HAL
open science

Quelle architecture pour l'Internet du futur ?

Walid Dabbous

To cite this version:

Walid Dabbous. Quelle architecture pour l'Internet du futur ?. Réseaux et télécommunications [cs.NI]. Université de Nice Sophia Antipolis, 2008. tel-00406587

HAL Id: tel-00406587

https://theses.hal.science/tel-00406587v1

Submitted on 23 Jul 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITE DE NICE-SOPHIA ANTIPOLIS

Mémoire

présenté pour obtenir le diplôme d'

Habilitation à Diriger des Recherches

en sciences

Spécialité : Informatique

par

Walid Dabbous

Quelle architecture pour l'Internet du futur ?

Soutenance prévue le 4 février 2008 à l'INRIA Sophia Antipolis devant le jury composé de :

Michel Riveill	Professeur UNSA	Président
Guy Leduc	Professeur ULg	Rapporteur
Guy Pujolle	Professeur UPMC	Rapporteur
Jean-Jacques Pansiot	Professeur ULP	Rapporteur
Abdelmadjid Bouabdallah	Professeur UTC	Examinateur
Philippe Nain	DR INRIA	Référent

A *Nayla* pour tout

Remerciements et plus

Les travaux décrits dans ce mémoire sont le fruit d'un long travail de collaboration avec un grand nombre de personnes (doctorants, chercheurs et ingénieurs). Nous ne pourrons pas les citer tous nominativement mais nous leur adressons ici nos remerciements. La mention des « magiciens[1] » du projet « Rodéo » s'impose en premier (Christophe Diot, Jean-Chrysostome Bolot et surtout Christian Huitema). Nous avons vécu ensemble une aventure scientifique très enrichissante. Après le départ de Christian en 1996, les réunions du « triumvirat » étaient devenues le champ de discussions scientifiques, organisationnelles, humaines et amicales. Thierry Turletti et Claude Castelluccia font aussi partie de la « bande ». C'est avec eux que nous avions élaboré toutes ces contributions sur les architectures de protocoles hautes performances et sur les applications multimédia. Les autres membres de « Planète » (Chadi Barakat, Vincent Roca et Arnaud Legout) sont les collègues avec lesquels l'aventure continue, relevée par l'enthousiasme qu'ils apportent à l'équipe. Nous n'oublierons pas de mentionner Emmanuel Duros, Patrick Cipière et Luc Ottavj impliqués à fond dans udlr. Une très belle aventure avait été amorcée, aventure qu'ils ont poursuivie avec UDcast. Une mention particulière devra être faite à tous les doctorants avec lesquels nous avons beaucoup appris : Frank Lyonnet, Antoine Clerget, Fethi Filali, Rares Serban, Miguel Ruiz Sanchez, Laurentiu Barza, Fatma Louati, Mohamad Malli et Mohamed Ali Kaafar. Nous avons eu une relation scientifique et humaine basée sur la confiance mutuelle et la recherche de l'excellence. Thierry Parmentelat et Mathieu Lacage ont aussi eu leur part du gateau : Leurs contributions sur les simulateurs et les plates-formes expérimentales sont d'une importance capitale pour nos travaux futurs.

Nous remercions également les membres du jury d'habilitation : merci à Philippe Nain d'avoir accepté d'être notre scientifique de référence, à Michel Riveill d'avoir accepté de présider le jury de soutenance, à Guy Pujolle, à Guy Leduc et à Jean Jacques Pansiot d'avoir accepté de rapporter sur ce travail et à Abdelmadjid Bouabdallah d'avoir accepté de participer au jury.

Enfin, à titre personnel, je remercie mon épouse Nayla pour son support constant le long de ces quinze dernières années.

[1] Le terme est emprunté à un article de François Baccelli dans Code Source.

Résumé

Nous présentons dans la première partie de ce mémoire un aperçu général de nos contributions dans le domaine des réseaux. Le principe de bout en bout et le concept Application Level Framing ont été des éléments structurants de l'architecture des réseaux dans la dernière décenie du vingtième siècle. Nous avons contribué à l'étude approfondie de l'architecture des protocoles et des applications réseaux en focalisant en particulier sur les aspects de passage à l'échelle et de support de nouvelles fonctionalités par les applications. Nos travaux sur les architectures de protocoles hautes performances représentent le premier axe de recherche décrit dans ce mémoire et font l'objet du premier chapitre. Un nombre important de travaux de recherche portait sur le support de la qualité de service et/ou du multipoint dans l'Internet. Nous avons aussi contribué dans ce domaine en adoptant une approche critique qui consistait à investiguer sur le passage à l'échelle des mécanismes de support de la qualité de service et de diffusion multipoint. Ces travaux sont décrits dans le deuxième chapitre de ce mémoire. L'hétérogénéité des réseaux et en particulier celles des nouveaux supports de transmission a compliqué le schéma : des solutions proposées pour des réseaux filaires n'étaient plus adaptés à un environnement hétérogène intégrant des liens de débit et de taux d'erreur différents. Nos contributions dans ce domaine et en particulier sur le support du routage unidirectionnel font l'objet du troisième chapitre. L'ossification du réseau a poussé les chercheurs à plaider pour la prise en compte de propositions architecturales « de rupture » tenant compte des intérêts souvent divergents des différentes parties prenantes. La difficulté de déployer de telles propositions sur le réseau a amené les chercheurs à proposer des contributions au niveau « overlay » et à concevoir des plates-formes d'expérimentation permettant de valider « sur le terrain » les contributions proposées. Nos activités actuelles sont centrées autour de cette thématique que nous décrivons dans le chapitre quatre. Dans la deuxième partie du mémoire nous présentons en détail trois contributions sélectionnées de façon à couvrir les différents domaines auxquels nous nous sommes intéressés dans les quinze dernières années.

Abstract

In the first part of this thesis, we present a *general* overview of our contributions in networking. The end to end principle and the Application Level Framing concept had major impact on network architecture in the last decade of the twentieth century. We studied protocols and applications architecture focusing on both scalability and new services aspects. Our activities on high performance protocol architecture represent therefore an important research direction and are described in the first chapter of this thesis. Many researchers also addressed multicast and quality of service support on the Internet. We also worked on this topic with a specific approach: we "believed" in multicast but not in QoS. Unfortunately neither one finally was deployed in a large scale on the Internet. Our contributions in this domain are described in chapter two. The network and in particular link heterogeneity is also an important factor: the solutions that were proposed to wired (and reliable) networks were not adequate for heterogeneous environments integrating links with different characteristics. Our contributions in this domain and in particular on unidirectional link routing are presented in the third chapter. The ossification of the Internet pushed many researchers to claim that "disruptive" architectural ideas should be considered. These new architectural considerations should take into account the interest of the different stakeholders in the future Internet. The difficulty to immediately deploy such disruptive ideas resulted in contributions in "overlay" layer. In parallel, work on the definition of a global environment to foster and evaluate networking innovations is being developed and deployed. Our activities in this domain are described in the fourth chapter of this thesis. The second part of this thesis contains the detailed presentation of three selected contributions covering the different topics of interest to us during the last fifteen years.

Tables des matières

Première Partie 7

Chapitre 1 Architecture de protocoles hautes performances 8

Chapitre 2 Les nouveaux services de l'Internet 14

Chapitre 3 Les nouveaux supports de transmission 25

Chapitre 4 Evolution incrémentale ou nouvelle architecture : Quelle solution à l'hétérogénéité des réseaux ? 32

Deuxième Partie 39

Chapitre 5 Génération Automatique d'Implantations Optimisées de Protocoles 40

Chapitre 6 Contrôle de Congestion en milieu Hétérogène 64

Chapitre 7 La Sécurité des Systèmes de Coordonnées Internet 90

Références bibliographiques 110

Liste de publications 115

Annexe 1 CV détaillé 123

Première partie

Présentation générale des travaux de recherche

Chapitre 1 **Architecture de protocoles hautes performances**

Chapitre 2 **Les nouveaux services de l'Internet**

Chapitre 3 **Les nouveaux supports de transmission**

Chapitre 4 **Evolution incrémentale ou nouvelle architecture : Quelle solution à l'hétérogénéité des réseaux ?**

Chapitre 1

Architecture de protocoles hautes performances

L'histoire de l'Internet commence avec le démarrage en 1969, de recherches visant à relier entre eux des ordinateurs dans différents centres de recherche en mettant en place un système de transmission permettant à un terminal unique d'avoir accès aux ordinateurs distants. Ce réseau de transmission, appelé Arpanet a vu le jour à l'Université de Californie à Los Angeles (UCLA) et reliait au début seulement trois ordinateurs. Les premiers essais en « grandeur nature » impliquant une quinzaine d'ordinateurs à UCLA, SRI, MIT, Harvard, etc., eurent lieu en 1971. Le travail sur les réseaux en France a démarré à cette époque par la mise en œuvre du réseau « Cyclades » [1.1]. Ce réseau avait adopté la technologie de transmission de données par datagramme (c'est-à-dire sans connexion réseau) similaire à celle de l'Arpanet mais il n'était pas relié à l'Arpanet.

Dès 1972, un groupe de travail (INWG) a été mis en place afin d'étudier une architecture permettant l'interconnexion des réseaux. Et en 1973, Vint Cerf et Bob Kahn inventèrent le concept d'Internet. L'idée était d'interconnecter les différents réseaux par des passerelles et de relayer les messages de réseau à réseau. Le protocole utilisé par les passerelles fut appelé le protocole IP (Internet Protocol). La première version du protocole IP fut publiée en 1978, mais la version devenue standard (version 4) a été achevée en 1981. L'utilisation du protocole IP permettant d'interconnecter des réseaux auparavant isolés, le développement de technologies de réseaux locaux rapides et peu onéreuses (réseaux Ethernet), et le développement d'applications multiples (courrier électronique, transfert de fichiers distants, etc.), ont rapidement rendu l'utilisation des réseaux « intéressante » puis « indispensable ». Ainsi, plus de 1000 ordinateurs étaient déjà raccordés à l'Arpanet en 1984.

En 1986, la NSF a mis en place un nouveau réseau, le NSFnet, qui agissait comme une épine dorsale (backbone) couvrant les États-Unis et reliant entre eux les différents réseaux déjà existants. Le débit auquel les messages pouvaient être envoyés sur ce réseau était de 56 000 bits par seconde (56 kb/s). Ce débit paraissait considérable à l'époque, bien qu'il soit nettement inférieur au débit d'accès disponible sur n'importe quel PC récent (sachant en plus que ce débit était partagé par tous les utilisateurs du réseau). Il a fallu attendre le 28 juillet 1988 pour que la France soit raccordée au NSFnet, par une liaison transatlantique mise en place par l'équipe « réseaux » de l'INRIA Sophia Antipolis, dirigée par Christian Huitema. L'Internet, qui est donc l'ensemble des réseaux connectés entre eux par le protocole IP, avait entre temps continué sa croissance exponentielle avec 10000 ordinateurs reliés en 1987. Cette croissance continue jusqu'à aujourd'hui : 2,5 millions d'ordinateurs étaient connectés en 1994, 20 millions en 1997 et 490 millions en juillet 2007. Étant donné le rythme de progression actuel, il est possible que l'Internet relie tous les ordinateurs du monde dont le nombre pourra dépasser la population mondiale (plus d'un ordinateur par personne) dans un futur proche.

Dans les années quatre-vingt, l'Europe était préoccupée par la standardisation du modèle ISO (Interconnexion de Systèmes Ouverts) de l'OSI (Organisation de Standardisation

Internationale). Malgré les longues discussions des années soixante-dix[1], les experts de l'OSI ne sont pas parvenus à trancher concernant un point fondamental : comment interconnecter les réseaux d'ordinateurs de façon globale. Deux services réseaux ont donc été « standardisés » : le service réseau orienté connexion (CONS) et le service réseau sans connexion (CLNS). Pour pallier le manque de fiabilité et le coût élevé du service orienté connexion (basé sur les connexions X.25), l'OSI a travaillé sur la standardisation de plusieurs protocoles de transport (TP0 à TP3). Un effort moins important a été fourni pour développer un protocole de transport adapté à un service réseau sans connexion (il s'agit du protocole TP4). Il a donc été envisagé d'interconnecter les deux « architectures » fournissant des services différents. Les efforts consistant à proposer des « convertisseurs de service » et de « protocoles » [R.93] n'ont pas permis de « sauver » le modèle orienté connexion et il est apparu clairement lors d'ateliers de travail regroupant des chercheurs des deux côtés de l'atlantique au début des années quatre-vingt-dix que la cohabitation entre l'Internet et le modèle orienté connexion de l'OSI était quasiment impossible. L'Europe avait donc un retard important à combler en ce qui concerne la conception et le développement de protocoles adaptés aux réseaux datagramme. Le projet OSI95 [1.2], démarré en 1990, était une tentative très ambitieuse visant à proposer une nouvelle mouture des protocoles ISO en 1995[2]. Plusieurs lacunes avaient été identifiées dans ce modèle : l'absence de mécanisme de contrôle de congestion adéquat au niveau transport, l'absence de support pour la communication de groupe (ou multipoint), les mauvaises performances des implantations des couches ISO transport et présentation. Ces défauts plombaient les protocoles ISO et empêchaient le support efficace d'applications multimédia sur des réseaux à haut débit, alors que du côté de l'Internet, Van Jacobson et Steve Deering avaient enrichi TCP et IP avec les algorithmes de contrôle de congestion et de diffusion multipoint.

Nous avons travaillé sur les algorithmes de contrôle de congestion pour le protocole TPX[3] étudié dans le cadre du projet OSI95. Avec l'augmentation du débit des liens, l'utilisation efficace des ressources du réseau passe par la possibilité des sources de « remplir le tuyau » (en opérant un contrôle de flux basé sur une valeur W suffisamment grande de la fenêtre). Pour éviter la congestion qui pourrait provenir d'une concentration du trafic et s'aggraver suite aux retransmissions de paquets de la part des sources, un algorithme de contrôle de congestion basé sur un démarrage lent a été proposé dans [1.3]. Selon cet algorithme, une source commence par une fenêtre minimale (1 paquet) et augmente la valeur de la fenêtre à la réception d'un accusé de réception (Ack) de données nouvelles et ce, jusqu'au dépassement d'un certain seuil au-delà duquel la source passe en mode « découverte de libération de ressources » en augmentant la fenêtre (ayant une valeur W) à la réception d'un accusé de réception par $1/W$. L'idée derrière cet algorithme est la suivante : afin de découvrir le niveau de charge adéquat la source augmente graduellement la valeur de la fenêtre (certes de façon exponentielle tous les RTT car la fenêtre est incrémentée à chaque réception de Ack, mais non plus directement de 0 à W_{max}, d'où le nom slow start de cet algorithme). Le seuil est supposé représenter le point opératoire, qui varie continuellement avec la charge du réseau. La phase dite linéaire sert donc à vérifier, en augmentant la fenêtre d'une façon moins agressive, la disponibilité de ressources libérées. Lors d'une perte de paquet soupçonnée par l'expiration de la temporisation de retransmission, on redémarre avec une valeur de la fenêtre égale à 1

[1] Voir http://www.en.wikipedia.org/wiki/CYCLADES

[2] La « première phase » du projet (90-92) était supposée être suivie par une deuxième phase de trois ans (92-95) mais le projet a été arrêté en 1992.

[3] A ne pas confondre avec XTP. TPX était conçu pour fournir le même service que TP4, mais en étant plus efficace à haut débit (TPY étant un autre protocole fournissant un service adapté aux applications multimédia). Pour plus détails voir le livre édité par André Danthine sur le projet OSI95 [1.2].

paquet. Plusieurs améliorations ont été apportées à cet algorithme, en respectant sa philosophie de base : les sources découvrent le point opératoire de façon « aveugle » en se basant sur l'information provenant des Acks considérés comme étant les battements de cœur du réseau et fournissant une auto-synchronisation permettant un fonctionnement à la fois efficace et robuste. Le signal de congestion (perte soupçonnée d'un paquet) ne permettait pas d'apprécier finement la gravité de la congestion et aucun contrôle de débit n'était effectué. Après une étude approfondie par simulation, nous avions montré que le débit en dent de scie d'une connexion utilisant le slow start n'était pas adéquat au support des applications multimédia nécessitant un débit plutôt stable (sur un intervalle de temps donné). Nous avions donc proposé d'enrichir le signal de congestion par une information sur le gradient du délai afin de prendre en compte la gravité de la congestion dans l'estimation du point opératoire [R.50]. L'utilisation d'un mécanisme de contrôle de congestion basé sur les délais a permis d'éviter les pertes de paquets « provoquées » par l'algorithme slow start et surtout de réduire la variabilité des délais, ce qui rend ce mécanisme plus adéquat pour les applications multimédia. Nous nous sommes basés sur ces travaux ultérieurement pour la conception de mécanismes de contrôle de flux applicatifs pour la vidéo conférence dans lesquels l'application ajuste le débit de codage en fonction des délais aller-retour des paquets sur le réseau. Il est à noter que l'utilisation d'information sur le délai pour l'ajustement des paramètres de la fenêtre a été finalement retenue dans TCP Westwood[1] [1.4]. Ceci permet un fonctionnement plus efficace dans le cas de réseaux haut débit hétérogènes.

