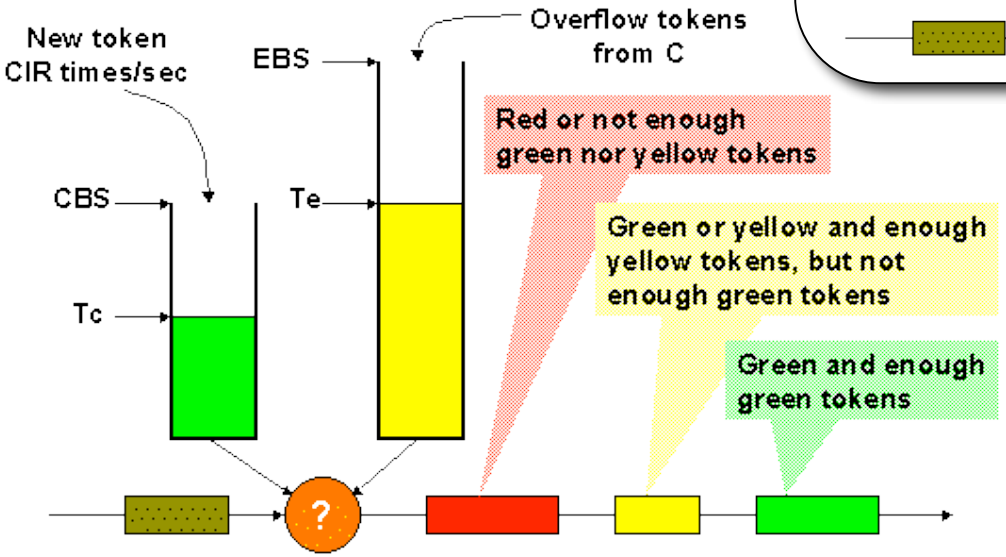
- *single rate Three Color Marker*
- RFC 2697
- Dos *Token Buckets*
- Parámetros: CIR, CBS *y EBS (Excess Burst Size*)
- Marca como *green, yellow or red*



Ejemplo: srTCM

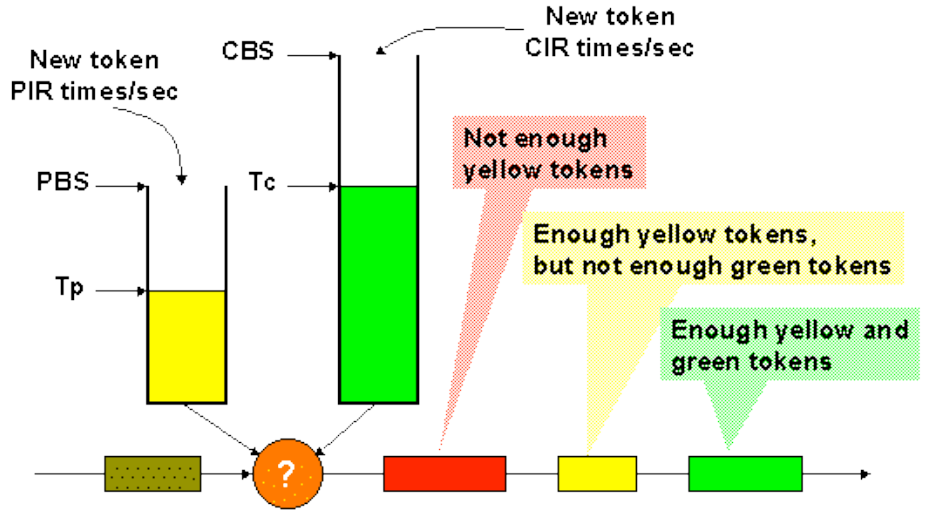
- Modo *Color Blind*:
  - If Tc(t) - B >= 0 *Green* (*Conform, DiffServ Codepoint AFx1*) else
  - If Te(t) - B >= 0 *Yellow* (*Exceed, AFx2*) else
  - *Red* (*Violate, AFx3*)
- Modo *Color Aware*:
  - If Green AND Tc(t) - B >= 0
    Green (*Conform*) else
  - If (G OR Y) AND Te(t) - B >= 0
    Yellow (*Exceed*) else
  - *Red* (*Violate*)

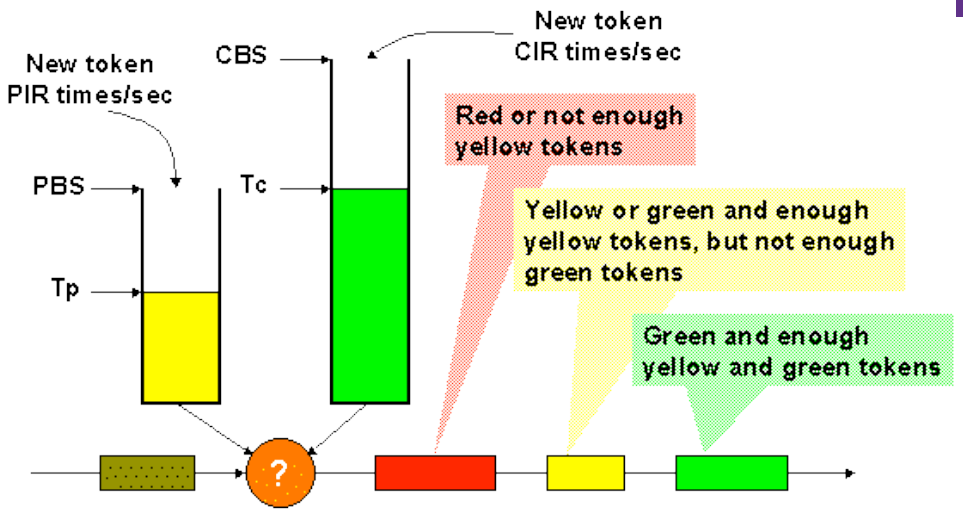




Ejemplo: trTCM

- *two rate Three Color Marker*
- RFC 2698
- PIR = *Peak Information Rate*
- PBS = *Peak Burst Size*
- *Color Blind*
- *Color Aware*
- Red : excede el PIR
- Yellow: excede el CIR





Scheduling

Scheduling

- Recursos compartidos
  - Buffer space
  - Capacidad en el enlace de salida
  - Tiempo de procesador
- Tipos de schedulers
  - Work-conserving
  - Non-work-conserving (no veremos)
- Schedulers sin prioridades
  - FCFS, RR, …
- Schedulers con prioridades
  - GPS, WFQ, SCFQ, WF2Q, …
- Características deseables
  - Sencillo de implementar
  - Reparto justo (*max-min fair share*) y protección
  - Performance bounds (deterministas o estadísticos)
  - Que permita implementar un CAC simple