La Qualité des services dans les réseaux

Fred Hémery

IUT Béthune
Département
Réseaux &
Télécommunications

TR3-2 QoS — 07/08

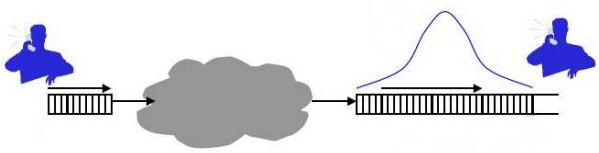
Plan

Introduction

- Le trafic sur les réseaux de données augmente d'année en année
- L'augmentation de la bande passante sur les réseaux tente de suivre la demande
- Cependant les applications ont de nouvelles exigences :
  - les applications multimédias qui associent la voix, les données et l'image (fixe ou animée).
- Evolution d'un réseau de conception simple (IP) reliant des terminaux intélligents (ordinateurs)
- Vers un réseau qui doit gérer en simultané les caractéristiques d'un réseau voix (téléphonie), un réseau données et un un réseau audiovisuel (vidéoconférence).
- Il est convenu de nommer les mécanismes qui vont assurer la mise en oeuvre de ces applications, en simultané, sur le même réseau : la **Qualité de Service** (ou QoS Quality of Service).

Introduction

- Application voix ou vidéo



- Application informatique

Définitions

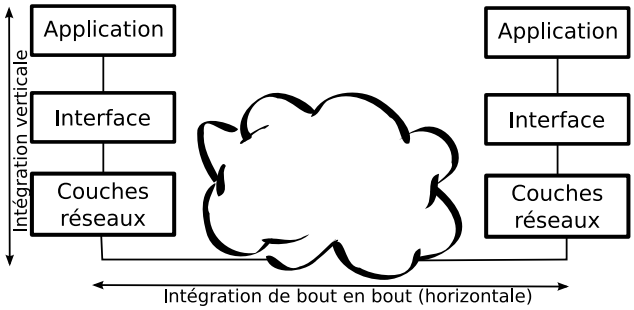
- La qualité de service d'un réseau désigne sa capacité à transporter dans de bonnes conditions les flux issus de différentes applications.
- Les applications peuvent générer des flux :
  - informatique (données, transaction, intéractif),
  - voix (stream audio),
  - images (vidéo : stream video)
- Les caractéristiques techniques :

fiabilité	le réseau doit être fiable et disponible (*reliability*)
bande passante	le réseau doit disposer d'une certaine bande passante (débit) pour absorber les trafics générés (*bandwidth*)
délai	le réseau doit permettre d'acheminer des données rapidement : on parle de **latence** (*delay*)
régularité	le réseau doit permettre d'acheminer des données de façon régulière : on parle de **gigue** (*jitter*)
taux d'erreurs	le réseau doit permettre d'acheminer des données sans perte (*loss ratio*)

Architecture de la QoS

- La mise en oeuvre d'une solution globale de QoS demande :
  - Des mécanismes de bout en bout (mécanismes horizontaux)
    - ⋆ Permet d'adapter le comportement des équipements traversés vis à vis du flux issu d'une application
    - ⋆ Analogie avec le service postal qui doit acheminer un pli express dans un ensemble de pli normaux (on signale sur le pli, le caractère urgent). Le non respect de la signalisation par l'un des centres de tri postal entraine une perte de la QoS.
  - Des mécanismes verticaux
    - ⋆ Permet aux applications de demander (via une interface apropriée) une QoS recherchée en utilisant les mécanismes de plus bas niveau (principe des couches ISO)
    - ⋆ Les mécanismes de QoS proposés par un routeur devront se référer aux mécanismes de QoS des liens de communication utilisés (ATM, Ethernet, ...)

Architecture de la QoS



Les besoins en QoS

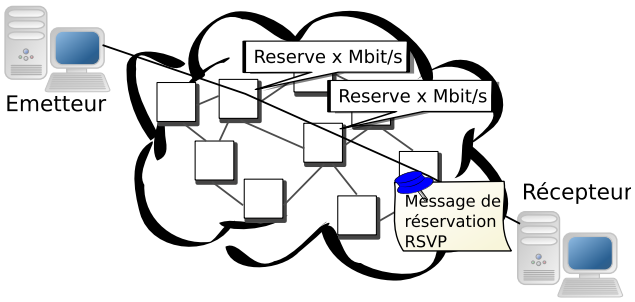
- Les applications informatiques dans la plus part des cas se contentent d'une priorité et d'une bande passante faible avec une bonne tolérance aux délais de transfert sur le réseau.
- Les applications interactives ont des exigences plus strictes (temps de réponse le plus court possible)
- Pour les applications qui utilisent la voix, le délai de traversée est un critère important
- Pour les applications vidéo, il faut gérer un mécanisme de multicast (un vers plusieurs) qui duplique le flux au plus près du destinataire.
- Les caractéristiques techniques de la QoS sont :
  1. Le besoin en bande passante, ils sont constants (mode stream) pour les applications audio/vidéo ou immédiat (mode burst) pour les applications qui utilisent toutes la bande passante ;
  2. Le besoin en délai de traversée du réseau, qui sont variables, aucun délai (transfert de fichiers, messagerie électronique,...) jusqu'aux applications avec fortes constraintes temporelle (isochrone) telles que la voix.

Les différentes solutions

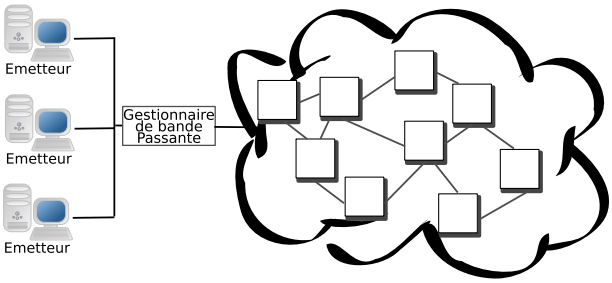
Trois approches sont identifiées pour faire face aux besoins de gestion de trafic

1. ne rien faire : gestion en best effort (solution par défaut utilisée aujourd'hui)
2. Mettre en place une gestion de trafic au sein du réseau. Il n'y a pas de politique unique, mais une en fonction des objectifs à atteindre.
   - permet de favoriser certains flux prioritaires par analyse préalable (demande implicite), ou à la demande (flux explicite).
   - Les flux moins prioritaires seront mis en file d'attente, ce qui nécessite pour le noeud du réseau la capacité de mettre en oeuvre : une classification, la mise en file d'attente et l'ordonnancement.
3. Mettre en place une gestion de trafic aux extrémités du réseau
   - Exclusivement liée au monde IP, on distingue principalement les gestionnaires de bande passante et les équipements de médiation application/réseau.

