Réseaux
Quality of Service

Sommaire

La qualité de service est une notion subjective qui peut agir sur:

- le débit (téléchargement ou diffusion vidéo);
- le délai (pour les applications interactives ou la téléphonie);
- la disponibilité (accès à un service partagé);
- le taux de pertes de paquets (pertes sans influence pour de la voix ou de la vidéo, mais critiques pour le téléchargement).

Description

Une séquence de paquets envoyée d'une source vers une destination est appelée un **flux.**

Sur un réseau en mode connecté, tous les paquets appartenant à flux suivent **la même route**.

Sur un réseau fonctionnant dans le mode sans connexion, les paquets peuvent emprunter **différents parcours**.

Les besoins de chaque flux peuvent être caractérisés par quatre paramètres principaux : fiabilité, temps d'acheminement, gigue et bande passante.

Ceux-ci déterminent ensemble la **qualité de service**, que le flux requiert.

Ci-dessous les différents niveaux d'exigence de QoS

Application	Fiabilité	Latence	Gigue	Bande passante
Email	Haute	Faible	Faible	Faible
Transfert de fichier	Haute	Faible	Faible	Moyenne
Web	Haute	Moyenne	Faible	Moyenne
Session distante	Haute	Moyenne	Moyenne	Faible
Audio	Faible	Faible	Haute	Moyenne
Vidéo	Faible	Faible	Haute	Haute
VoIP	Faible	Haute	Haute	Faible
Vidéoconférence	Faible	Haute	Haute	Haute

Les quatre premières applications (email, transfert de fichier, Web et Session distante) réclament une grande fiabilité.

Aucun bit ne doit être transmis incorrectement.

Cet objectif est atteint en calculant un total de contrôle pour chaque paquet et en vérifiant ce total sur le destinataire (**CRC**).

Si un paquet est endommagé pendant le transport, il n'est pas acquitté et l'émetteur le retransmet.

A l'inverse, la vidéo et l'audio tolérent les erreurs et aucuns contrôle n'est fait.

Description: latence

Les applications de transfert de fichiers, d'email et de vidéo, ne sont pas sensibles aux retards d'acheminement ou latence.

Un retard uniforme de quelques secondes sur tous les paquets n'a aucun impact négatif.

Les applications en temps réel, telles que la téléphonie et la vidéoconférence, ont des exigences strictes en matière de temps d'acheminement.

Si chaque parole d'une conversation téléphonique est transportée exactement avec un retard de 2 secondes, les interlocuteurs trouveront la communication inacceptable.

Les applications interactives, telles que la navigation sur le Web ou une session à distance, sont davantage sensibles aux variations du temps d'acheminement ou gigue.

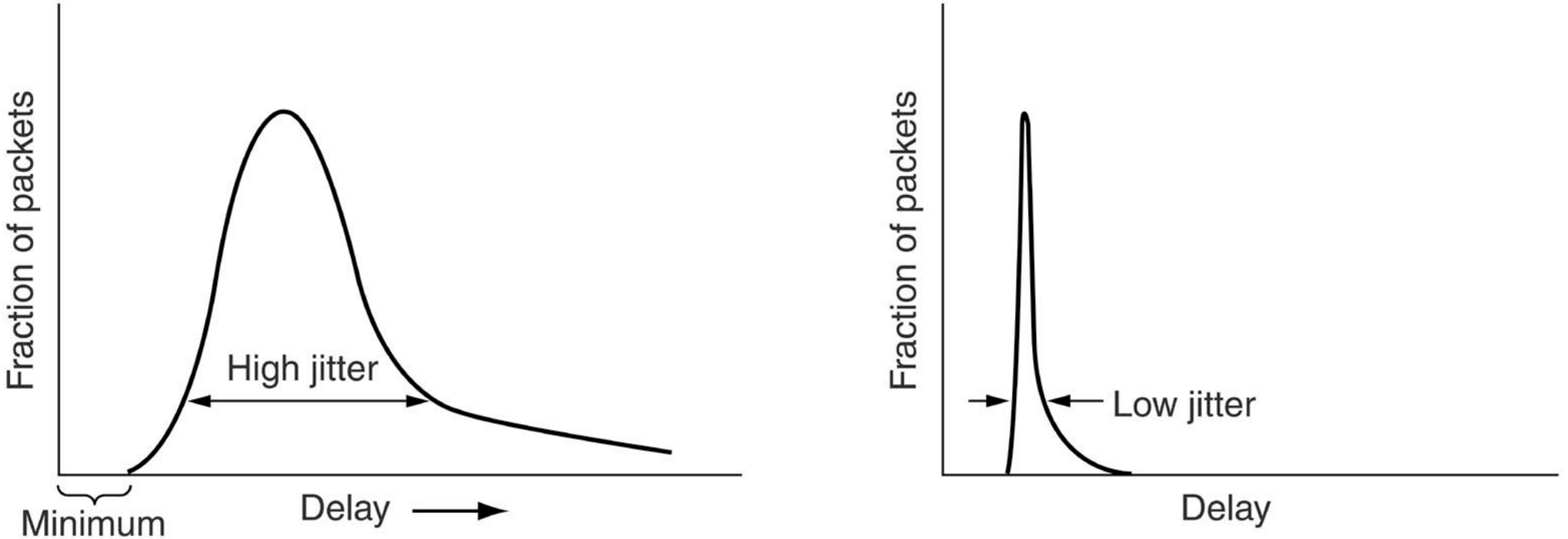
Une session sur un terminal souffre énormement de la gigue. Cela provoque une apparition des caractères à l'écran par courtes rafales.

La vidéo et surtout l'audio sont extrêmement sensibles à la gigue.

Si un utilisateur regarde une séquence vidéo par l'intermédiaire du réseau et que chaque image arrive avec un temps de retard aléatoire compris entre 1 et 2 s, le résultat sera terrible.

En ce qui concerne la lecture audio, même une gigue de quelques millisecondes est perceptible.

Description: gigue



La gigue peut être limitée en calculant le temps de transit attendu pour chaque saut sur l'itinéraire.

Lorsqu'un paquet arrive sur le routeur, celui-ci vérifie la variation en plus ou en moins par rapport au temps prévu.

Cette information est inscrite dans le paquet et adaptée à chaque saut.

Si le paquet est en avance, il est retenu suffisamment longtemps pour le rapprocher du délai prévu ; s'il est en retard, le routeur l'expédie le plus vite possible.

Les applications ont des exigences diverses en matière de bande passante.

Le courrier électronique et la connexion à distance peuvent se contenter de peu alors que la vidéo est très consomatrice.

