Déploiement de QoS

Riadh DHAOU

INPT/ENSEEIHT

Plan

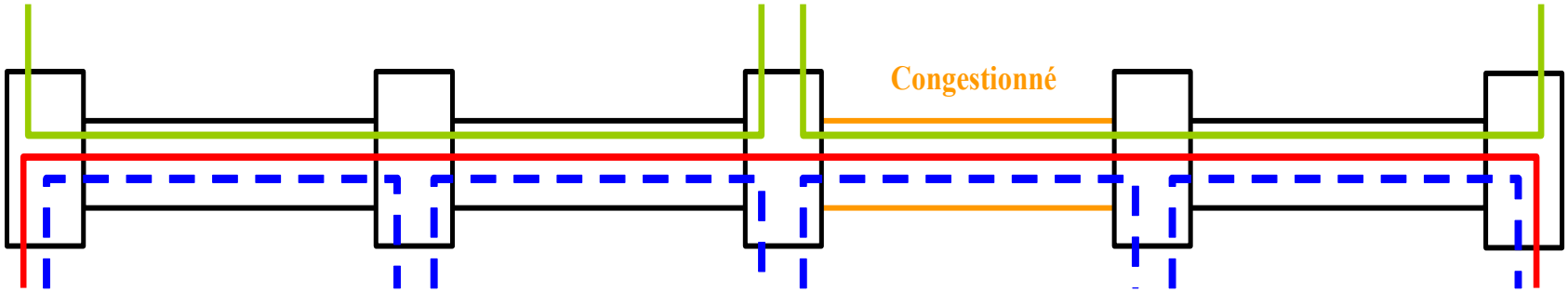
- Équité
- Qualité de Service
- Ingénierie de Trafic

Discussion: équité (“Fairness”)

- L’**équité** : un concept très subjectif.

- Dépend de la métrique utilisée:
  - Tous les paquets sont égaux?
  - Tous les utilisateurs sont égaux?
  - L’équité dépend-elle du nombre de sauts, de la distance, quantité de trafic concurrent,...?
  - Combien de personnes payent – à quel prix?

Exemple ‘Fairness’



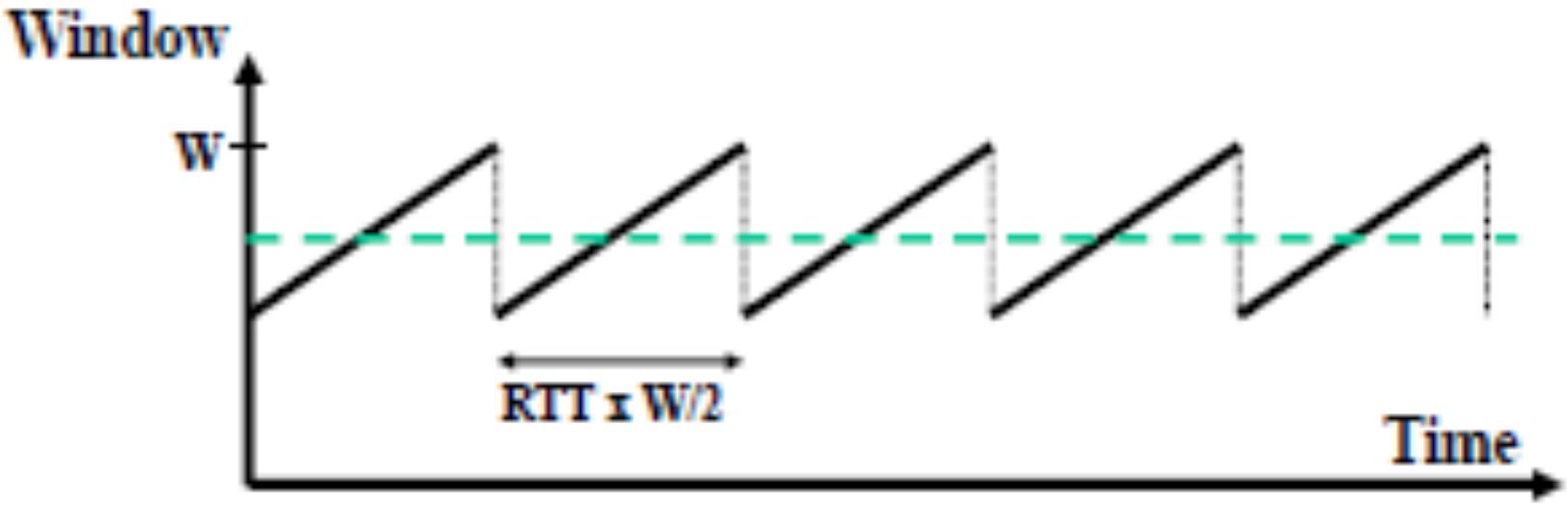
- Toutes les connexions doivent-elles avoir le même débit
  - Les connexions Rouges obtiennent un certain débit?
  - Les connexions Bleu obtiennent un meilleur débit?
  - Et les connexions Vertes?

Modélisation TCP

- Étant donné un mode de fonctionnement de TCP, peut-on prédire les performances que nous obtiendrons?

- Quels sont les facteurs importants
  - *Taux de perte (Loss rate)* : Affecte la fréquence de réduction de la taille de la fenêtre.
  - *RTT (Round Trip Time)*: Affecte le taux de croissance et le rapport entre le débit et la taille de la fenêtre
  - *RTO (Retransmission Timout)*: Affecte la performance durant la reprise sur erreur
  - *MSS (Maximum Segment Size)*: Affecte le taux de croissance

Calcul du débit TCP



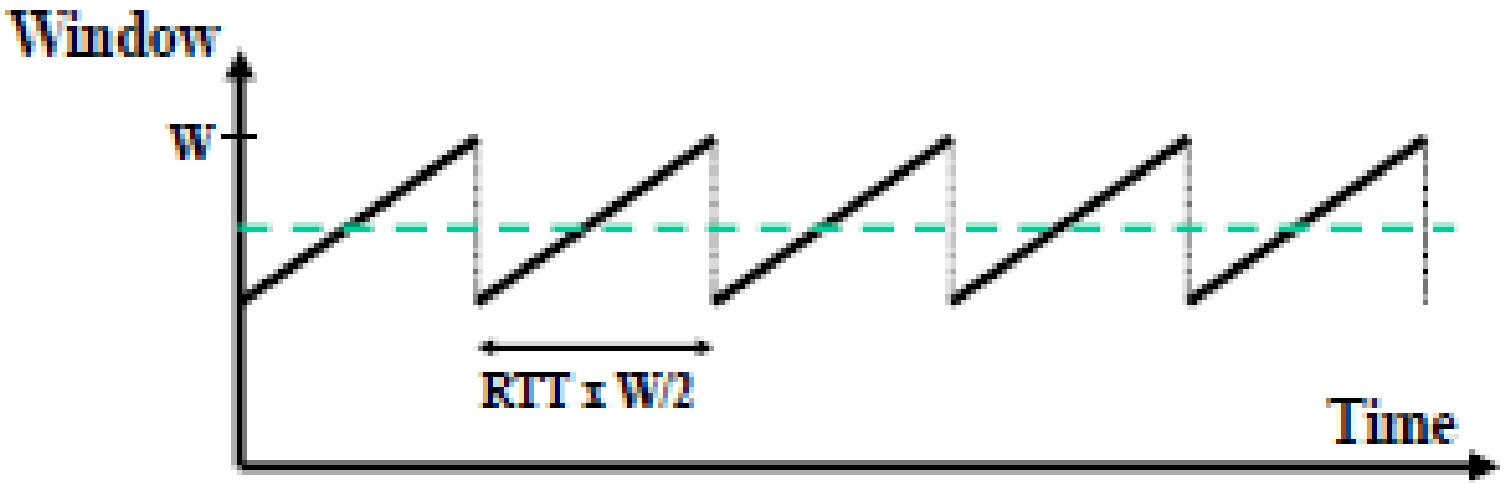
- Quel est le débit moyen ?
  - La taille moyenne de la fenêtre de congestion est ¾ W
  - On obtient le nombre de paquets envoyés par RTT
  - Multiplier par MSS pour obtenir le débit
- Débit = (¾ W * MSS) / RTT
- Mais, quelle est la valeur de W ?

Débit fondé sur un modèle de perte simple

- Quelques hypothèses additionnelles
  - RTT constant
  - Pas d'ACKs retardés
- W dépend du taux de perte: on perd un paquet par "fenêtre".
  - Les paquets transférés = (¾ W) * (W/2) = $3W^2/8$
  - 1 paquet perdu
    - $\Rightarrow$ taux de perte p = $8/3W^2$
    - $\Rightarrow W = \sqrt{\frac{8}{3p}} = \frac{4}{3}\sqrt{\frac{3}{2p}}$
  - BW = ¾ * W * MSS / RTT

$$BW = \frac{MSS}{RTT\sqrt{\frac{2p}{3}}}$$

Débit TCP



$$B = 1.22 \times \frac{MSS}{RTT \times \sqrt{loss}}$$

- **Est-ce équitable? Bien sur! “RTT fair”.**

Problèmes d'équité TCP

- De multiples flux TCP partageant un lien du goulot d'étranglement n'obtiennent pas nécessairement le même débit.
  - Facteurs comme RTT, petites différences de timouts, et instants de démarrage…affectent le partage de débit.
  - Spécifiquement le ratio entre les débits se stabilise.
- Modifier l'implantation du mécanisme de contrôle de congestion change l'agressivité de TCP et modifie la part de débit attribuée à chaque source.
  - Affectant l'"équité" relative aux autres sources
  - Changeant les timeouts, en ajoutant ou en diminuant certaines caractéristiques, ..
- Les utilisateurs peuvent prendre plus de débit en utilisant des flux parallèles.
  - Chaque flux obtient un partage du débit, favorisant les utilisateurs utilisant plusieurs flux par rapport à ceux utilisant un seul flux.

Partage équitable Max-Min

- Sur chaque lien, les flux sont divisés en deux groupes.
  - Les flux pour lesquels le goulot d'étrangement se situe sur un autre lien
  - Ceux pour lesquels le goulot d'étrangement se situe sur ce lien
- Le partage équitable max-min du débit R_{fair} d'un réseau est définit comme suit:
  - Les flux contraints par ce lien ont un débit $r = R_{fair}$
  - Les flux contraints par un autre lien ont un débit r, où
    - $r < R_{fair}$
    - r est le débit max-min de partage équitable du lien goulot d'étranglement
- Utilisé dans ATM:
  - Utilisant un feedback multi-valué binaire
  - L'Algorithme doit être distribué et adaptatif!