En plus de la proposition de mécanismes de contrôle de congestion, il fallait procéder de façon plus globale à la définition d'un *service* de transport et d'un *protocole* de transport standards pour les réseaux haut débit. Le protocole XTP a donc été étudié afin d'évaluer son adéquation en tant que « futur » protocole de transport haut débit de l'OSI. A l'origine, le protocole XTP était conçu pour une nouvelle architecture matérielle pour l'implantation de protocoles [1.5]. L'idée d'une implantation câblée a été abandonnée par la suite, mais les choix de conception du protocole méritaient une analyse approfondie. Le protocole proposait une simplification des mécanismes de contrôle d'erreurs motivée par l'augmentation des débits et l'amélioration de la qualité des liaisons[2], une intégration des couches réseau et transport en une seule couche dite de transfert (signifiant que les routeurs intermédiaires devraient participer aux fonctions de contrôle de flux, d'erreurs et de cadence), un alignement des champs des paquets sur les frontières de mots de 32 ou 64 bits, l'ajout d'une terminaison de paquet contenant le total de contrôle, etc. Une implantation logicielle, donc flexible, du protocole permettait d'étudier facilement l'impact de ces améliorations. Nous avons réalisé une implantation efficace du protocole XTP au niveau utilisateur sous Unix [R.81]. Même si nous n'étions pas d'accord avec le concept de couche de transfert, nous voulions évaluer en particulier l'efficacité du calcul du total de contrôle (checksum) basé sur des OU exclusifs (XOR) proposé par les concepteurs de XTP. Nos travaux ont montré qu'on pouvait obtenir des débits très élevés en intégrant les optimisations adéquates dans l'implantation logicielle et que la réalisation dans une machine de protocole matérielle n'était pas une condition de l'efficacité du protocole XTP. Cependant, en ce qui concerne la définition d'un service de transport standard pour le haut débit, les choses n'étaient pas très positives en faveur de XTP. Ce protocole consistait en une boîte à outils, programmable par l'application et permettant de réaliser une couche transport correspondant aux besoins des applications. La difficulté de

[1] Contrairement à l'épisode TCP Vegas considéré comme « contraire à la philosophie de base de TCP » car Vegas est basé sur un contrôle de débit (appelé aussi contrôle de cadence).

[2] Les concepteurs de XTP n'avaient pas anticipé la prolifération des liens sans fil dans l'Internet.

définir les besoins des applications et le succès de TCP intégrant le contrôle de congestion ont abouti à écarter le protocole XTP de la scène.

Le fonctionnement des protocoles de transport à haut débit a fait l'objet d'un grand nombre d'études. Cependant, le goulot d'étranglement dans la transmission et le traitement des données provenait surtout de la couche « présentation ». Les « méthodes d'accès » nécessitaient des accès et des manipulations des données assez coûteux en temps mémoire et en calcul. En même temps (au début des années 90), les performances des stations de travail avaient augmenté avec l'apparition des architectures à jeu d'instructions réduit (RISC), mais pas à la même cadence que les capacités des liens du réseau. L'accès mémoire RAM est devenu coûteux par rapport à l'accès à la mémoire cache et aux registres. Outre les fonctions de « contrôle », le traitement d'un protocole renferme aussi des fonctions orientées « données » dans lesquelles les données sont « lues » en mémoire, traitées ou « manipulées » et éventuellement « sauvegardées » de nouveau en mémoire (par exemple le codage de présentation, le calcul du total de contrôle au niveau transport, le chiffrement, la compression). Nos propres observations, confirmées par des mesures de performances publiées par ailleurs, ont montré qu'il était possible d'obtenir des gains de performance en intégrant les opérations de manipulation de données afin de minimiser les accès mémoires. Ceci dit, la complexité de certaines opérations de traitement de données hautement coûteuses (en particulier le codage de présentation) nous avait poussés à chercher à optimiser l'implantation de ces mécanismes de codage dans le compilateur ASN.1 Mavros[1]. L'optimisation des mécanismes de codage de présentation avait comme objectif d'améliorer les performances de toute la « pile protocolaire OSI » de façon à pouvoir transmettre les données en une seule opération d'envoi (ce qui revient à faire fonctionner les couches présentation et transport en particulier de façon intégrée). Nos travaux publiés dans [R.49] ont montré qu'il était possible de réduire considérablement le coût des mécanismes de présentation de données, ce qui rendait une implantation intégrée de toutes les couches protocolaires envisageables. Ceci dit, nous avions abouti aussi à une conclusion alarmante lors de cette étude : l'architecture matérielle des machines avait un impact non négligeable sur les performances et devrait être prise en compte pour la conception des protocoles et même pour la définition de l'architecture de protocole globale.

En effet, le modèle de référence ISO de l'OSI standardisé dans les années quatre-vingt, était conçu une décennie auparavant. L'organisation en couches reflétait donc l'architecture matérielle des années soixante-dix : le découpage (parfois) artificiel en couches répondait aux besoins de séparation des fonctionnalités entre les différents constructeurs des solutions matérielles de l'époque. Les notions même de service et de protocole découlent historiquement du modèle présentant les interfaces entre les différents fournisseurs de services. Il devrait y avoir un fournisseur pour l'accès au lien de transmission, un autre pour le routage dans le réseau, un troisième pour le contrôle de transmission et un quatrième pour le contrôleur de canal, etc., chacun de ces services pouvant être fourni par des constructeurs différents[2] : la couche physique et liaison correspondaient donc aux mécanismes mis en œuvre dans le modem et le coupleur (ou carte d'accès réseau). Les couches réseau et transport étaient généralement implantés dans une machine frontale, alors que la couche session correspondait au contrôleur du dialogue sur le canal entre l'ordinateur hébergeant l'application et le frontal. Les « méthodes d'accès » pour la présentation des données étaient

[1] Mavros est un des compilateurs ASN.1 ouverts les plus performants, sinon le plus performant au monde. L'INRIA reçoit jusqu'à aujourd'hui des demandes d'accès au code source pour des besoins de recherche. Ceci n'est pas étonnant quand on sait qui en est l'auteur…
[2] Pour respecter le O dans ISO.

intégrées dans l'application (comme par exemple X.409). Avec l'avènement des stations de travail au début des années quatre-vingt-dix, l'architecture des implémentations a changé surtout en ce qui concerne les couches dites « hautes » (transport et au-dessus). La couche transport est habituellement implantée dans le système d'exploitation et est accessible aux applications à travers l'interface socket. L'application réside sur la même machine au niveau utilisateur et intègre les mécanismes de présentation des données. La couche session n'a plus de place a priori dans ce schéma : les besoins de synchronisation des transferts de données sont connus et pourront être mieux maîtrisés par l'application elle-même. Il est plus convenable de laisser l'application piloter le déroulement des échanges de données au lieu de déléguer cette tâche de synchronisation à la « couche session ». D'autant plus que la localisation de cette couche entre transport et présentation impose des échanges des données asynchrones entre ces couches (ce qui empêcherait l'intégration des fonctions de manipulation de données souhaitée telle que cela est mentionné ci-dessus). Ceci aboutirait à une dégradation des performances à causes des copies multiples des données avant l'envoi sur le réseau.

Même dans le cas d'une mise en œuvre intégrée du concept des couches, on pourrait soupçonner les mécanismes de multiplexage et de segmentation de causer un fonctionnement inefficace. En effet, ces opérations cachent aux applications des informations importantes sur les couches basses et inversement, ce qui empêche d'optimiser les transferts. Par exemple, si l'application participe au contrôle de transmission des données, on pourrait utiliser une réponse à une requête pour remplacer un accusé de réception transport (comme c'est le cas dans le protocole VMTP). Cette approche permet d'éviter que des « mauvaises décisions » soient prises par le protocole de transport ou par l'application (comme par exemple fermer la fenêtre de transmission suite à la perte d'un paquet lors d'un transfert vidéo). Les deux concepts clés pour améliorer les performances sont donc la réduction de l'asynchronisme et l'implication des applications dans le contrôle de transmission. Par ailleurs, Clark et Tennenhouse avaient observé dans leur fameux papier Sigcomm [1.6] que l'amélioration des performances des systèmes de communication nécessite d'utiliser au mieux le maillon le plus lent de la chaîne. Les fonctions de manipulation de données (codage/décodage de présentation, chiffrement, etc.) représentent ce goulot d'étranglement. Comme la couche transport peut retarder considérablement la remise des données reçues « hors séquence » à l'application (en attendant « de boucher les trous »), il semble extrêmement utile que l'application puisse traiter ces données reçues hors séquence sans avoir à attendre les données retransmises.

La mise en trame par l'application ou (Application Level Framing) a donc été proposée comme principe clé pour la ré-architecture des protocoles. ALF est en fait le résultat naturel de plusieurs expérimentations réseau qui ont montré le besoin d'adopter les règles suivantes : (1) une transmission efficace ne peut être réalisée que si l'unité de contrôle d'erreur (la plus petite unité de retransmission après une perte) est égale à l'unité de transmission (la plus petite unité de données transmise sur le réseau) qui peut donc être perdue. Cette égalité n'est pas satisfaite en cas de support de protocoles de transport au dessus d'IP sur ATM. En effet, l'unité de contrôle d'erreur de bout en bout (le paquet IP) n'est pas égale à l'unité de transmission (la cellule ATM). Les pénalités évidentes résultant de la violation de cette règle sont des retransmissions inutilement importantes en cas de pertes et une utilisation inefficace de la mémoire. (2) l'unité de transmission devrait aussi être égale à l'unité de « traitement » de données par l'application. Sinon, les données reçues hors séquence pourraient s'accumuler dans les files d'attente ce qui revient à ralentir l'application. (3) l'expérience avec les services multimédia a montré qu'il est plus facile de développer des applications multimédia

adaptatives si l'unité de traitement est égale à l'unité de contrôle de transmission. Ne pas respecter cette règle revient à établir une séparation entre l'application et le contrôle de transmission qui pourrait résulter en de mauvaises décisions prises par la couche transport. Ces règles se résument en trois « Non » : non à la segmentation, non au multiplexage et non à la séparation en couches. Selon le principe ALF, l'application envoie ses données sous forme de trames autonomes (dites ADU ou Application Data Unit) ayant une signification pour l'application. Il est souhaité que les couches présentation et transport maintiennent les frontières de trames lors du traitement des données. Ceci est conforme à l'idée communément admise qui consiste à dire que les unités de données devraient être multiplexées à un seul niveau de la pile protocolaire [1.7]. Les entités émettrice et réceptrice de l'application définissent le contenu d'une ADU de façon à pouvoir traiter les ADU reçues hors séquence. L'ADU sera donc l'unité de manipulation des données afin de simplifier les traitements. Par exemple, un serveur d'image enverra des messages correspondants à des parties bien identifiées de l'image. Quand le client reçoit une telle trame, il pourra la traiter immédiatement en décompressant les données et en affichant les pixels correspondant à la partie reçue sur l'écran, ce qui permet de réduire le temps global de transmission. La taille de l'ADU devrait être inférieure à tous les MTUs (Maximum Transmission Unit) des sous réseaux traversés de façon à éviter complètement la segmentation.

ALF a été proposé en 1990. Ce concept a fait l'objet d'études approfondies par la communauté réseaux, en particulier dans le cadre du projet Hipparch [1.8]. La plupart des applications multimédia (audio et vidéo conférences, tableau blanc partagé, environnement virtuel à grande échelle, etc.) est basée sur ce concept. Les études et expérimentations réalisées ont montré l'intérêt de ce concept pour la mise en œuvre des applications réseaux. Ceci dit, la mise en œuvre de ce concept pour des applications différentes nécessite qu'on puisse générer pour chaque application une implémentation intégrée des mécanismes de protocoles fournissant le service requis par l'application. Une approche automatisée pour la génération du code du protocole adapté est donc très utile pour faciliter l'adoption de cette approche. Afin d'automatiser ce processus, il est nécessaire d'avoir un langage de spécification qui permettra de décrire les besoins de l'application et un compilateur de protocole permettant de générer un code efficace à partir des spécifications. Le projet Hipparch II avait pour objectif la réalisation d'un tel compilateur de protocole. Nous avons participé aux travaux sur le compilateur de protocole par la conception et la réalisation d'un optimiseur du code généré automatiquement à partir de spécifications formelles en Esterel. Ce travail publié dans Sigcomm'96 sera présenté en détail et fera l'objet chapitre 5.

Les résultats de ces travaux ont influencé l'état de l'art en ce qui concerne les architectures de protocoles hautes performances de façon fondamentale : après le « rêve » de l'OSI on s'est réveillé en réalisant que l'architecture matérielle avait un impact sur le modèle conceptuel définissant les fonctions protocolaires. Ceci est à l'encontre de l'idée même des systèmes « ouverts ». La complexité des applications et l'hétérogénéité des réseaux plaident pourtant pour une approche « standard ». Cette problématique est de nouveau d'actualité en ce qui concerne l'Internet du futur : avec la grande mobilité des utilisateurs, la connectivité ne peut plus être assurée de façon permanente (pour des raisons techniques ou même économiques) par la couche « inter-réseau ». Le modèle de transmission devrait passer alors de « store and forward » à « store, carry and forward ». Cette influence des réseaux sous-jacents sur l'architecture protocolaire s'avère donc être un élément structurel dans la conception des protocoles réseaux.

Chapitre 2

Les nouveaux services de l'Internet

Le principe « de bout-en-bout » est un des principes fondamentaux qui étaient à la base de l'architecture de l'Internet et ont guidé son évolution. Il stipule que le réseau doit être le plus simple possible et ne doit s'occuper que de l'acheminement de datagrammes, et que toutes les autres procédures liées à la communication doivent être réalisées à l'extérieur du réseau. Le réseau est encore plus simple s'il effectue un acheminement « au mieux » (« best effort »), c'est-à-dire sans garantir que les paquets atteignent leur destination. Dans ce cas, il n'y a pas besoin d'établir à l'avance un itinéraire, ni de réserver un circuit tout au long de la communication comme cela se fait dans un réseau téléphonique. Bien sûr, certains paquets peuvent être perdus lorsque le réseau est congestionné, c'est-à-dire quand ses ressources sont insuffisantes pour traiter l'ensemble des paquets en transit à un moment donné. C'est alors aux extrémités de déceler la congestion, et si nécessaire de retransmettre les paquets perdus. Cette approche permet une meilleure tolérance aux pannes au niveau inter-réseau, tout comme la simplification des routeurs IP permet l'interconnexion de différentes technologies de réseaux.

L'Internet a donc été conçu au départ pour offrir un service simple (connecter tous les ordinateurs du monde) de la façon la plus économique possible. Il n'a pas été conçu pour un type d'applications particulier. Ceci dit, il a bien été utilisé dans un premier temps par un petit nombre d'applications spécifiques (courrier électronique, transfert de fichiers, accès Web, etc.). Cependant, l'augmentation rapide des capacités des ordinateurs a fait que ceux-ci sont devenus capables au début des années quatre-vingt-dix de coder et de traiter des sons ou de la voix, ainsi que des images fixes ou de la vidéo. Il était donc naturel de penser à transmettre ces nouveaux types de données sur l'Internet. Les avantages potentiels sont en effet très nombreux, puisque la transmission de données multimédia de qualité permettrait d'offrir des services de téléphonie, de jeux distribués, ou de collaboration à distance combinant vidéoconférence et tableau blanc partagé (c'est-à-dire une espace partagé visible et modifiable par tous les utilisateurs) qui font intervenir la transmission de la voix, d'images animées, et de texte.