Architecture de la QoS



Architecture de la QoS



Niveaux de service de la QoS

La QoS à fournir est liée aux exigences des applications à mettre en oeuvre. On distingue trois niveaux principaux de service de QoS dans les réseaux :

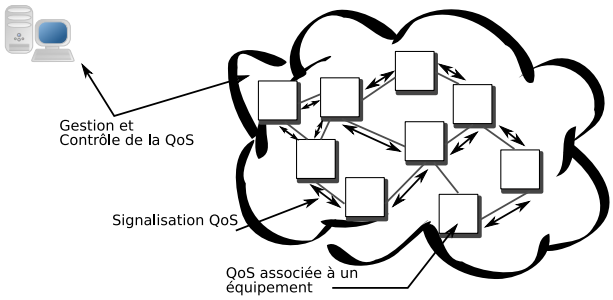
- **Les services aux mieux** ou *best effort* : en fait ne propose pas de QoS, mais un service primaire de connectivité entre deux points quelconques du réseau
- **Les services différenciés** ou *Differentiated Services* : proposent un traitement préférentiel de certains types de trafic. Il s'agit d'une préférence relative et non d'une garantie absolue.
- **Les services garantis** ou *Guaranteed Services* : proposent des caractéristiques de QoS garanties en utilisant des mécanismes de réservation de ressources sur le réseau.

Les composants de la QoS réseau

La Qos globale va nécessiter une architecture qui permette la coordination de chaque noeud du réseau.

- Mécanismes de QoS intégrés aux équipements du réseau : pour appliquer localement la politique de gestion de flux (file d'attente, . . .) et utilisation des caractéristiques de QoS des liens avec les autres équipements.
- Mécanismes de signalisation pour la QoS : permet de coordonner la QoS de bout en bout.
- Gestion et Contrôle de la QoS sur le réseau : administration centralisée (peut aussi être utilisée pour la facturation).

Les composants de la QoS réseau



Les composants de la QoS réseau

QoS associée à un équipement.

Permet de modifier le comportement de l'équipement en fonction des caractéristiques des paquets reçus. Cela nécessite pour chaque équipement du réseau :

- la classification : trier puis affectation dans une file d'attente
- le contrôle : vérifier que le paquet est valide
- la mise en file d'attente : partage de la bande passante
- l'ordonnancement : choix de la file d'attente en fonction de leur priorité

Les composants de la QoS réseau

Signalisation QoS.

La signalisation QoS permet d'appliquer de façon cohérente la QoS et cela de bout en bout du réseau sur chaque équipement traversé. La signalisation peut être de deux natures différentes :

- la signalisation *in band* : signalisation portée par l'information (dans une trame Ethernet on ajoute un champ qui exprime un niveau de priorité)
- la signalisation *out band* : la signalisation est transportée vers les équipements avant les données (lors de l'établissement d'un circuit virtuel dans le cas d'un réseau ATM)

La QoS doit s'appliquer de bout en bout et s'adapter aux liaisons hétérogènes traversées (Ethernet, ATM, . . .). En général la QoS est exprimée au niveau 3 (IP) et prise en charge au niveau 2 (liaison).

Les composants de la QoS réseau
Gestion et Contrôle de la QoS.

Il existe plusieurs niveaux de QoS (au mieux, différenciés, garantis).

- Comment affecter un service QoS à une application, un groupe d'utilisateurs, de façon statique ou dynamique ?
- Comment garantir le respect des services (utilisation d'un service garanti par une application non prioritaire) ?
- Facturation en fonction du niveau de QoS

Deux niveaux de gestion existent :

- Signalisation dynamique au niveau de l'utilisateur : l'application demande un service de QoS à l'aide d'un protocole de signalisation comme RSVP (*Resource Reservation Protocol*). L'opérateur devra avoir la possibilité de modifier dynamiquement la gestion des ressources du réseau.
- Signalisation en fonction du profil de l'utilisateur : Les caractéristiques du flux sont déterminées *a priori*.

Les protocoles de la QoS dans le monde IP

Les différents services de la QoS en Général

- au mieux
- différencié
- garanti

Les différents modèles de service de la QoS définis par l'IETF (*Internet Engineering Task Force*)

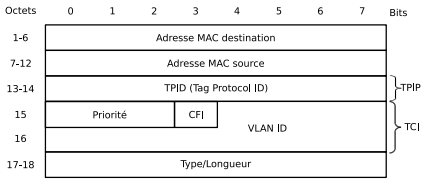
- le modèle *DiffServ* pour la QoS à services différenciés,
- le modèle *IntServ* pour la QoS à services garantis.

De plus l'IETF a défini des techniques pour gérer au mieux l'ensemble des lignes de communication. Ce sont des techniques d'ingénerie des trafics fondées sur MPLS (*Multi Protocol Label Switching*).

Plan

Les protocoles de la QoS avec Ethernet

- Utilise le protocole 802.1Q qui permet la mise oeuvre des VLANs (*Virtual LAN*)
- L'identification des VLANs permet d'affecter une QoS à chacun d'eux
- Les deux octets du champ TCI (Tag Control Information) contiennent :
  - 3 bits de priorité
  - 1 bit CFI (Canonical Format Indicator) pour le format d'adresse MAC
  - 12 bits pour le VLAN ID

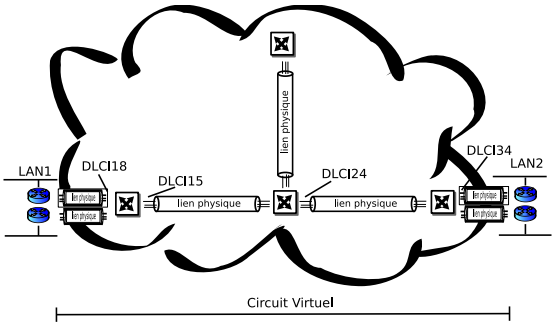


Les protocoles de la QoS dans un réseau Frame Relay

Rappels

- Simplification de la technologie de commutation de paquets *X*25
  - multiplexage statistique (optimisation de la bande passante)
  - réduction des contrôles d'erreurs
- Une connexion est établie entre deux points et forme un circuit virtuel
- Le chemin est enregistré dans les commutateurs à l'aide d'un identifiant **DLCI** (*Data Link Connection Identifier*)
- Un circuit virtuel est une succession de DLCI
- Les connexions sont point à point ou point à multipoint (il suffit d'ajouter un circuit virtuel permanent (CVP))
- Une connexion au réseau à relais de trames est définie par :
  - le débit du lien d'accés au réseau : **AR** (*Access Rate*)
  - une disponibilité du service
  - pour chaque circuit virtuel on indique :
    - le débit de transfert : **CIR** (*Committed Information Rate*)
    - Une possibilité de trafic supplémentaire **EIR** (*Excess Information Rate*)
    - Un délai d'acheminement (100*ms* en France, 300*ms* France-Amérique du sud)

Les composants de la QoS réseau



Les protocoles de la QoS dans un réseau Frame Relay

Relais de trame et QoS

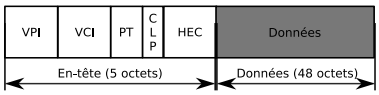
- Le contrôle de flux
  - Le contrôle à l'admission : négociation d'une qualité de service entre le réseau et l'utilisateur
  - Gestion dynamique de la fenêtre d'anticipation (prévenir la congestion)
  - Elimination sélective des trames (vérification de la conformité du trafic de l'utilisateur)
  - Mise en oeuvre de l'algorithme **EBF** (*Explicit Binary Feedback*) qui marque la trame si celle-ci traverse un commutateur congestionné
- QoS
  - Délai d'acheminement faible
  - Garantie d'acheminement élevée
- De nombreux opérateurs proposent le support de la voix sur leurs réseaux à relais de trames

Les protocoles de la QoS et ATM

- Débits supérieurs à 155Mbit/s
- Conçue pour le support natif de la QoS
- Doit assurer la convergence des réseaux existants (informatique, voix et vidéo)
- La technologie ATM est souvent appelée **B-ISDN** (*Broadband IDSN "Integrated Services Digital Network"*)
- Utilise une technologie de commutation hybride (commutation de circuit et de paquet) orientée connexion qui permet dans la phase de négociation le paramétrage de trafic et de qualité de service.

