Garanties de service pour les flots IP : un état de l'art

Résumé

Ce chapitre passe en revue un ensemble de mécanismes de gestion du trafic, ainsi que les principales architectures de garantie de service proposées dans la littérature. Nous posons les bases nécessaire à l'analyse des différentes propositions au vu de notre compréhension du trafic, de sa performance et du challenge que représente l'introduction de nouveaux mécanismes dans un réseau opérationnel. Une discussion des avantages et des faiblesses de chaque solution nous permet de positionner et de justifier l'approche alternative *Flow-Aware Networking* (FAN), qui sera détaillée dans le prochain chapitre.

Contributions :

- Proposition d'une méthodologie d'évaluation des solutions de QoS par rapport aux besoins des différentes types de flots ;
- État de l'art des principales architectures de QoS.

Sommaire

2.1 Le modèle *Best-Effort* actuel et ses limites

2.1.1 Réseau *Best Effort* et ordonnancement FIFO/DropTail

Le principe fondateur de l'Internet est d'offrir un réseau, fondé sur la brique IP, qui offre une connectivité de bout en bout sans aucune garantie de service [45]. On parle de réseau *Best Effort*. Les routeurs effectuent un routage simple des datagrammes IP en utilisant des files d'attente FIFO (*First In First Out*), dans lesquelles un paquet entrant est rejeté lorsque la file est pleine (mécanisme *DropTail*).

Il appartient ainsi aux couches supérieures, situées en bordure du réseau, d'assurer les fonctionnalités manquantes. C'est ainsi que le protocole de transport TCP a été proposé afin de réaliser un transfert fiable et équitable des données. On peut également citer des applications pair-à-pair qui font transiter le trafic au sein d'un overlay géré au niveau applicatif.

Cette architecture simple a permis l'émergence d'une multitude d'applications et a permis au réseau d'évoluer jusqu'à celui que nous connaissons aujourd'hui. Elle est toutefois remise en cause notamment pour son insuffisance à offrir un traitement avancé du trafic, ainsi que les garanties de service plus strictes requises par les nouvelles utilisations du réseau. Ces constatations ont motivé l'évolution des protocoles de transport, ainsi que l'introduction dans le réseau de mécanismes permettant un contrôle plus ou moins avancé du trafic.

2.1.2 Protocoles de transport

Les protocoles de transport utilisés majoritairement sur le réseau sont respectivement TCP (il en existe plusieurs variantes) et dans une moindre mesure UDP.

TCP (*Transmission Control Protocol*) permet d'assurer des transferts fiables, en mode connecté, entre deux points et est ainsi généralement utilisé pour supporter les flots élastiques. TCP à notamment pour objectif le partage équitable des ressources réseau disponibles ; il existe pour cela un grand nombre d'algorithmes dont nous présenterons les principaux dans la suite.

Dans une moindre mesure, on trouve aussi UDP (*User Datagram Protocol*) qui permet l'envoi non-fiable de données entre deux entités, et fonctionne en mode non-connecté. UDP est généralement associé avec les flots *streaming* et n'implémente pas de mécanisme de contrôle. Notons que les flots *streaming* peuvent également être transportés par TCP, et que l'on trouve également des protocoles adaptatifs basés sur UDP. Lorsque des flots UDP non-adaptatifs sont en compétition avec des flots TCP, ces derniers ont tendance à adapter leur débit et être défavorisés. Bien qu'ils subissent également des pertes, les flots UDP parviennent généralement à émettre à leur débit intrinsèque.

2.1.3 Comportement de TCP

La prédominance du protocole TCP explique pourquoi il se retrouve au cœur de nombreuses études, et que son fonctionnement puisse guider le dimensionnement de certaines ressources du réseau, voire être l'objet de mécanismes de gestion dédiés. Bien que nous privilégions des approches agnostiques aux procoles utilisés, il est nécessaire de présenter le fonctionnement de ce protocole afin de mieux comprendre ensuite la performance réalisée par les flots élastiques.

TCP a pour but d'établir, de manière décentralisée, une allocation efficace et équitable de la bande passante disponible sur les liens traversés. Le contrôle de congestion qu'il réalise détecte les événements de congestion dans le réseau et adapte en fonction le débit d'envoi des paquets. Il a été proposé dans l'article séminal de Van Jacobson [75] afin de faire face à la saturation du réseau (*congestion collapse*). Cette version originale de TCP (nommée TCP Tahoe), ainsi que les améliorations de l'algorithme de contrôle de congestion qui ont suivi (Reno et NewReno) s'appuient sur la détection des pertes de paquets. Elles se produisent lorsque les connexions TCP provoquent la saturation du buffer d'un des routeurs. Une alternative utilisée par TCP Vegas [37] est d'utiliser une estimation du délai subi par les paquets lors de la traversée des différentes files d'attente. Dans la suite, sauf mention contraire, nous réfèrerons implicitement à la version TCP NewReno.

Chaque paquet envoyé par TCP doit être acquitté par le récepteur. Le protocole maintient une fenêtre de congestion (*cwnd*), qui représente le nombre de paquets non encore acquittés, et qui permet de réguler le flux d'envoi. Le contrôle de congestion d'une connexion active (lorsque TCP est dans un mode dit *congestion avoidance*) repose sur l'algorithme AIMD (Additive Increase Multiplicative Decrease) qui gère l'évolution de *cwnd* : sa valeur est augmentée d'un pour chaque *cwnd* paquets acquittés, et diminuée de moitié lorsqu'une congestion est détectée. TCP possède d'autres modes de fonctionnement : *slow-start* intervient au début d'une connexion, afin de rapidement approcher le débit équitable ; *fast recovery* et *fast-retransmit* ont été proposées dans TCP Reno et NewReno afin de mieux

gérer les événements de congestion. Padhye *et al.* [120] ont établi la relation entre le débit atteint par une connexion TCP en fonction du taux de perte subit.

2.1.4 Limites de TCP

L'allocation équitable réalisée par TCP dépend de la coopération des sources [109]. Elle est ainsi vulnérable à celles n'implémentant pas TCP, parfois dans un but malicieux [1].

L'algorithme AIMD, sur lequel repose l'équité, souffre également de plusieurs imperfections. Il est par exemple inefficace sur des liens possédant un produit délai x bande passante élevé, et nécessite des buffers suffisamment dimensionnés (cette problématique sera abordée plus en détails dans le Chapitre 4). Plusieurs propositions de protocoles "haut débit" permettent à plusieurs flots longs en compétition d'obtenir de bonnes performances, avec un certain degré d'équité. Mais comme nous le verrons dans le Chapitre 5, elles sont souvent plus agressives et ne permettent pas une coexistence équitable avec les connexions TCP Reno, ce qui est une propriété recherchée (on parle de protocoles *TCP friendly*).