Exemple de partage équitable Max-Min

$$r_{fair} = \frac{C - \sum_{else} r_i}{n_{here}}$$

- Considérons des liens de 10 Mbs

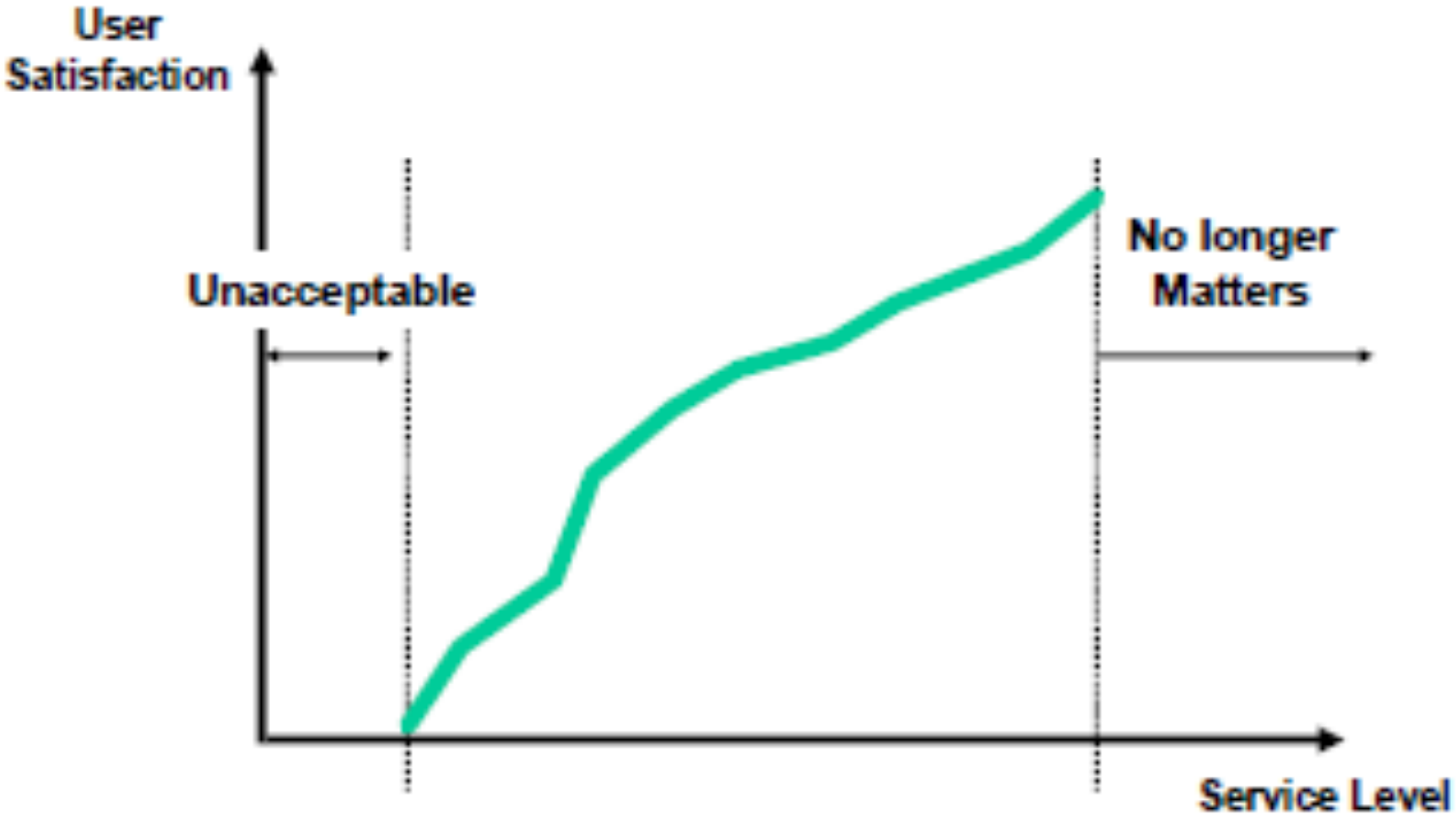
Plan

- Équité
- Qualité de Service
- Ingénierie de Trafic

Qu’est ce que la QoS?

- Internet actuel supporte un service « best effort » de délivrance de paquets
  - Suffisant pour la plupart des applications, mais quelques applications nécessitent ou peuvent bénéficier de « meilleurs » niveaux de service
- “Meilleure” qualité de service peut signifier la donnée de bornes sur un ou plusieurs paramètres.
  - Débit: transfert rapide de données, vidéo
  - Délais, gigue: téléphonie
  - Perte de paquets, taux d’erreurs binaires: services interactifs
- Peut signifier que l’utilisateur obtienne un “meilleur” traitement.
  - Mais aucune garantie (absolue) n’est donnée

Performance versus Satisfaction



Qualité de Service

- Définition classique de l'équité : traiter tous les utilisateurs *pareillement*
  - Par exemple, max-min fairness: tous les utilisateurs partageant le même lien goulot d'étranglement obtiennent le même débit.
- QoS: traiter les utilisateurs *différemment*.
  - Par exemple, quelques utilisateurs obtiennent des garanties de débit, alors que d'autres utilisent un service best effort
- Les deux ne sont pas contradictoires
  - Tout sauf être égal, les utilisateurs sont traités pareillement
  - Un traitement inégal est fondé sur des politiques:
    - Politiques *administratives* : rang ou position
    - Politiques *économiques* : payement supplémentaire pour un traitement préférentiel

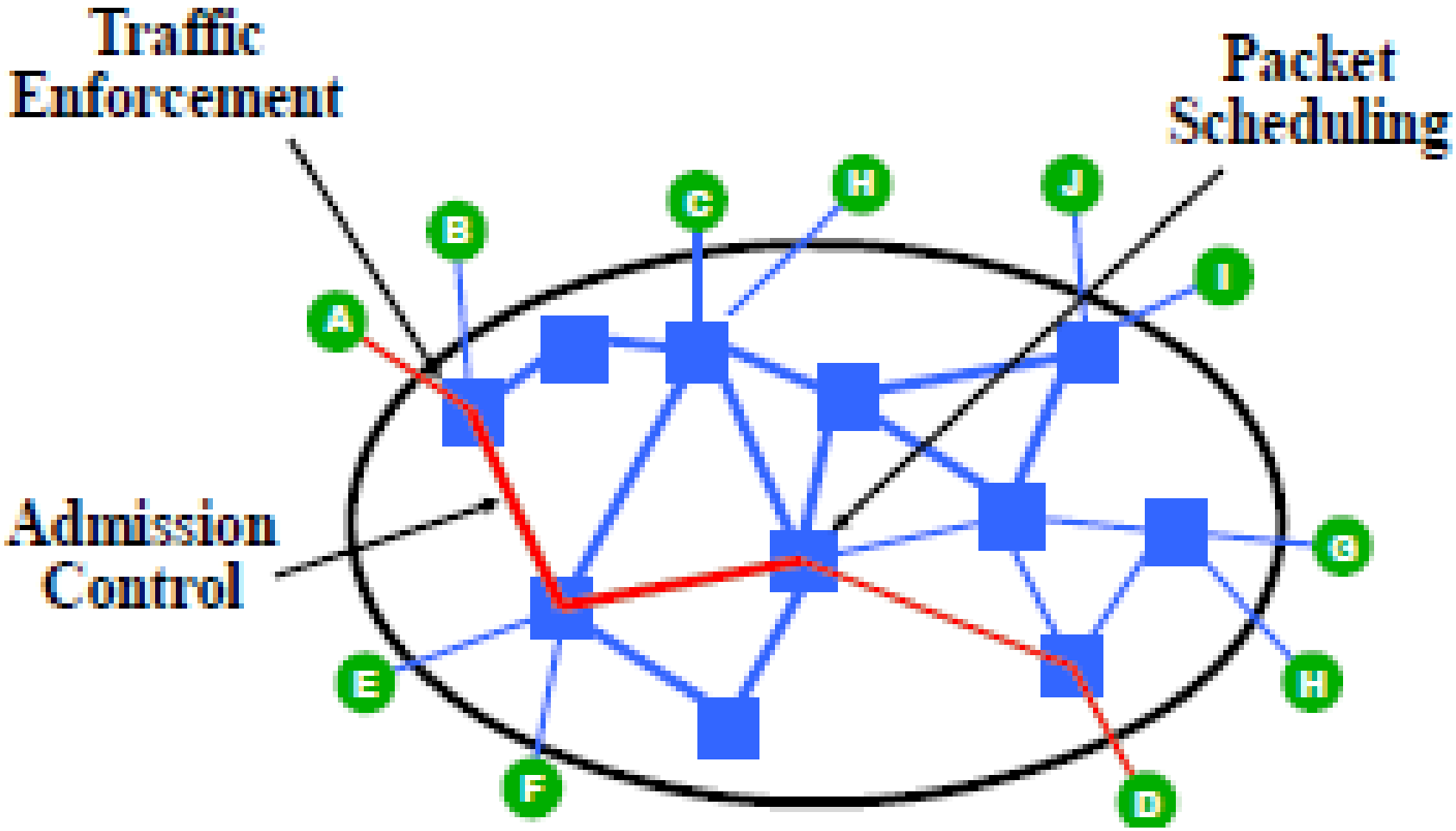
Comment fournir une QoS?

- Le ***contrôle d'admission*** (CAC) limite le nombre d'utilisateurs.
  - On ne peut donner de garanties si le nombre d'utilisateurs partageant les mêmes ressources (débit) est trop important.
  - Par exemple, réseaux téléphoniques commutés – tonalité occupé
  - Implique le rejet potentiel de la requête pour un service

- Le **renforcement du trafic** (Policing) limite la façon avec laquelle les utilisateurs injectent du trafic fondé sur des contraintes prédéfinies
  - S'assurer que les utilisateurs respectent leurs contrats de trafic
  - Les données injectée en dehors du contrat peuvent être rejetées (à l'entrée du réseau) ou peuvent être envoyées avec une priorité plus faible.

- L'**ordonnancement** (Scheduling) supporté dans les routeurs garantie que les utilisateurs obtiennent le partage du débit
  - Encore une fois, fondé sur des bornes pré négociées

Modèle de réseau à QoS



Déploiement de la QoS

Le déploiement de la QoS nécessite:

1. La définition d'architectures à QoS
2. La mise en place de mécanismes de QoS
3. L'utilisation de protocoles à QoS

Quelques méchanisms simples de QoS

- Classification :
  - Filtres de paquets
- Scheduling:
  - FIFO - First In First Out
  - PQ - Priority Queue (avec priorité)
  - WFQ - Weighted fair Queueing (avec pondération)
- Traffic enforcement :
  - Leaky buckets (sauts percés)
  - Shapers versus meters
- Admission control.