Il existent plusieurs technique pour gérer la QoS:

- Réservation en excès ;
- Mise en tampon ;
- Canalisation du traffic ;
- Algorithme du seau percé (leaky bucket) ;
- Algorithme du seau à jeton (token bucket) ;
- Réservation de ressources ;
- Contrôle d'admission ;
- Routage proportionnel ;
- Ordonnancement ;
- ...

Une solution simple est de fournir tant de capacité, d'espace de tampon et de bande passante sur les routeurs que les paquets circuleront sans difficulté.

L'ennui est qu'elle est coûteuse.

Mais avec le temps, les concepteurs auront une meilleure idée des quantités jugées suffisantes, et cette technique pourra devenir intéressante et même pratique.

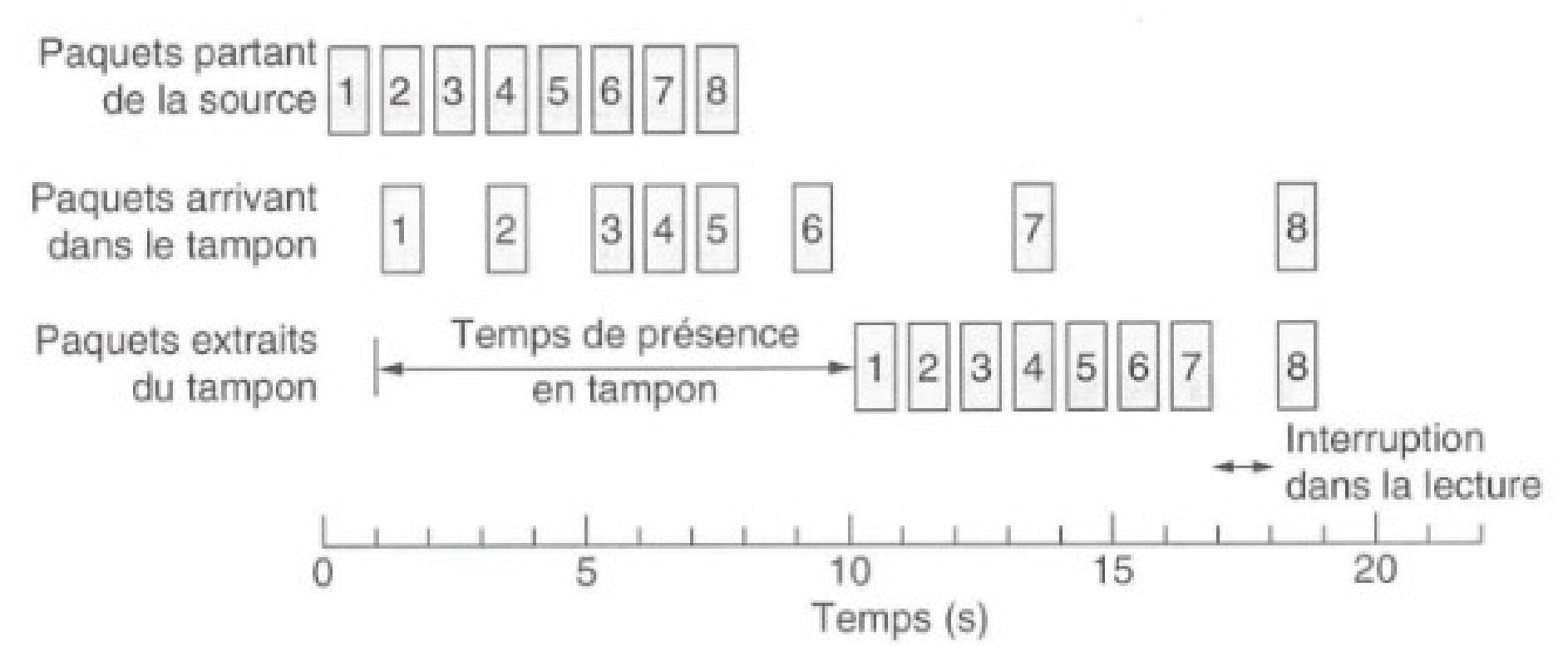
Dans une certaine mesure, le réseau téléphonique bénéficie de ressources en excès. Il est rare de décrocher le téléphone et de ne pas avoir de tonalité.

Il y a une telle capacité en réserve que la demande sera toujours satisfaite.

Les flux de paquets peuvent être placés en tampon sur le récepteur avant d'être remis.

L'emploi de tampons de réception n'a aucune incidence sur la fiabilité ou le débit mais augmente le délais en réduisant la gigue.

Cela peut être utile pour les applications audio ou vidéo à la demande pour lesquelles la gigue représente un gros problème.



À t=10 s, la lecture commence, les paquets 1 à 6 sont déjà dans le tampon et peuvent être extraits à intervalles réguliers pour pouvoir assurer une lecture régulière.

Malheureusement, le paquet 8 a été tellement retardé qu'il n'est pas disponible au moment où il doit être lu.

La lecture est alors suspendue jusqu'à son arrivée, provoquant une interruption désagréable.

Dans l'exemple précédent, la source transmet des paquets à intervalles réguliers. Mais dans d'autres situations, ils pourraient être émis irrégulièrement et provoquer une congestion sur le réseau.

La transmission non uniforme de paquets est un phénomène courant si le serveur gère plusieurs flux à la fois tout en autorisant l'exécution d'autres tâches, telles que l'avance ou le retour rapides, l'authentification utilisateur, etc.

la **canalisation de trafic** *(traffic shaping),* permet de réguler le trafic côté serveur, et non plus côté client.

La canalisation du trafic concerne la régulation du rythme moyen auquel les données sont transmises, et par là même, des rafales.

Cette technique est à opposer aux protocoles par fenêtres d'anticipation, qui **limitent la quantité des données** émises en une fois, **mais pas le rythme** auquel elles sont transmises.

Lorsqu'une connexion est établie, l'utilisateur et le sous-réseau, c'est-à-dire le client et l'opérateur, se mettent d'accord sur un certain comportement du trafic pour un circuit.

Cet engagement entre eux est souvent appelé le **contrat de niveau de service,** ou *SLA (Service Level Agreement).*

Dans la pratique, cela se traduit par un client annonçant :

« Mon trafic est de ce type, pouvez-vous le transporter ? »

Si l'opérateur est d'accord, il est confronté à un autre aspect du problème : comment savoir si le client respectera ses engagements et que faire si ce n'est pas le cas ?