Les protocoles de la QoS et ATM

- Les informations sont découpées en cellules de longueur fixe : 53 octets et le flux des cellules emises n'est pas constant (*Asynchronous Transfert Mode : ATM*)
- L'en-tête comporte une double identification **VPI** (*Virtual Path Identifier*) et **VCI** (*Virtual Circuit Identifier*), un VPI pouvant contenir plusieurs VCI.
- Les commutateurs ATM ont la capacité de faire de commutation de circuit et de la commutation de chemin.
- Un circuit virtuel ATM est constitué d'une succession de couples VPI/VCI (les DLCI du réseau à relais de trames)



Les protocoles de la QoS et ATM

Technologie orientée connexion, à chaque phase d'appel

- le réseau vérifie qu'il peut accepter l'appel avec la qualité de service demandée
- le réseau détermine la route (ensemble de commutateurs traversés)
- le réseau crée le circuit virtuel (suite de couples VPI/VCI affectés aux commutateurs traversés)

Les circuits virtuels commutés **SVC** (*Switched Virtual Circuit*) sont établis grâce à une procédure de signalisation au travers du réseau

- signalisation aux extrémités via une interface utilisateur **UNI** (*User to Network Interface*)
- signalisation à l'intérieur du réseau **NNI** (*Network to Network Interface*)

Les protocoles de la QoS et ATM

Technologie orientée connexion (suite)

- La signalisation suit le protocole *Q*.2931. l'appel se fait à l'aide de messages **SETUP** qui contiennent les paramètres **IE** (*Informations Elements*) :
  - Destinataire : *Called Party Number*, adresse au format *E*.164
  - Source : *Calling Party Number*
  - Bande passante désirée : *ATM User Cell Rate* en nombre de cellules par seconde
  - Classe de service : *Quality of Service*
  - type d'AAL (*ATM Adaptation Layer*) : *AAL type* adaptation au réseau environnant (encapsulation, segmentation et ré-assemblage)
- Un protocole de routage propre au réseau ATM : **PNNI** est utilisé pour établir le circuit virtuel
- Les VCI sont unidirectionnels alors que les VPI sont bidirectionnels

Les protocoles de la QoS et ATM

- A chaque appel d'un utilisateur (application), le réseau vérifie si il peut l'accepter sans perturber le trafic existant
- Les caractéristiques du trafic font l'objet d'un contrat entre l'utilisateur (application) et le réseau
- Le réseau met en oeuvre une surveillance du trafic émis par l'utilisateur, si il y a une différence (trafic plus important) :
  1. le réseau marque les cellules (bit CLP dans l'en-tête) émises en trop comme pouvant être détruites (en cas de problème)
  2. ces cellules sont détruites en priorité en cas de congestion
  3. si le réseau n'est pas congestionné, les cellules sont acheminées jusqu'à un seuil maximal de dépassement, au-delà elles seront détruites

Les protocoles de la QoS et ATM

La QoS est supportée en natif par la technologie ATM. Elle s'exprime sous la forme de paramètres qui ont été regroupés pour définir des classes de service.

- Les paramètres de trafic
  - **PCR** (*Peak Cell Rate*) : débit maximal souhaité,
  - **SCR** (*Sustainable Cell Rate*) : débit moyen soutenu,
  - **MBS** (*Maximum Burst Size*) : taille maximale d'un pic de trafic,
  - **MCR** (*Minimum Cell Rate*) : débit minimal souhaité
- Les paramètres de qualité de service
  - **CTD** (*Cell Transfert Delay*) : délai de traversée du réseau,
  - **CDV** (*Cell Delay Variation*) : variation du délai ou gigue,
  - **CDVT** (*Cell Delay Variation Tolerance*) : tolérance de gigue,
  - **CLR** (*Cell Loss Ratio*) : taux de perte de cellule
- Contrôle de flux

Les protocoles de la QoS et ATM

Les classes de service permettent de regrouper les paramètres

CBR (*Constant Bit Rate*) Service pour les connexions avec une bande passante fixe (valeur de PCR). Pour les applications temps réel sensibles au délai d'acheminement(valeur de CTD) et à la gigue (valeur de CDV) comme le transport de la voix, de la vidéo, ...

VBR-RT (*Variable Bit Rate*) Service pour les connexions sensibles aux contraintes temporelles (CTD et CDV) pour la voix et la vidéo compressée. Les paramètres fixés sont : PCR,SCR, MBS ;

VBR-NRT Service pour les connexions sans contraintes temporelles ayant un trafic irrégulier. Les paramètres fixés sont : PCR,SCR, MBS. l'application fixe le CLR et le CTD ;

ABR (*Available Bit Rate*) Service pour les connexions faisant varier leur trafic. Les paramètres fixés sont : PCR, MCR. Le contrôle de flux est défini pour contrôler le débit de l'utilisateur

GFR (*Guaranteed Frame Rate*) Service pour les connexions utilisant l'AAL5 (transport de trames type IP), les cellules sont détruites par trame en cas de congestion

UBR (*Unspecified Bite Rate*) Service pour les applications non critiques

Les protocoles de la QoS et ATM

Pour assurer les classes de service ATM met en oeuvre des fonctions de contrôle de trafic.

CAC (*Connection Admission Control*) les actions qui sont effectuées pour vérifier si un appel peut être accepté ou rejeté. Si il est accepté, les ressources du réseau sont réservées en conséquence et en fonction de la classe de service demandée.

UPC (*Usage Parameter Control*) les actions qui sont effectuées pour vérifier le trafic émis par l'utilisateur pour éviter tout surcroît de trafic non planifié

Feedback Control (Contrôle de congestion) les actions qui sont effectuées pour réguler le trafic ABR.

- mécanisme **EFCI** (*Explicit Forward Congestion Indication*) : le commutateur congestionné valide le bit AFCI dans l'en-tête des cellules. L'état du bit sera vu par les commutateurs voisins ;
- mécanisme *Relative Rate* : le commutateur congestionné emet des cellules spéciales **RM** (*Ressource Management*) vers la source du trafic ;
- mécanisme *Explicit Rate* : la source ne peut augmenter son trafic sans accord du réseau

Plan

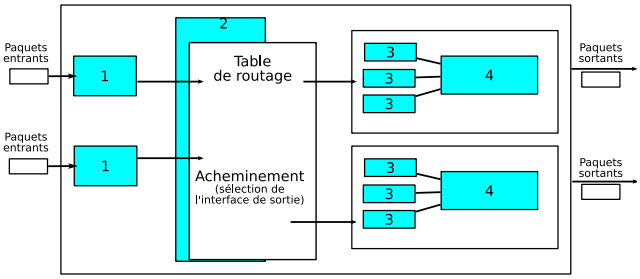
Les protocoles de la QoS liés à IP

- IntServ : Modèle de QoS à service garanti
  - volonté en 1994 de l'IETF de prendre en charge la QoS temps réel et le contrôle du partage de la bande passante sur les liens du réseau.
  - demande des informations supplémentaires dans les routeurs
  - un mécanisme spécifique d'initialisation est nécessaire pour appeler le service (identification du flux)
  - utilise un protocole de signalisation RSVP
- DiffServ : Modèle de QoS à service différencié
  - Evolution du modèle IntServ pour les plus grands réseaux
- MPLS : modèle d'optimisation de trafic
  - volonté de l'IETF d'améliorer les débits en optimisant le traitement des paquets IP dans le réseau
  - utilise le concept de commutation multiniveau (associer les avantages du routage et ceux de la commutation)