Le contrôle de *cwnd* en fonction des pertes subies par les connexions est également la cause d'autres imperfections de TCP. D'une part, il ne s'agit que d'une vision limitée des capacités du réseau, uniquement aux instants de congestion (action corrective). D'autre part, une perte de paquet ne signale pas nécessairement une congestion. Les réseaux optiques possèdent par exemple un taux d'erreur qui génèrera des pertes, et la réduction de *cwnd* et donc de débit ainsi causée ne sera compensée que tardivement, à cause de la lenteur de AIMD sur un lien à fort débit. Enfin, lors des événements de saturation, de nombreuses connexions TCP réduisent leur *cwnd* en même temps, ce qui entraîne une moins bonne utilisation des ressources du lien.

Nous remarquons également que les débits d'accès des utilisateurs sont aujourd'hui encore relativement faibles par rapport aux capacités de cœur de réseau. Ainsi le goulot d'étranglement se situe généralement dans le réseau d'accès, et les mécanismes de TCP n'interviennent pas dans le cœur de réseau qui reste transparent, sauf pour quelques flots bien particuliers. Toutefois, l'augmentation des débits d'accès avec l'arrivée de la fibre optique pourrait déplacer ce problème plus en amont.

2.1.5 Vers un déploiement d'architectures de qualité de service ?

La présence de nouvelles applications du réseau, comme la téléphonie ou la vidéo, requiert d'importantes garanties de service pour lesquelles le modèle Best Effort actuel, reposant majoritairement sur TCP, semble insuffisant. Un certain nombre de propositions mettant en œuvre de tels mécanismes ont été proposées afin d'apporter des garanties de service dans le réseau. Pourtant, les opérateurs sont toujours hésitants à introduire des mécanismes de gestion de trafic plus avancés. Le manque de preuves concernant l'efficacité de telles solutions, auquel s'ajoute leur complexité de mise-en-œuvre en est souvent la cause. Au lieu de cela, les opérateurs appliquent un surdimensionnement systématique des ressources afin de garantir une performance raisonnable lorsque le réseau n'est pas fortement chargé. Un dimensionnement qui assure une charge des liens inférieure à 40% est par exemple appliqué, pour faire face à une éventuelle augmentation de la charge en cas de panne et re-routage du trafic sur un autre lien.

Une telle gestion requiert des opérateurs des investissements réguliers coûteux, et semble être de plus en plus remise en cause avec la montée des applications gourmandes en ressources, comme la vidéo ou le pair-à-pair. On voit ainsi de nombreux réseaux déployer des mécanismes complexes d'inspection de trafic et de bridage ad-hoc. L'augmentation des capacités dans les réseaux d'accès (fibre optique, etc.) pourrait également être une motivation suffisante pour reconsidérer le déploiement de mécanismes de gestion de trafic. D'autant que certains mécanismes peuvent, comme nous le verrons, apporter une performance prévisible, présentant un avantage compétitif pour un opérateur et lui simplifiant sa tâche de dimensionnement.

2.2 Garanties de service pour les flots IP

Nous commençons par présenter un certain nombre de notions générales qui nous semblent utiles pour la compréhension du trafic et l'évaluation de sa performance, pour ensuite établir un ensemble de critères nous permettant de comparer les architectures existantes de garantie de service.

1. Par exemple, en ne réduisant par son débit d'envoi lors d'une congestion, une connexion peut bénéficier d'un débit plus élevé que les autres connexions partageant le lien qui réduisent leur débit.

2.2.1 Granularité du trafic IP

La modélisation du trafic peut être réalisée à différents niveaux de granularité, représentés sur la figure 2.1.

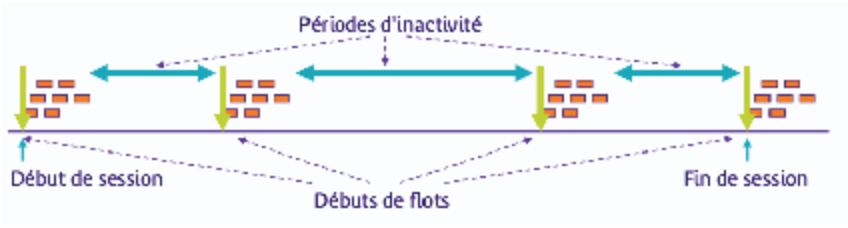

FIGURE 2.1 – Représentation des différentes échelles de trafic.

Le *paquet* IP est l'unité élémentaire traitée par la couche réseau. L'échelle de temps correspondante est de l'ordre de la microseconde ou de la milliseconde en fonction du débit des liens considérés. L'étude du trafic à cette échelle est rendu complexe par sa propriété d'autosimilarité [98]. Elle résulte de la distribution de la taille des flots, qui possède une variance infinie, ainsi que des rafales de paquets induites par TCP [124].

Le *flot* consiste en un ensemble de paquets groupés dans le temps, appartenant à une même application. Il correspond typiquement au transfert d'un objet numérique sur le réseau, dont la performance sera ressentie par l'utilisateur (par exemple son temps de transfert). En pratique, l'identification des flots dans le réseau repose sur le fait que les paquets ont des valeurs communes pour certains champs de leur en-tête IP et TCP/UDP[2], et sont groupés dans le temps.

Les flots se présentent au réseau en *sessions*. Il s'agit d'une succession de flots entrecoupés de temps de réflexion (*think times*) de l'ordre de quelques secondes, et reflétant l'activité d'un utilisateur (session de navigation, e-commerce, etc.). Les sessions sont en pratique plus difficiles à identifier que les flots.

2.2.2 2 classes d'applications : S et E

En dépit de la diversité des applications transportées par le réseau, il est possible d'en distinguer deux groupes ayant des propriétés et des besoins de garanties de qualité de service différents [140, 131].

flots élastiques : ils sont induits par le transfert de documents numériques (pages web, fichiers, ...), et utilisent généralement la couche de transport TCP. Ils représentent la majeure proportion du trafic Internet. Leur dénomination vient du fait que leur débit s'adapte aux ressources disponibles dans le réseau. Le maintien d'un débit moyen à long terme est suffisant pour garantir un temps de réponse satisfaisant pour l'utilisateur.

flots streaming : ils sont produits par les applications audio et vidéo, et sont typiquement transportés sur UDP. Ils sont caractérisés par leur durée intrinsèque (le signal à transmettre). Certaines applications sont également caractérisées par leur débit, alors que d'autres sont adaptatives en fonction du niveau de congestion rencontré, qui détermine la qualité reçue. Contrairement aux flots élastiques qui ne requièrent qu'une garantie de service à long terme, leur qualité de service est ici affectée par le débit instantané reçu. Ils peuvent également nécessiter, au niveau paquet, de faibles délais ainsi qu'un taux de pertes négligeable.