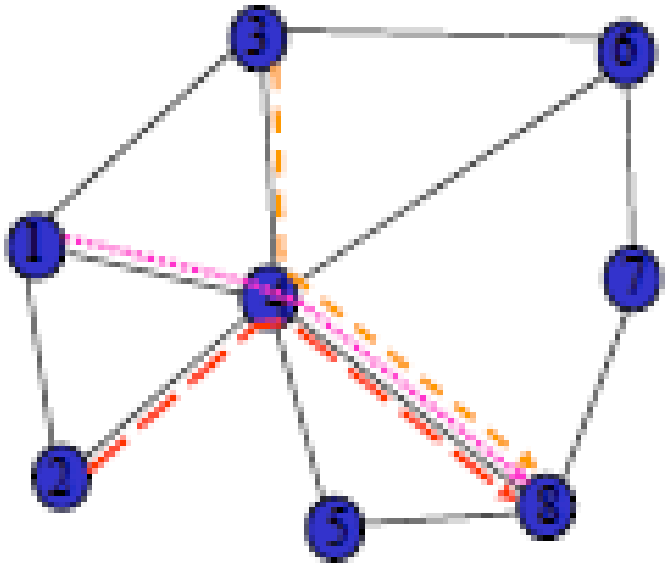
Plan

- Équité
- Qualité de Service
- Ingénierie de Trafic

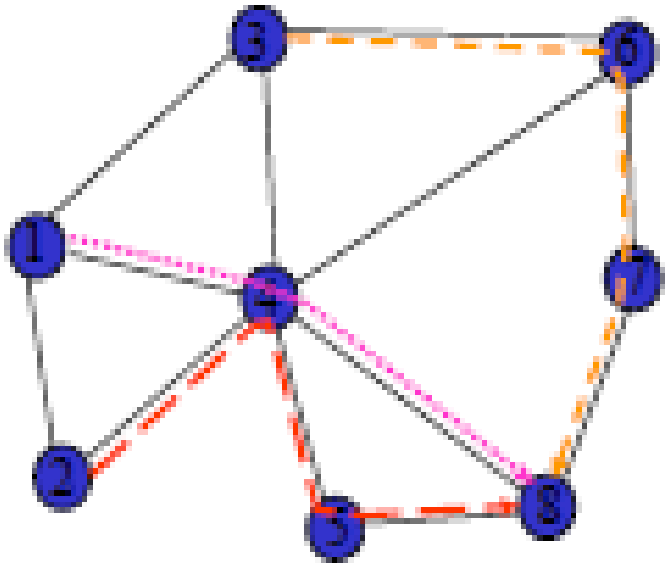
Ingénierie de trafic

- Gestion exercée à un niveau flux ou agrégat de paquets
- Répartition des flux dans le réseau afin de parvenir à une utilisation efficace des ressources (débit)
- Les algorithmes de plus court chemin pour routage d'un flux donné n'est plus suffisant.
  - Ne tient pas compte des besoins du flux, par exemple, en terme de débit
  - Ne tient pas compte de la concurrence entre différents flux
- Doit tenir compte de demandes agrégées de différent flux

Ingénierie de trafic : Exemple



Shortest path routing congests link 4 to 8



Better flow allocation distributes flows more uniformly

Pourquoi MPLS?

- Les circuits sont (parfois) utiles
  - Le routage IP conventionnel sélectionne un chemin, ne permet pas un choix de route, la commutation est fondée uniquement sur l'adresse de la destination
  - La commutation de label permet une flexibilité au niveau du routage
- *L'ingénierie de trafic*: établit plusieurs chemins séparés afin de tenir compte des besoins en terme de performance des flux de trafic agrégés
- Les « Labels » permettent une commutation plus rapide

Déploiement de QoS dans une architecture TCP/IP – Cas de l'Internet Actuel

Service de type « Best-Effort »

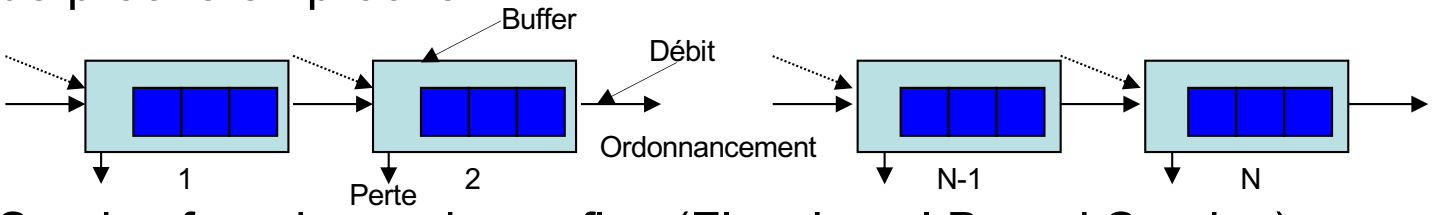
- Architecture d'Internet classique fondée sur une seule classe de service « best-effort »
  - Même traitement pour tous les paquets (tous les utilisateurs, toutes les applications)
  - Aucune garantie
  - Services fournis = ce que peut fournir le réseau à l'instant T
- Le réseau doit fournir une meilleur qualité de service à certains paquets?
  - Pourquoi?
  - Pourquoi pas?

Qualité de Service (QoS)

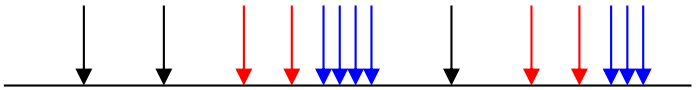
- Fournir un service de communication efficace : une vue anticipée

- Catégories de service : Débit, Latence, Gigue, Pertes,…
- Si affectées par la congestion: le service devient Best-effort

- Autres considérations
  - La performance de bout en bout est une accumulation de performances de proche en proche



  - Service fournis au niveau flux (Flow Level Based Service)
    - Flux = séquence de paquets partageant les mêmes caractéristiques de service
  - Pourquoi pas au niveau paquet?
    - Complexité,
    - Adaptation (des applications, des utilisateurs, …)
    - Validité des modèles de trafic



Exemple de problème de niveau réseau: congestion

- Comment traiter la congestion ?
  - A quel niveau?

- 3 facteurs importants
  - Performance de l'application
  - Débits requis pour assurer la performance
  - Complexité / Coût des mécanismes nécessaires

Deux visions différentes

Visions	**Conservatrice**	**Progressiste**
Idées	Ne pas changer l'architecture IP	L'architecture réseau doit évoluer
Arguments	Les ressources doivent être suffisantes	Il y aura toujours un manque récurrent de ressources réseau → La pénurie doit être gérée
Approche	Approche de bout en bout Amélioration du service par les hôtes – Adaptation des applications – Contrôle applicatif	Les protocoles de transport ne suffisent pas. Le réseau doit fournir des services – de délai – de débit
Solutions	**Dimensionnement du réseau** Augmenter la capacité de transfert: Supports, Routeurs Prédire le trafic: Ingénierie de trafic	**Gestion des ressources** Comment ? IntServ/RSVP DiffServ
	Quelles solutions sont moins coûteuses? Aucune certitude! « big and dumb » vs. « small and smart »	

Comment fournir un meilleur service?

- *Routage* vs *Commutation*

- *Ordonnancement* vs *Rejet*
  - Ordonnancement avec priorités
    - Les paquets les plus favorisés obtiennent les plus faibles délais
  - Rejet avec priorités
    - Les paquets les plus favorisés obtiennent le plus faible taux de perte

- *Service Relatif* vs *Service Absolu*
  - Les mécanismes avec priorités ne peuvent donner des garanties globales (absolues) que si la charge globale est régulée

Du service *relatif* au service *absolu*

- ***Service relatif***
  - Rendre un meilleur service à une classe de trafic signifie dégrader le service pour les autres classes de trafic
  - Garantie par classe

- ***Service global***
  - Garantie à une granularité fine (sur la base de flux)
  - Contrôle d'admission par flux requis (réservations)
  - Changement philosophique majeur
    - L'admission par flux est un changement drastique de l'Internet

Réservation vs *Best-effort*

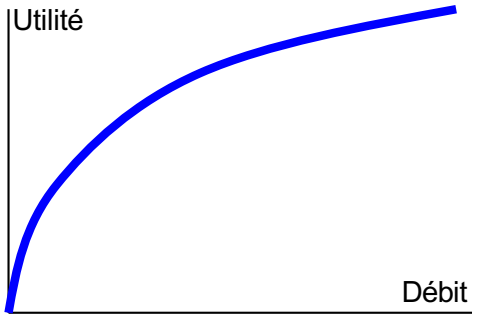
- **Question de base: Que devons nous faire?**
  - Accepter tous les flux (BE)?
  - Refuser certains flux afin de préserver le service pour les flux acceptés (R)?

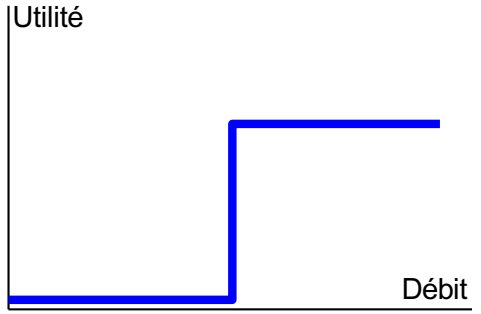
- **Comment pouvons nous décider ?**
  - Lequel des deux choix permet aux applications une meilleur performance?

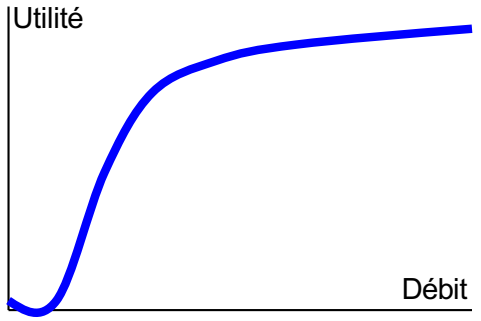
- **La modélisation des performances au niveau Application**
  - Ne donne pas de simples fonctions de délai/gigue/perte
  - Dépend de la perception de l'utilisateur: e. x., qualité de l'image, etc.
  - Dépend du comportement adaptif des applications
    - Adaptation du débit d'émission
    - Adaptation du codage (pour masquer les erreurs)
    - Adaptation des points de synchronisation (playback point)
  - Caractérisée par relation de dépendance entre la performance et le débit

Classes d'Applications

- Applications "Elastic": transfert de données
  - Tolérantes aux délais
  - Tolérantes à la perte
- Applications temps réel (RT "Real-Time"): media streaming
  - Rigides
    - Ne peuvent tolérer aux pannes/distorsions
    - Ne peuvent placer des points de synchronisation
  - Adaptatives
    - Tolérantes aux pannes
    - Peuvent déplacer les points de synchronisation
  - Point de synchronisation ou ("playback point")
    - Récepteur prend le temps de traiter le contenu en play back







Quelle est la meilleure approche?

- Deux Options:

  (1) permettre à tous les flux d'utiliser le lien (BE)

  (2) limiter les utilisations (R)

- **Laquelle est la plus efficace? (la plus utile)**
  - Elastic: BE est la meilleure (utilité concave)
  - RT: R est la meilleure (convexe autour de l'origine)
    - Rigide: évident
    - Adaptative: seulement de petites régions de convexité

Comment fournir une QoS ?