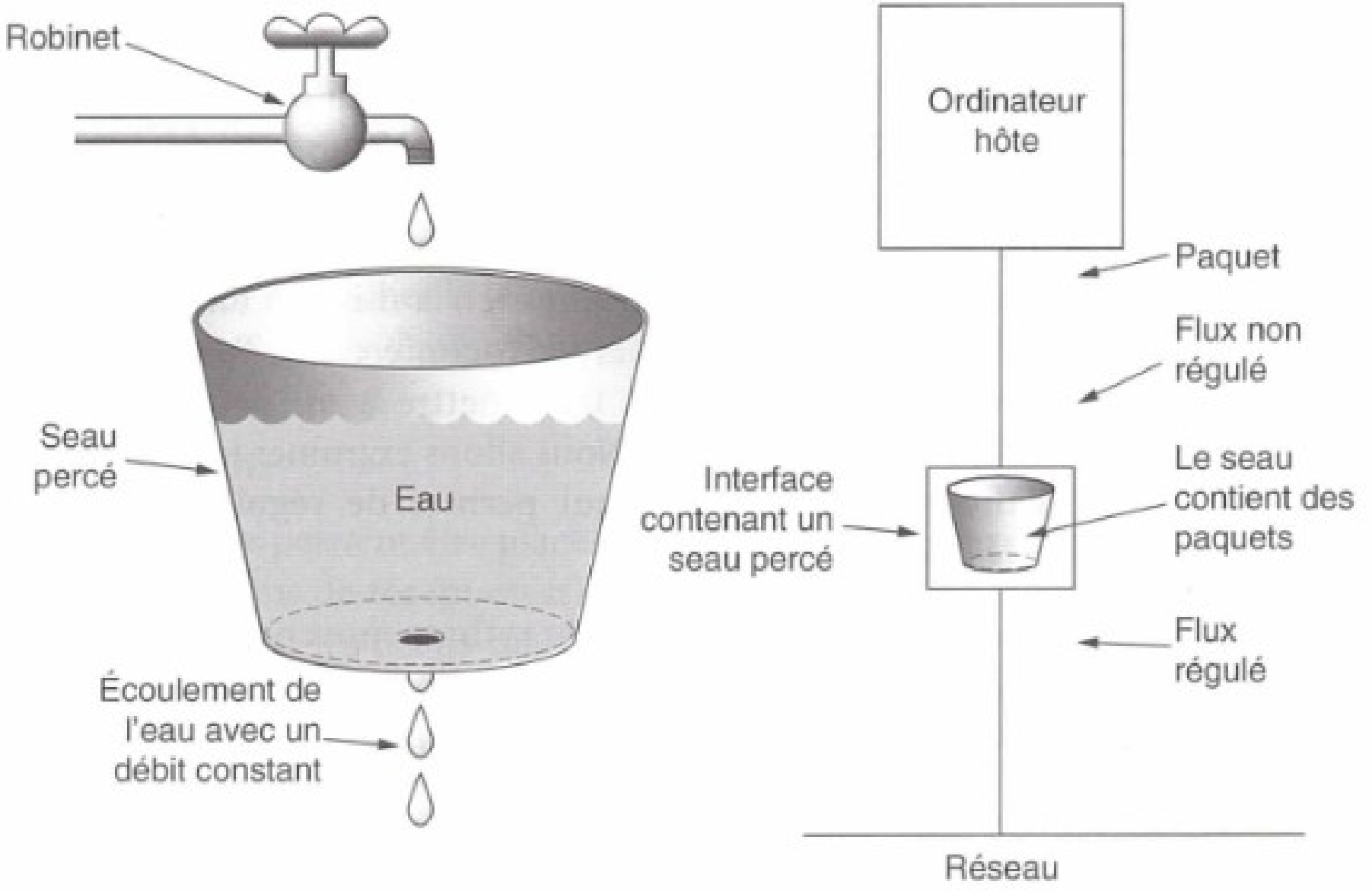
Pour cela, il doit adopter une **stratégie de surveillance du trafic** *(traffic policing).*

Imaginez un seau dont le fond est percé d'un petit trou.

Quelle que soit la vitesse à laquelle vous le remplissez, la vitesse d'écoulement par le trou reste constante : p s'il y a de l'eau, et zéro s'il n'y en a pas.

Et lorsque le seau est plein, continuer de verser de l'eau provoque un débordement et une perte du liquide, c'est-à-dire que l'eau perdue n'apparaîtra pas dans le flux de sortie qui passe par le trou.

Techniques de gestion de la QoS: leaky bucket



L'implémentation de l'algorithme du seau percé est simplement une file d'attente de taille fixe.

Un paquet qui arrive est accepté si la file n'est pas pleine, sinon, il est éliminé.

À moins que la file ne soit vide, un paquet est transmis par top d'horloge.

Si un ou plusieurs processus sur l'hôte tentent d'envoyer un paquet lorsque le nombre maximal admis est déjà atteint, le nouveau paquet est supprimé.

L'algorithme du seau percé impose un **écoulement rigide à un débit moyen**, quelles que soient la sporadicité et la longueur des rafales.

Pour certaines applications toutefois, il est parfois préférable d'accélérer l'écoulement lorsque de grandes rafales arrivent.

Il faudrait donc disposer d'un algorithme plus souple, de préférence qui ne perde jamais de données, comme **l'algorithme du seau à jetons** *(token bucket).*