2.2.3 Régimes opérationnels des liens

Afin de comprendre la performance réalisée par les flux, il est utile de distinguer trois régimes de partage de bande passante, représentés en figure 2.2. Dans le dessin du haut, les flots sont représentés par des boites de hauteur égale à leur débit exogène, et dont la position horizontale est déterminée par le temps de début et la durée du flot. La position verticale de la boîte sur le lien est choisie aléatoirement. Au dessous, on représente l'évolution dans le temps du débit global des flots en cours.

2. Ce sont le protocole de transport et l'adresse IP accompagnés soit des ports TCP/UDP pour IPv4, soit du *flow label* pour IPv6

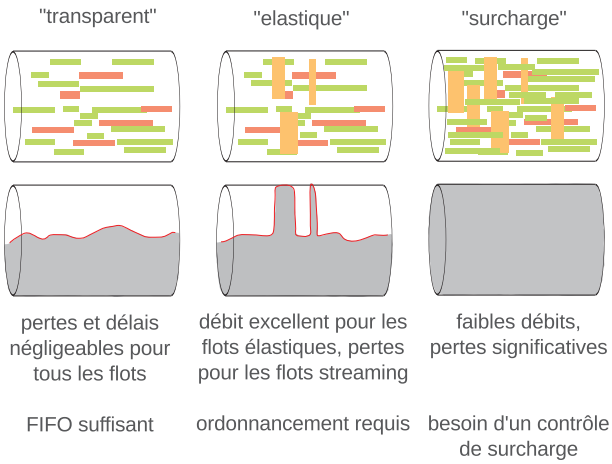

FIGURE 2.2 – Trois régimes d'opération du lien : "transparent", "élastique" et "surcharge"

Le cas de gauche transcrit un régime "transparent" dans lequel tous les flots ont un débit limité et la somme de leurs débits reste inférieure à la capacité du lien (avec une forte probabilité). La limite en débit peut être imposée par un lien d'accès, ou encore par l'allocation faite sur un autre lien. Les pertes de paquets ainsi que les délais sont alors très faibles. C'est le régime dans lequel se trouvent la plupart des liens du réseau Internet actuel, en raison du surprovisionnement effectué par les opérateurs et des débits d'accès relativement faibles de la plupart des utilisateurs.

Le régime "élastique" représenté au milieu de la figure se produit lorsque des flots au débit potentiellement non limité s'ajoutent au trafic généré par les flots goulottés. Une telle situation se produit même lorsque la charge globale est inférieure à la capacité du lien. Ce sera typiquement le cas d'un lien d'accès. Il est également possible que des flots limités en débit se combinent pour saturer les ressources du lien : par exemple dans un réseau de collecte, où les flots sont limités par le débit d'accès de l'utilisateur, voire dans le cœur de réseau avec l'arrivée des technologies à très haut débit comme la fibre optique. Lors des instants de congestion, les buffers des liens se remplissent et débordent, causant des délais et des pertes impactant notamment les flots *streaming*. Les délais peuvent être significatifs si la taille des buffers est importante. En plus d'un dimensionnement adéquat des ressources, il semble nécessaire d'utiliser un mécanisme d'ordonnancement ou de gestion de la file d'attente afin de préserver la QoS des flots dans ce régime (eg. différenciation). Notons que les liens de peering entre opérateurs réseaux, ou que les liens d'interconnexion d'un *datacenter* sont typiquement en régime élastique.

Le cas de droite correspond à un régime de "surcharge" où l'offre de trafic (produit du taux d'arrivée des flots et de la taille moyenne d'un flot) dépasse la capacité disponible sur le lien. Dans ces conditions, le nombre de flots augmente très rapidement et leur débit tend alors vers zéro. Les algorithmes de contrôle de congestion tels que ceux de TCP sont incapables de gérer ce genre de congestion. Il en résulte une QoS dégradée pour tous les utilisateurs, uniquement stabilisée par l'abandon des connexions. Des mécanismes de gestion de cette surcharge tels qu'un contrôle d'admission sont nécessaires afin d'éviter une telle configuration.

2.3 Techniques d'amélioration de la QoS

Les insuffisances présentées plus haut de la situation actuelle ont donné lieu à un ensemble de propositions. Celles-ci reposent sur l'extension des mécanismes en bordure de réseau, comme le développement de protocoles TCP plus adaptés, et/ou l'intégration au réseau de mécanismes avancés de gestion du trafic.

Le développement de nouveaux protocoles de transport n'étant pas l'objet de notre étude, nous n'entrerons pas dans trop de détails, au delà de la description faite à l'occasion du Chapitre 5. Il est toutefois difficile, comme le souligne la RFC 2309 [35], de contrôler dynamiquement le trafic depuis la bordure de réseau, sans indication supplémentaire que les pertes de paquets. Ce document recommande ainsi l'introduction de mécanismes d'ordonnancement et de contrôle des files d'attente dans le réseau (AQM, introduits ci-dessous).

Nous passons en revue ici de telles contributions, avant de nous intéresser à des mécanismes plus avancés tels que le contrôle de la surcharge ou la différenciation de trafic. Leur analyse nous permettra de mieux comprendre la performance des architectures plus complètes qui seront étudiées à la fin de ce chapitre.

2.3.1 Allocation de ressources et contrôle de congestion

Gestion des files d'attente

L'ajout de mécanismes de gestion des files d'attente au sein des routeurs (*Active Queue Management*, AQM) représente un complément au contrôle de congestion de bout en bout [53]. Ces mécanismes permettent de gérer la taille de la file d'attente en éliminant des paquets si nécessaire. Leur vision locale du trafic permet d'informer de manière plus ou moins complexe les protocoles de transport de l'état de congestion du lien, et de remédier aux problèmes dûs à la politique *DropTail*. On retrouve ici les éléments de la terminologie ATM : *congestion-indication* et *explicit-rate* [132].

Exemple d'AQM : prévention de la congestion

RED [57], l'AQM le plus connu, rejette les paquets de manière probabiliste en fonction de la longueur de la file d'attente, afin de forcer TCP à adapter son débit. La variante *Weighted RED* (WRED) permet de définir plusieurs classes de trafic en fixant le seuil de rejet. La probabilité de rejet ne dépend pas de l'activité des flots et souffre ainsi de problèmes d'équité. De plus, RED ne permet pas d'éliminer les pertes de paquets.