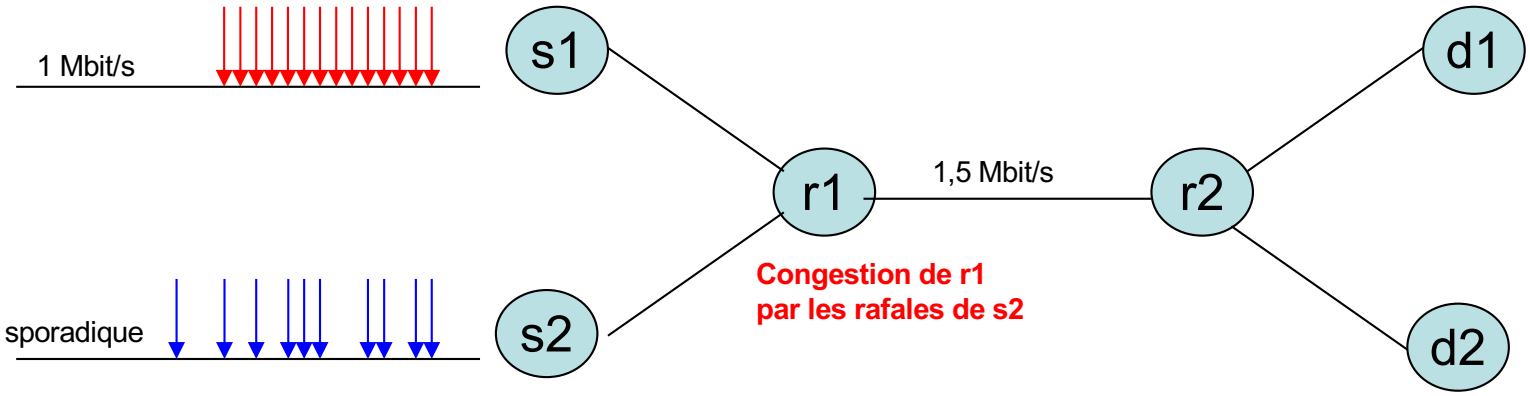
- *Réservation de Ressources* en fonction des flux
  → Absorption de la Congestion

- Mener un ensemble d'actions « salutaires »
  – Garder le flux à gauche du point d'inflexion
  → Évitement de la congestion

- Pour quelles applications ?
  – Les applications capables de prédire leurs débit d'émission
    - Media Streaming
  – Les applications real-time rigides
    - Soit fonctionnent sans aucun problème durant toute la session soit ne fonctionnent pas du tout

Composants de la QoS (1)

- **Modèle simple (1)**



- Solution
  - Distinguer les paquets des deux flux
    - → Donner une priorité aux paquets du média continu
  - Traiter les paquets en fonction de leur contrainte de service
  - Éviter les interactions entre les flux: Isoler les flux

Composants de la QoS (2)

- 1er Composant : *Classification*
  - Le routeur doit pouvoir distinguer les différentes classes de service
  - Classification faite en fonction d'une marque dans le paquet

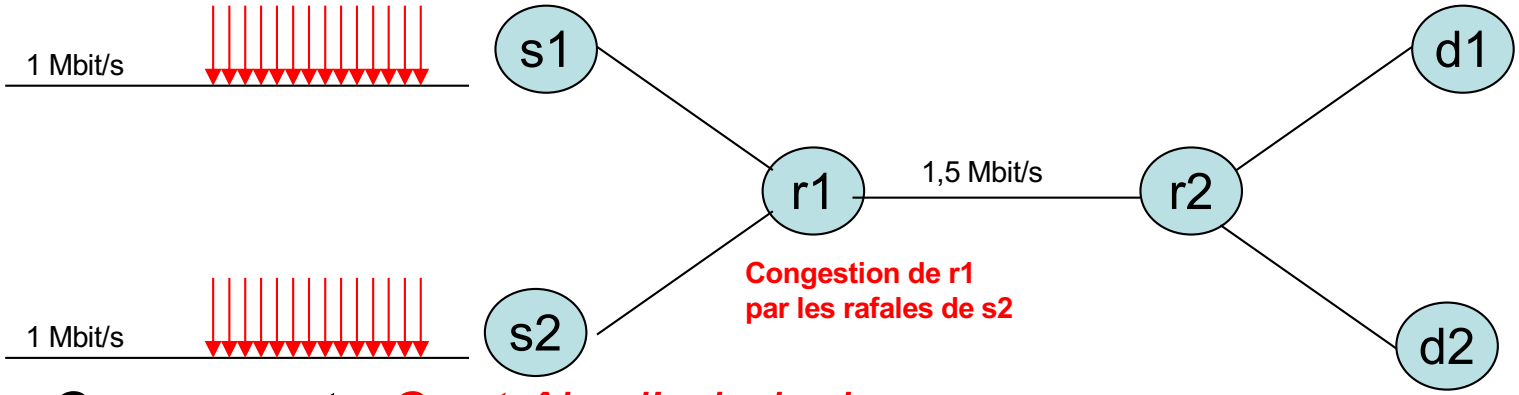
- 2ième Composant : *Réservation*
  - Isoler une classe par rapport à une autre
  - Réservation de débit
    - Descripteur de trafic
  - Sous contrainte
    - Utilisation des ressources le plus efficacement possible

Et si le flux à contrainte temporelle émet plus que prévu ?

- 3ième Composant : *Contrôle de trafic* (dit « d'accès »)
  - Surveillance et remise en forme
    - A l'entrée du réseau

Composants de la QoS (3)

- **Modèle simple (2)**
  - Plus assez de débit, la qualité est dégradée pour les deux flux
  - Il vaut mieux bloquer un flux



- 4ième Composant : *Contrôle d'admission*
  - Signalisation des besoins

- Pour une optimisation des ressources : Routage contraint
  - Choix de la route adaptée aux services requis
  - Répartition du trafic dans le réseau

Contrôle en boucle ouverte

- **Principe**
  - 2 phases pour chaque flux (établissement de l'appel, transmission des données)
  - Signalisation des besoins et des demandes: Plan de contrôle (ex: RSVP, Reservation Protocol)

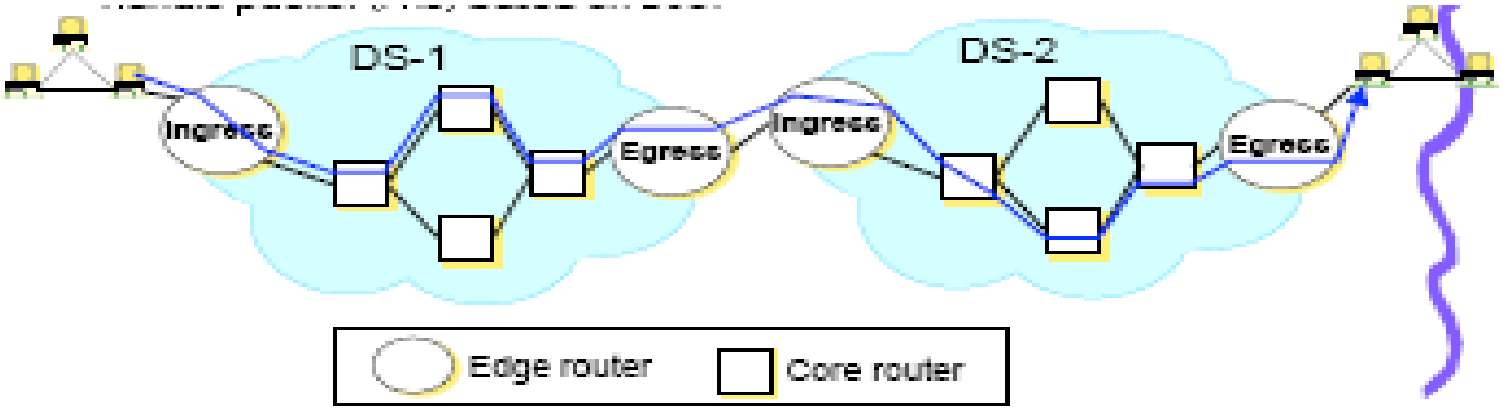
- **Établissement de l'appel**
  - Négociation des ressources entre la source et le réseau
    - Description du trafic de la source: descripteur de flot
  - Le réseau accepte ou refuse l'appel: contrôle d'admission
  - Établissement du contrat de trafic: réservation
    - Si la source se conforme à ce contrat alors le réseau garantie l'absence de congestion
- **Transmission des données**
  - La source émet son trafic dans la limite des paramètres de sa description
  - Le réseau contrôle la conformité de flux de la source: Contrôle d'accès
    - Régulation: remise en forme
    - Surveillance: prélèvement ou marquage
  - Actions prises par le réseau pour fournir la QoS au flot
    - Traitement des priorités (ordonnancement, discipline d'attente)

DiffServ (Differentiated Services)

- **Objectif:** offrir différents niveaux de service
  - Organisés autour de domaines
  - « Edge routers » et « core routers »

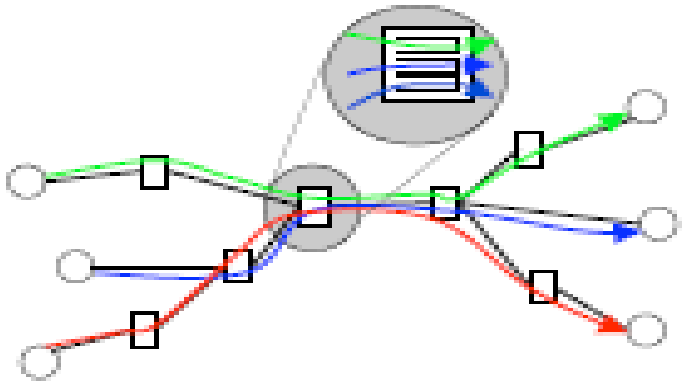
- **Edge routers**
  - Tirent les paquets en classes (fondé sur une variété de facteurs)
  - Exercent du « Traffic Policing/shaping »
  - Positionnent les bits (DSCP) dans les entêtes de packet

- **Core routers**
  - Traitent les paquets (PHB) en utilisant DSCP (PHB : Per Hop Behavior)



IntServ (Integrated Services)

- **Objectif:** supporter une grande variété de services dans une architecture unique
- Les garanties de service sont de bout en bout sur la base du flux
  - Le flux est une abstraction tenant compte de la QoS
- Les réservations sont faites par les terminaux (end-points)
  - Le réseau ne se soucie par des besoins des applications
  - Réservations faites par les récepteurs
- Les routeurs le long de la route maintiennent un état du flux
  - L'état sert à fournir le service adéquat
  - L'état maintenus par les routeurs sont souvent rafraîchis par les terminaux: Soft-state



IntServ

- Chaque source (flux) décrit ses caractéristiques de trafic et ses besoins en QoS (par exemple, délai de bout en bout) en utilisant RSVP

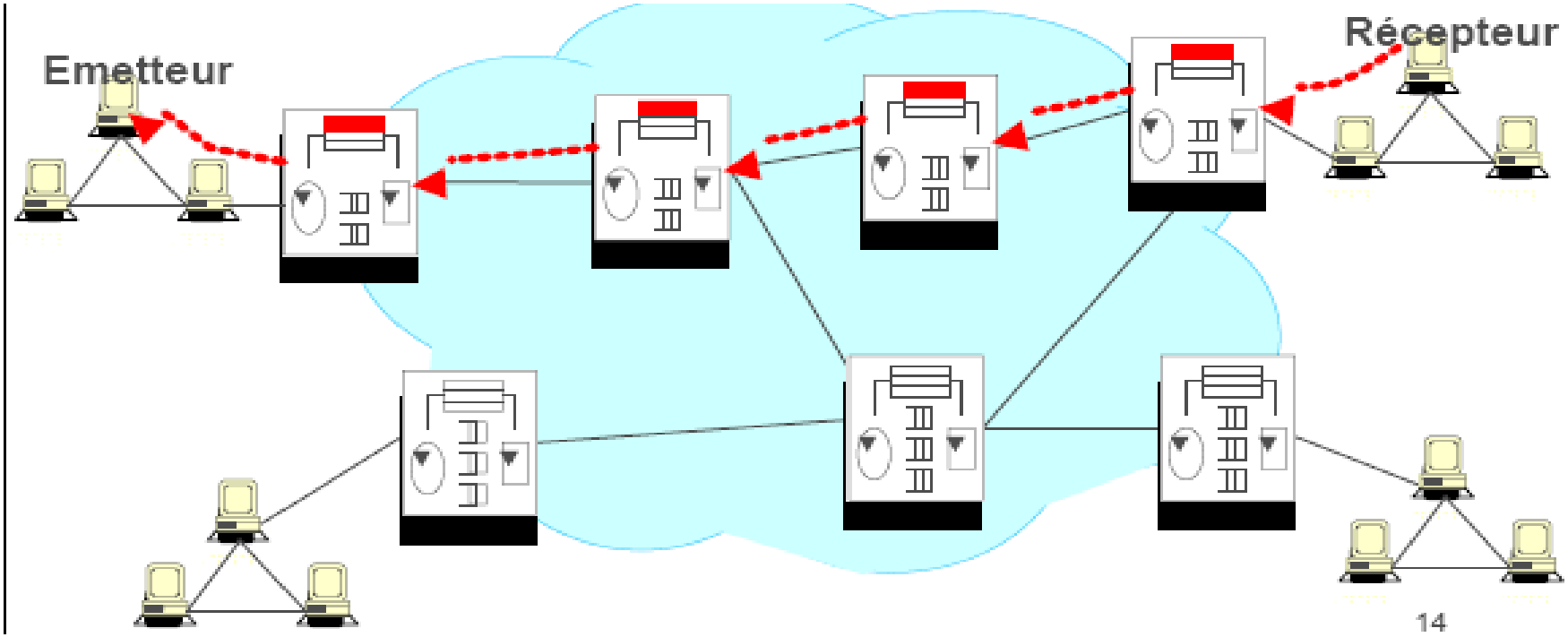
- Chaque routeur à la réception de ce message ajoute les caractéristiques du flux dans une table (pas de réservation)

- Quand le message arrive à destination, le destinataire indique ses contraintes et envoie un message de réservation de ressources

- On essaye alors de réserver dans chaque routeur le débit et la taille mémoire nécessaires pour garantir les contraintes.