La transmission de données multimédia sur l'Internet n'est pas une simple extrapolation de la transmission de données textuelles, car les besoins des applications multimédia sont différents de ceux des applications classiques de transfert de données. Ces dernières applications ne font en général intervenir que deux utilisateurs, une source et une destination. D'autre part, leur utilité pour un utilisateur ne dépend pas, ou peu, du temps qu'il a fallu pour transférer les données. Au contraire, les applications multimédia mettent généralement en jeu un nombre important d'utilisateurs (e.g. l'enseignement à distance, la télédiffusion, ou les environnements virtuels partagés). De plus, il est très important que les données audio ou vidéo émises par un utilisateur atteignent rapidement les autres utilisateurs. Donc, d'une part les applications multimédia interactives ont besoin d'une transmission efficace de groupe (entre plusieurs utilisateurs), et d'autre part elles sont sensibles à la qualité du service fournie par le réseau.

Steve Deering avait défini en 1991 un service de diffusion multipoint pour l'Internet : un « groupe » multipoint est défini par un seul identificateur de groupe, que l'on appelle adresse multipoint. Une source désirant envoyer des données aux membres du groupe envoie ses données à l'adresse multipoint du groupe. Les utilisateurs qui désirent participer aux activités d'un groupe s'abonnent à l'adresse multipoint du groupe. L'acheminement efficace des paquets d'une source vers toutes les destinations d'un groupe est calculé et effectué par certains routeurs de l'Internet en utilisant des algorithmes de routage multipoint. Le MBone[1] (Multicast backBone, ou épine dorsale multipoint de l'Internet) est un réseau de recouvrement qui contient l'ensemble des routeurs permettant une telle transmission. En cas de déploiement universel du multipoint[2], tous les routeurs de l'Internet devront effectuer du routage multipoint et le MBone sera identique à l'Internet. La notion d'adresse multipoint est importante car elle permet aux sources de ne pas avoir à connaître les destinations (les membres d'un groupe) avant de pouvoir leur envoyer des données. Ceci est très utile par exemple pour la diffusion de type télévision sur Internet. De plus, il n'est pas une condition que les destinations connaissent l'identité des sources avant de pouvoir recevoir des données de ces sources. Ces propriétés rendent le processus de transmission multipoint indépendant de la taille du groupe, et permettent de s'affranchir de la mise en place de connexions ou circuits entre les participants (ce qui n'est pas le cas dans une audio ou vidéoconférence sur réseau orienté connexion). Ce point est très important, car il permet aux techniques développées sur Internet de fonctionner aussi bien pour une audioconférence entre un petit nombre d'utilisateurs que pour la retransmission d'une émission d'une source vers un grand nombre d'utilisateurs (comme pour une diffusion TV). Vu l'importance du sujet, nous avons publié une étude bibliographique sur les algorithmes et protocoles de routage multipoint [R.9]. Cette étude a eu un large impact dans la communauté structurant ainsi un domaine assez vaste. Nous aborderons dans la section suivante les évolutions du routage multipoint et les raisons de l'absence du déploiement universel de cette technique.

La deuxième caractéristique qui distingue les applications multimédia réside dans le fait que la « qualité » perçue par l'utilisateur dépend de façon importante du service fourni par le réseau. Comme l'Internet fournit un service d'acheminement sans garantie aucune sur les performances dû au fait que les ressources du réseau (comme les liens de transmission) sont partagées par tous les utilisateurs, le support des applications multimédia sur l'Internet représente donc un problème non trivial. Deux approches sont alors possibles. Il est possible de les résumer sous la forme « les applications s'adaptent au réseau » ou « le réseau s'adapte aux applications ». La première approche consiste à dire que l'Internet fournit un service qui n'est pas adapté aux besoins des applications multimédia, et donc qu'il faudrait modifier ce service avant de pouvoir utiliser ce type d'applications. En particulier, il s'agit de modifier le réseau pour offrir un ou des services offrant des garanties de qualité ou du moins une qualité améliorée. Cette approche nécessite la mise en place dans le réseau de nouveaux mécanismes sur lesquels nous avons travaillé et qui seront décrits plus loin dans la section 2.2.

2.1 Applications Adaptatives

[1] L'INRIA Sophia était le premier site français connecté au MBone en novembre 1992 et ce, à travers un tunnel entre jerry.inria.fr et mbone.cit.cornell.edu. Nous avions géré l'établissement des tunnels pour d'autres sites en France (Paris, Rennes, vous apprécierez le routage triangulaire) jusqu'à ce que le FMbone soit mis en place fin 1993.

[2] Déploiement qui n'est toujours pas réalisé à l'échelle mondiale.

La deuxième approche consiste à dire que le service « au mieux » offert par l'Internet, même s'il n'est apparemment pas idéal pour les applications multimédia, est de toute façon le seul service disponible au niveau de l'inter-réseau. Il s'agit donc de minimiser l'impact négatif de ce service sur la qualité des données multimédia reçues par les destinations. Ceci est fait en pratiqué par l'utilisation de mécanismes de contrôle. Trois caractéristiques du réseau ont un impact important sur la qualité, à savoir le délai, la bande passante disponible, et les pertes. A chacune de ces caractéristiques sera donc associé un mécanisme de contrôle.

Considérons d'abord les mécanismes de contrôle de délai. Le délai mis par un paquet d'une conversation audio pour traverser le réseau dépend du nombre et de la taille des paquets envoyés par les autres utilisateurs qui partagent les mêmes ressources (liens de transmission, tampons mémoire dans les routeurs). Le réseau ne favorise pas certains paquets plutôt que d'autres. Il est donc impossible à cette connexion audio d'avoir un impact direct sur le délai qui sera mis par ses paquets pour traverser le réseau, à moins de changer la façon dont sont traités les paquets dans le réseau. Par contre, il est facile de contrôler les variations de délai en ajoutant un tampon mémoire à chaque destination. Un paquet arrivant à la destination est mis en attente provisoire dans le tampon, ce qui lui permet d'attendre les paquets suivants en retard, afin qu'ils soient rejoués de façon synchrone.

Les mécanismes de contrôle de débit cherchent à faire en sorte que le débit émis par un utilisateur (une source de données) soit égal à la bande passante disponible (ou plus généralement aux ressources disponibles) dans le réseau pour cet utilisateur. Il faut pouvoir d'abord estimer cette bande passante et ensuite ajuster le débit de la source en fonction de l'estimation. En pratique il est difficile d'estimer directement une bande passante, et on estime plutôt l'état plus ou moins congestionné du réseau en observant le nombre de paquets perdus aux destinations. Plus ce nombre est grand, plus le réseau est chargé, plus faible est la capacité disponible, et plus faible doit donc être le débit vidéo. La modification du débit vidéo se fait en contrôlant soit le nombre d'images émises par seconde, soit la qualité visuelle de chaque image transmise. On obtient donc un mécanisme basé sur une boucle de contre-réaction : à chaque instant, la source vidéo essaye d'envoyer le débit maximum (c'est-à-dire la meilleure qualité d'image possible) étant donné l'état plus ou moins congestionné du réseau. On dit alors que l'application vidéo est adaptative. La mise en œuvre de mécanismes d'adaptation nécessite quand même que la source soit au courant des pertes observées par les destinations. Un problème additionnel se pose pourtant dans le cas d'une diffusion vidéo à plusieurs destinataires, à savoir quel débit la source devrait choisir. Une solution possible pourrait être d'adapter le débit de la source à la destination ayant le taux de perte le plus élevé. Ceci revient à dégrader la qualité pour toutes les destinations dès lors qu'une liaison vers une destination particulière est surchargée. Une autre solution est que la source code les données de façon hiérarchique (c'est-à-dire selon leur importance décroissante) et de faire en sorte que seule l'information importante soit transmise sur les liaisons surchargées.

Les mécanismes de contrôle de débit décrits ci-dessus ajustent le débit d'une source audio ou vidéo en fonction de l'état du réseau, et ils tendent à minimiser le nombre de paquets perdus. Mais ils n'empêchent pas toutes les pertes. Il faut donc un mécanisme supplémentaire de contrôle des pertes qui minimise l'impact visuel ou auditif des paquets perdus. Pour les applications (comme les éditeurs de texte ou les outils de travail collaboratif partagés entre plusieurs utilisateurs) qui ont besoin de recevoir tous les paquets émis par une source, il est nécessaire de retransmettre les paquets perdus jusqu'à ce qu'ils soient reçus convenablement. Pour les applications (comme les applications audio ou vidéo) qui peuvent tolérer un certain taux de perte, il est suffisant d'envoyer de l'information redondante qui permettra de

reconstruire les paquets perdus sans qu'ils aient besoin d'être retransmis (ce qui peut prendre beaucoup de temps et donc nuire à l'interactivité). On peut par exemple inclure dans chaque paquet de l'information (donc redondante) sur le paquet précédent. En cas de perte du $n^{\text{ième}}$ paquet, l'information de redondance sur ce paquet qui se trouve dans le $(n+1)^{\text{ième}}$ paquet permettra, lors de la réception de ce $(n+1)^{\text{ième}}$ paquet, de reconstruire le paquet n. On peut donc de cette façon corriger les pertes isolées de paquets. On peut étendre ce mécanisme pour corriger la perte de deux paquets consécutifs en ajoutant dans le paquet n de l'information sur le paquet n-2.

Nous avons travaillé sur le support des applications multimédia sur l'Internet dans le cadre des projets Mice[1], Merci[2] et Meccano[3]. Nous citons ci-après avec un peu plus de détail deux contributions : la première sur le support des applications de vidéoconférence sur l'Internet avec des liens sans fil et la deuxième sur le support des environnements virtuels à grande échelle avec un service multipoint de type SSM.

Dans le cadre d'une collaboration avec le LEP et avec NEC nous avons travaillé sur l'intégration de mécanismes d'adaptation aux liens sans fil dans l'application RendezVous développée à l'INRIA Sophia Antipolis [2.1]. Dans un environnement sans fil, le taux d'erreur de bit sur les liens n'est pas négligeable. On pourrait considérer la protection totale au niveau de la couche liaison par l'ajout de données de redondance, mais cette solution est souvent assez coûteuse en bande passante. L'idée de gérer les erreurs au niveau applicatif paraît donc comme une alternative assez intéressante dans le cas d'une application de vidéo conférence. Nous pourrions en fait exploiter la nature du signal transmis afin de concevoir des mécanismes de redondance adaptés aux liens sans fil. Ce « paramètre » a un impact sur les mécanismes de codage vidéo, sur le contrôle de congestion (il faudrait pouvoir distinguer les erreurs dues à la corruption de données des erreurs dues à la congestion) ainsi que sur l'architecture de la solution globale de bout en bout. En ce qui concerne le codage vidéo, les applications de vidéo conférence « classique » sur Internet telle que IVS (Inria Video conference System) proposaient de transmettre sur le réseau des unités pouvant être traitées par l'application de façon indépendante et ce afin d'être robuste vis-à-vis des pertes de paquets (selon le concept ALF). Dans une application basée sur H.261 cela revient à organiser les groupes de blocs de façon à ce qu'ils appartiennent à un même paquet transmis sur le réseau. Se contenter de cela dans un environnement sans fil n'est pas suffisant, une erreur de bit dans un paquet provoque la perte de toutes les données dans ce paquet, ce qui dégradera les performances de l'application de vidéo conférence de façon considérable. Il faudrait effectuer le mécanisme de découpage en unités autonomes encore plus finement de façon à ce que l'ADU (l'unité de traitement la plus petite au niveau applicatif) corresponde à ce qui serait perdu en cas d'erreur de bit sur le lien sans fil. Cette approche requiert que l'on passe à l'application les paquets corrompus reçus à l'interface réseau afin que l'application puisse en extraire toutes les données utiles (i.e. les unités de données non corrompues dans le paquet). On pourrait même définir des mécanismes de protection par l'ajout de redondance FEC au niveau des ADUs (en effectuant par exemple un OU logique sur les blocs modifiés des K images précédentes avec ceux de l'image courante). On pourrait aussi isoler la FEC dans un flot complémentaire (composé dans ce cas du OU logique des blocs modifiés des K images précédentes uniquement). Cette séparation permet la transmission de la FEC sur un groupe multipoint différent que celui du flot de base. En adoptant ce mécanisme pour chacune des couches du signal vidéo nous avons pu expérimenter l'intérêt de cette solution « de bout en

[1] Voir http://www-mice.cs.ucl.ac.uk/multimedia/projects/mice/
[2] Voir http://www-mice.cs.ucl.ac.uk/multimedia/projects/merci/
[3] Voir http://www-sop.inria.fr/rodeo/meccano/

bout » dans le cas d'un réseau hybride filaire/sans fil (i.e. sans l'utilisation de transcodeurs à l'entrée du réseau sans fil).

Les environnements virtuels sont des applications réseaux multi-utilisateurs dans lesquelles les utilisateurs (ou participants) peuvent se déplacer dans un monde virtuel, communiquer avec d'autres participants et interagir avec leur environnement. On les qualifie d'environnements virtuels à grande échelle (Large Scale Virtual Environments en anglais) lorsque le nombre de participants est très grand (des dizaines voire des centaines de milliers). Les jeux en réseaux sont un exemple de telles applications. Si le réseau supporte un service de transmission en multipoint, l'application peut en bénéficier pour augmenter l'efficacité du module de communication. Grâce au multipoint natif, l'envoi de données vers plusieurs participants peut se faire de manière transparente pour le participant avec une seule opération d'envoi, dans l'hypothèse où les communications ne nécessitent pas de transmission fiable. IP multipoint permet de fournir un service de transmission multipoint en mode sans connexion vers un « groupe » de participants. La notion de groupe est très importante car elle permet d'avoir un point de rendez vous « logique » dans le réseau. C'est comme si le « monde » était transformé en un espace de communication ouvert où l'on peut facilement établir des communications de groupe avec des participants non pré-identifiés mais ayant indiqué localement leur intérêt pour ces communications. Dans ce modèle appelé Any Source Multicast (ASM), le réseau prend en charge l'établissement d'arbres de diffusion multipoint ainsi que le relais des paquets vers les participants intéressés.

A priori, le modèle ASM semble être un modèle adéquat pour les environnements virtuels : les participants ont besoin de communiquer avec d'autres participants qu'ils ne connaissent pas à l'avance. La notion de groupe dans ce modèle permet de réaliser cette indirection de manière implicite. Cela dit, un problème majeur reste alors à régler : les participants ne sont pas intéressés par tous les échanges dans le monde virtuel et il est donc primordial de rajouter un mécanisme de filtrage des données reçues pour chaque participant. Si tout le monde virtuel était représenté par un seul groupe multipoint, les participants seraient alors submergés par tous les échanges dans le monde. Une solution consiste à découper le monde en zones représentées chacune par un groupe multipoint distinct. Un participant qui se déplace dans le monde virtuel et qui arrive dans une zone donnée s'abonne au groupe multipoint correspondant à cette zone. Ceci permet de diminuer la quantité d'information à priori « inutile » reçue par le participant. Les zones correspondent ainsi aux centres d'intérêts des participants. Cette approche est d'autant plus nécessaire que le nombre de participants est grand. Elle est donc essentielle dans le cas des environnements virtuels à grande échelle pour lesquels le contrôle de trafic est un besoin critique. Plusieurs problématiques annexes doivent alors être considérées : quelle taille choisir pour les zones ? Comment gérer l'hétérogénéité des participants en terme de capacité et de besoins en communication différents ? Quelles sont les limites du système ? Etc. Une architecture de communication pour les environnements virtuels à grande échelle basée sur ASM a été conçue et mise en œuvre par des membres de l'EPI Planète [2.2].