Feng *et al.* [51] suggèrent que la taille de la file d'attente ne permet pas d'estimer correctement le niveau de congestion, puisque le trafic n'est pas de type Poisson [126, 98]. Leur proposition, BLUE, se base sur les taux d'inactivité du lien et de pertes de paquets. L'idée de découpler la mesure de congestion de la taille de la file d'attente est également présente dans REM [97], PI [72, 71] et AVQ [144].

Malgré les propriétés d'ajustement automatique des algorithmes plus récents tels que BLUE, ARED [58] ou PI, la plupart des AQM sont complexes à paramétrer et n'offrent pas une performance prévisible.

Marquage de paquets

Afin d'éviter le rejet d'un paquet, qui gaspille les ressources du réseau, Floyd [56, 130] propose d'ajouter un ou plusieurs bits ECN (*Explicit Congestion Notification*) dans les entêtes TCP/IP des paquets, qui indiquent un état de congestion imminent. Cette modification préventive nécessite une standardisation afin d'obtenir la coopération des routeurs intermédiaires, notamment entre plusieurs domaines administratifs, ainsi que des protocoles de transport à chaque bout de la connexion.

Flots non-coopératifs

La plupart de ces mécanismes reposent sur une collaboration des utilisateurs, et ne prennent en compte ni les protocoles autres que TCP, ni les flots malveillants. La difficulté principale est d'éviter de maintenir un état pour chaque flot en cours, ce qui poserait des problèmes de passage à l'échelle pour des liens de cœur de réseau.

RED with penalty box [106] détermine les flots non-coopératifs à partir d'un historique des derniers événements de perte, et leur impose une limite en débit (CBQ [54]). Flow RED [101] se base sur la place occupée par chaque flot dans le buffer. Stabilized RED [117] estime le nombre de connexions actives. SFB [51] maintient une probabilité de rejet pour les flots de manière compacte grâce à l'utilisation des filtres de Bloom, sur le même principe que BLUE (taux de perte de paquets et utilisation du lien).

CHOKe [121] utilise le contenu du buffer en y choisissant un paquet tiré au hasard ; si ce paquet appartient au même flot que le paquet entrant, les deux sont rejetés. L'algorithme AFD [122] maintient une structure de données annexe afin d'estimer le débit des flots qui émettent au dessus du débit équitable, et d'adapter la probabilité de rejet en conséquence.

Pour les algorithmes utilisant le contenu de la file d'attente, la question de leur sensibilité à la taille de cette dernière reste à déterminer. Le paramétrage reste problématique dans tous les cas et consiste souvent en des règles empiriques.

Solutions mixtes transport/AQM

Plutôt que de signaler une congestion, XCP [81] propose aux sources une évolution incrémentale de *cwnd* sur plusieurs RTT[3], qui remplace ainsi la brique AIMD de TCP. L'évolution lente de XCP, ou des protocoles basés sur AIMD, n'offre pas une performance satisfaisante pour les flots courts, qui représentent la majorité du trafic. RCP [47] propose d'envoyer explicitement aux sources des flots un débit à réaliser afin d'émuler un comportement *Processor Sharing* (PS). Ce débit est un débit équitable calculé périodiquement par le routeur à partir du débit des flots et de la taille de la file d'attente. Un

3. *Round Trip Time*, une mesure du délai aller-retour entre deux hôtes

inconvénient sérieux de propositions telles que XCP et RCP est de requérir la coopération de l'ensemble des routeurs du réseau.

2.3.2 Ordonnancement

Differents types d'ordonnancement

L'ordonnancement au sein des routeurs permet également de réguler les flux de paquets en choisissant lequel sera envoyé prochainement. L'allocation est effectuée par le routeur, ce qui permet de gérer plus simplement les flots non-réactifs, et ainsi de ne pas dépendre du protocole de transport, de sa rapidité à s'adapter, voire de sa coopération. Il existe de nombreuses alternatives au schéma FIFO, qui consistent généralement au maintien de plusieurs files d'attentes (virtuelles ou non). Cette classification peut se faire par flot, ou par classes.

L'émission d'un paquet des différentes files d'attente se fait soit en priorité – on parle de *Priority Queueing* (PQ) – ou en fonction de mécanismes tels que RR (*Round Robin*) [160], WRR (*Weighted Round Robin*) [157] ou DRR (*Deficit Round Robin*) [155], qui réalisent des mécanismes de tourniquets entre les files d'attente afin qu'aucune ne monopolise les ressources du lien. L'ordonnancement idéal est souvent considéré être de type *fair queueing* (FQ) [156], qui consiste à attribuer le même débit à chaque flot, et il en existe de nombreuses implémentations.

Fair Queueing

FQ réalise un partage *max-min* de la bande passante et présente de nombreux avantages d'isolation et d'équité entre les flots. Il permet notamment de laisser au protocole de transport la liberté d'exploiter au mieux les ressources qui lui sont allouées sans gêne pour les autres flots.

Une vision plus détaillée des avantages présentés par l'introduction de FQ sera présentée dans le Chapitre 5, où nous étudierons les interactions entre les protocoles et FQ. Ses propriétés de différentiation sont également exploitées, et notamment dans la proposition de contrôle d'admission pour l'architecture FAN que nous présentons dans le Chapitre 3. Enfin, Keshav [87] montre comment il est possible d'exploiter dans le protocole de transport l'information implicitement fournie par l'ordonnanceur qui émet les flots au débit équitable.

FQ n'est souvent pas considéré comme une alternative réalisable du fait qu'il nécessite de maintenir une file d'attente par flot. Plusieurs propositions permettent d'obtenir un algorithme approximatif [107, 145] mais elles restent encore à étudier plus profondément. Nous verrons dans le Chapitre 3 une proposition d'un algorithme réalisable et extensible de FQ, appelée PFQ (*Priority Fair Queueing*), qui a été faite préalablement au début de cette thèse, dans le cadre de l'architecture *Flow-Aware Networking*.

Equité des débits

L'équité en débit n'est pas nécessairement un objectif absolu. Elle est d'ailleurs remise en cause dans [40], au profit d'une égalité de la congestion causée dans le réseau. Annoncée comme plus juste pour un utilisateur, elle permet notamment d'éviter le recours à plusieurs flots parallèles afin d'augmenter la quantité de bande passante reçue. Une telle architecture est présentée plus loin dans la Section 2.4.1.