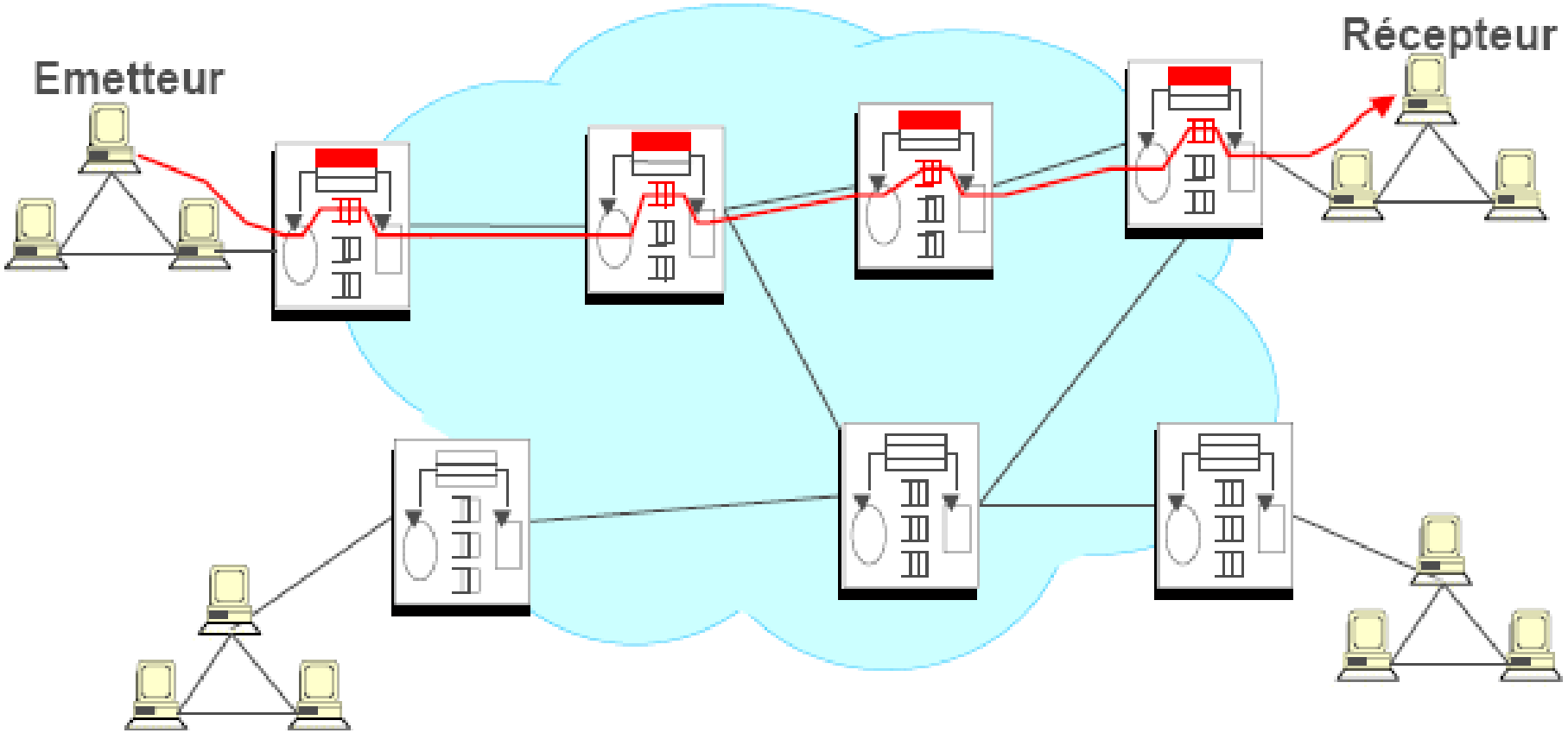
IntServ - Exemple

- Délai et Débit garantis par flux
- Allocation de ressources
  - Contrôle d'admission par flux
- État de chaque flux - Chemin Stable

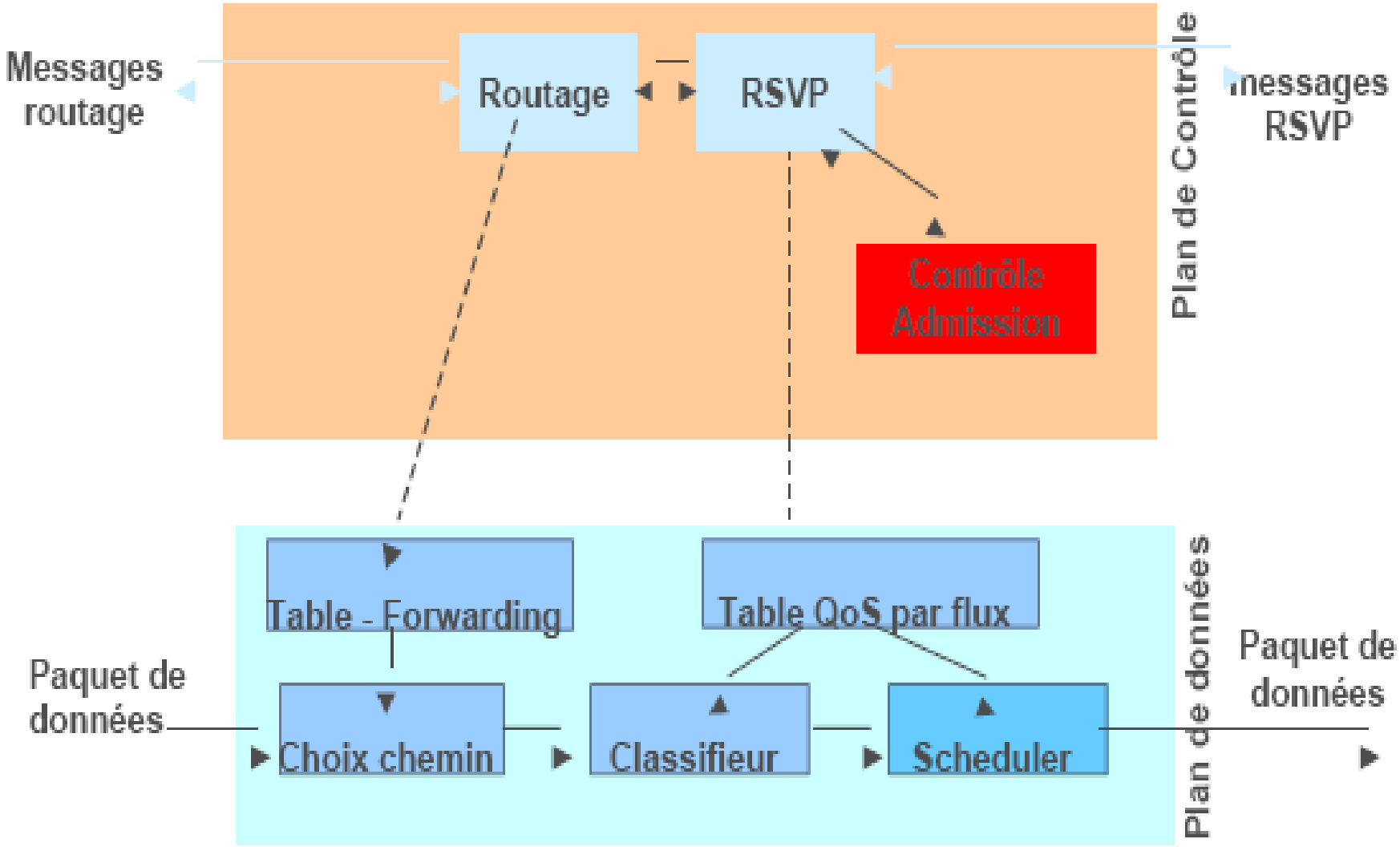


IntServ - Exemple

- Scheduling et Gestion de buffer par flux



IntServ - Schéma général



IntServ - Classes de Service

Le service peut être vu comme un contrat entre le réseau et le client:
Service de bout en bout

3 classes de Service

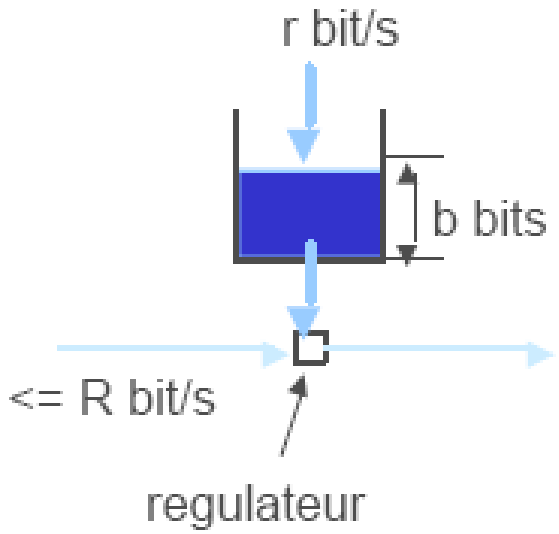
- Service Garanti (Applications temps réel)
  - Réseau => Client : borne supérieure sur le délai pour chaque paquet
  - Client => Réseau : pas plus de données envoyées que spécifié
  - Contrôle d'Admission : repose sur une analyse du pire cas
  - Classification/Scheduling par flux
- Charge contrôlée (Contraintes de Délai)
  - Réseau => Client : performance ~ à 1 réseau best-effort peu chargé
  - Client => Réseau : pas plus de données envoyées que spécifié
  - Contrôle d'admission : mesures de trafic agrégé
  - Ordonnancement sur une agrégation de flux possible
- Best-Effort (Applications "élastiques")