Mais le modèle ASM n'a pas connu un déploiement universel en raison d'une complexité protocolaire importante[1] et de l'absence de modèle économique intéressant pour les fournisseurs de services Internet. Le modèle Source Specific Multicast (ou SSM) a été proposé dans l'espoir d'un meilleur déploiement. Dans ce modèle, la notion très « élégante » de groupe multipoint qui sert de rendez vous logique disparaît. Les concepteurs de ce modèle

[1] Les connaisseurs apprécieront la liste IGMP/MLD, DVMRP/MOSPF/CBT/PIM-DM/PIM-SM, MSDP, MADCAP/AAP/MASC/BGMP, MBGP, GLOP, eGLOP, etc.

considèrent que l'établissement de communications de groupe à l'échelle planétaire n'est pas réalisable dans l'état actuel de la technologie. Contrairement au modèle ASM qui permet à n'importe quel utilisateur du réseau d'envoyer des données vers le groupe, avec SSM on doit se contenter d'un canal de communication qui émane d'une source bien identifiée. Ce service paraît à première vue moins convenable que le service ASM pour les environnements virtuels. En effet, l'absence de la notion de point de rendez vous logique au niveau du service réseau nécessite l'introduction d'une signalisation pour identifier et mettre en correspondance les participants « voisins ». Un protocole de communication pour les environnements virtuels basé sur SSM a donc besoin de fournir plus de services que dans la cas ASM ce qui se traduit aussi naturellement par plus de signalisation sur le réseau. Nous avons travaillé sur la définition d'une architecture de communication pour les environnements virtuels au dessus de SSM. Ces travaux [R.72] ont montré que la possibilité de filtrage des informations en provenance d'un participant particulier (inexistante dans le modèle ASM) ouvre des perspectives plus intéressantes pour l'implantation d'un contrôle de trafic à grande échelle pour des participants qui ont des capacités et des besoins différents. L'architecture de communication basée sur SSM peut donc être plus sophistiquée que celle basée sur ASM, mais elle permet de fournir des services améliorés pour ce type d'applications. Nous avons étudié le compromis entre la signalisation additionnelle engendrée par l'absence de point de rendez vous logique dans le modèle SSM et la meilleure gestion du trafic de données par les mécanismes de contrôle de congestion à granularité fine permise par ce modèle.

Ces travaux nous ont permis de confirmer un point désormais connu mais néanmoins assez important : la complexité des mécanismes requis pour fournir une fonctionnalité donnée au niveau de l'application dépend du service disponible au niveau du réseau. L'absence du déploiement du multipoint a donc accéléré la conception et le déploiement de réseaux de recouvrement (overlay) permettant le support de la diffusion multipoint sans aucune intervention au niveau IP afin de répondre aux besoins des applications.

2.2 Evolutions des services de l'Internet

Certaines applications ont des besoins très stricts de garanties de performance. Pensons par exemple à une application de télé-chirurgie : peu de personnes seraient volontaires (tout au moins du coté « patient ») sans être sûres que chaque mouvement du chirurgien et du bistouri seront bien transmis, et rapidement, d'un point à l'autre du réseau. De même, les applications de jeux distribués sur l'Internet nécessitent que les délais entre participants soient faibles. En effet, il est important dans les jeux de combat comme Doom que les roquettes tirées par un joueur atteignent le joueur cible, et l'atteignent juste après le temps que dure leur vol, plutôt qu'après un délai important correspondant au temps de transit d'un paquet à travers un Internet congestionné.

Certains industriels avaient cependant tendance à généraliser ces besoins de garantie de performance à une grande panoplie d'applications. Ils affirmaient qu'à partir du milieu de années quatre vingt dix, l'Internet n'était plus « un jouet entre les mains des chercheurs » et qu'il fallait absolument « mettre de l'ordre dans tout cela en fournissant (des équipements avec) un support de la qualité de service dans le réseau ». Une grande pression s'est alors exercée du côté de l'IETF : des groupes de travail traitant de la qualité de service ont été mis en place, les industriels y participaient avec des délégations très importantes[1]. Il est à noter

[1] Dans certaines réunions du groupe diffserv on comptait 500 participants.

que ce sujet n'était pas nouveau à l'époque : il y avait eu auparavant les discussions historiques sur le datagramme ou le circuit virtuel, le protocole ST II défini mais non utilisé, les différents travaux de recherche proposant une architecture réseau avec un meilleur support de la qualité de service (e.g. Virtual Clock de Lixia Zhang [2.3]). Tout cela était présent à l'esprit des participants à ces réunions. Le sujet a donc été abordé de nouveau de façon approfondie.

Il était clair qu'il fallait un ordonnancement différent du « Premier Arrivé Premier Servi » si l'on voulait contrôler les délais dans le réseau. En effet, le délai observé par un paquet dépend du temps d'attente dans les routeurs et le politique PAPS ne permet pas de servir un paquet audio (donc plus urgent) avant un autre paquet qui se trouve devant dans la file audio. Les travaux dans ce domaine ont montré qu'il était possible, en changeant les mécanismes utilisés dans les routeurs de l'Internet, de fournir des garanties de services. Ceci est réalisé en pratique via la mise en place de mécanismes dits d'attente équitable (« fair queueing » en anglais) dans les routeurs, qui permettent d'allouer explicitement des ressources du routeur et des liens qui lui sont attachés à telle ou telle type de trafic. Nous ne détaillons pas l'état de l'art prolifique traitant de ce sujet.

Reste à définir les « types de trafic » qui devraient bénéficier d'un traitement individualisé adéquat dans les routeurs. Dans une première approche dite int-serv, il s'agissait d'appliquer l'ordonnancement global des ressources réseaux « par flot » autrement dit pour chaque connexion de chaque utilisateur. Ceci requiert la mise en place d'un mécanisme qui permette aux applications de dire au réseau qu'elles ont besoin de telles ou telles ressources, et pour combien de temps et ce pour chaque flot. Les défenseurs de ce modèle disaient « pourquoi pas ? Le réseau téléphonique fournit une garantie de qualité de service individualisée pour chaque appel téléphonique et ce à l'échelle mondiale ». Mais la différence réside dans le fait que les besoins des applications de l'Internet ne pouvaient pas être représentés sous la forme *n*.64 Kbps. La gestion globale efficace[1] des ressources réseau de façon à satisfaire les besoins des applications impliquait une complexité qui dépassait de loin les limites imposées par la technologie de l'époque[2]. Il y avait donc une barrière technologique au support de la qualité de service par flot dans l'Internet[3].

Le support généralisé de la qualité de service « par flot » s'étant heurté à de sérieux problèmes de passage à l'échelle, l'IETF a lancé (dans le cadre du groupe diff-serv) des travaux sur des mécanismes ne nécessitant pas des traitements (ordonnancement dans les routeurs, signalisation de bout en bout) individualisés pour chaque flot. L'idée de base est donc d'appliquer au niveau de chaque routeur le même « comportement local » à tous les flots appartenant à un agrégat de trafic. L'identification d'un tel agrégat peut se faire par l'intermédiaire du champ TOS dans l'entête du paquet par exemple, sans effectuer un traitement individualisé par flot. Cette approche revient à fournir des services différenciés dans le réseau, c'est à dire un service spécifique appliqué à chaque « classe » de trafic. Les études du groupe diff-serv ont abouti à la définition de deux profiles de « comportement local » EF (Expedited Forwarding) et AF (Assured Forwarding). Le premier permet de fournir

[1] Les travaux de Parekh et d'autres montrent comment calculer les bornes sur les délais en fonction du paramétrage des mécanismes d'ordonnancement. Ceci dit, les bornes calculées était inutilisables dans un contexte pratique. En gros, il fallait surréserver avec des facteurs de surréservation pouvant aller jusqu'à 100 si on voulait « garantir » les délais pour la connexion considérée. Il est évident que ce modèle ne pouvait pas être mis en œuvre de façon « scalable ».
[2] N'en déplaise aux « Packet Star » et compagnie.
[3] Sans parler du besoin de facturation à l'usage et donc d'authentification, services non déployés à l'époque.

un service de liaison louée virtuelle, le deuxième une séparation en plusieurs classes de service différenciés.

Dans le cadre du projet RNRT Intradiff[1] et d'une collaboration avec CS Telcom, nous nous sommes intéressés à l'évaluation de la scalabilité des mécanismes de support de qualité de service (ordonnancement, gestion active des files d'attente, signalisation) dans un contexte de services différenciés. Notre objectif était d'évaluer la complexité d'implanter ces mécanismes ainsi que les performances globales attendues en termes de délai, de gigue et de disponibilité de la bande passante. Nos travaux [R.27], [R.28] ont montré que le réglage des paramètres des mécanismes d'ordonnancement et de gestion de la file d'attente a un impact déterminant sur les performances de ces mécanismes et qu'il était extrêmement difficile de trouver le bon réglage des paramètres dans le cas d'applications ayant des besoins variés sur de grands réseaux.

D'autre part, le dimensionnement (*provisioning*) et la configuration du réseau sont deux problèmes particulièrement intéressants cités dans les documents du groupe diff-serv. Le « dimensionnement » consiste à déterminer et à allouer (physiquement ou logiquement) les ressources requises aux différents points dans le réseau. La « configuration » consiste à distribuer les bons paramètres aux équipements réseaux afin de réaliser les objectifs de dimensionnement. Le dimensionnement et la configuration seront notés sous l'appellation gestion de la qualité de service. Cette gestion peut être statique ou dynamique. Dans le cas statique, la gestion de la QoS dans le réseau peut être effectuée « manuellement » par l'administrateur de réseau en fonction de la topologie et de la matrice de trafic, ou bien via un mécanisme de signalisation invoqué uniquement au moment de l'établissement des contrats avec les clients. La gestion dynamique est effectuée « automatiquement »; elle est basée soit sur une signalisation soit sur des mesures. Certains opérateurs voulaient étudier la faisabilité et l'efficacité d'une telle gestion dynamique qui semblait importante pour les services qualitatifs (services sans garanties fermes au dessus de AF), car on ne peut pas prédire le trafic de façon précise, mais aussi pour les services quantitatifs (services avec garanties fermes au dessus de EF) si on veut fournir un dimensionnement flexible, ou bien des TCA (Traffic Conditioning Agreement) dynamiques. Nous avons travaillé sur ce sujet dans le cadre des projets Intradiff et Arcade. Nos travaux ont montré qu'on pouvait construire un réseau basé uniquement sur un dimensionnement statique dès lors que l'on considère une répartition équilibrée et relativement stable des trafics. Cependant, ce mode de dimensionnement ne tenant pas compte des routes, la saturation d'un nœud liée à la convergence de multiples trafics, peut entraîner une dégradation notable de la qualité. Ceci se produit en particulier si l'on a affaire à des clients ayant des capacités d'accès très différentes, ou si l'on considère que certaines passerelles vers d'autres réseaux ou vers des sites de serveurs sont des nœuds de congestion possibles. Nous avons alors étudié dans [R.39] des protocoles de signalisation qui permettent de compléter l'approche « services différenciés » pour d'une part, continuer à garantir un niveau de qualité de service constant quand le réseau approche de ses limites et d'autre part optimiser l'utilisation des ressources du réseau. Suite à ces travaux, il nous a paru clairement qu'il était extrêmement difficile de déployer ce genre de mécanismes dans le réseau et nous avons préféré nous focaliser sur d'autres approches que nous avions aussi étudiées en parallèle.

Une des approches alternatives suivie par un certains nombre de chercheurs et que nous avons justement explorée consistait à concevoir des mécanismes de relais de paquets très rapide

[1] Voir http://www-sop.inria.fr/planete/intradiff/

dans les routeurs IP de façon à réaliser très efficacement la fonction de principale d'un routeur : l'acheminement des datagrammes IP, sans ajouts de nouvelles fonctionnalités dans les routeurs. Dans le cadre de cette problématique qui visait à construire des « routeurs gigabit », nous nous sommes intéressés en particulier au problème de recherche optimisée dans la table de routage IP des routeurs de cœur réseau. Avec l'introduction du routage sans classe (CIDR) pour mieux utiliser l'espace d'adressage IP et pour réduire la taille des tables de routage, les algorithmes de recherche de route vers une destination donnée devraient être modifiés. En effet, avec CIDR les entrées dans la table de routage consistaient en des préfixes de longueur variable correspondant à des « réseaux » plus ou moins grands et permettant un routage spécifique (préfixe long) ou global pour une agrégation de réseau (préfixe plus court). Les exceptions et spécificités étant aussi enregistrées dans la même table de routage, il fallait trouver une structure de données adéquate résultant en bonne organisation des préfixes et des algorithmes de recherche permettant de trouver le plus rapidement possible l'entrée dans la table de routage ayant la plus longue partie commune avec l'adresse de la destination (Longest Prefix Match). Ce domaine de recherche a été très actif vers la fin des années quatre-vingt-dix. Nous avons effectué un résumé de l'état de l'art du domaine incluant une taxonomie ainsi qu'un une comparaison des différents algorithmes de recherche de préfixe le plus long et de mise à jour des tables sur une plate-forme de test commune que nous avons développée. Nos résultats ont montré l'existence d'un compromis entre la rapidité de la recherche d'un préfixe dans la table et celle de la mise à jour par ajout ou suppression d'un préfixe de la table. Le lecteur intéressé par ce domaine pourra consulter [R.4].

Une autre approche consiste à concevoir des mécanismes de gestion des files d'attente des routeurs couplés éventuellement avec un contrôle de congestion appliqué de bout en bout dans l'objectif d'assurer une meilleure équité dans le traitement des flots dans un environnement hétérogène. En effet, le contrôle de congestion de bout en bout « TCP-courtois » impose une perception de l'état du réseau et un schéma d'adaptation unique à toutes les sources. Ces deux aspects ne sont pas réalistes pour les applications multimédia multipoint surtout dans un environnement de réseau dynamique. Proposer des mécanismes à l'intérieur du réseau et assouplir le principe de « toute l'intelligence à la périphérie » nous a semblé donc utile. Constatant que la plupart des mécanismes existants fournissant un partage équitable des ressources nécessite une gestion de tampons complexes et le maintien d'états sur les flots dans les routeurs, nous avons proposé un mécanisme simple permettant de pallier ces deux problèmes. Le mécanisme appelé TUF [R.38] réalise un partage équitable de la bande passante sans maintenir d'états « par flot », en utilisant une discipline de service FIFO. Les routeurs auront juste à jeter le paquet ayant l'étiquette (ou « tag ») la plus élevée en cas de congestion (et non pas le dernier reçu comme Drop Tail, ni un paquet choisi aléatoirement dans la file comme RED). Le « génie » de la solution réside dans le choix de la valeur des étiquettes en fonction de la réactivité des flots de façon à assurer un partage équitable de la bande passante. En résumé TUF propose de réaliser un objectif (l'équité entre les flux) en se basant sur une fonctionnalité simple dans les routeurs (jeter le paquet ayant le « tag » le plus élevé de la file FIFO) et sur un étiquetage des paquets décrivant la réactivité des flux en cas de congestion. Cette contribution sera présentée avec plus de détail et fera l'objet du chapitre 4.

Nous avons mené une autre étude dans le même domaine : assurer l'équité avec TUF nécessite la connaissance des modèles d'adaptation des flux, chose qui pourrait être difficile à réaliser. En redéfinissant un « objectif » plus modeste : assurer une meilleure protection entre les flux que dans le cas de la file FIFO, nous avons proposé un mécanisme de gestion de file d'attente qui permet de fournir cette isolation sans maintenir des états « par flot » dans le

routeur. Nous avons d'abord observé que les fonctions de la file d'attente de sortie dans un routeur consistent d'une part à réaliser le multiplexage des paquets des différents flots et d'autre part à absorber les salves des flots individuels. En se basant sur une identification des flots actifs (les flots ayant un paquet dans la file d'attente), nous avons proposé d'accepter un nombre limité de paquets pour chaque flot actif, ce qui permet de laisser de la place pour accepter un paquet d'un nouveau flot. Cette approche revient donc à protéger la fonction « multiplexage » du routeur de l'absorption des salves. Nos travaux ont montré [R.26] que le mécanisme proposé, appelé MuxQ, permet de fournir une très bonne isolation des flots et présente des performances similaires à DRR et CSFQ sans avoir besoin pour autant de maintenir des états « par flot » dans le routeur ni de changer les en-têtes de paquets IP.