SRPT [69] (*Shortest Remaining Processing Time*) est un ordonnancement préemptif qui donne la priorité aux documents les plus courts : le serveur est supposé connaître le volume de données restant à être transféré de chaque document, et dédie la totalité de sa capacité à servir celui représentant le minimum. La performance des flots longs est en effet relativement indépendante de celle réalisée par les flots courts, il est ainsi intéressant de leur donner priorité : [69, 10]. Cette discipline de service est connue pour améliorer le temps de service des documents, notamment lorsque leur taille possède une distribution à queue lourde (eg. [132]). Si cet ordonnancement semble améliorer globalement la performance, sa mise en œuvre reste cependant complexe, et il conviendrait de déterminer plus précisent son comportement au sein du réseau.

D'autres propositions existent, que nous n'avons pas prétention à énumérer. Nous verrons cependant dans le prochain chapitre qu'une telle équité des débits est fortement recommendée et nous permet le développement de modèles de performance simples et robustes.

2.3.3 Contrôle de la surcharge

Impact de la surcharge

Le contrôle de congestion effectué par TCP (ou par un mécanisme au niveau paquet au sein d'un routeur) n'empêche toutefois pas le réseau d'entrer en situation de surcharge, lorsque la demande en débit dépasse la capacité. Le débit individuel des flots décroit vers zéro, ce qui entraine une accumulation des connexions et rend le système instable : le temps moyen de transfert des flots tend vers l'infini. Notons qu'il s'agit d'une instabilité au niveau flot, puisque les tailles finies des buffers causent des pertes qui stabilisent le système au niveau paquet.

En réalité, ce phénomène est mitigé par les *timeouts* subis par les connexions TCP, ainsi que par un phénomène d'impatience de la part des utilisateurs ou applications. Les conséquences d'une telle surcharge, due aux retransmissions et aux transferts abandonnés, sont présentées dans [137, 34].

Contrôle d'admission

S'il est parfois contesté dans le cœur de réseau, le besoin d'un contrôle d'admission [133, 104] est généralement reconnu, notamment pour protéger la performance des flots *streaming*. Shenker et al [140] l'illustrent par exemple au travers de fonctions d'utilité : il existe un débit minimal en-dessous duquel le flot n'est pas utile. Roberts et al. [137, 14, 18, 20] le recommandent également pour les flots élastiques (surtout en cas de tarification à l'usage [135]). Un tel contrôle d'admission peut être réalisé de différentes manières, reposant sur une signalisation du trafic, ou sur des mesures effectuées localement (on parle de contrôle d'admission implicite).

La notion de contrôle d'admission est souvent mal perçue, notamment dans le cœur de réseau, du fait qu'il est difficile de justifier à un utilisateur du rejet d'un flot. Il faut bien préciser que de tels mécanismes supposent un dimensionnement satisfaisant au préalable par l'opérateur, qui analyse le trafic passé et tente de prévoir les évolutions possibles dans le futur, les cas de panne, etc. Les mécanismes de QoS n'ont pas pour ambition de s'y substituer, mais plutôt de gérer des situations exceptionnelles de surcharge, comme une panne suivie d'un re-routage de trafic, où la performance est généralement dégradée de manière significative (phénomènes de *flash crowd*).

Autres mécanismes

Le contrôle d'admission est transparent en ce qu'il peut faire partie intégrante des autres mécanismes de gestion de la surcharge : multi-chemins et routage adaptatif [119], ou encore équilibrage de charge [27].

Il est parfois possible que le lien devienne surchargé malgré l'utilisation d'un contrôle d'admission (suite à des erreurs de mesure, un changement de profil des flots acceptés, etc.). Nous en verrons un exemple dans le Chapitre 6 lors de situations de *flash crowd*. Il convient alors de considérer des mécanismes de préemption de flot, qui consistent à interrompre un certain nombre de flots en cours.

2.3.4 Différenciation et classes de trafic

Une autre approche consiste à grouper les flots en différentes classes (CBQ, *Class-Based Queueing*) [154] et leur offrir un traitement différencié à des fins de tarifications, ou encore pour augmenter l'utilité du réseau. Comme nous le verrons dans le Chapitre 7, il peut être nécessaire de distinguer plusieurs classes de service dans le réseau d'accès ou de collecte, afin de protéger les flux de voix ou les flux TV à fort débit par exemple [25].

Dans le réseau de cœur, il est possible d'exploiter les différentes contraintes de QoS entre les flots *streaming* (faibles délais et pertes) et élastiques (débit moyen minimal) en servant les premiers en priorité, sans affecter les derniers. Dans un tel contexte, Bonald et Massoulié [28] exhibent des configurations de réseaux instables même lorsque le système n'est pas saturé ; un mécanisme de contrôle de la surcharge est alors d'autant plus nécessaire.

La classification du trafic permet d'offrir différents niveaux de transparence pour chaque type de flot (en termes de délais ou de taux de pertes pour les flots *streaming*, ou de bande passante pour les flots élastiques). Toutefois, Benameur *et al.* [20] suggèrent que ce type de différenciation n'est pas efficace, puisque la QoS effectivement ressentie par les flots est soit excellente, soit très mauvaise et il est difficile de définir plusieurs niveaux intermédiaires. La différence est visible en surcharge uniquement, et n'est pas manifeste si les flots ont un débit d'accès limité. Roberts *et al* [131] montrent que l'attribution de poids aux flots élastiques (modèle *Discriminate Processor Sharing*) est peu intéressante et rend leur performnce sensible à la distribution de leur taille. Il en est de même pour les flots *streaming*, et il devient difficile de prédire la performance obtenue.

Comme nous l'avons vu précédemment, l'utilisation d'une discipline de service SRPT peut être intéressante. Elle revient à différencier les flots en fonction de leur taille. Bansal *et al.* [11] montrent que cet ordonnancement est plus efficace que *Processor Sharing*. Le risque de famine des flots longs nécessite cependant un contrôle de la charge des autres flots, même si cet effet est minimisé pour des distributions de tailles de flots à queue lourde.

L'usage d'un contrôle d'admission permet en outre de garantir une bonne qualité de service pour l'ensemble des flots acceptés, et ainsi de reporter l'impact de la congestion sur un petit nombre de flots rejetés. On réalise ainsi une différenciation par la disponibilité du réseau [17] (qui peut être également déclinée en plusieurs classes).

De nombreuses méthodes existent pour identifier les flots à des fins de différenciation. La plus connue est certainement la méthode DPI (*Deep Packet Inspective*) qui inspecte le contenu de chaque paquet jusqu'au niveau applicatif éventuellement. Elle est généralement utilisée afin d'identifier une application. Il est également possible de distinguer le trafic en fonction de sa source ou de sa destination, ou encore de reposer sur une signalisation et la vérification a posteriori de sa conformité.