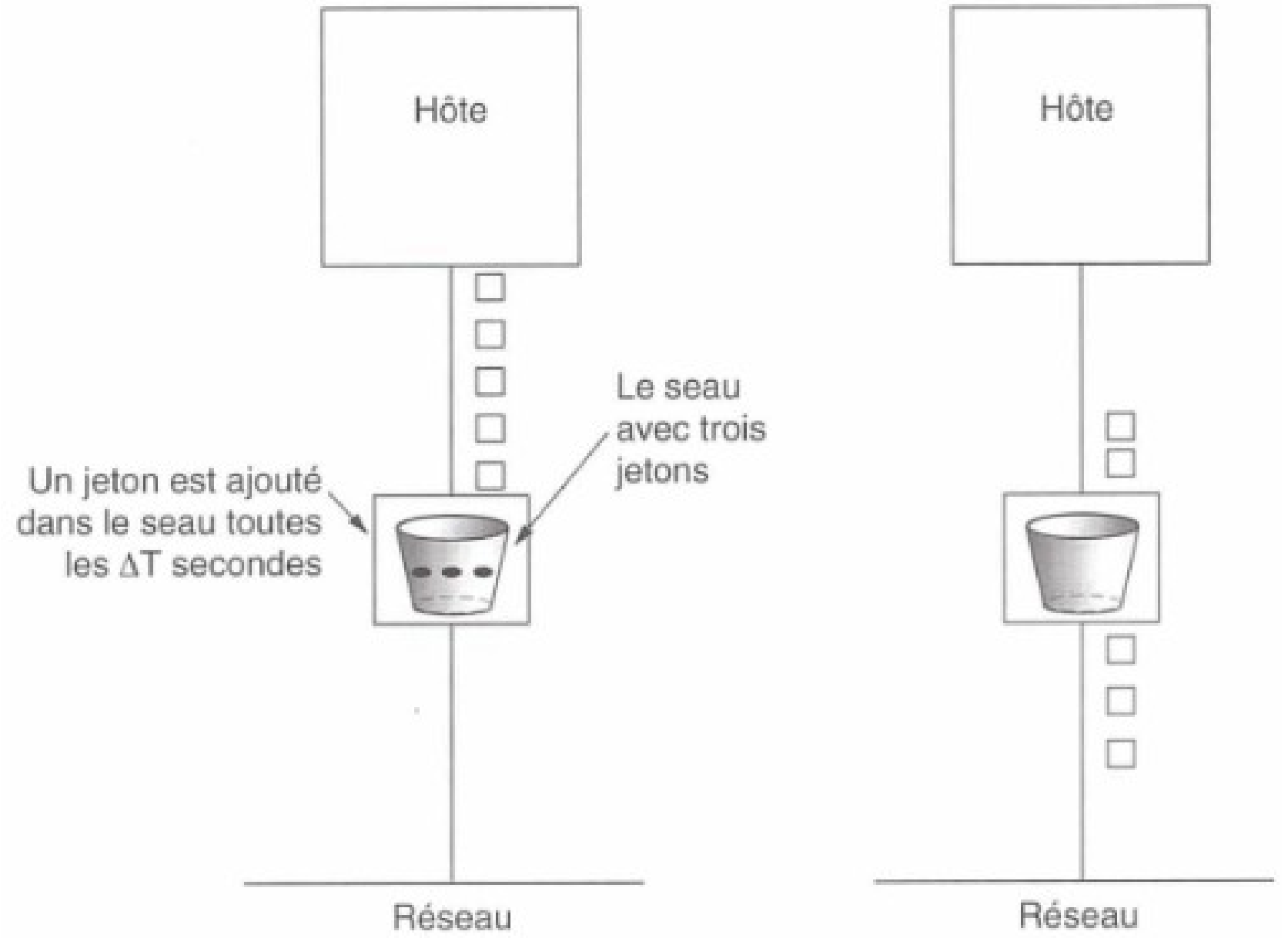
Dans cet algorithme, le seau percé contient des jetons générés par une horloge à un rythme d'un jeton toutes les ΔT secondes.

Imaginons un seau avec trois jetons et cinq paquets en attente de transmission.

Pour qu'un paquet soit transmis, il doit capturer et détruire un jeton.

Trois des cinq paquets sont passés à travers, mais les deux autres sont bloqués et doivent attendre la génération de deux jetons supplémentaires.

Techniques de gestion de la QoS: token bucket



Le seau percé ne permet pas à un hôte au repos d'économiser des droits de transmission pour envoyer des rafales plus importantes ultérieurement.

Le seau à jetons permet de le faire jusqu'à concurrence de la taille maximale du seau, α.

Cette propriété permet ainsi d'envoyer en une fois des rafales pouvant comprendre jusqu'à α paquets.

Cela permet un **meilleur temps de réponse** face aux transmissions sporadiques et importantes.

Le seau à jetons jette des jetons lorsqu'il se remplit mais ne supprime pas de paquets, alors que le seau percé perd des paquets lorsqu'il est plein.

Calcul de la durée des rafales:

Soit L secondes la longueur d'une rafale, C octets la capacité du seau, D_e octets le débit en entrée du seau, D_s octets le débit maximal en sortie.

On peut dire qu'une rafale contient: $C + LD_e$

On peut dire que le nombre d'octets par rafale est : LD_s

Donc: $C + LD_e = LD_s \rightarrow L = C / (D_e \times D_s)$

Pouvoir réguler le comportement du trafic en entrée est un bon départ pour garantir la qualité de service.

Pour réguler efficacement, cela signifie implicitement que tous les paquets d'un flux doivent suivre la même route.

Il faut établir une voie du type circuit virtuel, de la source vers la destination, et tous les paquets qui appartiennent au flux doivent suivre cette route.

Une fois qu'on dispose d'une route spécifique pour un flux donné, il est alors possible de réserver les ressources suivantes le long du parcours: **bande passante**, **tampons mémoire** et **cycles d'UC**.

Bande passante:

Si un flux requiert 1 Mbit/s et que la ligne de sortie possède une capacité de 2 Mbit/s, il ne sera pas possible de faire passer trois flux sur cette ligne.

Réserver de la bande passante signifie ne pas **sur-réserver** une ligne de sortie.

Mémoire tampon:

Lorsqu'un paquet arrive, il est déposé sur la carte d'interface.

Le logiciel du routeur doit ensuite le copier dans un tampon mémoire.

Ce tampon est placé en file d'attente de transmission sur la ligne de sortie choisie.

Si aucun tampon n'est disponible, le paquet est supprimé.

Pour assurer une bonne qualité de service, certains tampons doivent être réservés pour un flux spécifique, de sorte que celui-ci ne soit pas en concurrence avec d'autres flux pour l'obtention de tampons.

On pourrait déduire que si le traitement d'un paquet prend 1 us, un routeur peut traiter un million de paquets par seconde.

C'est faux!

Il y a toujours des périodes d'inactivité en raison de fluctuations statistiques de la charge.

Si le processeur a besoin de chacun de ses cycles pour réaliser son travail, la perte de quelques cycles seulement provoquerait un retard qu'il ne pourrait **jamais** rattraper...

Une fois que la canalisation de flux est appliquée grâce à la réservation des ressources, on est sûr que le traffic suit une seule routesur laquelle les capacitées sont réservées.

Le routeur peut alors accepter ou refuser un flux.

La décision ne se fait pas seulement sur simple comparaison entre la quantité de resources demandées et celle disponible.

Certaine application sont plus tolérente que d'autre, par exemple, une lecture vidéo à 30 images/sec pourra permettre une dégradation jusqu'à 25 images/sec si les ressources ne sont pas suffisantes.

De nombreuses parties peuvent être impliquées dans la négociation du flux:

- l'émetteur,
- le récepteur
- tous les autres routeurs entre eux sur l'itinéraire

Les flux doivent être décrits précisément en termes de paramètres spécifiques qui peuvent être négociés.