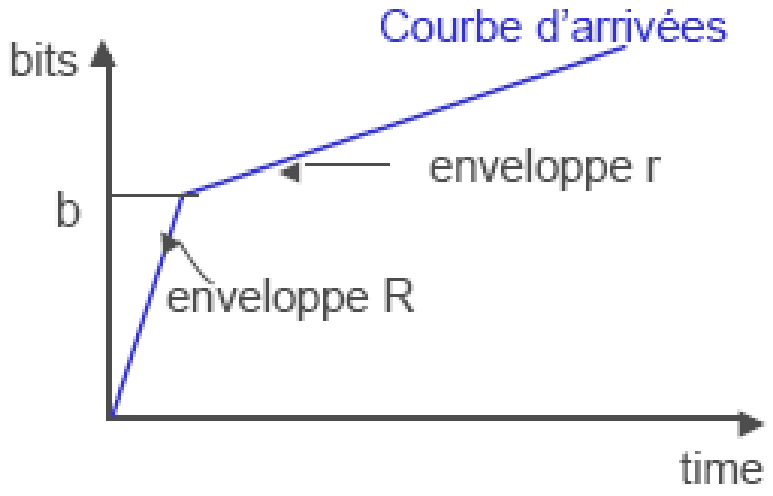
Message PATH et RESV

- Message PATH
  - Flowspec: Spécification du Trafic (token bucket)
  - Chaque routeur stocke:
    - L'adresse du noeud précédent
    - Émetteur et son Flowspec
- Message RESV
  - Suit le chemin inverse au message PATH
  - Flowspec + QoS désirée (e.g., queueing delay)
  - Routeur déroule :
    - Contrôle d'admission
    - Met à jour la réservation (quantité de ressources)

Spécification: Token Bucket

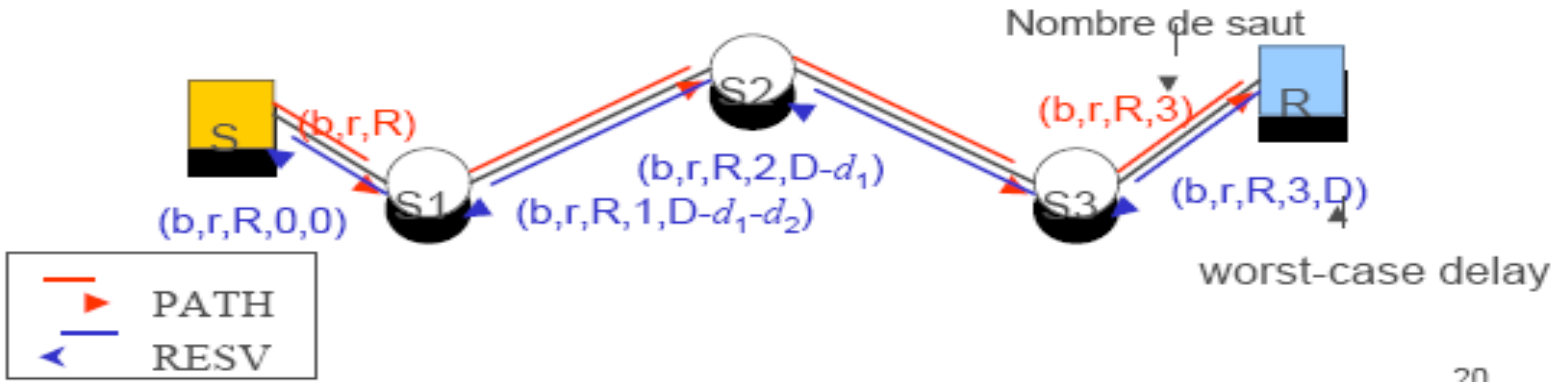
- 2 paramètres (r, b)
  - r : débit d'arrivée des jetons
  - b : taille de la file des jetons
- Débit d'arrivée <= R bit/s (e.g., R capacité du lien)
- 1 bit est transmis s'il y a un jeton
- Courbe d'arrivée – quantité maximale de bits transmis en une durée t





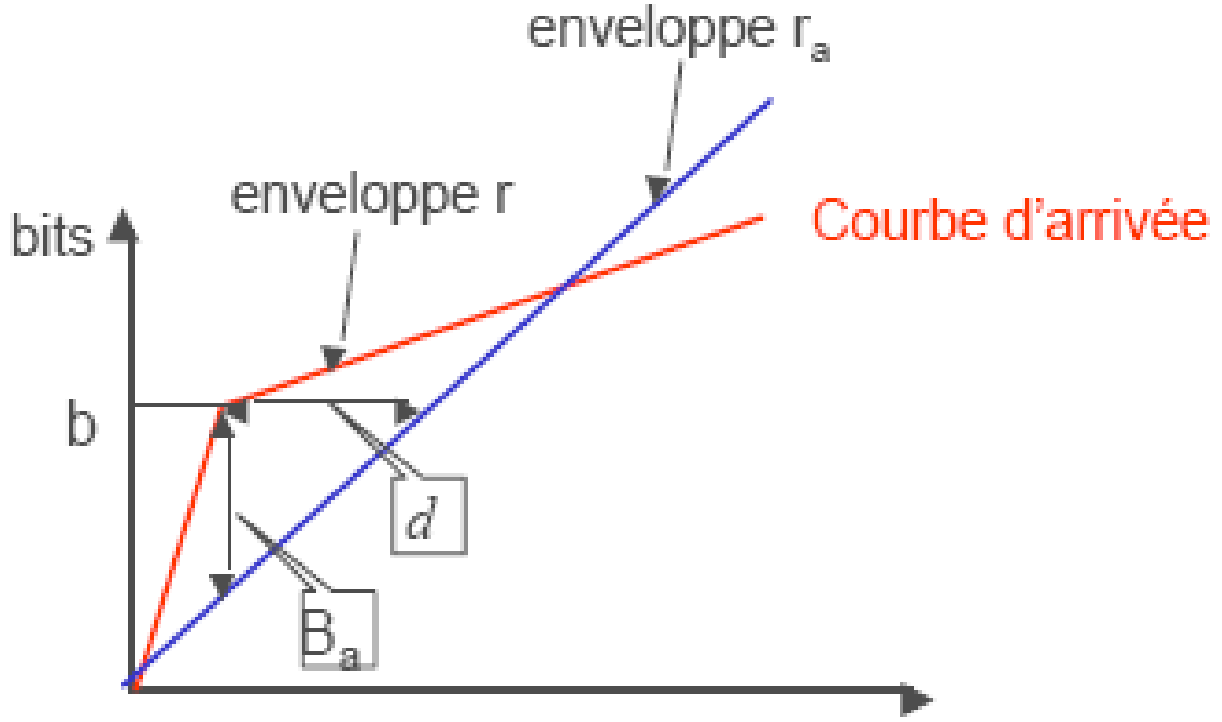
Réservation de bout en bout

- Quand R reçoit le message PATH il connaît
  - Caractéristique de Trafic (tspec): (r,b,R)
  - Nombre de sauts
- R renvoie cette info + le délai pire cas D qu'il peut supporter dans le message RESV
- Chaque routeur le long du chemin fournit un délai garanti (par saut) et forwarde le message RESV données mises à jour



Réservation pour 1 noeud

- Étant donné (b,r,R) et le délai d
- Allouer le débit r_a et la taille mémoire B_a qui garantisse *d*



“Soft State”

- Un état est associé à une session et une durée de vie
  - l’état est perdu quand la temporisation expire
- L’émetteur et le récepteur rafraîchissent périodiquement l’état en s’envoyant des messages PATH/RESV et en réarmant les temporisations
- **Avantage**
  - Les paquets de signalisation peuvent éventuellement être perdus (pas besoin de transmission fiable)
  - S’adapter aux changements de routes

RSVP et le routage

- RSVP est prévu pour fonctionner avec de nombreux protocoles de routage
- Service de routage minimal
  - RSVP demande à l'algorithme de routage dans chaque noeud comment router le message PATH
- Routage à QoS
  - Sélection de route RSVP repose sur les paramètres de QoS
- Routage Explicite

IntServ - Récapitulatif

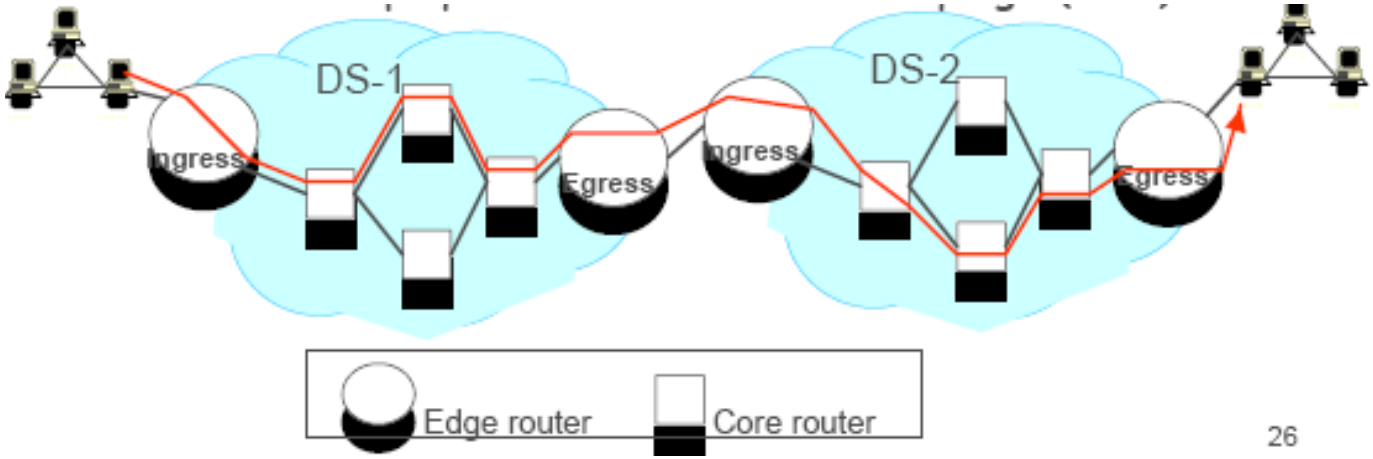
- **Avantages** :
  - Définition claire de la QoS et du service
- **Inconvénients** :
  - Une source peut avoir du mal à déterminer ses paramètres de token bucket
  - Passage à l'échelle (trop nombreux flux à gérer)
  - Fournir un "débit" r impose une politique d'ordonnancement compliquée
- **Conclusion :** difficultés de déploiement à grande échelle

La différentiation de Services : DiffServ

- Objectif : fournir de la QoS de façon “scalable”
- Principe :
  - Agrégation de flots en classes.
  - Per Hop Behavior :
    - EF (Expedited Forwarding) : délai faible. Cette classe est la plus prioritaire
    - AF (Assured Forwarding) : divisée en sous-classes, correspond sensiblement à des niveaux de priorités
    - Best Effort
  - Routeurs servent des classes et plus des flux Þ scalable
  - Pour émettre du trafic, une source établit un SLA (Service Level Agreement) avec un routeur de bordure
  - Routeurs de bordure et fournisseur de Bande Passante (BB) gère la bande passante, assurant par exemple que le trafic EF ne constitue pas plus de 10 % du débit total.