Dans le même domaine, nous avons étudié le problème de partage de ressources dans le réseau entre les flux point à point et multipoint et entre les flux multipoint eux-mêmes. L'état de l'art concernant le contrôle de congestion multipoint multi-débit proposait une approche orientée récepteur dans laquelle les données envoyées par une application multipoint sont transmis sur des couches différentes et chaque récepteur s'abonnait à un certain nombre de couches en fonction du débit TCP-courtois entre la source et lui. Nous avons proposé une autre alternative selon laquelle les routeurs du réseau implémentent des mécanismes facilitant la cohabitation en flots point à point et multipoint, ce qui permet aux ISP d'établir des règles de partage des ressources entre flots point à point et multipoint ; règles qu'ils peuvent paramétrer et dont ils peuvent contrôler la mise en œuvre eux mêmes. Nous avons alors proposé de partager les ressources entre les flots point à point et multipoint de façon à ce que la globalité des flux multipoint actifs au niveau d'un routeur obtiennent une part de la bande passante et les flux point à point le reste. La part allouée au trafic multipoint est une fonction du débit TCP courtois de la session multipoint, du nombre de flux point à point et multipoint actifs au niveau du routeur (avec éventuellement un seuil à ne pas dépasser pour le trafic multipoint). Ce mécanisme de partage fournit une grande flexibilité aux ISP désirant déployer le multipoint dans leur réseau. En plus de ce partage entre point à point et multipoint réalisé par un mécanisme d'ordonnancement dont les paramètres sont réglés en fonction des règles du partage voulu, nous avons proposé un mécanisme de gestion de la file d'attente du trafic multipoint qui consiste, en cas de congestion, à jeter en priorité les paquets des groupes multipoint ayant moins de récepteurs en aval que les autres. Cette approche requiert la connaissance au niveau de chaque routeur du nombre de récepteurs en aval pour un groupe multipoint donné[1]. Les résultats publiés dans [R.2] montrent que ces « outils » peuvent être paramétrés pour fournir le partage requis. En plus, le contrôle de congestion multipoint multi-débits multicouches orienté récepteur peut être facilité par notre mécanisme de gestion de file d'attente dans les routeurs : comme les couches de bases auront plus d'abonnés elles seront moins pénalisées en cas de congestion, ce qui fournit un rejet sélectif des paquets avec différentes priorités sans pour autant requérir une affectation explicite de priorités aux différentes couches transmises.

Ces contributions pour améliorer le service fourni par le réseau ont été confrontées à une très grande inertie au niveau du déploiement. Ni le multipoint, ni les mécanismes de gestion de file d'attente et encore moins les mécanismes d'ordonnancement n'ont été déployés à grande échelle dans le réseau. Il était donc illusoire de persister dans cette direction, d'autant plus qu'un autre changement radical étant en train de se produire : l'Internet se déployait par-dessus diverses technologies de liens y compris sans fil et satellites. Ce changement

[1] Dans [R.29] nous avons proposé un mécanisme permettant calculer le nombre de récepteurs dans une session multipoint.

compliquait encore plus le support de nouveaux services dans le réseau car il en augmentait fortement l'hétérogénéité.

Chapitre 3

Les nouveaux supports de transmission

L'Internet s'est déployé très largement en intégrant une multitude de supports de transmission tels que les liens satellites, les liaisons sans fil terrestres (GSM, 802.11, UMTS, WiMax), les fibres optiques, la paire torsadée, le câble HFC, les liens ATM, etc. Ce déploiement a été facilité par le principe de bout en bout qui stipule que les procédures liées au contrôle de transmission (contrôle d'erreur et de flux) doivent être effectuées à l'extérieur du réseau. Ce principe a permis la simplification des routeurs IP assurant l'interconnexion (en mode sans connexion) des différentes technologies de réseaux. Les protocoles des couches réseau et transport de l'Internet ont été conçus en se basant sur ce principe, afin de supporter une large plage de technologies ayant des caractéristiques très variées. Pourtant, certaines liaisons ont des caractéristiques spécifiques qui ont un impact très important sur les performances des protocoles de l'Internet. Parmi ces spécificités au niveau physique ou au niveau liaison on trouve : un taux d'erreur de transmission élevé (liaisons sans fil), un délai de transmission élevé (liaisons satellite géostationnaire), un délai de propagation variable (liaisons satellite LEO), l'asymétrie ou l'unidirectionnalité de la liaison (satellite ou câble), ainsi que le support de fonctionnalités redondantes avec les couches supérieures (GSM, ATM ou Frame Relay). L'application stricte du principe de bout en bout se heurte donc à l'existence de telles liaisons. Les problèmes qui en découlent sont multiples :

- le non fonctionnement de certains protocoles (comme par exemple ARP, DVMRP et autres sur liaison unidirectionnelle)
- la forte dégradation des performances de certains protocoles (tels que TCP et IGMP sur des liaisons à délai élevé ou variable, TCP sur HFC ou xDSL),
- la difficulté de concevoir des mécanismes d'adaptation de bout en bout (à cause de la grande variabilité des caractéristiques des liaisons),
- l'interdépendance des mécanismes de contrôle de congestion au niveau liaison et transport (TCP sur ATM)
- la mise en correspondance des mécanismes de support de la qualité de service au niveau IP et au niveau liaison (diff-serv sur ATM ou sur Frame Relay, IP sur satellite).

Nous nous sommes intéressés à ces sujets en focalisant sur l'étude de l'impact des nouveaux supports de transmission sur le fonctionnement et les performances des protocoles de l'Internet et en particulier sur les protocoles de routage et de transport point à point et multipoint.

3.1 Impact sur les protocoles de routage

Notre intérêt à ce sujet a commencé au milieu des années quatre-vingt-dix dans le cadre d'une convention de recherche avec Eutelsat. Le problème qui nous était posé par les ingénieurs du département technique[1] se résumait ainsi : avec l'avènement des technologies de diffusion numérique (telle que DVB-S qui est le standard de facto pour la diffusion de télévision), les chaînes de télévision occuperont moins de bande passante sur les liens satellite et il y aura la

[1] Avec l'implication active de Nghia Pham.

possibilité de « faire de l'IP ». Il était connu que transmission « satellite » et « délai » ne faisaient pas bon ménage, mais le problème posé comprenait une composante plus difficile : l'unidirectionnalité des liaisons satellites. En effet, si le coût d'une antenne d'émission était exhorbitant (600 KF à l'époque), le prix d'une antenne parabolique de réception était dérisoire. On pouvait donc envisager une solution d'accès par satellite à l'Internet. La demande d'Eutelsat était d'analyser les problèmes soulevés par cette configuration et de proposer des solutions « standards » si possible.

L'analyse du problème a montré l'intérêt des liens satellite pour l'accès Internet mais aussi et surtout pour la diffusion de données dans l'Internet. En effet, les satellites couvrent une large zone géographique et le signal renvoyé permet de fournir une « diffusion naturelle » vers un grand nombre d'utilisateurs. En plus, dans un réseau congestionné comme l'Internet, la possibilité de court-circuiter les liens chargés du réseau pour faire parvenir un contenu haut débit à un grand nombre d'utilisateurs, semble très séduisante. Mais le problème provient de la nature unidirectionnelle des liens : s'il est possible pour un utilisateur de recevoir des paquets IP d'un satellite avec une simple parabole, il est lui impossible de renvoyer des paquets directement vers le satellite. Or, une grande majorité des applications (courrier électronique, web, audio et vidéoconférence, jeux) supposent un échange de données de façon bidirectionnelle entre les participants. D'autre part, les protocoles de routage dynamique (point à point et multipoint) ne fonctionnent pas dans le cas d'une liaison unidirectionnelle. En plus, le relais des données multipoint par RPF (Reverse Path Forwarding)[1] ne pourra pas être assuré par le récepteur satellite qui reçoit les données sur une liaison différente de celle qu'il utilise pour joindre la source. Tous ces problèmes ont été présentés dans le cadre d'un premier BoF[2] au 36ème IETF à Montréal en juin 1996 puis à San Jose en décembre 1996. Suite à cette deuxième réunion informelle, il a été décidé de mettre en place un groupe de travail dans le domaine du routage appelé udlr (UniDirectional Link Routing). Deux options ont été longuement discutées par le groupe de travail : modifications des protocoles de routage RIP, OSPF et DVMRP, établissement d'un tunnel entre les récepteurs et l'émetteur satellite à travers le réseau terrestre (voir [R.46], [R.87] ainsi que les Internet-Drafts [R.61] et [R.62]). Finalement il a été décidé d'adopter une solution basée sur un mécanisme d'encapsulation qui masque l'aspect unidirectionnel de la liaison. Le support de ce mécanisme permet aujourd'hui à des routeurs placés au pied d'un récepteur de reconnaître les routeurs des antennes d'émission et d'établir des tunnels qui assureront un fonctionnement *normal* des protocoles de routage dynamique. Ce mécanisme, couplé à un service de multiplexage embarqué sur le satellite[3] permet de couvrir une vaste région avec des dizaines d'émetteurs et de fournir une réception à haut débit à des milliers d'utilisateurs. La solution est décrite dans le RFC 3077 [R.63] publié en mars 2001 avec le statut de « proposed standard » à l'IETF. Plusieurs personnes impliquées activement dans le développement de la technologie udlr ont fondé la start-up UDcast en juin 2000 (www.udcast.com).

Ces travaux ont ouvert la voie à plusieurs études concernant les protocoles de routage et de transport sur un réseau intégrant des liens satellite. Ces études comportaient une forte composante expérimentale basée sur la disponibilité d'une antenne d'émission installée à

[1] Selon RPF, un routeur multipoint relaie le paquet sur toutes ses liaions (sauf sur la liaison par laquelle il a reçu le paquet) si et seulement si il a reçu ce paquet par la liaison qu'il utilise pour joindre la source de ce paquet.
[2] Un BoF (Birds of a feather) est une réunion informelle à l'IETF d'un groupe de personnes intéressées par un sujet donné. Il peut donner lieu ou pas à la mise en place d'un groupe de travail.
[3] Tel que le système Skyplex conçu par Eutelsat et qui permet de multiplexer au niveau du satellite les flux montants de différentes stations et de transmettre en downlink un flux DVB standard, donc démodulable par les set-top box du marché.

l'INRIA Sophia Antipolis avec une capacité d'émission réservée vers le satellite[1] HotBird5 à 13°E fournie par Eutelsat et ce, pour plusieurs années dans le cadre de la collaboration autour de udlr. Nous mentionnons ces travaux en les situant dans leur contexte et en renvoyant le lecteur intéressé vers les publications ou rapports plus détaillés.

Dans le cadre des projets @IRS[2] et Meccano nous avons étudié les problèmes liés au support d'applications multimédia multipoint sur un réseau intégrant un lien satellite avec udlr. La liaison satellite a des caractéristiques particulières : le délai de propagation aller-retour est de l'ordre de 500 ms, la bande passante est onéreuse et assez limitée, mais permet le support d'un service de diffusion à un grand nombre de récepteurs. Les applications envisagées dans le cadre du projet (DIS ou Distributed Interactive Simulation, audio et vidéo conférence, ...) génèrent des flux avec des besoins de qualité de service différents. Le support de ces flux sur la même liaison satellite nécessite un mécanisme de partage de la bande passante permettant de fournir un service adéquat à chacun des flots applicatifs. Nous avons étudié deux solutions [3.1] pour la gestion de la qualité de service. La première est basée sur une gestion de la qualité de service au niveau IP (avec des files WFQ ou CBQ et mécanismes de contrôle de congestion RED ou WRED par exemple). La liaison satellite est considérée alors comme une liaison HDLC au-dessus de laquelle les paquets IP sont transmis sans avoir un contrôle de l'émission sur le canal au niveau liaison. En d'autres termes, un paquet IP transmis à la couche liaison sera transmis sur le canal en entier avant de pouvoir transmettre un autre paquet venant éventuellement d'un autre flux. La deuxième solution requiert la mise en œuvre d'un mécanisme d'allocation du canal satellite au niveau liaison. Ceci revient à découper la bande passante disponible en « sous-canaux ». Un flux IP avec qualité de service spécifique serait alors orienté vers le sous-canal correspondant.

Par ailleurs, les applications considérées requièrent un service multipoint optimisé. Nous nous sommes penchés sur le réglage des protocoles du routage multipoint (IGMP, PIM). Avec udlr, la liaison satellite est considérée comme une liaison « classique » de l'Internet. Vu que le mécanisme d'encapsulation standardisé fonctionne au niveau liaison, les stations d'émission satellite et « tous » les récepteurs sont considérés comme appartenant à un Ethernet bidirectionnel « gigantesque ». Si l'on place les routeurs multipoint au pied des antennes d'émission, il faudrait opérer le protocole IGMP (Internet Group Management Protocol) pour la gestion des abonnements aux groupes multipoint sur ce « réseau local ». Le protocole IGMP exploite la diffusion bidirectionnelle du réseau local sous jacent en échangeant des requêtes/réponses entre le routeur désigné et les hôtes intéressés par les contenus diffusés sur les différents groupes. En plus, les réponses sont randomisées dans le temps et les hôtes annuleront leur réponse s'ils reçoivent une réponse identique avant l'expiration de leur temporisation. Or la liaison satellite n'est pas « vraiment » un réseau Ethernet. Les délais entre les stations d'émission et les récepteurs sont très importants et le nombre des récepteurs peut être très grand. Même si le mécanisme d'encapsulation permet de renvoyer les réponses des récepteurs sur le lien satellite, le fonctionnement d'IGMP risque d'être compromis par une « implosion » de messages de réponses vers les stations émettrices si le choix du délai d'attente n'est pas judicieux. Nous avons mis en œuvre une solution [3.2] basée sur des horloges exponentielles : peu de récepteurs choisiront alors un délai faible et la majorité choisiront un délai plus grand que le temps nécessaire pour recevoir la première réponse par la voie des airs. Les différents paramètres peuvent être réglés de façon à optimiser le fonctionnement dans le cadre d'un réseau satellite avec un grand nombre de récepteurs.

---

[1] Maintenant ce satellite s'appelle EuroBird2 et il est à 25,5° E.
[2] Voir http://www-rp.lip6.fr/airs/ et http://www-sop.inria.fr/rodeo/rrizo/AIRS/DescrTech-airs.html

Certaines études prédisaient un accroissement important des flux multipoint dans l'Internet. Ceci soulève la question du choix d'un support de transmission optimal : le satellite géostationnaire par sa couverture étendue est le candidat idéal. Par ailleurs, certains industriels travaillaient sur la préparation d'une nouvelle génération de processeurs de type DVB-S, avec un traitement à bord du satellite et une diffusion « multi-faisceaux ». Nous avons travaillé sur l'adaptation des protocoles multipoint à ce contexte de satellite nouvelle génération. Le satellite permet dans ces conditions, grâce à un support spécifique des protocoles de routage multipoint à bord du satellite, d'économiser la bande passante en réalisant la duplication à bord de façon plus fine (uniquement vers les faisceaux dans lesquels il y a des abonnés au groupe multipoint considéré). L'objectif de nos travaux était donc de concevoir un mécanisme permettant la commutation des paquets au niveau du satellite et d'effectuer le filtrage des paquets au niveau des récepteurs de façon efficace et ce, en respectant les contraintes de découpage des fonctions entre la « charge utile » et le « segment sol ». Deux modes de commutation ont été proposés : Dans le premier, la commutation entre les groupes multipoint et les faisceaux descendants est basée sur une table pré-calculée et transmise au bord du satellite. Dans le deuxième, la correspondance est réalisée au niveau terrestre et la commutation se fait au bord du satellite directement sur la base des numéros de ports de sortie liés aux faisceaux descendants et contenus dans le paquet. Nous avons développé un protocole de convergence permettant de réaliser la correspondance entre la signalisation des protocoles de routage multipoint de l'Internet (e.g. PIM-SM) et la signalisation DVB utilisée au niveau du satellite afin de mettre à jour les différentes tables (correspondance, commutation, filtrage) requises pour le fonctionnement global du système. Ces travaux ont été réalisés dans le cadre du projet Dipcast et publiés dans [R.3].

Nous nous sommes intéressés aussi aux critères de choix du type d'arbre d'acheminement pour le multipoint au-dessus des liaisons satellites. Deux modes de diffusion sont possibles avec le protocole PIM-SM : une diffusion utilisant un arbre partagé par tous les récepteurs et dont la racine est le point de rendez-vous du groupe multipoint, et une diffusion utilisant un arbre basé sur la source et spécifique à chaque récepteur dans le groupe. Les documents décrivant le protocole[1] stipulent qu'un récepteur commencera par recevoir les données sur l'arbre partagé et qu'il pourra commuter ultérieurement à un arbre spécifique basé sur la source. Ceci permet d'améliorer la qualité de service (e.g. les délais de transmission) en choisissant systématiquement le routage par les chemins minimums entre la source et les différents récepteurs. Mais les critères de décision de commutation de l'arbre partagé à l'arbre basé sur la source n'étaient pas mentionnés. Nous avons étudié ce problème et nous avons proposé des critères pour la décision de commutation basés sur une connaissance des ressources réseau disponibles et sur une coordination entre les différents récepteurs d'un groupe multipoint. Ces travaux sont publiés dans [R.33]. D'autres études ont été menées concernant le passage à l'échelle des protocoles de routage au-dessus de udlr ainsi que les aspects opérationnels et de démonstration de udlr, etc.