De tels procédés sont généralement soustraits à une tarification différenciée, et contraires à la neutralité des réseaux. Des approches telles que SRPT (sur la taille des flots), ou la différenciation faite par le *fair queueing* (sur leur débit, voir Chapitres 3 et 5) sont au contraire compatibles avec cette neutralité.

2.4 Analyse des principales architectures de qualité de service

Devant la nécessité d'offrir des garanties de service au trafic, et face aux insuffisances d'un réseau *Best-Effort*, un certain nombre d'architectures ont été proposées combinant un ensemble de mécanismes présentés dans la section précédente. Nous avons volontairement limité notre présentation à celles que nous jugeons les plus importantes, ou qui présentent des caractéristiques intéressantes afin de positionner l'approche Flow-Aware Networking (FAN) qui est l'objet de cette thèse. Ce sont : (1) IntServ, (2) DiffServ, (3) Flow-Aware Diffserv, (4) DPS/SCORE, (5) Flow-State Aware et variantes, (6) Congestion Exposure, et enfin (7) Flow-Aware Networking.

Nous présentons d'abord succinctement les points importants de chacune de ces architectures, avant de procéder à leur analyse. Pour cela, nous nous basons sur notre compréhension du trafic et de ses besoins en termes de performance tels que présentés plus haut dans ce chapitre. Nous procédons à une analyse systématique des fonctionnalités supportées afin de fournir des garanties pour les différents types de flots, ainsi que des mécanismes mis-en-œuvre pour leur réalisation.

Il est intéressant de remarquer qu'un grand nombre de ces propositions, comme FAN, considèrent le trafic, ou du moins la performance, au niveau d'un flot [4]. Joung [79] et Wojcik [164] font également une analyse de ces approches, mais ils se limitent à un petit nombre d'aspects tels que la définition des flots ou des classes de service, le contrôle d'admission, la gestion des files et la signalisation. Nous renvoyons le lecteur à ces références pour plus de détails.

2.4.1 Présentation des principales architectures

Intserv

IntServ [165] est l'une des premières architectures proposées afin de fournir des garanties de service pour les flux IP. Elle repose sur le protocole RSVP [36], qui permet pour chacun de réserver des ressources réseau (bande passante et buffer) au sein des routeurs successifs. Trois principales classes de trafic sont définies et la plus prioritaire permet d'assurer un débit (déterministe) pour les flots *streaming*. Sans le spécifier, IntServ recommande un contrôle d'admission pour les classes les plus prioritaires, et nécessite l'introduction d'un ordonnancement approprié (WFQ/PQ). Un mécanisme de préemption est également prévu.

Intserv nécessite le support de l'ensemble des routeurs de bout en bout. Le besoin d'établir une signalisation pour chaque flot rend difficile le passage à l'échelle hors d'un réseau d'accès, notamment lorsqu'un grand nombre de flots est en compétition. De plus, la caractérisation du trafic lors de la réservation, ainsi que la vérification de sa conformité restent problématiques.

DiffServ

DiffServ [67] est proposé comme une alternative plus extensible à IntServ, où le contrôle individuel des flots (par exemple par RSVP) n'est effectué qu'en bordure de réseau. Le reste du temps, chaque flot

4. la définition d'un flot est parfois plus floue, ou considère quelques champs supplémentaires de l'en-tête IP, sans toutefois affecter les remarques faites ici

est associé à un agrégat au travers d'une marque faite dans l'en-tête IP du paquet [5]. Il suffit alors à un routeur d'inspecter la marque apposée pour classifier le paquet et lui offrir un traitement différencié. Trois classes principales sont prévues (éventuellement subdivisées), correspondant à celles de IntServ.

DiffServ convient dans une certaine mesure pour un réseau d'opérateur qui souhaite proposer à ses clients des services à valeur ajoutée (Triple Play [6] par exemple), puisqu'à la fois les serveurs et les clients sont situés dans le même domaine.

L'évaluation de la performance offerte aux flux *streaming* en fonction des ressources réservées à la classe prioritaire est cependant difficile, et ce notamment dans un contexte inter-domaine où chaque opérateur peut implémenter et configurer DiffServ différemment. En outre, le manque d'un contrôle individuel des flots au sein du réseau fait qu'il est difficile d'offrir des garanties aux flux sauf à appliquer un facteur de surdimensionnement. La performance requiert une bonne connaissance de la matrice de trafic entre les différents points du réseau, et est ainsi sensible aux changements de routage.

Nous verrons dans le chapitre suivant que la distinction d'un grand nombre de classes de trafic n'est pas utile puisque la performance est soit très bonne si le lien n'est pas en surcharge, soit très mauvaise si le lien est en surcharge : l'ensemble des flots des classes les moins prioritaires subissent alors la congestion.

Flow-Aware DiffServ et autres extensions

Une faiblesse supplémentaire de l'architecture DiffServ est que le taux de service de chaque classe n'est pas proportionnel au nombre de flots en cours. Il est ainsi possible que des classes prioritaires peu chargées offrent un meilleur service que les classes plus prioritaires. Li *et al.* [99] abordent ce problème en introduisant des estimateurs permettant d'adapter dynamiquement le taux de service au nombre de flots en cours dans chaque classe.

Un certain nombre d'autres propositions étendent DiffServ afin de mieux gérer des situations de surcharge, et ainsi moins reposer sur un surdimensionnement des liens.

Pre-Congestion Notification

Un exemple est PCN (*Pre-Congestion Notification*), qui repose sur l'ajout de marques de congestion au sein du domaine Diffserv, afin que les nœuds de bordure puissent décider du contrôle des flots. Ce mécanisme s'applique aux classes les plus prioritaires de DiffServ et permet la réalisation d'un contrôle d'admission et d'un processus de terminaison de flots.

Un tel déploiement, *Controlled Load* [41], propose d'appliquer deux marques différentes en fonction de la taille atteinte par la file d'attente. Les paquets reçoivent une marque de pré-congestion lorsqu'un seuil d'admission est dépassé (par exemple par un mécanisme de type ECN sur une file virtuelle de capacité réduite). Ces signaux sont agrégés afin de décider de l'admission des nouveaux flots. Lorsqu'un second seuil est dépassé, les paquets reçoivent une marque de congestion, qui permettra de décider de la préemption éventuelle de certains flots. Les auteurs de [44] introduisent une alternative n'utilisant qu'un seul type de marquage pour définir les seuils d'admission et de préemption de flots.