Les Routeurs QoS

- Un routeur prenant en charge la QoS doit mettre en oeuvre les quatre fonctions suivantes :
  1. Le classificateur : établit les caractérisques , recherche d'une entrée dans la table de routage, déterminer l'interface de sortie et sa file d'attente
  2. Le contrôle et le marquage : détermine si le flux entrant est conforme au trafic prévu sinon il est détruit ou marqué (susceptible d'être détruit)
  3. La gestion des files d'attente : contrôle de congestion
  4. L'ordonnanceur : achemine les paquets en fonction de la classification et du contrôle de trafic en sortie

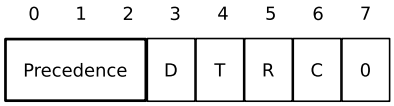
Les Routeurs QoS



Les Routeurs QoS

La classification

- classification des paquets dans un but de QoS en fonction :
  - d'informations contenues dans le paquet, ou
  - déduites par le routeur
- Classification à partir du **TOS** (*Type Of Service*) ou **DSCP** (*Differentiated Service Code-Point*) dans le modèle DiffServ



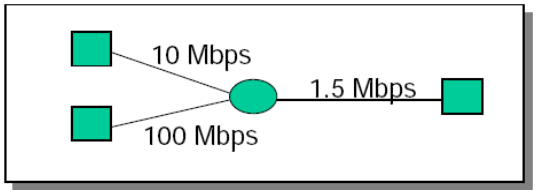
- Classification multichamp : adresse IP source, destination, TOS, port source, destination dans les en-têtes TCP, UDP
  - peut entrainer un phénomène de gigue
  - problème pour les paquets cryptés
  - informations pas toujours fiables (peuvent être modifiées par l'utilisateur)

Les Routeurs QoS

Le Contrôle, marquage et lissage de trafic

- Détermine si le flux reçu est conforme à un profil de trafic convenu à l'avance
- Le marquage permet de traiter le trafic non conforme
- Le lissage a pour but d'éliminer les crêtes de trafic pour les transmettre ultérieurement (induit un délai supplémentaire) et permet la mise en oeuvre d'une politique de trafic : la spécification du taux avec lequel un flux est autorisé à envoyer des paquets dans le réseau. S'exprime avec les critères :
  - débit moyen
  - débit crête
  - taille maxi de la rafale (nombre de paquets qui peuvent être transmis dans un intervalle de temps très court)

Les Routeurs QoS : Le contrôle de congestion



- Une congestion peut survenir lorsque trop de paquets sont injectés dans le réseau et prennent des routes similaires. Il y a alors augmentation du temps d'attente et risque de perdre des paquets.
- Une congestion peut aussi survenir du fait de la différence de puissance de traitement d'un routeur à l'autre.

Les Routeurs QoS : Le contrôle de congestion

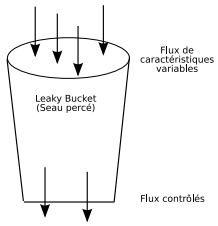
Les principes

- Réactif
  - lorsque la congestion est détectée, informer les noeuds en amont et en aval,
  - puis, marquer des paquets, rejeter des paquets, traiter les paquets prioritaires.
- Préventif
  - diffusion périodique d'informations d'états (taille des buffers)
  - contrôle continue de la source (*Leacky Bucket*, *Token Bucket*. . .),
  - contrôle de flux, contrôle d'admission.
- De bout en bout
  - pas de retour du réseau
  - la congestion est estimée grâce à l'observation des pertes et des délais de bout-en-bout
- Assisté par le réseau
  - bit d'annonce de congestion (SNA, DECbit, TCP/ECN, FR, ATM)

Les Routeurs QoS : Le contrôle de congestion

Le mécanisme du "seau qui fuit" (*Leaky Bucket*)

- Quelle que soit la vitesse de remplissage du seau, la vitesse découlement par le trou reste constant ρ s'il y a de l'eau et zéro sinon.
- Lorsque le seau est plein, ajouter de l'eau provoque un débordement et la perte du liquide.
- Ecoulement constant quelles que soient la fréquence et la longueur des rafales
  - Un ordinateur produit des données à 25Mo/s
  - Un routeur ne peut pas accepter ce débit sur de longue période (plutôt 2Mo/s)
  - Les données arrivent par rafales d'une durée de 40ms toutes les secondes
  - On peut utiliser un seau percé d'un débit constant $\rho = 2Mo/s$ et une

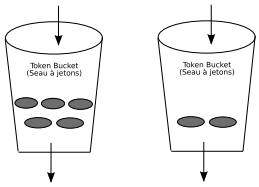


Les Routeurs QoS : Le contrôle de congestion

Le mécanisme du "seau à jetons" (*Token Bucket*)

- Permet de faire varier le débit en fonction des rafales qui arrivent
- Le seau produit un jeton toutes les ΔT secondes
- Pour être transmis, un paquet capture et détruit un jeton
- Si les jetons ne sont pas utilisés, ils peuvent être stockés jusqu'à une taille maxi du seau n. Cela permet d'émettre en une fois jusqu'à n paquets en rafale sur le réseau

- Pour une rafale de $1Mo$, on obtient en sortie un débit à $25Mo/s$ pendant $11ms$ puis le reste des données à $2Mo/s$.
- La durée de la rafale(S) à plein débit(M) est fonction de la capacité C du seau et du débit en sortie(ρ) ($C + \rho S = MS$), donc $S = \frac{C}{M-\rho}$



Les Routeurs QoS : Le contrôle de congestion

Les autres mécanismes

- Pour prévenir la congestion, il faut agir sur la source responsable de l'engorgement
  - **RED** (*Random Early Detection*) : lorsque dans une file d'attente on atteint un certain seuil (seuil_min), les paquets placés au-delà du seuil peuvent être détruits de manière aléatoire. Plus le niveau monte et plus les paquets ont une probabilité d'être détruits.
    - ⋆ fonctionne bien avec le protocole TCP. L'émetteur va limiter son trafic (réduction de la fenêtre)
  - **WRED** (*Weighted Random Early Detection*) : même principe avec choix en fonction d'un type de trafic
  - **RIO** (*Random Early Detection with In/Out*) :même principe avec un seuil plus bas pour les paquets qui ne respectent pas le contrat (Out Profile) et un seuil plus haut pour ceux qui le respectent (In Profile)
  - **ARED** (*Adapted Random Early Detection*) : élimination plus forte lorsque le nombre de flux TCP augmente
  - **FRED** (*Flow Random Early Detection*) : élimination des paquets en fonction de la réaction de l'application (réaction plus au moins rapide)

Les Routeurs QoS : L'ordonnancement

- L'ordonnancement consiste à vider les files d'attente vers l'interface de sortie du routeur
  - **FIFO** (*First In First Out*) : Stocke les paquets quand le réseau est saturé, puis vide la file dans l'ordre d'arrivée
  - ***Priority Queuing*** : La file d'attente la plus prioritaire sera vidée plus rapidement que les autres
  - **CBQ** (*Class-Based Queuing*) : Permet à plusieurs applications de partager l'utilisation du réseau avec des spécifications minimales (bande passante, délai, . . .)
  - **WFQ** (*Weighted Fair Queuing*) : Le vidage des files d'attente est effectué par une pondération (weighted) équitable (fair) des flux. On réalise un entrelacement des flux.