Les satellites de télécommunication géostationnaires représentent donc un support de transmission dont l'intégration dans l'Internet a soulevé un grand nombre de problèmes protocolaires et architecturaux. Les constellations de satellites en orbite basse ont été aussi considérées comme support de transmission pour du trafic IP et pour une intégration avec les autres protocoles de l'Internet. Nous avons étudié dans le cadre du projet RNRT « Constellations » le problème du routage IP dans les constellations de satellites (avec des liens inter-satellites). Comme les satellites en orbite basse (typiquement de 700 à 1400 km) se

[1] Voir RFC 2217 : http://www.ietf.org/rfc/rfc2117.txt

déplacent à des vitesses relativement élevées par rapport aux terminaux terrestres, la topologie du réseau change régulièrement ce qui soulève des problèmes pour les protocoles de routage tels que RIP ou OSPF qui se basent sur l'échange d'informations sur la topologie : le coût de tels protocoles serait en fait exhorbitant à cause de la topologie dynamique du réseau. Néanmoins, certaines caractéristiques des constellations (prédictibilité, périodicité du segment spatial, régularité et le nombre constant des nœuds dans le réseau satellite) rendent le routage dans cet environnement moins insurmontable qu'il n'y paraît dans un premier temps. Deux stratégies pour le routage intra-constellations ont été proposées : la première exploite la nature prédictible et périodique de la topologie de la constellation en pré-calculant des tables de routages pour toutes les configurations possibles. Ces tables seront chargées à bord de tous les satellites et utilisées de façon à maintenir des chemins à travers une topologie « virtuelle » fixe. Cette approche vise à masquer la mobilité des nœuds à des protocoles orientés connexions tels que ATM et qui pouvaient être supportés sur la constellation. La deuxième approche exploite la caractéristique de régularité : les informations concernant le routage sont maintenues dans un nœud virtuel fixe par rapport au sol. Cette approche nécessite donc l'échange des tables entre les satellites lors de leur mouvement et semble être plus convenable pour le support des datagrammes IP. En ce qui concerne le routage entre la constellation et le réseau terrestre, trois alternatives ont été étudiées : l'utilisation de tunnels, de la traduction d'adresse (NAT) ou du routage inter-domaine (en considérant la constellation comme un système autonome indépendant). La faisabilité des ces approches a été étudiée dans [R.5] en prenant en compte les aspects technologiques de l'époque (support de ATM et de MPLS). Les résultats de l'étude ont montré que des solutions protocolaires sont envisageables pour le routage au bord des constellations en dépit des fortes réserves qui étaient répandues sur ce sujet. Le problème concernant les constellations est venu d'ailleurs : des considérations technologiques et économiques concernant la réalisabilité des liens inter-satellite ont remis en question les grands projets de constellations. Cette étude nous a été pourtant bénéfique au niveau méthodologique car il s'agissait de trouver des solutions protocolaires dans des environnements dynamiques avec des caractéristiques et des contraintes spécifiques.

3.2 Impact sur les protocoles de transport

L'intégration de nouveaux supports de transmission change donc la donne pour les protocoles de routage mais aussi pour les protocoles de contrôle de transmission. Commençons par les protocoles de transport multipoint fiable. Nous avions travaillé auparavant sur la conception de la mise en œuvre de tels protocoles pour une application de tableau blanc partagé [R.83] en particulier dans le cadre d'une collaboration avec SGS Thomson. Mais ici il s'agit d'adapter ces protocoles au contexte satellitaire avec la facilité de la diffusion et la difficulté de la voie de retour. La diffusion fiable est un problème assez complexe. Il s'agit de contrôler la congestion dans le cas d'émetteurs/récepteurs ayant des conditions hétérogènes, sans pourtant perdre une partie de l'information et de préférence en un temps minimal. Dans le cas des liaisons satellites unidirectionnelles, on doit minimiser l'utilisation de la voie de retour terrestre. Pour cela, nous avons conçu un mécanisme de diffusion multipoint fiable basé sur la transmission de paquets de redondance (FEC) et permettant d'éviter l'envoi de demandes de retransmission par les récepteurs. Ce protocole a été testé par une expérimentation sur la liaison montante dont nous disposions. Nous avons par ailleurs testé les performances de ce mécanisme dans un environnement hybride (avec des récepteurs munis de carte de réception satellite et d'autres connectés via le MBone), dans le cadre de la plate-forme

d'expérimentation du projet COIAS[1]. Ces travaux ont été aussi utilisés par le W3C afin de mettre à jour par satellite les pages sur des sites miroirs.

Dans le cadre d'une collaboration avec Hitachi, et en se basant sur nos travaux précédents, nous avons proposé un protocole multipoint fiable pour une application de transfert de fichier sur réseau satellite vers un grand nombre de récepteurs. La spécificité de ce protocole décrit dans [3.3] réside dans l'utilisation d'un mécanisme hybride FEC avec des retransmissions. L'émetteur envoie d'abord le fichier en entier avec les paquets de redondance au niveau transport. Cette redondance permet d'éviter des retransmissions pour un certain profil de pertes de paquets. Quand l'émetteur termine la transmission du fichier, il envoie un message de contrôle pour demander l'état de tous les récepteurs qui répondent en envoyant leur état et en incluant le cas échéant les numéros de paquets qui n'ont pas été reçus correctement. Cette opération est répétée un certain nombre de fois avant de décider de la fermeture de la « session » et de l'état final pour chacun des récepteurs (a-t-il reçu le fichier ou pas ?). Des travaux ultérieurs ont donné lieu au protocole TICP [3.4] de collecte d'information dont l'objectif à l'origine était de collecter les informations des récepteurs sur la réception complète ou pas du fichier transmis.

Une autre étude intéressante concerne le contrôle de transmission (contrôle d'erreurs et contrôle de congestion) multipoint fiable par satellite. Afin de répondre aux besoins hétérogènes des récepteurs en termes de bande passante, il a été proposé dans la littérature de répartir les données à transmettre sur plusieurs canaux et d'utiliser des mécanismes de redondance au niveau paquet. La FEC peut même être transmise sur un des canaux de façon à permettre son utilisation à la demande des récepteurs. Ce couplage permet à chaque récepteur de s'abonner à un nombre de canaux correspondant à ses conditions (bande passante disponible sur le chemin entre la source et le récepteur, taux d'erreur sur ce chemin, etc.). Ceci revient à appliquer le concept RLM défini pour les applications audio et vidéo à la transmission multipoint fiable. Si les récepteurs souhaitent rester « TCP-courtois » il convient de choisir le nombre de canaux de façon à ce que le débit agrégé reçu corresponde au débit qu'aurait eu une connexion TCP entre la source et le récepteur. Or les solutions proposées pour la répartition des données dans des canaux pour la transmission multipoint fiable [3.5] imposaient uns distribution exponentielle des débits sur les différents canaux. Une telle répartition ne permet pas tout le temps de sélectionner des canaux dont le débit agrégé est assez proche du débit TCP-courtois, ce qui aboutit à une réaction imprécise et lente à la congestion dans le réseau. Nous avons proposé une organisation des données à transmettre dans les différents canaux de façon à minimiser le temps de réception pour chaque récepteur sans pour autant avoir une répartition exponentielle des débits sur les différents canaux. L'idée de base est de transmettre différents « bouts » du fichier sur les différents canaux en commençant sur chaque canal à partir d'un endroit différent dans le fichier à transmettre[2]. Un récepteur peut s'abonner à un nombre quelconque de ces canaux. Il peut en particulier choisir des groupes de canaux de façon à doubler son débit à chaque fois s'il le veut. De nouveau on voit clairement l'intérêt du concept ALF : on organisant les données de façon « significative » pour l'application, nous pouvons optimiser la transmission nettement mieux que dans le cas d'une architecture en couches séparées et indépendantes. Ces travaux ont été publiés dans [R.43].

---

[1] http://www.cs.ucl.ac.uk/research/coias/

[2] Si on prend l'exemple d'une transmission avec 8 canaux, la transmission sur le premier canal commence au début du fichier, sur le deuxième canal à la moitié, sur le troisième au quart, sur le quatrième aux trois-quarts, sur le cinquième au huitième, sur le sixième aux cinq-huitièmes, sur le septième aux trois-huitièmes et sur le dernier aux sept-huitièmes.

Considérons maintenant les protocoles de transport point à point tels que TCP. Plusieurs algorithmes d'adaptation de bout en bout dont le slow start ont été intégrés dans le protocole TCP. Ces algorithmes ont pour but d'éviter la congestion dans le réseau. Cependant, avec la très grande hétérogénéité des supports de transmission (délai des liaisons satellitaires, taux d'erreur de bits élevés des liaisons sans fil, etc.), les contraintes pour l'adaptation de bout en bout sont plus difficiles à respecter.

Nous nous sommes intéressés au cas de TCP sur liaison satellite géostationnaire. Le délai important de la liaison dégrade les performances de l'algorithme du slow start. Il n'est pourtant pas envisageable d'arrêter le support du slow start pour les connexions TCP qui traversent une liaison satellite : ceci aboutirait à une congestion sur les autres liaisons du réseau traversées par ces connexions. Il fallait donc trouver des mécanismes permettant de régler le problème soit de bout en bout, soit par l'introduction de proxy d'amélioration des performances. Ces alternatives ont été étudiées dans le cadre du projet TRANSAT en collaboration avec Alcatel et en réponse à un appel d'offre de l'ESA. Les résultats de cette étude ont montré l'intérêt de l'utilisation de la FEC au niveau transport pour améliorer les performances de TCP.

Par ailleurs, nous avons effectué une étude sur les facteurs affectant les performances de TCP dans un environnement hétérogène. Au lieu de considérer un type particulier de liens (satellite, réseaux dans fil, fibres optiques, ADSL, etc.), nous avons abordé le problème en effectuant une classification en fonction des paramètres déterminants pour les performances : produit bande passante – délai important, délais de transmission importants, pertes de paquets dues à la corruption et non à la congestion et asymétrie de la bande passante. Pour chacun de ces facteurs nous avons analysé l'impact sur les performances de TCP et comparé les différentes solutions proposées dans l'état de l'art. Cette étude a été publiée dans [R.6].

Enfin, nous nous sommes intéressés en particulier aux performances de TCP pour un trafic bidirectionnel dans un environnement asymétrique. En effet, le contrôle de congestion TCP a été conçu en considérant que le chemin de retour des accusés de réception n'est en aucun cas congestionné. Ceci n'est pas vrai sur des liens à forte asymétrie tels que les liens ADSL ou satellite sur lesquels le trafic de données « montant » (i.e. empruntant le lien dans le sens du débit faible) interfère avec les accusés de réception censés prendre ce même chemin vers la source. La fameuse « auto-synchronisation » (self-clocking en anglais) fournie par les ACKs TCP se trouve alors altérée. Nous avons proposé un mécanisme d'ordonnancement adaptatif permettant de gérer le trafic bidirectionnel dans un environnement avec des liens asymétriques. Fonctionnant à l'entrée du lien à débit faible, ce mécanisme est basé sur une séparation du trafic en deux classes : une pour les accusés de réception et une autre pour les paquets de données. Les poids déterminant le partage de la bande passante du lien entre les deux classes sont réglés en fonction du trafic de façon à maximiser une fonction d'utilité qui pourrait être définie soit par l'utilisateur, soit par l'opérateur du réseau. Les résultats de l'étude [R.24] montrent que notre mécanisme permet d'obtenir une bonne utilisation des ressources disponibles en maximisant la satisfaction de l'utilisateur dans un tel environnement asymétrique.

L'hétérogénéité de l'Internet n'a pas cessé d'augmenter. Les performances des protocoles et parfois leur fonctionnement en était perturbé. Des améliorations ont été proposées dans un processus évolutif et continu. Jusqu'où tiendra le réseau ? Peut-on continuer avec des sparadraps ou bien faudrait-il une approche plus radicale ?

Chapitre 4

Evolution incrémentale ou nouvelle architecture: Quelle solution à l'hétérogénéité des réseaux ?

4.1 Considérations architecturales

Le principe de bout en bout stipule que les fonctions de contrôle de transmission (détection et correction d'erreurs de transmission, contrôle de flux) devraient être assurées au niveau des nœuds extrémité (dits aussi *systèmes finaux* ou machines hôtes) et non par les *routeurs* du cœur de réseau. Ce principe a permis d'adopter une architecture de communication sans états dans le réseau et a facilité le déploiement à très grande échelle de l'Internet par le biais du protocole IP. Cette architecture protocolaire simple adaptée aux applications « classiques » telles que le courrier électronique et le transfert de fichiers ne permet pas de fournir des garanties de qualité de service par le réseau. Le réseau n'était donc pas conçu pour supporter des applications multimédias telles que la vidéo conférence à cause de la variabilité de l'état du réseau (délai de transmission des paquets de données, débit disponible et taux de perte de paquet). Ces applications devraient donc intégrer des mécanismes d'adaptation permettant de masquer les imperfections du réseau. Ces problèmes ont été au cœur de nos préoccupations pendant plusieurs années et nous avons contribué à la conception de mécanismes de contrôle de transmission pour les applications multimédias multipoints sur Internet telles les applications audio et vidéo conférence ainsi que le développement de l'environnement virtuel partagé à grande échelle (telle que l'application V-Eye).

La situation est devenue plus compliquée depuis quelques années : plusieurs tentatives d'enrichissement des services fournis par l'Internet (le routage multipoint, le support de la qualité de service, la mobilité) n'ont pas abouti à un large déploiement (au niveau inter-réseaux), et une tendance à l'ossification au niveau du réseau a été constatée. Les problèmes sont exacerbés par une hétérogénéité accrue aussi bien au niveau des liens (satellite, fibre optique, sans fil, ADSL, etc.) avec des débits, délais et taux de pertes différents, qu'au niveau des équipements (routeurs, mobiles, capteurs, etc.) avec des capacités différentes au niveau de l'unité centrale, de la mémoire et de la batterie. En même temps, les évolutions technologiques ne s'arrêtent pas et l'on dispose maintenant de réseaux ad-hoc (i.e. sans infrastructure fixe), réseaux de capteurs et de réseaux robustes aux grands délais (DTN ou Delay Tolerant Networks). L'intégration de ces réseaux dans l'Internet représente un défi majeur car certaines contraintes de ces réseaux remettent en question une des caractéristiques fondamentales de l'Internet : la disponibilité permanente de la « connectivité » assurée par la couche IP.

Par ailleurs, les travaux de recherche de la communauté « réseaux » pendant les trente dernières années ont suivi une approche *évolutive* selon laquelle des solutions ont été proposées pour pallier les problèmes de communication au fur et à mesure de leur apparition (e.g. le contrôle de congestion pour le protocole TCP). Ces solutions « ponctuelles » ont permis de maintenir l'Internet en état de marche malgré la multiplicité des intervenants au niveau de la gestion (réseaux d'accès, inter-réseaux). Ceci dit, cette approche évolutive a abouti à une complexification du réseau par le rajout de mécanismes allant à l'encontre de son

architecture de base (e.g. NAT). Ces problèmes ont été discutés longuement pendant les années 2000-2004 (publications [4.1, 4.2], conférences invitées [4.3, 4.4], rapports de workshops [4.5, 4.6, 4.7]). Le résultat communément admis de ces discussions est d'affirmer que la sortie de cette impasse ne peut pas être réalisée par une approche « compatible avec l'existant » et qu'il fallait une approche de rupture dans laquelle une *nouvelle architecture* réseau est à proposer.

Quitte à avoir une solution de rupture, il convient de « bien recommencer » (clean slate approach). Il s'agit donc d'identifier les défis correspondants aux limitations actuelles de l'Internet auxquelles la nouvelle architecture devrait répondre et de définir une approche méthodologique pour parvenir à un déploiement effectif et à grande échelle des nouvelles « innovations ».

Le premier défi correspond au support de la sécurité dans le réseau : l'Internet a été conçu en supposant des utilisateurs « coopératifs » et des applications simples. L'échelle et l'hétérogénéité du réseau ont augmenté énormément. Les protocoles actuels supposent donc un comportement coopératif et ne permettent pas de se protéger contre un comportement « anormal » (malicieux ou pas). Il faudrait changer l'architecture du réseau de sorte à améliorer la robustesse des protocoles face à un comportement anormal, à pouvoir résoudre rapidement les problèmes de sécurité identifiés et à isoler et à protéger les utilisateurs des attaques etc. Pour cela, il faudrait répartir les responsabilités entre les routeurs, les systèmes d'exploitation et les applications et établir un compromis entre « responsabilité » et « protection de la vie privée ».