Un jeu de ces paramètres est appelé **spécification de flux**.

À mesure que la spécification se propage le long du parcours, chaque routeur l'examine et modifie les paramètres selon les ressources disponibles.

Les modifications ne peuvent que réduire le flux.

Lorsque la spécification atteint l'autre extrémité, les paramètres peuvent être établis.

La spécification de flux, décrite dans les RFC2210 et 2211, présente 5 paramètres:

- Débit du seau à jeton ;
- Taille du seau à jeton ;
- Débit de pointe ;
- Taille minimale d'un paquet ;
- Taille maximale d'un paquet.

Imaginons qu'un routeur puisse traiter 100 000 paquets par seconde.
Si un flux de 1048576 octets (1 Mo/s) se présente avec des tailles minimale et maximale de paquets de 512 octets.
Le routeur calcule qu'il pourrait recevoir 2 048 paquets / sec.
Il devrait lui réserver ≈ 2% de ses ressources processeur.
Si une règle sur le routeur interdit d'allouer plus de 50 % des ressources du processeur et que celui-ci est déjà occupé à 49 %, le flux doit être rejeté.

Des calculs semblables doivent être réalisés pour les autres ressources.

La plupart des algorithmes de routage tentent de trouver le chemin idoine vers chaque destination.

Une autre approche serait de répartir le trafic vers chaque destination entre plusieurs itinéraires.

Puisque les routeurs ne possèdent généralement pas une vision globale de la charge à l'échelle du réseau, la seule méthode viable pour atteindre cet objectif est qu'ils se servent d'informations disponibles localement.

Un moyen simple est de diviser le trafic en parts égales, ou proportionnellement à la capacité des différentes lignes de sortie.

Il existe toutefois des algorithmes plus sophistiqués...

Si un routeur gère plusieurs flux de paquets, le danger subsiste de voir un flux s'accaparer une trop grande part de ses ressources au détriment des flux concurrents.

Le traitement des paquets dans l'ordre de leur arrivée signifie qu'un émetteur zélé pourra s'approprier la plupart des ressources des routeurs au travers desquels ses paquets passent, privant ainsi les autres émetteurs d'une meilleure qualité de service.

Pour contrecarrer ce genre de tentative, divers algorithmes d'ordonnancement des paquets ont été inventés.

L'un des premiers algorithmes à avoir été inventé est l'algorithme **d'attente équitable** *(fair queueing).*

Les routeurs dispose de files d'attente distinctes pour chaque ligne de sortie et pour chaque flux.

Lorsqu'une ligne est au repos, le routeur analyse tour chaque file d'attente en y prélevant chaque fois le premier paquet.

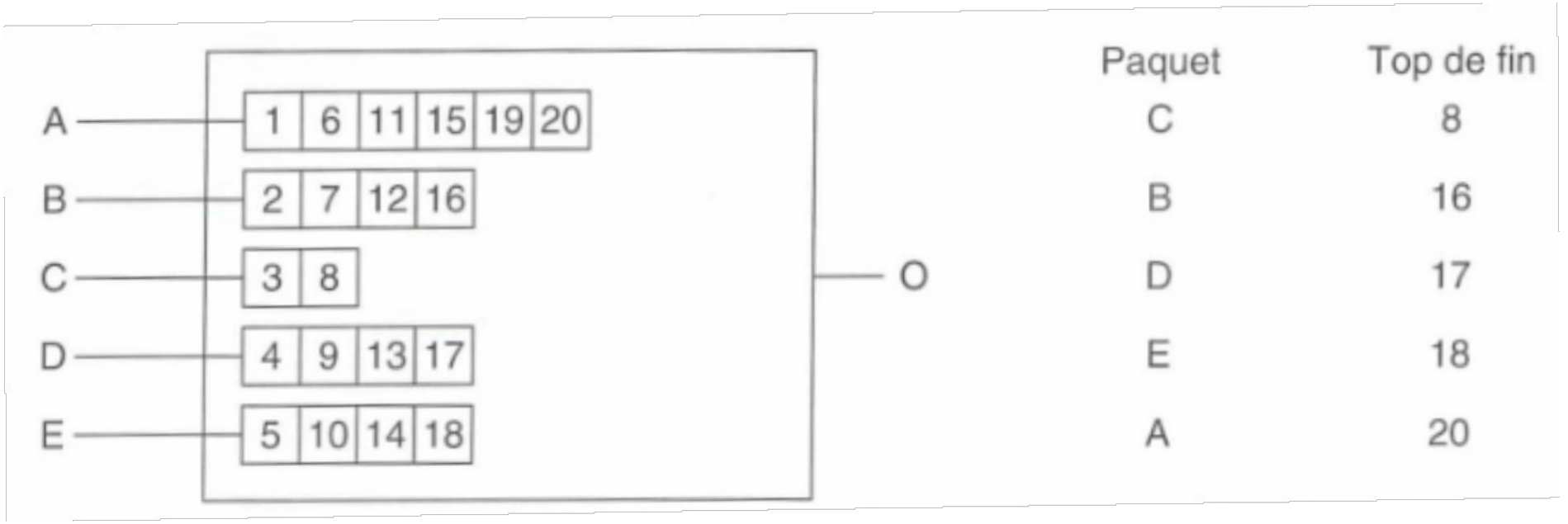
De cette façon, s'il y a α prétendants pour une ligne de sortie donnée, chacun d'eux aura la main tous les α paquets.

Cet algorithme présente néanmoins un problème : il attribue plus de bande passante aux hôtes qui expédient de grands paquets qu'à ceux qui en envoient des petits.

Une amèlioration apportée est le traitement des files d'attente est réalisé par octet plutôt que par paquet.

Chaque octet de chaque file est analysé à tour de rôle jusqu'à localiser le top d'horloge sur lequel chaque paquet se termine.

Les paquets sont ensuite classés en fonction de cette information et envoyés dans cet ordre.



Le problème de cet algorithme est d'accorder à tous les hôtes la même priorité.

Il est souvent utile d'octroyer davantage de bande passante aux serveurs de films vidéo qu'aux serveurs de fichiers ordinaires.

Cet algorithme est celui **d'attente équitable pondérée** *(weighted fair queueing) et* est largement utilisé.

Ces services s'adressent à la fois aux applications unicast et multicast.

Un exemple de la première catégorie serait un utilisateur regardant un clip vidéo à partir d'un site d'informations.