Differentiated Services (Diffserv)

- Découpages en Domaines
- “Edge routers”
  - Lissage et contrôle par agrégat
  - Marque les paquets (champ TOS d’IPv4) => classes
- “Core routers”
  - Traite les paquets en utilisant le marquage (PHB)



DiffServ

- Plusieurs types de service (reposant sur les PHB)
  - Service Assuré (Olympic - décliné en 3 niveaux) utilise AF
  - Service Premium utilise EF
  - Best-Effort
- 2 informations binaires
  - Bit P
  - Bit A

Services Assuré et Prémium

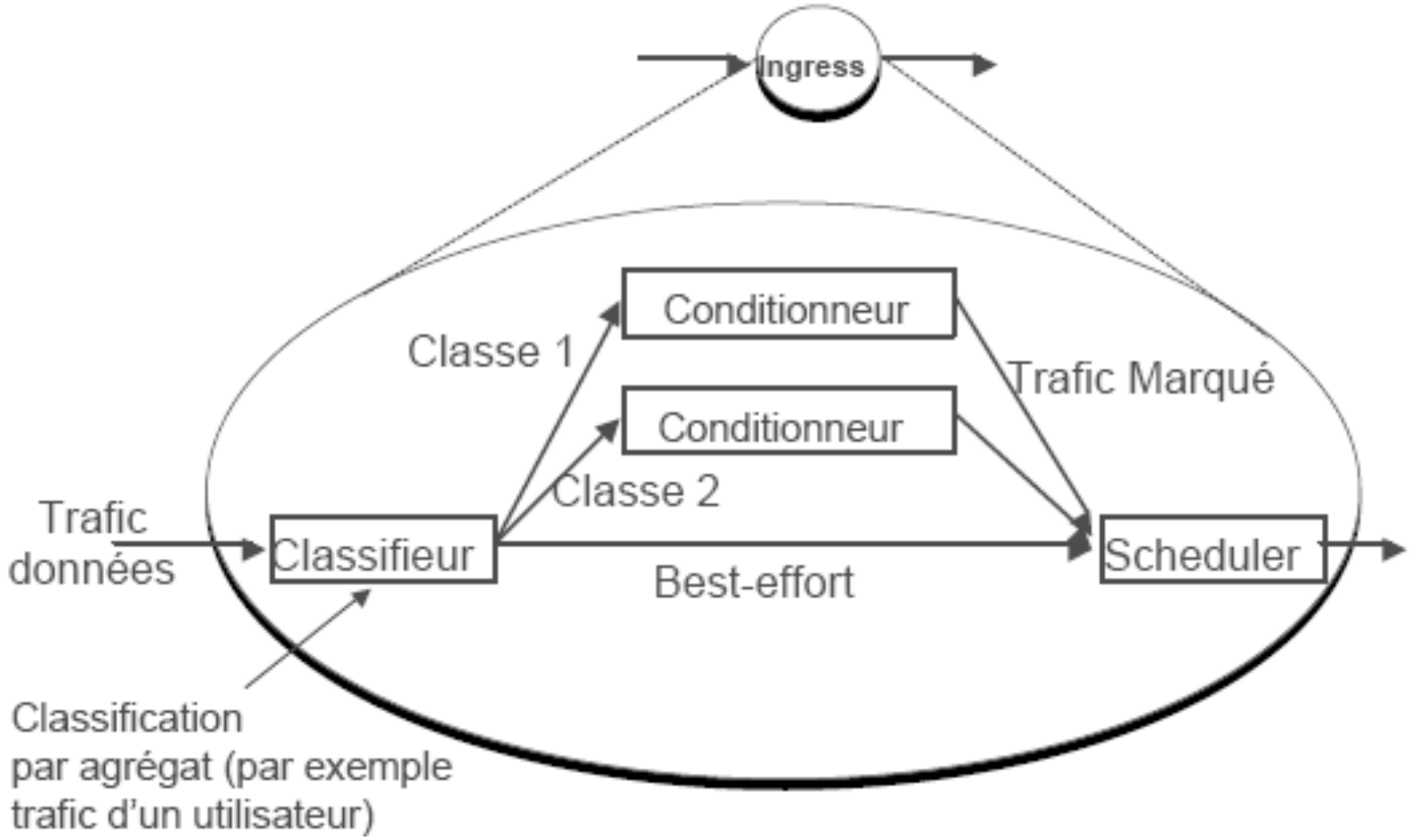
Service Assuré

- Définit 1 profil utilisateur i.e. combien de trafic assuré il peut envoyer dans le réseau (toutes destinations confondues)
- Réseau : fournit un taux de perte inférieur au best-effort
  - En cas de congestion, on perd d'abord les paquets Best-Effort
- Usager : Pas plus de paquets 'service assuré' que son profil
  - Sinon le trafic en excès est déclassé en Best Effort

Service Prémium

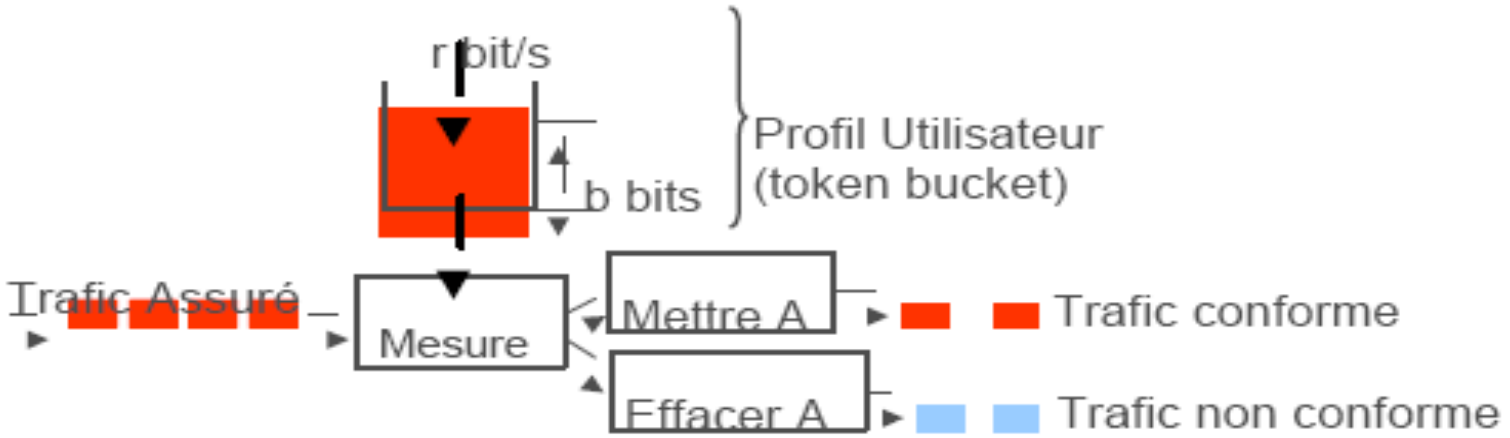
- "Tuyau virtuel" entre 1 "ingress" et 1 "egress" router
- Réseau : garantit pas de perte ET faible délai
- Utilisateur : n'envoie pas + que la taille du "tuyau"
  - Trafic en excès retardé et supprimé quand la file déborde

Routeur de bordure



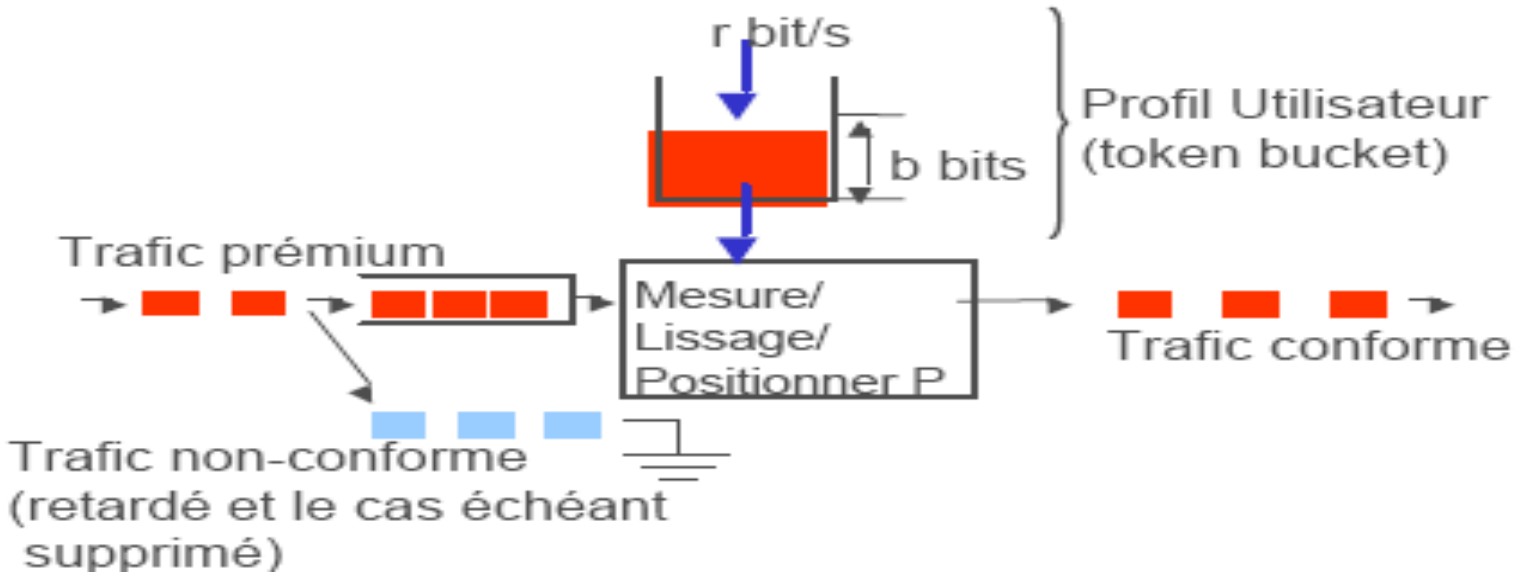
Mesure/Marquage

- Service assuré
- Trafic conforme marqué:
  - Bit A positionné
- Trafic non conforme (en excès) n'est pas marqué
  - Bit A effacé (s'il était positionné) dans chaque paquet ; trafic traité comme du best effort



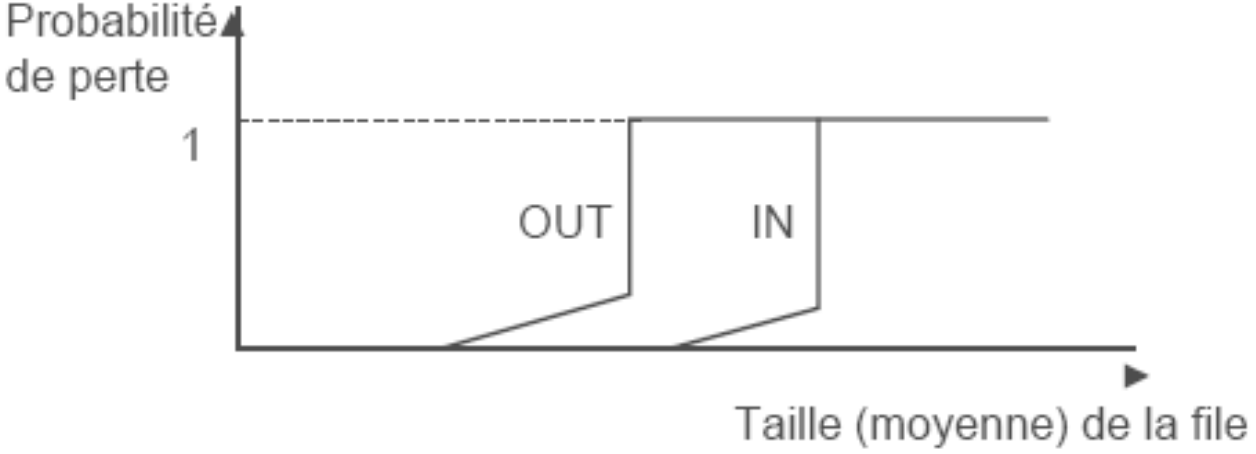
Mesure-Marquage-Lissage

- Service Premium
- Trafic conforme marqué (Bit P positionné)
- Trafic non-conforme retardé / supprimé (buffer plein)