Le modèle IntServ : Définitions

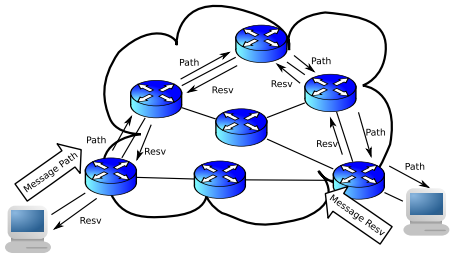
- Les éléments qui constituent le réseau **IntServ** (*Integrated Service*) sont des **NE** (*Network Elements*) qui peuvent être de type :
  - QoS-capable : élément qui offre un ou plusieurs service IntServ
  - QoS-aware : élément qui offre l'interface sans les services IntServ (offre des services comparables)
  - non-QoS : élément qui n'offre pas IntServ
- Le modèle IntServ définit deux types de services :
  1. le service garanti (*GS Guaranteed Service*) : émule un circuit virtuel dédié, avec bande passante garantie et délai d'acheminement limité
  2. la charge controlée (*CL Controlled Load*) : service *best-effort* dans un contexte non surchargé
- Un routeur prenant en charge les services IntServ doit mettre en oeuvre les fonctions d'un routeur Qos

Le protocole RSVP

- Le protocole **RSVP** (*ReSerVation Protocol*) est le protocole de signalisation et le contrôle de reservation pour IntServ (ou autres)
- RSVP est décrit dans les RFC :
  - RFC2205 Protocol Specification,
  - RFC2208 Applicability Statement,
  - RFC2209 Message Processing
- RSVP n'est pas un protocole de routage mais travaille en collaboration avec eux
- Permet la réservation dynamique de ressources, ce qui demande
  - l'application doit être capable d'exprimer ces besoins
  - les systèmes (serveurs, stations, périphériques) doivent prendre en compte ces besoins et permettre des les exprimer (API comme Winsock 2)
  - le protocole doit réserver les ressources sur le réseau
  - les commutateurs et les routeurs de réseau doivent comprendre les requêtes de réservation et assurer les contrats de QoS

Le protocole RSVP

- Le protocole RSVP établit une connexion logique appelée **session** qui est identifiée par trois éléments :
  - adresse_destination : adresse IP destination (unicast ou multicast)
  - identifiant_protocole : identifiant du protocole sur IP
  - port_destination : port TCP ou UDP



Le protocole RSVP : Principe de fonctionnement

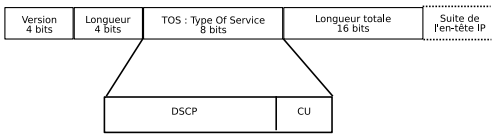
- l'emetteur spécifie le trafic en terme de bande passante (min, max), délai et gigue dans un descripteur de trafic **TSPEC** qui est placé dans un message **RSVP-PATH** avec une spécification de trafic **ADSPEC** qui sera modifiée sur le chemin. L'ensemble est envoyé vers le destinataire.
- Le message est routé vers le destinataire à l'aide d'un algorithme de routage (RIP, OSPF, . . .)
- Chaque routeur traversé stocke les informations relatives au chemin constitué ("**PATH-STATE**"). Cela permettra le retour du message RSVP vers l'émetteur. De plus le routeur peut modifier la spécification **ADSPEC** pour l'adapter aux restrictions courantes (bande passante réellement disponible)
- le recepteur utilise les spécifications **TSPEC** et **ADSPEC** pour déterminer les paramètres de retour. La réservation effective se fait par l'émission d'un message **RSVP-RESV** qui contient une spécification de la requête avec le type de service désiré(CL, GS) et une caractérisation des paquets pour cette reservation (**filter_spec**). La combinaison RSPEC et filter_spec forme une description du flux (*flow descriptor*)

Le protocole RSVP : Principe de fonctionnement (suite)

- Le message **RSVP-RESV** revient vers l'émetteur à l'aide du **PATH**
- Chaque routeur qui reçoit un message **RESV** le contrôle (authentification) et effectue la réservation et l'allocation de ressources. Si la requête n'est pas satisfaite (pas assez de ressource) une erreur est retournée au recepteur, sinon le routeur enregistre le flux et envoie un message **RESV** au routeur suivant.
- Le dernier routeur envoie un message de confirmation à l'émetteur
- Si l'émetteur ou le récepteur ferme la session RSVP, les réservations sont perdues
- le protocole RSVP maintient les réservations périodiquement de façon à rendre les allocations en cas de panne d'un routeur ou d'un lien sur le chemin
- le protocole RSVP est unidirectionnel
- Les informations de réservation transportées par le protocole RSVP sont indépendantes du protocole lui-même (RSVP peut être utilisé en dehors de IntServ)

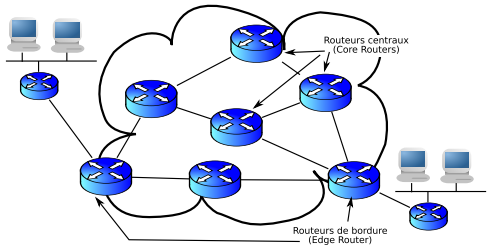
Le modèle DiffServ

- Permet de fédérer les flux de plusieurs applications en un seul
- Permet d'augmenter la taille du réseau
- Introduit la notion de **classe** de trafic
- Introduit la notion de priorité relative entre paquets en fonction du **PHB** (*Per Hop Behavior*) défini sur le routeur
- Le champ **TOS** (*Type Of Service*) est utilisé dans un contexte DiffServ et est appelé **DS**
  - le champ **DSCP** (*Differentiated Services CodePoint*) permet de selectionner le PHB à appliquer au paquet
  - le champ **CU** (*Currently Unused*)



Le modèle DiffServ

- Les opérations complexes (classification, contrôle et marquage de l'en-tête des paquets) se font à l'entrée du réseau : sur les noeuds de bordure (*boundary nodes*)
- Les noeuds intérieurs (*interior nodes*) traitent les paquets en fonction de la classe codée dans l'en-tête au travers du **DSCP** et du **PHB**



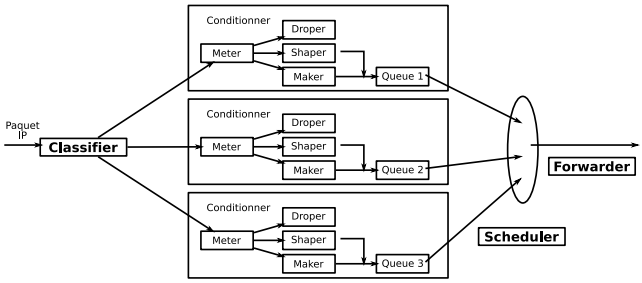
Le modèle DiffServ : Notion de domaine et de région

- Un domaine DiffServ (*DS Domain*) est une zone administrative avec un ensemble commun de politiques d'approvisionnement du réseau et de définitions de PHB.
- Une région DiffServ (*DS Region*) est un ensemble contigu de domaines DiffServ qui peuvent offrir des politiques d'approvisionnement et de PHB différents.
- Les Noeuds frontières (*DS boundary nodes*) sont les équipements de bordure du domaine
- Les noeuds intérieurs (*DiffServ Interior Nodes*) qui appliquent un comportement approprié (PHB) aux paquets IP en fonction de leur valeur de DSCP.