Un exemple de la seconde serait un ensemble de stations de télévision numérique diffusant des programmes sous forme de flux de paquets IP se destinant à plusieurs abonnés situés à différents endroits.

Le protocole principal de l'IETF pour l'architecture de services intégrés est le **protocole de réservation de ressources**, ou **RSVP** *(Resource reSerVation Protocol).*

Il est **uniquement** utilisé pour faire des réservations ;

Le protocole RSVP permet à plusieurs émetteurs de transmettre des données à plusieurs groupes de destinataires.

Il autorise les récepteurs individuels à changer librement de canal, et optimise l'exploitation de la bande passante tout en empêchant les situations de congestion.

Le protocole utilise le routage multicast et chaque groupe reçoit une adresse.

Lorsqu'un émetteur souhaite diffuser des données se destinant à tous les membres d'un groupe, il place l'adresse du groupe dans chaque paquet qu'il envoie.

L'algorithme standard de routage multicast construit ensuite un arbre recouvrant englobant tous les membres du groupe.

Cet algorithme ne fait pas partie du protocole RSVP.

La seule différence par rapport à un routage multicast traditionnel est un envoi périodique d'informations supplémentaires au groupe, qui permet d'indiquer aux routeurs situés le long des itinéraires de maintenir certaines structures de données dans leur mémoire.

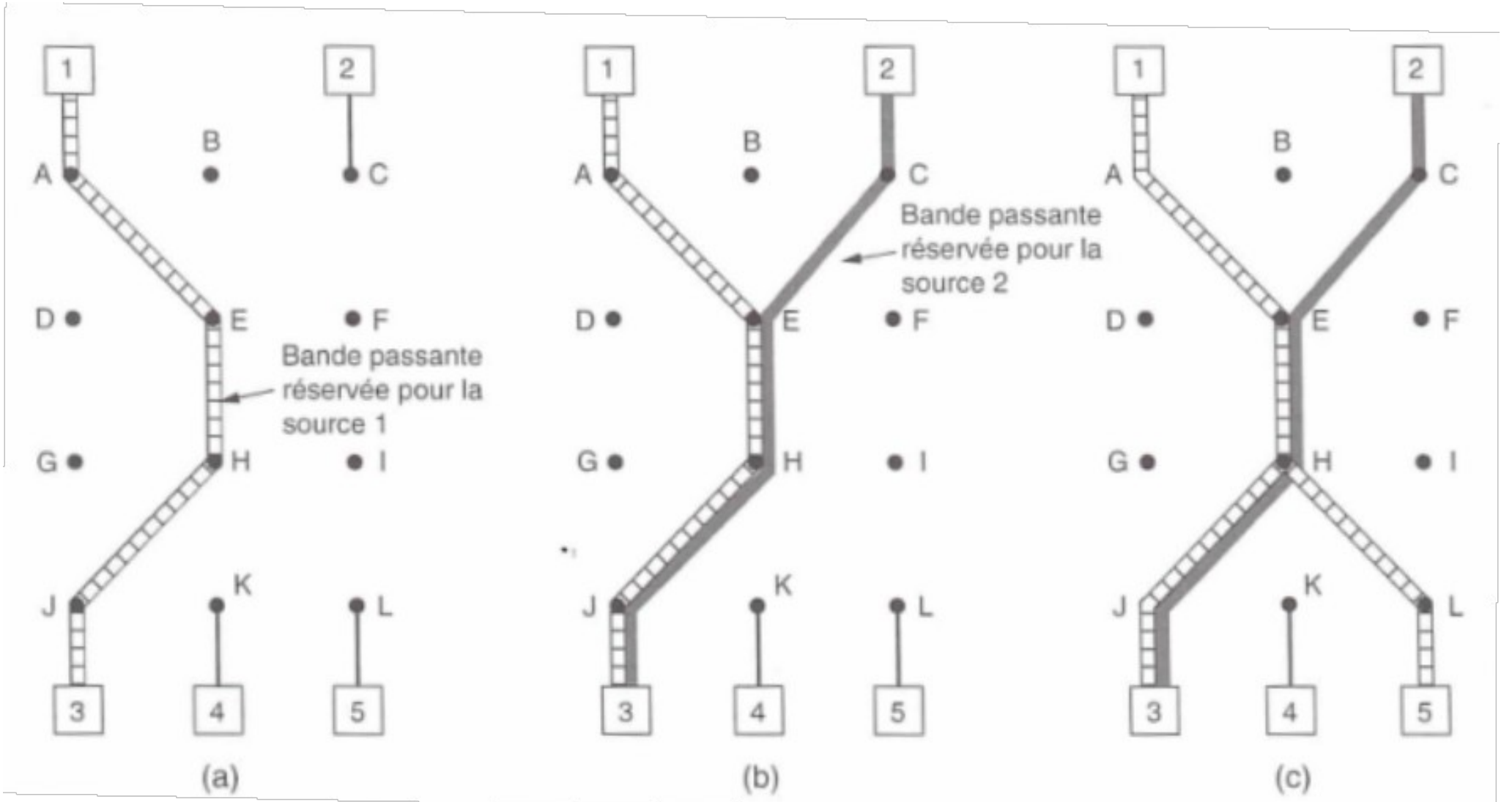
Pour bénéficier d'une meilleure réception et éviter toute congestion, les récepteurs d'un groupe peuvent envoyer un message de réservation de ressources le long de l'arbre recouvrant, qui sera propagé jusqu'à l'émetteur.

Le message est diffusé au moyen de l'algorithme RPF de comparaison de chemin inverse étudié plus haut.

À chaque saut, le routeur concerné prend note de la demande et réserve la bande passante requise.

Si elle se révèle insuffisante, une notification d'erreur est retournée au processus ayant généré la demande.

Lorsque le message parvient à la source, la bande passante est alors réservée tout le long de l'itinéraire entre l'émetteur et le récepteur.


1
2
A
B
C
D
E
F
G
H
I
J
K
L
3
4
5
Bande passante réservée pour la source 1
Bande passante réservée pour la source 2
(a)
(b)
(c)

Lors de l'enregistrement d'une réservation, un récepteur peut spécifier une ou plusieurs sources de diffusion.

Il peut également indiquer si ces choix sont fixes pour la durée de la réservation ou s'il souhaite conserver la possibilité de changer de sources ultérieurement.

Les routeurs utilisent cette information pour optimiser les prévisions de bande passante. En particulier, deux récepteurs ne peuvent être établis comme pouvant partager un canal que s'ils sont d'accord pour ne pas changer de source par la suite.

Cette stratégie est adoptée pour offrir une situation totalement dynamique où bande passante réservée et choix de la source sont dissociés.