Le couplage actuel entre la notion d'adresse (IP) et l'identificateur de machine représente une autre source de problèmes : certains opérateurs implémentent des politiques (de sécurité par exemple) faisant des hypothèses implicites concernant les adresses des machines. Il faudrait dissocier l'identificateur d'une machine de sa position géographique. Une conception « propre » concernant le nommage et la localisation de façon à améliorer la sécurité, la flexibilité et l'efficacité de l'acheminement des données dans le réseau est donc essentielle.

Une autre limitation provient d'hypothèses telles que la « connectivité permanente », la faible mobilité des systèmes finaux et l'utilisation de l'adresse destination pour le routage, qui ne sont pas vérifiées dans les réseaux de capteurs ou les DTNs. L'interfaçage de ces types de réseaux à l'Internet se fait alors à travers de passerelles ce qui induit une perte de fonctionnalités et de performance. La prise en compte de ces aspects dans l'architecture du réseau reste un problème ouvert : il n'est même pas sûr qu'un ensemble donné d'algorithmes, de protocoles et d'applications puisse fonctionner dans des réseaux « interactifs » (i.e. avec connectivité permanente) et des réseaux « robustes aux grands délais » (i.e. avec connectivité intermittente).

Une autre difficulté consiste à prendre en compte les aspects économiques et en particulier les intérêts divergents des différents intervenants de façon à protéger les investissements publics et privés. Il est important en fait de noter que l'architecture réseau implique une structuration de l'industrie et de son économie. L'architecture actuelle de l'Internet ne permet pas à un utilisateur de contrôler le chemin suivi par les données qu'il envoie ou qu'il reçoit[1]. Lever cette restriction aboutira à une disponibilité et une performance meilleures (e.g. par le choix

[1] A l'instar du choix du « long distance carrier » effectué depuis des décennies par les utilisateurs du réseau téléphonique aux US.

d'un chemin opérationnel). Cette ouverture incitera les ISP (FAI) à fournir des services « différenciés ». Il faudrait alors traiter les problèmes techniques (stabilité du réseau, conflits de préférences entre l'utilisateur et l'ISP) et économiques (facturation, contrat inter-ISP, SLA « cascadés » ou « opposables ») qui en découlent.

Par ailleurs, l'Internet a toujours fonctionné avec le principe de « concentré de technologies » : les routeurs implémentent les fonctions du plan de données et du plan contrôle selon une algorithmique intégrée (pas de plan de contrôle séparé). Cette approche aboutissait à des solutions « complexes » mais néanmoins efficaces. Mais les réseaux actuels appartiennent à des entités concurrentes et les fonctions qu'ils doivent supporter sont nettement plus sophistiquées. On aimerait avoir moins d'opacité et pouvoir effectuer par exemple le partage de charge, la détection d'anomalies, le diagnostic des fautes, optimisations des applications et autres fonctions de gestion du réseau.

Comment répondre aux besoins des applications et à l'hétérogénéité des technologies ? Dans l'Internet actuel c'est le modèle du sablier qui consiste à fournir les services de communication (le haut du sablier) au dessus d'un protocole simple et universel (IP) fonctionnant au-dessus d'une grande variété de technologies (le bas du sablier). Ceci revient à mettre le moins de fonctions dans le réseau. En d'autres termes, les composantes universelles de l'architecture sont réduites au minimum. Si on ne peut déployer les nouvelles idées architecturales au niveau « IP », il convient de « remonter au premier niveau d'interopérabilité » en insérant une couche de recouvrement (overlay). Cette approche met en œuvre des overlays répondants à des besoins spécifiques des applications en leur fournissant des fonctions de niveau réseau comme le routage, le modèle de service, la manipulation/codage des données, etc. Une autre possibilité serait de considérer le réseau comme un « substrat » au-dessus duquel « plusieurs architectures réseau » puissent coexister. Cette séparation amènerait de nouvelles opportunités en séparant les opérateurs de ressources physiques et leurs premiers clients : « les réseaux virtuels ». Ces deux approches traitent le problème à des niveaux différents mais focalisent toutes les deux sur la conception d'un « support » au-dessus duquel différents services et architectures pourront fonctionner avec les mêmes objectifs de répartition des fonctionnalités, de réduction au minimum les parties universelles, de stabilité, de « déployabilité », etc. Quitte à décider ultérieurement où l'on devrait implémenter les mécanismes en fin de compte.

Dans la continuation de nos contributions précédentes sur l'optimisation des transferts pour les applications multipoint, nous avons travaillé sur l'amélioration des performances pour les réseaux de recouvrement (overlay) fournissant un service de diffusion au niveau « applicatif ». En effet, les propositions précédentes de réseaux de recouvrement pour la transmission multipoint ont démontré l'importance d'exploiter les informations de la topologie du réseau sous-jacent dans la construction de la structure de la couche de recouvrement. Toutefois, la plupart de ces propositions reposent sur l'établissement d'une structure maillée entre les nœuds participants au réseau overlay suivi par un processus incrémental de raffinements afin d'optimiser l'arbre de diffusion multipoint et d'améliorer ainsi les performances du système. Ces approches ne passent pas de ce fait à l'échelle, induisent un surcoût de communication élevé, et nécessitent un temps de convergence important avant d'atteindre une structure stable. Nous avons proposé une approche basée sur une structure de regroupement hiérarchique des nœuds du réseau overlay avec un algorithme de *localisation* qui dirige graduellement les nouveaux venus vers les « chefs de groupe » les plus « proches » ce qui permet de construire une structure efficace sans pour autant induire un surcoût élevé. La solution proposée est robuste, passe à l'échelle et prend en compte la

topologie physique. Nous avons mené des simulations et des expérimentations sur PlanetLab afin d'en évaluer les performances. Les résultats publiés dans [R.16] prouvent d'une part que le processus de localisation consomme très peu de ressources en termes de délais et de bande passante, et d'autre part que le réseau overlay construit est efficace pour supporter des applications multipoint à très grande échelle.

Nous avons délibérément passé sous silence dans le paragraphe précédent la notion de « proximité ». Dans les réseaux de recouvrement, la proximité d'un « pair » est déterminée en général par le délai réseau entre les deux nœuds en question. Nous avons étudié la possibilité de déterminer la proximité au niveau « applicatif » en se basant sur une fonction d'utilité qui représente la qualité du point de vue de l'application. Par exemple, pour une application de transfert de fichier, il s'agit du temps global de transfert de fichier qui dépend du délai réseau entre le client et le serveur de fichiers, mais aussi de la bande passante disponible entre les deux participants. Selon ce schéma, la structuration du réseau overlay sera basée sur la proximité « au niveau applicatif » sans que cela corresponde nécessairement à la proximité « réseau ». Cette approche permet de construire des réseaux optimisés pour une application donnée, mais nécessite l'inférence des paramètres de la fonction d'utilité de façon qui passe à l'échelle. Nous avons investigué sur cette problématique et les résultats sont publiés dans [R.17] et [R.20].

Si l'on revient maintenant à la notion de proximité basée sur les délais réseaux, la solution proposée pour l'optimisation des transferts pour les applications multipoint repose sur une approche réactive : lors de l'arrivée d'un nouveau nœud, on localise le meilleur emplacement dans la structure par des échanges explicites puis on le redirige vers cet emplacement. Si le nombre des applications overlay croît –ce qui correspond à une tendance forte dans la voie vers le nouvel Internet–, il sera intéressant de fournir un service de la couche de recouvrement permettant aux nœuds de disposer de « coordonnées » réseaux qui seront utilisées pour la structuration du réseau overlay. Cette approche proactive implique un certain coût de mise en œuvre, mais permettra d'éviter les échanges explicites pour situer un nœud par rapport à d'autres dans le réseau. Nous avons travaillé dans ce domaine en particulier sur la sécurisation des systèmes de coordonnées. Ce travail publié dans [R.12] fera l'objet du chapitre 7.

4.2 Vers une méthodologie d'expérimentation réseau

Les travaux de recherche sur une nouvelle architecture de l'Internet doivent absolument avoir une forte composante expérimentale pour permettre aux chercheurs de mettre en œuvre, déployer et évaluer les innovations architecturales. Il est très important que ces expérimentations soient effectuées à l'échelle du réseau et qu'elles supportent un trafic provenant de « vrais » utilisateurs. Ceci incite à la réalisation d'une plate-forme expérimentale à étendue globale et ayant diverses fonctionnalités.

Tout d'abord, il faudrait que la plate-forme puisse supporter plusieurs nouveaux services et des architectures réseaux différentes. Ceci nécessite qu'il y ait le moins de restriction possible sur les architectures qui peuvent être supportées et les fonctionnalités qui sont fournies par la plate-forme. Pour cela, il faudrait que la plate-forme inclue des liens et des nœuds divers et permette la connexion de n'importe quel équipement.

Par ailleurs, la plate-forme devrait fournir un degré élevé de configurabilité (comme le font les plate-formes de recherche) et autoriser le support d'utilisateurs réels (comme c'est le cas

dans les réseaux de production). Ceci est essentiel pour évaluer les innovations de façon « réaliste » et pour mettre en place une communauté d'utilisateurs ayant un intérêt dans une nouvelle fonctionnalité, ce qui pourrait stimuler le transfert de technologie vers l'Internet commercial.

Pour remplir ces objectifs, la plate-forme devrait fournir un environnement au-dessus duquel on pourrait opérer des réseaux expérimentaux différents. Il s'agit alors de fournir un ensemble de ressources ainsi qu'un moyen d'attribuer et de configurer ces ressources aux différentes « expérimentations » en garantissant tant que possible l'isolation entre elles. Les ressources physiques, les fonctions de gestion, les processus de gouvernance et l'implémentation de la plate-forme peuvent changer dans le temps. Chaque expérimentation dispose d'un sous-ensemble des ressources de la plate-forme appelé « part » (ou slice en anglais). Il est à noter que les utilisateurs requerront des niveaux variés d'isolation, de connectivité, de dynamisme et de contrôle dans leur slice.

Ces concepts de base mènent à une liste de caractéristiques souhaitées de la plate-forme d'expérimentation : la neutralité vis-à-vis des services et architectures (du moins à partir de la couche 3), le support de la diversité au niveau des équipements terminaux, la facilité d'accès aux utilisateurs, la pérennité des l'infrastructure et l'existence d'incitations aux différentes institutions pour contribuer, d'un processus de gouvernance précisant les politiques d'attribution de slices et d'une architecture logicielle qui met en œuvre ces politiques.

Nous nous sommes intéressés à l'évaluation expérimentale des protocoles réseaux depuis le début de nos travaux de recherche dans le domaine en 1987. Notre méthodologie consistait à réaliser des implémentations des mécanismes et protocoles proposés et à tester ces implémentations sur le réseau « réel » (soit en LAN[1] ou à travers l'Internet). Nous avions d'ailleurs contribué activement dès 1995 à la mise en place de plate-formes de tests « réseau » nationales avec les autres acteurs universitaires majeurs en France (plate-forme Mirihade puis Safir). Nous étions impliqués aussi dans le projet VTHD qui visait à établir une plate-forme expérimentale « Vraiment à Très Haut Débit ». Nos travaux sur ces plate-formes nous ont permis d'illustrer l'importance du support du trafic réel et de ne pas se contenter d'un trafic « généré » pour « remplir les tuyaux ». Ces travaux nous ont amené à la réflexion décrite ci-dessus concernant les plate-formes expérimentales et nous ont incités à nous intéresser de près au réseau PlanetLab [4.8] qui consiste en un overlay permettant de réaliser plus facilement des expérimentations sur l'Internet.

Nous avons donc participé au montage du projet OneLab [4.9] et nous y participons actuellement. Il s'agit d'une extension des concepts de plate-forme PlanetLab permettant : d'augmenter la diversité des ressources gérées par la plate-forme pour inclure les liens sans fil physiques et des liens « émulés », d'augmenter les services fournis au niveau de la plate-forme avec une meilleure visibilité du réseau et des ressources physiques sous-jacentes, et d'améliorer le passage à l'échelle de la plate-forme en la réalisant comme une fédération de plate-formes locales ou régionales tout en définissant les politiques de gouvernance appropriées à ce mode de fonctionnement fédéré.

Nous avons récemment soumis une suite au projet OneLab (nommée OneLab2) et ce, en réponse au deuxième appel à projets du septième programme cadre de la Commission

[1] Ce qui nous a souvent valu les foudres « amicales » du service « Système et Réseaux » de l'INRIA Sophia Antipolis (e.g. quand la diffusion multipoint expérimentale que nous menions consommait 20% des ressources du réseau local de production de Sophia).

Européenne, dans le cadre de l'unité FIRE[1] (Future Internet and Research Experimentations) mise en place spécialement par la Commission pour financer des travaux de recherche concernant de nouveaux paradigmes de communication avec une approche expérimentale. Dans le cadre de ce projet, nous nous proposons de contribuer à l'élaboration d'une méthodologie d'expérimentation et d'une analyse comparative des résultats (benchmarking en anglais). Nos objectifs dans ce domaine consistent à fournir les outils et environnements nécessaires pour réaliser facilement des expérimentations en contrôlant les paramètres réglables et en sauvegardant les valeurs des paramètres du contexte de façon à pouvoir ultérieurement comparer les résultats d'expérimentations différentes. Une composante intéressante sera basée sur l'intégration d'outils de simulation et d'émulation dans le schéma de validation global. En utilisant des simulateurs intégrant du code réel, on se rapproche au mieux des conditions expérimentales. Il est alors envisageable de lancer certaines évaluations dans le simulateur et sur la plate-forme expérimentale en répétant ce processus si besoin est, afin de réaliser l'étalonnage et de pouvoir comparer en fin de compte les résultats des simulations et des expérimentations. Ce sujet est d'autant plus important que certaines configurations expérimentales (comme dans les réseaux sans fil) sont parfois extrêmement difficiles à reproduire[2] et impliquent l'utilisation de ressources onéreuses. L'utilisation d'un simulateur couplé à la plate-forme expérimentale de façon à émuler le fonctionnement d'un « réseau sans fil » contrôlable semble alors être une approche très intéressante. Elle permet de fournir une composante importante du schéma global d'évaluation. Il est souhaitable aussi de pouvoir utiliser les mêmes scripts pour lancer des simulations et des expérimentations sur la plate-forme. Tout cela aboutira à une méthodologie d'expérimentation globale pour les innovations réseaux que nous projetons de définir dans les années prochaines.

4.3 Et demain alors ?

Le réseau téléphonique était centré sur la notion de « lignes » et de chemins. L'Internet a été conçu pour rendre efficace la conversation entre deux systèmes finaux. L'utilisation actuelle du réseau revient dans la plupart des cas à chercher des informations sans se soucier de leur localisation. Ceci correspond à un mode de « dissémination » et non à la réalisation de conversation entre deux machines. Mettre en œuvre la correspondance entre les besoins de dissémination avec les primitives de communication fournies par le réseau complique les choses : le réseau n'a pas accès aux contenus et ne peut donc pas participer à l'application de politiques de sécurité (éviter le spam) ou à une meilleure efficacité (diffusion de données demandées par plusieurs utilisateurs). Il convient donc d'investiguer sur une nouvelle architecture réseau « orientée données » dans laquelle les données ont des noms stables indépendamment de leur localisation. Les fonctions de sécurité seront liées aux données et non aux canaux de communication. Les données pourront être transportées sur n'importe quel support physique (sans exiger un fonctionnement en mode interactif). Avec cette architecture « orientée données », le réseau pourra mieux aider l'utilisateur à réaliser les fonctions de communications requises. En adoptant le mode « dissémination » l'utilisateur ne reçoit que les données en réponse aux requêtes posées. Ceci permet de fournir « une qualité de service » sans pour autant exiger le support de mécanismes compliqués pour la qualité de service de bout en bout dans le réseau. Par ailleurs, un contenu populaire ne surchargera pas le réseau car l'identité des données sera visible et manipulable au niveau du réseau. Nous nous proposons d'étudier ce type d'architecture en proposant des solutions pour les mécanismes de

[1] Voir la page http://cordis.europa.eu/fp7/ict/fire
[2] A moins de les réaliser dans des « **F**ages de **C**araday » tel que le préconise ironiquement Christophe Diot.

communication et de sécurité et en expérimentant les fonctionnalités et les performances sur des plate-formes expérimentales globales.