Le modèle DiffServ : Notion de domaine

- Dans un domaine on distingue :
  - les équipements frontières (*boundary nodes*)
  - les équipements internes (*interior nodes*)
- Les équipements frontières mettent en oeuvre 4 mécanismes
  1. Une classification des trafics (*traffic classifier*) qui permet de sélectionner des paquets dans un flot en fonction d'informations dans l'en-tête. Il y a deux classifications possibles :
     - ⋆ **BA** (*Behavior Aggregate*) établie en fonction du champ DS (DSCP)
     - ⋆ **MF** (*Multi-Field*) établie à partir de plusieurs champs de l'en-tête (adresse source ou destination, champ DS, identificateur du protocole, le numéro de port source, destination)
  2. un conditionnement des trafics (traffic conditionning) assuré par 4 composants : le metreur (*Meter*), le marqueur (*Marker*), le lisseur (*Shaper*) et le suppresseur (*Dropper*)
  3. un ordonnancement (scheduling)
  4. un mécanisme d'acheminement (forwarding)

Le modèle DiffServ



Le modèle DiffServ : Le PHB

- le PHB est la description externe du comportement d'un routeur face à un trafic particulier. Par exemple : garantir une bande passante minimum de 20% d'un lien
- Les PHB sont mis en oeuvre à l'aide de gestion de files d'attente et de régulation de flux
- Un PHB sera appliqué en fonction de la valeur du champ DS (DSCP) d'un paquet
- Les PHB ayant un comportement proches sont regroupés (*PHB Groups*)
- Pour des PHB standardisés on associe une valeur de DSCP recommandée
  - Un ensemble de 32 **RECOMMENDED** Codepoints(Pool 1) (sur les 64 possibles du champ DSCP) est assigné par l'**IANA** (*Internet Assigned Number Authority*)
  - Un ensemble de 16 Codepoints(Pool 2) est reservé à une utilisation locale
  - Un ensemble de 16 Codepoints(Pool 3) Locale et IANA

Espace	Valeur DSCP	Utilisation
1	xxxxx0	IANA
2	xxxx11	Locale
3	xxxx01	Locale/IANA

Le modèle DiffServ : Le PHB

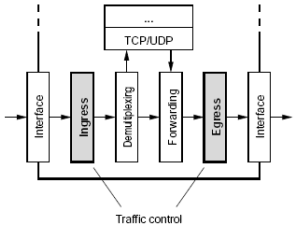
- Le comportement *best effort* est défini comme le PHB par défaut. La valeur de DS (DSCP) correspondante est 000000
- Le PHB *Expedite Forwarding* (EF) encore appelé *Premium Service* fournit un service assimilé à une ligne louée virtuelle avec une garantie de bande passante, taux de perte, délai et gigue faible
  - Le DSCP correspondant au service EF est 101110
- Le PHB *Assured Forwarding* (AF) fournit un service garantissant un acheminement des paquets IP avec une haute probabilité. Il regoupe plusieurs PHB.
  - Définit N classes indépendantes de M niveaux de priorités (*Drop precedence*) différentes
  - Actuellement $N = 4$ et pour chaque classe $M = 3$

Drop precedence	Classe 1	Classe 2	Classe 3	Classe 4
Low	$AF11 = 001010$	$AF21 = 010010$	$AF31 = 011010$	$AF41 = 100010$
Medium	$AF12 = 001100$	$AF22 = 010100$	$AF32 = 011100$	$AF42 = 100100$
High	$AF13 = 001110$	$AF23 = 010110$	$AF33 = 011110$	$AF43 = 100110$

Plan

Principe de base

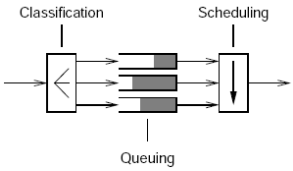
- Le noyau Linux propose des fonctions pour la classification et l'ordonnancement du trafic réseau



- Ingress : permet de faire une destruction des paquets indésirables et une classification préliminaire
- Egress : permet de la faire la classification, la mise en file d'attente et l'ordonnancement

Principe de base

- La classification analyse le contenu du paquet et l'identifie à une classe
- En fonction de la classe du paquet, il est placé dans une file d'attente
- Avant d'être éventuellement choisi pour être transmis sur l'interface de sortie

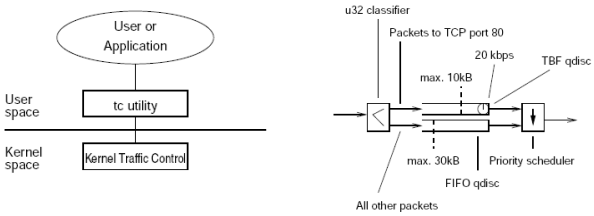


Principe de base

- La gestion de la file d'attente est associée au mécanisme d'ordonnancement pour donner les "*queuing desciplines*"
- Les plus utilisées sont :
  - ***drop-tail FIFO*** : supprime tous les nouveaux paquets au-delà d'une certaine limite
  - **RED FIFO**
  - **TBF** (*Token Bucket Filter*) : lisse le trafic en émettant des paquets à une certaine fréquence
  - **PRIO** : file d'attente avec priorité
  - **HTB** (*Hierarchical Token Bucket*) : distribue la bande passante entre les différentes classes de paquets
- La classification utilise un filtre "*u*32" qui peut analyser toutes les données du paquet
- la classification peut aussi prendre en compte les métriques du réseau (fréquence d'arrivée d'une classe de paquet)

Principe de base

- L'accès à la gestion du trafic sous linux se fait par l'intermédiaire de la commande "`tc`".




```
tc qdisc add dev eth0 root handle 1: prio
tc qdisc add dev eth0 parent 1:1 handle 20: tbf \
   rate 20kbit burst 2kb limit 10kb mtu 1500
tc qdisc change dev eth0 parent 1:2 bfifo limit 30kB
tc filter add dev eth0 parent 1:0  protocol ip u32 \
   match ip protocol 6 0xff match tcp dst 50 0xffff classid 1:1
```


Mise au point sur les unités dans `tc`

- Afin d'éviter toute confusion, voici les règles utilisées par tc pour la spécification de la bande passante :

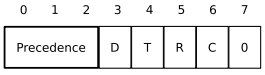
```
mbps = 1024 kbps = 1024 * 1024 bps =>
                                  byte/s (octets/s)
mbit = 1024 kbit => kilo bit/s.
mb = 1024 kb = 1024 * 1024 b => byte (octet)
mbit = 1024 kbit => kilo bit.
```