Une fois qu'un récepteur a réservé des ressources, il peut basculer vers une autre source tout en conservant la portion de parcours existante qui est alors valide pour la nouvelle source.

Avantages:

Les algorithmes de services intégrés sont capables de fournir une bonne qualité de service à un ou plusieurs flux car ils réservent les ressources requises le long des itinéraires.

Inconvènients:

Ils présentent toutefois l'inconvénient de nécessiter une configuration avancée pour établir chaque flux, une caractéristique qui les empêche de bien s'adapter lorsqu'il faut établir des milliers ou des millions de flux.

Ils maintiennent aussi en interne sur les routeurs l'état de chaque flux, ce qui les rend vulnérables lorsqu'un routeur tombe en panne.

Enfin, les changements requis dans le code du routeur sont conséquents, et l'établissement des flux requiert des échanges complexes entre routeurs

Les services différenciés peuvent être offerts par un ensemble de routeurs formant un domaine administratif (eg. FAI).

Le domaine définit un ensemble de **classes de services** avec leurs règles de transmission correspondantes.

Lorsqu'un client signe un contrat de services différenciés, la qualité de service fournie à ses paquets est en rapport avec le service choisi.

Lorsque des paquets pénètrent dans un domaine, ils peuvent contenir un champ ***Type de service***.

Le trafic d'une certaine classe peut être soumis à certaines exigences de comportement et de débit, comme ce que permet de faire l'algorithme du seau percé.

Classe de services EF ***(Expedited Forwarding)***:

Le choix des classes de services est l'affaire de chaque opérateur. Mais l'IETF travaille sur la définition de classes de services **indépendantes** du réseau.

Il existe deux classes de services : service ordinaire et service accéléré.

La plus grande majorité du trafic est traitée avec un service ordinaire, mais une petite fraction doit être traitée avec le service accéléré.

Ces paquets doivent alors être capables de traverser le sous-réseau comme s'il n'y avait pas d'autres paquets présents.

Une façon d'implémenter cette stratégie est de programmer les routeurs pour qu'il y ait deux files d'attente par ligne de sortie:

- une pour les paquets ordinaires ;
- l'autre pour les paquets accélérés.

Lorsqu'un paquet arrive, il est placé dans la file d'attente appropriée.

Le mécanisme d'ordonnancement devrait utiliser une technique telle que l'attente équitable pondérée.

Par exemple, si la charge se compose de 10 % de trafic accéléré et de 90 % de trafic ordinaire, 20 % de la bande passante pourraient être dédiés à la première classe de trafic, et le reste à la seconde classe.

Le trafic accéléré bénéficierait de deux fois plus de bande passante et d'un faible délai d'acheminement.

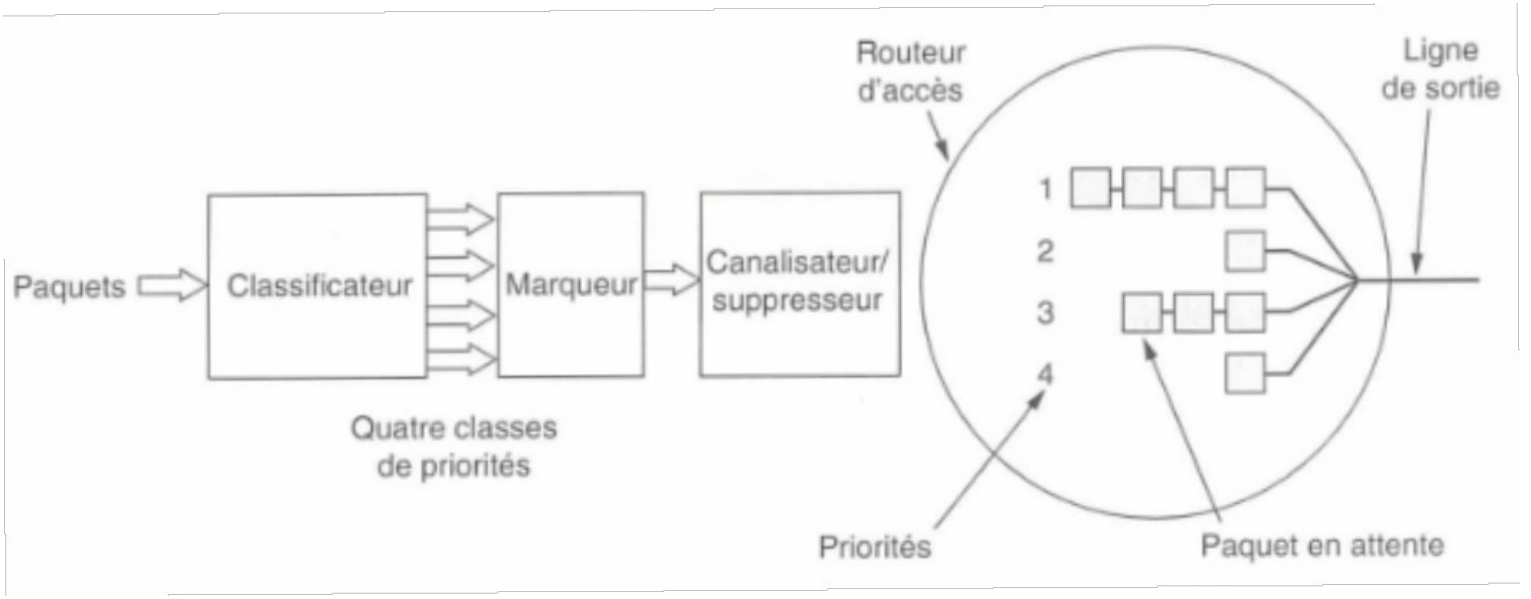
Classe de services AF *(**Assured Forwarding**):*

Il existe une stratégie de classes de services plus élaborée, la transmission garantie, ou AF *(Assured Forwarding).*

Elle spécifie quatre classes de priorités, chacune bénéficiant de ses propres ressources.

De plus, il définit trois types de probabilités de suppression de paquets en cas de congestion : faible, moyenne et haute.

Combinées ensemble, ces deux catégories de propriétés définissent douze classes de services.



La première étape consiste à classer les paquets en fonction des quatre classes de priorités.

Elle peut être réalisée sur l'hôte émetteur, ou sur le routeur d'accès (le premier).

L'avantage d'effectuer la classification sur l'hôte émetteur est qu'il est possible de savoir à quel flux appartient quel paquet.