Ces mécanismes nécessitent de standardiser à la fois les conditions de marquage et de congestion. Les mécanismes de marquage reposent généralement sur des files virtuelles avec RED, ou sur des seaux à jetons (*token bucket*) [161], dont la performance est sensible à des caractéristiques détaillées sur le trafic. Il est de plus difficile de prédire la performance réalisée en fonction des seuils de marquage, d'admission ou de préemption.

Stateless CORE & Dynamic Packet State

L'approche SCORE (*Stateless CORE*) propose un compromis entre une architecture où chaque routeur doit garder un état par flot (comme IntServ), et une architecture sans état (comme DiffServ). Les routeurs de bordure reconnaissent les flots et effectuent les traitements les plus complexes. L'information nécessaire est alors encodée au sein des paquets afin de permettre aux routeurs de cœur de prendre les décisions appropriées (*Dynamic Packet State*). Nous présentons deux réalisations de cette architecture : CSFQ et CJVC.

CSFQ (*Core Stateless Fair Queueing*) [145] consiste a émuler un réseau *fair queueing* en insérant des estimations du débit des flots dans les en-têtes de paquets, afin d'adresser les problèmes d'extensibilité de DRR tout en offrant une performance similaire. La tâche des routeurs de cœur consiste à évaluer le débit équitable sur le lien, et rejeter les paquets entrant de manière probabiliste en fonction du ratio entre le débit du flot encodé dans le paquet, et ce débit équitable.

5. réutilisation du champ ToS, dénommé DSCP dans DiffServ
6. Ce terme dénote les offres fournissant à la fois des services Internet, de TV et de téléphonie

CJVC (*Core Jitter Virtual Clock*) [82] est une autre adaptation des mêmes mécanismes afin d'offrir des garanties de service par flot (délai et bande passante) en implémentant un mécanisme de contrôle d'admission. Il s'agit de l'adaptation dans l'architecture SCORE de *Jitter Virtual Clock* qui émule le fonctionnement de IntServ.

Au delà des modifications nécessaires dans les en-têtes de paquets, il n'est pas clair comment ces différents algorithmes peuvent être combinés pour fournir une architecture réalisant à la fois CSFQ et CJVC par exemple. La robustesse d'une telle architecture dépend fortement du routeur de bordure qui effectue l'encodage des paquets. Il est proposé que ce soient les utilisateurs eux-mêmes qui marquent les paquets : une vérification statistique peut alors être utilisée par les routeurs de bordure, avec isolation des hôtes malveillants, mais cela reste un mécanisme complexe à réaliser.

Flow Routing

Les sociétés Caspian et Anagran ont commercialisé des routeurs travaillant au niveau flot. La contribution essentielle est une technologie permettant de maintenir en mémoire l'état de l'ensemble des flots en transit (rendue possible d'après eux par la réduction du coût des mémoires). Cela leur permet de réaliser un routage par flot, où la décision de routage est prise uniquement pour le premier paquet. Un tel comportement serait moins gourmand en temps processeur que l'approche classique qui inspecte chaque paquet.

Ces solutions prouvent la faisabilité de maintenir une table de flots au sein du routeur, et ont permis la proposition de l'architecture Flow-State Aware, présentée ci-après.

Flow-State Aware

L'architecture Flow-State Aware (FSA) préconise un ensemble de classes de trafic qu'un réseau devrait supporter, et a fait l'objet d'une recommandation UIT [1].

Si la plupart restent classiques, on remarquera la classe *Conditionally Dedicated Bandwidth* (CDBW). CDBW propose une variante moins stricte de contrôle d'admission qui permet de concentrer les pertes de paquets en cas de congestion sur un petit ensemble de flots [112], par exemple les flots dernièrement admis. FSA réalise cette fonctionnalité à l'aide d'un indicateur de priorité à deux états (*discard first* et *discard last*), qui détermine un ensemble de flots qui subira la congestion en premier. On peut les voir comme une relaxation de la notion de rejet ou d'admission.

FSA offre ainsi certaines classes de trafic pour lesquelles un flot peut démarrer immédiatement, sans attendre de décision du réseau, en étant admis conditionnellement. Il pourra ensuite évoluer vers un statut plus prioritaire. Ce mécanisme ressemble au *cell loss priority tagging* dans ATM [143]. Cette approche permet également de ne pas nécessiter de réservation de ressources pour les classes autres que *Guaranteed Rate* et ainsi d'obtenir une utilisation plus efficace des ressources du lien.

FSA souligne l'importance de gérer la qualité de service du trafic au niveau des flots individuels, mais recommande l'utilisation de mécanismes DiffServ pour le cœur de réseau, à des fins de passage à l'échelle : l'état d'un agrégat[7] contient alors la somme des caractéristiques individuelles des flots.

Alors que la spécification n'impose aucun type de signalisation, on peut considérer l'usage de la solution Flow State présenté ci-dessus, qui repose sur une signalisation intra-bande légère [113]. Cela permet une réalisation plus extensible que RSVP par exemple, mais nécessite en contrepartie un traitement matériel de la structure de QoS annoncé dans l'en-tête du paquet, afin de tenir les haut débits.

La plupart des classes de trafic reposent également sur des profils de flots, et la recommandation n'indique pas comment vérifier leur conformité, ce qui reste encore un problème complexe. Le problème de l'inter-domaine persiste également.

Congestion Exposure

Congestion Exposure est une architecture en cours de standardisation à l'IETF, qui consiste à rendre visible pour le réseau la congestion engendrée par les flots. Elle est illustrée par la proposition *re-feedback* qui force la source à annoncer au sein de ses paquets le taux de congestion qu'elle pense causer au réseau, par exemple au travers du bit ECN [159] des en-têtes TCP et IP (on parle de *re-ECN*).

Les routeurs détectent le niveau de congestion, par l'utilisation de mécanismes de contrôle de la file d'attente, et positionnent des marques ECN dans les paquets. Les récepteurs répètent cette marque dans le champ IP, qui est alors vue comme une dette pour l'émetteur qui doit alors puiser dans son crédit pour marquer également les paquets qu'il envoie. Les routeurs de bordure proches de l'émetteur

7. un ensemble de flots groupés dans la même classe, au sens DiffServ

peuvent alors rejeter les paquets dont la balance de crédit est en déficit par rapport aux marques de congestion reçues.

Dans cette architecture, aucune notion de flot n'est maintenue dans le réseau, et le contrôle est entièrement effectué de bout en bout. Elle représente une alternative à l'équité des débits qui est généralement considérée comme un objectif en soi de la plupart des solutions de qualité de service, remise en cause par Briscoe [39].