- En interne, les nombres sont stockés en bps (octet/s) et b (octet). Mais `tc` utilise l'unité suivante lors de l'affichage des débits :

```
1Mbit = 1024 Kbit = 1024 * 1024 bps => octets/s
```


La commande `tc` : la file d'attente `pfifo_fast`

- C'est la file d'attente par défaut (premier rentré, premier sortie)
- Cette file à trois "bandes", pour chacune des bandes les règles FIFO s'appliquent.
- Tant qu'il y a des paquets dans la bande 0 (la plus prioritaire), la bande 1 (et encore moins la bande 2) ne sera pas traitée.
- Le champ TOS est utilisé pour placer les paquets dans les bandes. (les paquets avec le bit "délai minimal" activé sont placés dans la bande 0)



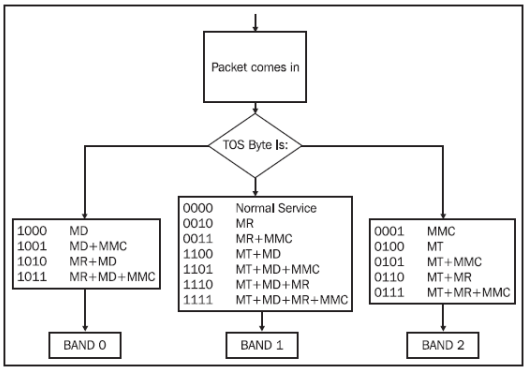
Binaire	Décimal	Signification
1000	8	Minimise le Délai (Minimize delay) (md)
0100	4	Maximalise le Débit (Maximize throughput) (mt)
0010	2	Maximalise la Fiabilité (Maximize reliability) (mr)
0001	1	Minimalise le Coût Monétaire (Minimize monetary cost) (mmc)
0000	0	Service Normal

La commande `tc` : la file d'attente `pfifo_fast`

- On place le paquet dans une bande (0,1 ou 2) en fonction de la valeur du TOS.

TOS	Bits	Signification	Priorité Linux	Bande
0x0	0	Service Normal	0 Best Effort	1
0x2	1	Minimise le Coût Monétaire (mmc)	1 Filler	2
0x4	2	Maximalise la Fiabilité (mr)	0 Best Effort	1
0x6	3	mmc+mr	0 Best Effort	1
0x8	4	Maximalise le Débit (mt)	2 Masse	2
0xa	5	mmc+mt	2 Masse	2
0xc	6	mr+mt	2 Masse	2
0xe	7	mmc+mr+mt	2 Masse	2
0x10	8	Minimise le Délai (md)	6 Interactive	0
0x12	9	mmc+md	6 Interactive	0
0x14	10	mr+md	6 Interactive	0
0x16	11	mmc+mr+md	6 Interactive	0
0x18	12	mt+md	4 Int. Masse	1
0x1a	13	mmc+mt+md	4 Int. Masse	1
0x1c	14	mr+mt+md	4 Int. Masse	1
0x1e	15	mmc+mr+mt+md	4 Int. Masse	1

La commande `tc` : la file d'attente `pfifo_fast`



Le TOS en fonction de l'application

Application	Contexte	TOS	Explications
TELNET		1000	(minimise le délai)
FTP			
	Contrôle	1000	(minimise le délai)
	Données	0100	(maximalise le débit)
TFTP		1000	(minimise le délai)
SMTP			
	phase de commande	1000	(minimise le délai)
	phase DATA	0100	(maximalise le débit)
Domain Name Service			
	requête UDP	1000	(minimise le délai)
	requête TCP	0000	
	Transfert de Zone	0100	(maximalise le débit)
NNTP		0001	(minimise le coût monétaire)

La commande tc : Filtre à seau de jetons (*Token Bucket Filter*)

Rappel du mode de fonctionnement :

- Les données arrivent dans TBF avec un débit **EGAL** au débit des jetons entrants. Dans ce cas, chaque paquet entrant a son jeton correspondant et passe la file d'attente sans délai.
- Les données arrivent dans TBF avec un débit **PLUS PETIT** que le débit des jetons. Seule une partie des jetons est supprimée au moment où les paquets de données sortent de la file d'attente, de sorte que les jetons s'accumulent jusqu'à atteindre la taille du tampon. Les jetons libres peuvent être utilisés pour envoyer des données avec un débit supérieur au débit des jetons standard, si de courtes rafales de données arrivent.
- Les données arrivent dans TBF avec un débit **PLUS GRAND** que le débit des jetons. Ceci signifie que le seau sera bientôt dépourvu de jetons, ce qui provoque l'arrêt de TBF pendant un moment. Ceci s'appelle "une situation de dépassement de limite" (*overlimit situation*). Si les paquets continuent à arriver, ils commenceront à être éliminés.

La commande tc : Filtre à seau de jetons (*Token Bucket Filter*)

- Il ne fait que laisser passer les paquets entrants avec un débit n'excédant pas une limite fixée administrativement.
- Les paramètres
  - limit le nombre d'octets qui peuvent être mis en file d'attente en attendant la disponibilité de jetons. *latency* exprime un temps maxi.
  - burst Taille du seau, en octets. C'est la quantité maximale, en octets, de jetons dont on disposera simultanément.
  - mpu L'Unité Minimale de Paquet (Minimun Packet Unit) détermine le nombre minimal de jetons à utiliser pour un paquet (même si celui-ci est de taille nulle).
  - rate Le paramètre de la vitesse.
  - peakrate Le débit de crête (*peak rate*) peut être utilisé pour spécifier la vitesse à laquelle le seau est autorisé à se vider.
- Un exemple de configuration

```
tc qdisc add dev ppp0 root tbf rate 220kbit latency 50ms burst 1540
```


La commande tc : File d'attente stochastiquement équitable (*Stochastic Fairness Queueing : SFQ*)

- Le trafic est divisé en un grand nombre de files d'attente FIFO : une par conversation (une session TCP ou un flux UDP). Le trafic est alors envoyé dans un tourniquet, donnant une chance à chaque session d'envoyer leurs données tour à tour.
- En fait il y a un nombre limité de files d'attente. Un algorithme de hachage répartit les sessions dans les différentes files d'attente.
- Les paramètres
  - perturb Reconfigure le hachage une fois toutes les "*pertub*" secondes.
  - quantum Nombre d'octets qu'un flux est autorisé à retirer de la file d'attente avant que la prochaine file d'attente ne prenne son tour. Par défaut, égal à la taille maximum d'un paquet (MTU).
- Un exemple de configuration

```
tc qdisc add dev eth0 root sfq perturb 10
```


La commande tc : Terminologie

- Gestionnaire de mise en file d'attente (*qdisc*) (*Queueing Discipline*) : Un algorithme qui gère la file d'attente d'un périphérique, soit pour les données entrantes (*ingress*), soit pour les données sortantes (*egress*).
- Gestionnaire de mise en file d'attente basé sur des classes (*Classful qdisc*) : Un gestionnaire de mise en file d'attente basé sur des classes contient de multiples classes. Certaines de ces classes contiennent un gestionnaire de mise en file d'attente supplémentaire, qui peut encore être basé sur des classes, mais ce n'est pas obligatoire.
- Classes : Un gestionnaire de mise en file d'attente basé sur les classes peut avoir beaucoup de classes, chacune d'elles étant internes au gestionnaire. Une classe peut à son tour se voir ajouter plusieurs classes. Une classe peut donc avoir comme parent soit un gestionnaire de mise en file d'attente, soit une autre classe.
- Classificateur (*Classifier*) : Chaque gestionnaire de mise en file d'attente basé sur des classes a besoin de déterminer vers quelles classes il doit envoyer un paquet. Ceci est réalisé en utilisant le classificateur.