La deuxième étape consiste à marquer les paquets en fonction de la priorité définie. (utilisation du champ *TOS)*

La troisième étape consiste à faire passer les paquets à travers un filtre de **canalisation/suppression** qui peut retarder ou éliminer certains paquets pour donner aux quatre flux un comportement acceptable, par exemple en utilisant les techniques du seau percé ou celles du seau à jetons.

S'il y a trop de paquets, certains pourront être supprimés à ce stade en fonction de leur catégorie.

Il y a également la possibilité d'utiliser des stratégies plus élaborées de mesure ou de rétroaction.

Le routage par étiquette est fondé sur les informations de l'étiquette plutôt que sur l'adresse de destination.

En utilisant l'étiquette comme index d'une table interne, les lignes de sortie appropriées peuvent être trouvées au moyen d'une simple recherche.

Cette technique permet de réaliser un routage très rapide, et toutes les ressources requises peuvent être réservées le long de l'itinéraire (Ressemble au Circuits Virtuels).

Cette méthode de commutation a reçu diverses appellations:

- **commutation par étiquettes** *(label switching),*
- **commutation par marquage** *(tag switching)*
- *…*

l'IETF l'a normaliser sous la désignation **MPLS** *(MultiProtocol Label Switching),* soit commutation multiprotocole par étiquettes.

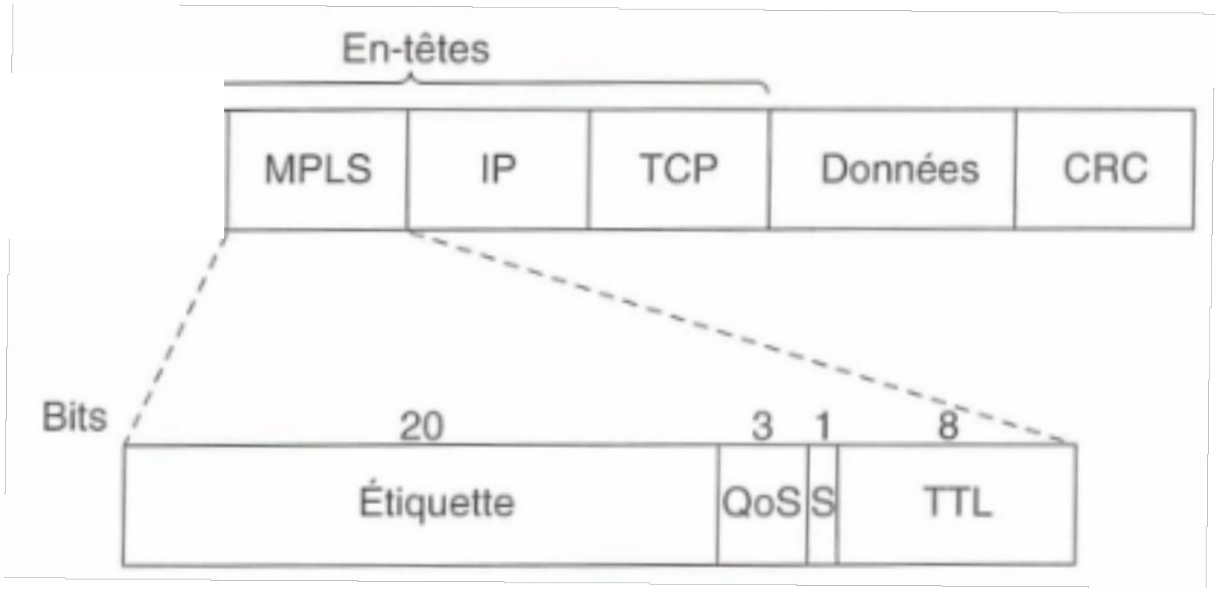
Le premier problème qui se pose avec cette méthode est de savoir où placer l'étiquette.

Puisque les paquets IP n'ont pas été conçus pour des circuits virtuels, il n'y a pas de champ disponible dans l'en-tête pour y inscrire un numéro de circuit virtuel.

Pour cette raison, il a fallu ajouter un nouvel en-tête MPLS **avant** l'en-tête IP.

L'en-tête MPLS générique se compose de quatre champs:

- le plus important étant le champ *Etiquette* qui contient l'index ;
- le champ *QoS* indique la classe de services ;
- le champ *S* sert à l'empilement de plusieurs étiquettes dans les réseaux hiérarchiques, s'il contient 0, le paquet est rejeté. Cette fonction prévient la formation de boucle infinie en cas d'instabilité du routage.
- le champ TTL (Time To Live).



L'étiquette sert d'index dans une table pour déterminer la ligne de sortie à utiliser ainsi que la nouvelle étiquette à employer.

Le changement d'étiquette est pratiqué dans tous les sous-réseaux de circuits virtuels, car sa signification est locale.

Deux routeurs différents peuvent ainsi envoyer des paquets avec une étiquette semblable à un autre routeur afin qu'ils soient expédiés sur une même ligne de sortie.

Pour que les flux soient identifiables à l'autre bout, les étiquettes doivent être adaptées à chaque saut.

Une différence par rapport aux circuits virtuels traditionnels est le niveau d'agrégation. Il est fréquent de voir les routeurs agréger plusieurs flux qui se terminent au niveau d'un routeur ou d'un LAN donné, et leur attribuer la même étiquette.

Les flux ainsi groupés sous une même étiquette sont dits appartenir à une même classe d'équivalence de transmission, ou **FEC** *(Forwarding Equivalence Class).*

Lorsqu'un routeur démarre, il vérifie pour quels flux il représente la destination finale, par exemple quels sont les hôtes destinataires qui se trouvent sur son LAN.

Il crée une ou plusieurs classes **FEC** pour ces flux, alloue à chacune une étiquette, et transmet les étiquettes à ses voisins.

Ceux-ci les inscrivent dans leur table d'acheminement puis envoient de nouvelles étiquettes à leurs voisins, et ainsi de suite jusqu'à ce que tous les routeurs aient pris connaissance du parcours.

Les ressources peuvent également être réservées à mesure que l'itinéraire est construit pour garantir une qualité de service appropriée.

MPLS peut opérer sur plusieurs niveaux en même temps.

Au niveau le plus haut, chaque opérateur de réseau sur l'itinéraire de la source vers la destination peut être vu comme une sorte de **métarouteur**.

Un paquet peut porter en lui une pile d'étiquettes.

Le bit ***S*** permet à un routeur qui enlève une étiquette de savoir s'il y en a d'autres (***S*** = 0) ou non (***S*** = 0).

Dans la pratique, cette fonctionnalité est principalement exploitée pour implémenter des réseaux virtuels privés et des tunnels récursifs.