On en trouve les prémisses dans le concept d'Internet autogéré proposé par Kelly [83], qui montre que l'ajout d'une telle marque de congestion permet aux utilisateurs en bout de réseau de gérer de façon optimale l'allocation des ressources, et notamment de garantir de faibles délais grâce à de petits buffers. Le paiement d'une faible taxe proportionnelle au taux de congestion engendré, vu comme incitatif, a en fait été mal perçu par la communauté, notamment parce qu'il pouvait être un encouragement à sous-dimensionner volontairement les ressources du réseau.

2.4.2 Discussion et positionnement de FAN

L'architecture FAN sera présentée en détail dans le chapitre suivant. Dans cette section, nous signalons les caractéristiques de chaque proposition, et nous en servons pour positionner l'architecture FAN.

Granularité des propositions

La reconnaissance des flots est poussée plus ou moins loin dans le réseau en fonction des propositions. *Congestion Exposure* est similaire à l'approche *Best Effort* actuelle et considère le trafic uniquement au niveau paquet ; il appartient aux hôtes en bordure de réseau de gérer leur performance.

DiffServ ne matérialise les flots qu'en bordure de réseau ("domaine DiffServ"), et traite uniquement des agrégats dans le cœur en fonction d'une marque simple apposée dans l'en-tête de paquet. C'est également le cas de l'approche DPS/SCORE, qui maintient un état par flot (plus complexe) sur les nœuds de bordure uniquement, et transporte l'information nécessaire pour le cœur de réseau dans l'en-tête du paquet. De telles approches ont été conçues suites aux problèmes d'extensibilité de l'architecture IntServ, dans laquelle il convenait de signaler et de garder en mémoire l'état de chaque connexion.

Les approches Caspian/Anagran, FSA, FAN quand à elles identifient les flots sur chaque équipement. Les réalisations faites par Caspian et Anagran sont la preuve de la possibilité technique de réaliser un état par flot, qui apporte des avantages en terme de routage et de monitoring.

IntServ repose sur une signalisation lourde basée sur RSVP, FSA sur une signalisation intra-bande légère, alors que FAN se base uniquement sur des mesures implicites du trafic.

De plus, seul le contrôle d'admission de cette dernière requiert un état pour tous les flots, et il est possible de relaxer cette contrainte ; l'algorithme de *fair queueing* ne nécessite de retenir que les flots actifs, qui sont en nombre beaucoup plus limité, indépendamment de la capacité du lien.

Classes de service et différenciation

Congestion Exposure qui repose sur un réseau *Best Effort* ne fait aucune distinction de classe. Au contraire, la plupart des autres architectures reposent sur une distinction relativement complexe d'un certain nombre de classes (trois pour IntServ, trois éventuellement subdivisées pour Diffserv, cinq pour FSA) qui reçoivent soit une priorité fixe, soit une part fixe ou ajustable de bande passante. DPS/SCORE propose plutôt un mécanisme permettant d'émuler la présence de certaines classes de service. Ces classes sont généralement utiles en cas de congestion, les moins prioritaires étant dégradées.

Pour le cœur de réseau, FAN ne fait aucune distinction de classe, mais permet la différenciation implicite des flots *streaming* et élastique, qui possèdent des critères différents de qualité de service. Les flots *streaming* sont servis en priorité et les autres utilisent la bande passante laissée disponible. En régime nominal, la distinction de plusieurs classes de service n'est pas utile. En surcharge, il semble ainsi préférable d'éliminer l'excédent de charge par un contrôle d'admission. De ce fait, la différenciation ne se fait pas par un lien plus ou moins transparent, mais par un taux d'accessibilité au lien. La congestion ne dégrade pas tous les flots mais touche uniquement ceux qui sont rejetés. Il est possible de définir plusieurs classes de service possédant des taux d'accessibilité différents.

Équité, isolation et flots non-réactifs

Aucune architecture n'assure l'équité de débit entre les flots, à l'exception de FAN qui intègre un ordonnancement *fair queueing*, et DPS/SCORE qui peut l'émuler[8]. Cette équité n'est d'ailleurs

8. Il n'est pas précisé s'il est possible d'émuler plusieurs mécanismes simultanément

pas forcément recherchée pour *Congestion Exposure*. Faute d'un tel ordonnancement, la gestion des flots non-réactifs est soit inexistante, soit nécessite des mécanismes complexes de traitement. Dans *Congestion Exposure*, cette gestion repose sur un mécanisme de "facturation" de la congestion générée à l'utilisateur.

Seules les architectures qui implémentent un contrôle d'admission peuvent garantir l'isolation des flots (IntServ, DiffServ+PCN, FSA et FAN). La performance des flots ne peut être réellement protégée que dans les architectures travaillant à la granularité du flot comme IntServ, FSA et FAN. Les autres reposent sur la coopération des utilisateurs en bout de réseau. Nous soulignons que le traitement des flots *streaming* est réalisé implicitement par FAN.

Contrôle de la surcharge

Si un contrôle de la surcharge est prévu, son implémentation n'est que conseillée et rarement explicitée : IntServ, DiffServ+PCN, FSA. On trouve cependant plusieurs propositions de contrôle d'admission et de préemption basées sur RSVP ou PCN.

FAN intègre un mécanisme de contrôle d'admission, local, implicite et basé sur des mesures. Son amélioration fait l'objet du Chapitre 6. Comme nous l'avons évoqué précédemment, il peut faire l'objet de mécanismes avancés de re-routage, et complété par des mécanismes de routage multi-chemin ou d'équilibrage de charge.

Discussion

Parmi les architectures que nous venons de présenter, peu implémentent l'ensemble des mécanismes nécessaires à garantir la performance des flots *streaming* et élastiques. Elles ne reposent généralement pas sur un modèle simple et robuste permettant de connaitre la performance réalisée par ces flots, et nécessaire au respect des contrats de services avec les utilisateurs.

Comme nous allons le voir dans le chapitre suivant, FAN repose sur des descripteurs simples du trafic et des flots (la charge moyenne et le débit crête). La performance est dictée par des modèles efficaces et robustes aux évolutions du trafic, qui permettent une performance prévisible, et fournissent une ligne directrice pour le dimensionnement des réseaux et de leurs ressources.

Enfin, parce qu'elle ne repose sur aucun marquage ni signalisation, FAN permet une implémentation simple, neutre du point de vue des utilisateurs et des applications, ne nécessitant aucune standardisation. Son déploiement peut être progressif, ne pose aucun obstacle à assurer la performance des flots dans l'inter-domaine. Elle offre des composants que nous jugeons nécessaires aux évolutions futures des réseaux, des protocoles et de leurs utilisations.