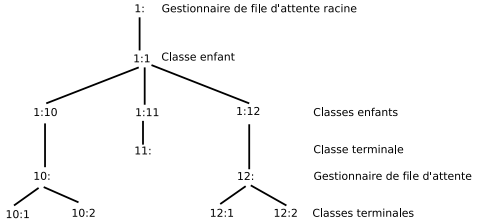
La commande tc : Terminologie

- Filtre (*Filter*) : La classification peut être réalisée en utilisant des filtres. Un filtre est composé d'un ensemble de conditions qui, si elles sont toutes vérifiées, envoie le paquet vers une classe.
- Ordonnancement (*Scheduling*) Un gestionnaire de mise en file d'attente peut, avec l'aide d'un classificateur, décider que des paquets doivent sortir plus tôt que d'autres. Ce processus est appelé ordonnancement (*scheduling*), et est réalisé par exemple par le gestionnaire pfifo_fast mentionné plus tôt.
- Mise en forme (*Shaping*) : Le processus qui consiste à retarder l'émission des paquets sortants pour avoir un trafic conforme à un débit maximum configuré. La mise en forme est réalisée sur *egress*. Familièrement, rejeter des paquets pour ralentir le trafic est également souvent appelé Mise en forme.

La commande tc : Terminologie

- Chaque interface à un gestionnaire de mise en file d'attente **racine** de sortie (*egress root qdisc*), pfifo_fast par défaut.
- Chaque gestionnaire et classe est repéré par un **descripteur** (*handle*)
- Ces descripteurs sont constitués de deux parties : un nombre **majeur** et un nombre **mineur** : <major> :<minor>.
- Habituellement le gestionnaire racine est repéré par 1 :, ce qui est équivalent à 1 : 0. Le nombre mineur d'un gestionnaire de mise en file d'attente est toujours 0.
- Les classes doivent avoir le même nombre majeur que leur parent.
- Le nombre majeur doit être unique à l'intérieur d'une configuration *egress* ou *ingress*.
- Le nombre mineur doit être unique à l'intérieur d'un gestionnaire de mise en file d'attente et de ses classes.

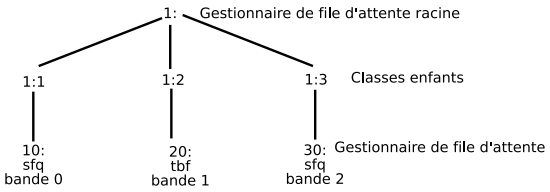
La commande tc : Terminologie



La commande tc : Le gestionnaire de mise en file d'attente PRIO

- Le gestionnaire PRIO peut être vu comme pfifo_fast où chaque bande est une classe
- Par défaut trois classes sont créées (avec gestionnaire de mise en file d'attente FIFO)
- Chaque fois qu'un paquet doit être retiré d'une file d'attente, la classe :1 est d'abord testée
- Paramètres
  - ***bands*** : Nombre de bandes à créer. Chaque bande est en fait une classe.
  - ***priomap*** : Par défaut fonctionne comme pfifo_fast.

La commande tc : Le gestionnaire de mise en file d'attente PRIO

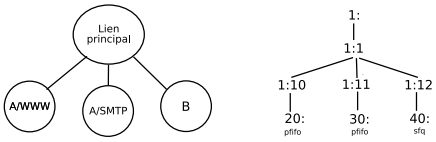



```
# tc qdisc add dev eth0 root handle 1: prio
## Ceci crée ⋆instantanément⋆ les classes 1:1, 1:2, 1:3
# tc qdisc add dev eth0 parent 1:1 handle 10: sfq
# tc qdisc add dev eth0 parent 1:2 handle 20: tbf \
  rate 20kbit buffer 1600 limit 3000
# tc qdisc add dev eth0 parent 1:3 handle 30: sfq
```


La commande tc : Le gestionnaire de mise en file d'attente HTB (Hierarchical Token Bucket)

- Permet de garantir une bande passante par classe avec la possibilité de spécifier la quantité de bande passante qui pourra être partagée.
- Lorsqu'une classe n'utilise pas la totalité de sa bande passante, les autres classes peuvent en bénéficier.
- Paramètres
  - rate : détermine quelle fraction de bande passante est attribuée à une classe
  - ceil : quantité de bande passante maximum qu'une classe peut avoir.
  - burst : déterminent de combien d'octets une classe peut dépasser le taux avant d'être arrêtée
  - prio : donne une priorité aux classes. Plus le nombre est bas, plus la priorité est haute

La commande tc : Le gestionnaire de mise en file d'attente HTB (Hierarchical Token Bucket)




```
tc qdisc add dev eth0 root handle 1: htb default 12
tc class add dev eth0 parent 1: classid 1:1 htb rate 100kbps ceil 100kbps
tc class add dev eth0 parent 1:1 classid 1:10 htb rate 30kbps ceil 50kbps
tc class add dev eth0 parent 1:1 classid 1:11 htb rate 10kbps ceil 30kbps
tc class add dev eth0 parent 1:1 classid 1:12 htb rate 60kbps ceil 100kbps

tc filter add dev eth0 protocol ip parent 1:0 prio 1 u32 \
   match ip src 1.2.3.4 match ip dport 80 0xffff flowid 1:10
tc filter add dev eth0 protocol ip parent 1:0 prio 1 u32 \
   match ip src 1.2.3.4 flowid 1:11

tc qdisc add dev eth0 parent 1:10 handle 20: pfifo limit 5
tc qdisc add dev eth0 parent 1:11 handle 30: pfifo limit 5
tc qdisc add dev eth0 parent 1:12 handle 40: sfq perturb 10
```


La commande tc : La classification des paquets avec des filtres

- la classification peut se faire sur une partie quelconque du paquet
  - adresse source/destination
  - port source/destination
  - le protocole
  - marque posée par un routeur (iptables)
  - le champ TOS

```
tc filter add dev eth0 parent 1:0 protocol ip prio 1 u32 ..
tc filter add dev eth0 parent 1:0 protocol ip prio 1 u32 match ip src 1.2.3.0/24
tc filter add dev eth0 parent 1:0 protocol ip prio 1 u32 match ip sport 80 0xffff flowid 10:1
tc filter add dev eth0 parent 1:0 protocol ip parent 1:0 prio 10 u32 \
   match ip tos 0x10 0xff flowid 1:4
```


La commande tc : La classification des paquets avec des filtres

Voici une liste(non exhaustive) des types de filtres qui sont disponibles

- fw en fonction de la marque posée par un firewall
- u32 en fonction des champs dans le paquet
- route en fonction de la route que va prendre le paquet
- rsvp en fonction du protocole rsvp

Les classificateurs ont des arguments en commun :

- protocol : Le protocole que ce classificateur traitera
- parent : La classe auquel le classificateur est attaché
- prio : La priorité de ce classificateur. Les plus petits nombres seront testés en premier
- handle : référence dont la signification dépend du filtre