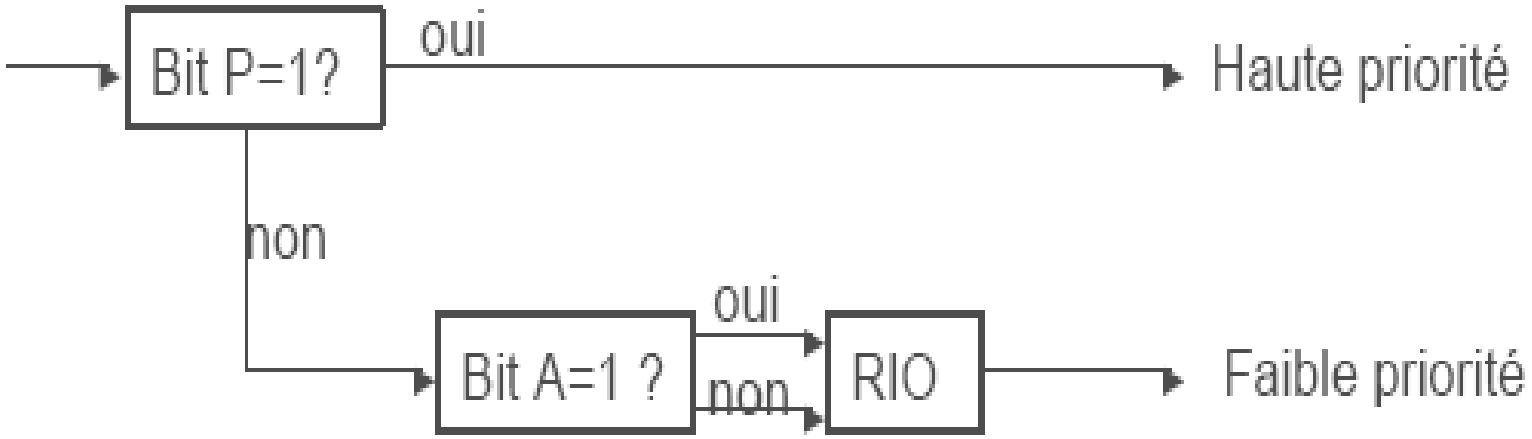
Scheduler

- Utilisé par tous les routeurs
- Pour le service Premium – priorité stricte
- Pour le service Assuré – RIO (RED with In and Out)
  - Suppression en priorité des paquets non-conformes
    - Pour le trafic non-conforme tenir compte de tous les paquets
    - Pour le trafic conforme ne tenir compte que des paquets conformes



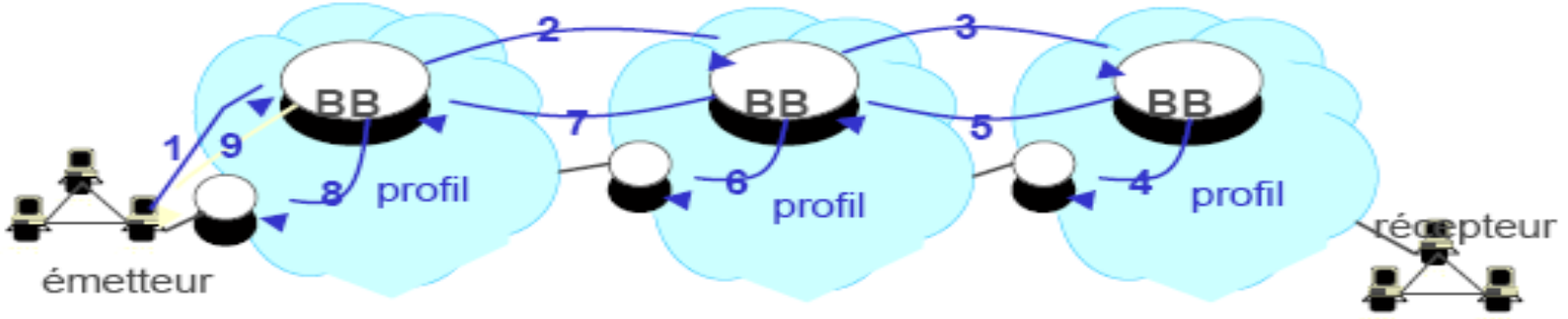
Scheduler

- Trafic envoyé avec une priorité haute
- Trafic assuré passe à travers RIO et est envoyé avec une priorité faible



Contrôle du chemin

- Chaque domaine possède un Bandwidth Broker (BB)
  - Allocation de bande passante entre entrée/sortie
- Réalise le contrôle d'admission pour le domaine
- BB pas facile à implanter
  - Connaissance complète du domaine
  - Goulot d'étranglement ("Single point of failure")



Comparaison

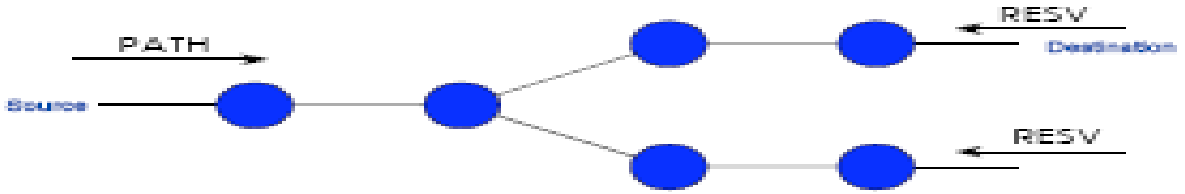
	Best-Effort	Diffserv	Intserv
Service	Pas d'isolation Pas de guaranties	Isolation et garanties par agrégat	Isolation et garanties par flux
portée	Bout en bout	Domaine	Bout en bout
Complexité	Pas de setup	Setup à long terme	Setup par flux
Passage à l'échelle	++ (les routeurs ne maintiennent que les tables de routage)	+ (routeurs de bordure par agrégat, routeurs internes par PHB)	-- (chaque routeur maintient l'état de chaque flux)

Signaling protocol: RSVP

- Goal:
  - Performs signaling to set up reservation state for a session
- Signaling on session basis
  - A session is a simplex data flow sent to a unicast or a multicast address, characterized by
    - <IP dest, protocol number, port number>
  - Multiple senders and receivers can be in same session
- Things to notice
  - Receiver initiated reservation
  - Decouple routing from reservation

RSVP: Principe

- La signalisation est constituée d'un flux de messages path et resv
  - Pas de réservation pour ce flux
  - Remise sans garantie et non acquittée

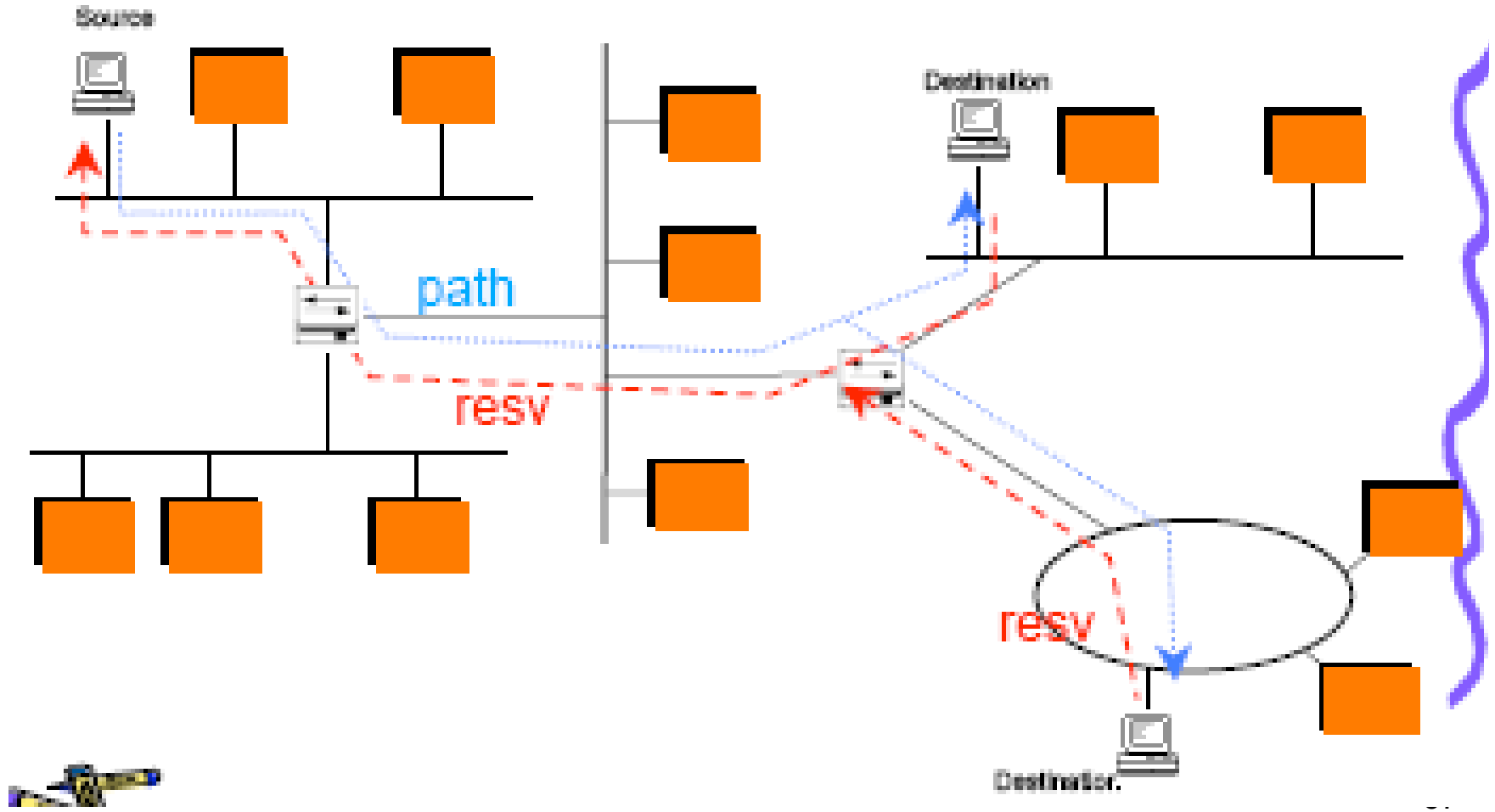


  - Chaque routeur RSVP traversé par un flux RSVP mémorise un état de ce flux
- Soft state
  - Notion de contexte applicatif
  - Rafraîchi par les messages resv
  - Après un certain délai L de non réception, l'état est détruit
- Libération immédiate de la réservation
  - Messages teardown (démolition)

PATH and RESV messages

- PATH message
  - specifies source traffic characteristics (Tspec)
    - Use token bucket
  - Set up the path state each router including the address of previous hop
    - Route Pinning
- RESV message
  - Specifies the reservation style, QoS desired (RSpec)
  - Source traffic characteristics (from PATH)
  - Filter specification, i.e., what senders can use reservation
  - Based on these routers perform reservation
  - Set up the reservation state at each router

How it works



En résumé

- Des garanties de services sont fournies si le flot est contraint et que le service assure un niveau minimum
- Un service de QoS fonctionne en boucle ouverte
- Le débit se définit avec une sporadicité
  - Contrôle du débit avec un leacky bucket/token bucket
- Usage pour l'Internet bloqué
  - Contrôle d'admission
  - Complexité des mécanismes
  - Coût des mécanismes par rapport à la bande passante