Deuxième partie

Approfondissement de trois contributions

Chapitre 5 **Génération Automatique d'Implantations Optimisées de Protocoles**

Chapitre 6 **Contrôle de Congestion en milieu Hétérogène**

Chapitre 7 **La Sécurité des Systèmes de Coordonnées Internet**

Dans cette deuxième partie, nous présentons de façon détaillée trois parmi nos contributions. Le choix a été fait de façon à couvrir différents thèmes d'intérêts. Nous ne nous sommes donc pas focalisés sur un seul thème. Ces contributions ont été situées dans le contexte de nos travaux dans la première partie de ce mémoire. La première contribution concerne la génération automatique d'implantations optimisées de protocoles. La deuxième un mécanisme fournissant un contrôle de congestion équitable dans un environnement hétérogène. Quant à la troisième, il s'agit d'une collaboration récente portant sur la sécurité des systèmes de coordonnées Internet.

Chapitre 5

Génération Automatique d'Implantations Optimisées de Protocoles

Un compilateur de protocoles prend en entrée une spécification formelle d'un protocole et génère automatiquement son implantation. Les compilateurs de protocoles produisent généralement des implantations très peu performantes en termes de vitesse et de taille de code. Dans ce chapitre, nous montrons que la combinaison de deux techniques rend possible la génération automatique d'implantations performantes. Ces techniques sont i) l'utilisation d'un compilateur synchrone qui génère à partir de la spécification modulaire un automate minimum (au lieu de plusieurs automates indépendants), et ii) l'utilisation d'un optimiseur qui améliore la structure de l'automate et génère une implantation C optimisée. Nous avons développé un compilateur de protocoles qui combine ces deux techniques. Ce compilateur prend en entrée une spécification de protocole écrite en Esterel. La spécification est compilée en un automate intégré par la première passe du compilateur Esterel. L'automate est ensuite optimisé et transformé en une implantation C par un optimiseur appelé HIPPCO. HIPPCO améliore les performances et réduit la taille du code en optimisant simultanément la vitesse d'exécution du chemin commun et la taille du chemin rare. Nous évaluons dans ce chapitre les performances de notre approche en utilisant des implantations du protocole TCP générées à partir d'une spécification Esterel. Nous comparons les différentes versions du code générées avec le code de la version BSD de ce protocole. Les résultats sont très encourageants. Les implantations générées par HIPPCO exécutent jusqu'à 25% moins d'instructions que la version BSD, tout en gardant des tailles comparables.

5.1 Introduction

La conception d'un compilateur qui produit automatiquement à partir de la spécification formelle d'un protocole son implantation optimisée a été le sujet de plusieurs travaux de recherches dans la communauté « réseaux » [5.15, 5.18, 5.12]. Malheureusement les compilateurs existants produisent du code trop lent pour être utilisé dans un environnement opérationnel. L'objectif du travail présenté dans ce chapitre est de développer un compilateur de protocoles qui prend en entrée une spécification formelle d'un protocole et génère une implantation assez performante pour être utilisée pour la conception d'applications. Plus concrètement, l'objectif de notre compilateur est de produire du code qui:

- soit aussi rapide ou plus rapide que les meilleures implantations manuelles,
- ayant une taille aussi petite ou plus petite que les meilleures implantations manuelles,

Les motivations de ces objectifs sont évidentes : tant que les compilateurs de protocoles produiront du code peu performant, ils ne seront pas utilisés pour la production d'applications réelles. Notre approche pour aborder ce problème consiste à utiliser un compilateur existant et à développer un nouvel optimiseur et générateur de code. Cet outil utilise un ensemble d'optimisations que nous allons décrire en détail. Les résultats obtenus avec notre outil sont

très prometteurs. En effet, nous arrivons à compiler une spécification du protocole TCP en une implantation qui est :

- 25% plus rapide que l'implantation manuelle BSD de TCP,
- ayant une taille seulement 25% plus grande que celle de l'implantation BSD.

Notre principale conclusion est qu'il n'existe pas de limitation intrinsèque provoquée par la génération automatique d'implantations de protocoles. Les mauvaises performances des compilateurs existants peuvent être expliquées par l'utilisation de techniques inappropriées. Au contraire, l'application automatique d'optimisations conduit généralement à de meilleurs résultats que leur application manuelle. Nous espérons que ce travail renforcera l'intérêt porté à la génération automatique de protocoles et permettra d'augmenter les performances des compilateurs de protocoles.

5.2 Etat de l'art

Le travail présenté dans ce chapitre fait référence aux domaines de la génération automatique d'implantations de protocoles, des optimisations des protocoles et plus généralement de celui des optimisations des implantations de programmes.

Les implantations de protocoles sont réalisées à partir de leurs spécifications formelles. Ces réalisations peuvent être (1) *manuelles* (les implantations sont développées manuellement), ou (2) *automatiques* (un outil compile les spécifications en implantations).

Bien que l'approche manuelle soit la plus couramment utilisée, un intérêt croissant est apporté à l'approche automatique. Cette dernière a l'avantage d'assurer la conformité des implantations par rapport à leurs spécifications et de réduire considérablement leur phase de développement. Ceci est particulièrement intéressant dans le cadre de l'approche ALF selon laquelle il faudrait générer des protocoles spécifiques adaptés aux besoins des applications. Quelques outils ont été développés pour générer automatiquement des implantations à partir des langages formels LOTOS [5.23, 5.15], Estelle [5.12] et SDL [5.18]. Cependant, leur usage est resté assez limité. Les implantations générées sont souvent très partielles. Les interfaces inter-couches de la pile de communication doivent, très fréquemment, être implantées manuellement. De plus, les codes générés sont généralement moins performants et plus gros que les implantations manuelles [5.22].

Une équipe de l'université de Madrid a développé un prototype qui génère des implantations de protocoles à partir de leur spécification en LOTOS [5.15]. Les implantations générées par cet outil sont principalement utilisées pour vérifier les fonctionnalités des spécifications LOTOS [5.23]. Une spécification LOTOS est constituée de modules qui communiquent de façon asynchrone [5.15]. Avec le compilateur de l'université de Madrid, chaque module est implanté par un processus qui communique et se synchronise aux autres à l'aide de mécanismes asynchrones fournis par le système d'exploitation. Les performances des implantations générées sont assez médiocres et la taille des implantations conséquentes. Dans [5.15], il est relaté une expérimentation, dans laquelle une compilation d'une spécification d'un protocole simple (à la TP0) génère un code de 500 kilo-octets.

Le compilateur Nist-Estelle [5.9] a été développé pour générer rapidement des implantations utilisées pour des simulations. La structure de ces implantations est identique à celle des implantations générées par l'outil précédent : les modules de la spécification sont implantés

par des processus indépendants qui communiquent de façon asynchrone. L'analyse détaillée de ces implantations montre que 60% à 80% du temps d'exécution de ces systèmes est utilisé pour la communication et la synchronisation des différents modules [5.12].

Nous utilisons, en revanche, une approche synchrone. La spécification est compilée en un automate intégré éliminant ainsi le coût de la communication asynchrone. D'autre part, nous appliquons une série d'optimisations sur l'automate généré afin d'augmenter la vitesse d'exécution et de réduire la taille du code.

La plupart des optimisations utilisées par HIPPCO sont connues dans la littérature. Par exemple, l'utilisation des techniques d'*insertion en ligne des appels de fonctions* et d'*extraction du code rare* est présentée dans [5.17]. Nos contributions consistent donc d'une part à automatiser l'application de ces optimisations dans le contexte d'un compilateur de protocoles et d'autre part, à montrer que l'application de ces optimisations à un niveau plus élevé que les compilateurs traditionnels, permet d'améliorer les performances du code des protocoles. Alors que les optimisations sont généralement appliquées au niveau du programme C ou au niveau de l'éditeur de lien [5.17, 5.19], nous générons du code C optimisé. Cette approche rend notre code portable et notre outil complémentaire aux compilateurs existants.

De nombreux chercheurs ont travaillé, ces dernières années, sur des techniques de restructuration de programmes pour améliorer leur utilisation des caches. La majorité de ces travaux utilise des informations de profilage pour restructurer les codes objets des programmes afin d'améliorer la localité du code. McFarling [5.16] propose des algorithmes et des heuristiques qui utilisent les données de profilage pour exclure du cache certaines instructions afin d'améliorer les performances du cache. Pettis et Hansen [5.19] décrivent des techniques qui compactent les instructions les plus fréquemment utilisées. De plus, ils proposent de réordonner les fonctions pour réduire les conflits de cache qu'elles génèrent. Dans [5.10], les auteurs proposent de réduire la pollution du cache en restructurant le code. Contrairement, aux autres travaux, aucun profilage n'est utilisé. Les optimisations utilisent des analyses statiques et des analyses de dépendance du programme.

Certaines optimisations qui utilisent le profilage sont implantées dans quelques compilateurs existants. Le compilateur C pour les processeurs Alpha (distribués avec OSF-1) contient certaines de ces optimisations. Une étude sur l'impact de ces optimisations, exécutées par un compilateur commercial (HP C compiler) sur les codes des protocoles TCP et UDP, est décrite dans [5.21]. Ce compilateur implante les algorithmes présentés dans [5.19]. Bien que les débits des protocoles augmentent de 300 à 500 Kb/s, les effets de ces optimisations sont assez limités. Une raison en est que ces optimisations sont effectuées au niveau des procédures. Aucune optimisation inter-procédurale n'est effectuée.

Les optimisations que nous proposons dans la section 4 sont basées sur les techniques présentées dans [5.16, 5.19, 5.21]. Elles diffèrent cependant sur plusieurs aspects :

- Aucune phase de profilage n'est nécessaire. Les optimisations sont appliquées par une analyse statique du code à partir des prédictions spécifiées dans les modules de base. Cette approche rend la phase de compilation plus efficace et réduit le coût de collection du profilage.
- La structure en arbre des automates générés par le compilateur Esterel rend l'application des optimisations beaucoup plus simple et efficace.

- Des optimisations inter-modulaires sont effectuées. Les optimisations sont effectuées directement sur les arbres intermédiaires générés par le compilateur Esterel. Ce compilateur génère à partir d'un ensemble de modules un arbre intégré. L'abstraction module n'existe plus à ce niveau de la représentation.
- Les optimisations sont effectuées à un niveau plus haut que le niveau code objet. L'objectif de nos optimisations est de générer des implantations de bonne qualité qui puissent être traitées efficacement par les compilateurs de bas niveaux (compilateurs C).

Le reste de ce chapitre est structuré comme suit : notre environnement de développement automatique de protocole, qui est composé de la première passe du compilateur Esterel et de HIPPCO, l'optimiseur, est présenté dans la section 5.3. Les sections 5.4 et 5.5 détaillent les optimisations d'HIPPCO dont le but est d'augmenter la vitesse et de réduire la taille du code généré. Pour valider notre approche, nous avons exécuté un ensemble d'expérimentations. Ces expérimentations sont décrites dans la section 5.6. La section 5.7 analyse les résultats. La section 5.8 conclut ce chapitre.

5.3 Un environnement de développement automatique de protocoles

Dans cette section, nous décrivons l'architecture générale de l'environnement de développement de notre compilateur de protocoles. Cet environnement comporte quatre parties : le langage de spécification, le compilateur, l'environnement d'exécution et le langage cible. Le langage de spécification de notre environnement est le langage Esterel [5.1, 5.2]. Le compilateur est composé de deux étages : le premier est la première passe du compilateur Esterel qui génère, à partir de la description formelle un automate. Le deuxième est HIPPCO, l'optimiseur que nous avons développé. HIPPCO optimise l'automate et génère une implémentation C optimisée. Le langage C a été choisi comme langage cible pour des raisons de portabilité. L'architecture de cet environnement est présentée par la figure 5.1 et détaillée dans les sous-sections suivantes.

5.3.1 Le langage Esterel

Le langage Esterel est un langage *synchrone* utilisé pour la spécification de systèmes réactifs. Esterel considère que les réactions sont instantanées et donc que ses entrées et sorties sont synchrones. Grâce à ces évolutions instantanées, les réactions successives ne peuvent pas se chevaucher, elles sont dites *atomiques*. Les modules des spécifications Esterel communiquent par diffusion instantanée: un signal émis par un module est reçu instantanément par les autres modules de la spécification. Ces propriétés permettent la compilation d'une spécification Esterel en un automate à états finis déterministe.

Esterel a été choisi dans le projet HIPPARCH comme langage de spécification pour deux raisons essentiellement. Premièrement, il possède toutes les abstractions nécessaires pour spécifier des protocoles[1]. En effet, il permet la spécification des exceptions, des tests, des opérations mathématiques de base et du parallélisme. Deuxièmement, son approche

[1] Il s'agit bien sûr de la partie « contrôle » des protocoles. On ne parle pas ici de la copie ou de la manipulation des données.

synchrone conduit généralement à des systèmes plus performants que les langages utilisant une approche asynchrone, car elle supprime les communications interprocessus.

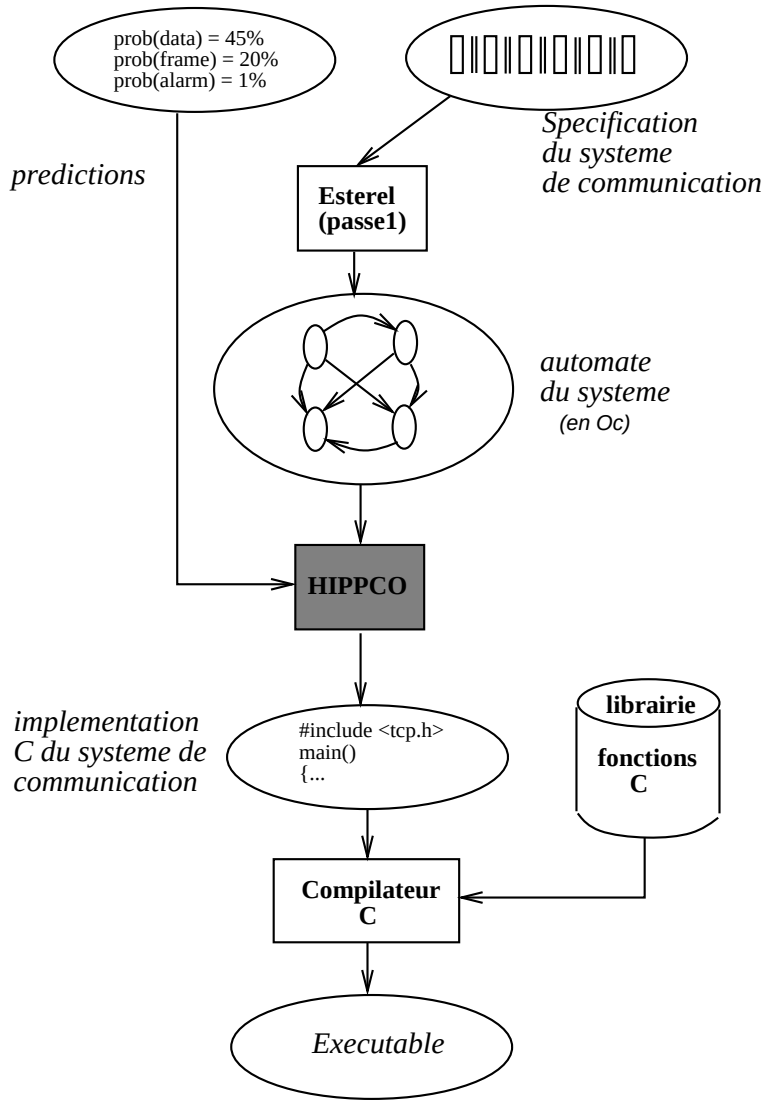


Figure 5.1 : L'environnement de développement automatique de protocoles

Les langages de spécification de protocoles « standards », tels que SDL ou Estelle, utilisent des approches asynchrones. La sémantique de SDL est celle de machines d'états concurrentes qui communiquent en s'échangeant des messages à travers des canaux de communication. La communication est asynchrone et les messages stockés dans une file d'attente unique à chaque machine d'état. En revanche, la sémantique d'Esterel est celle d'une seule machine d'états, composée de tous les modules de la spécification. Les communications entre les modules de la spécification se font de façon synchrone par des références mémoires. La figure 5.2 illustre la différence entre les approches synchrone et asynchrone.

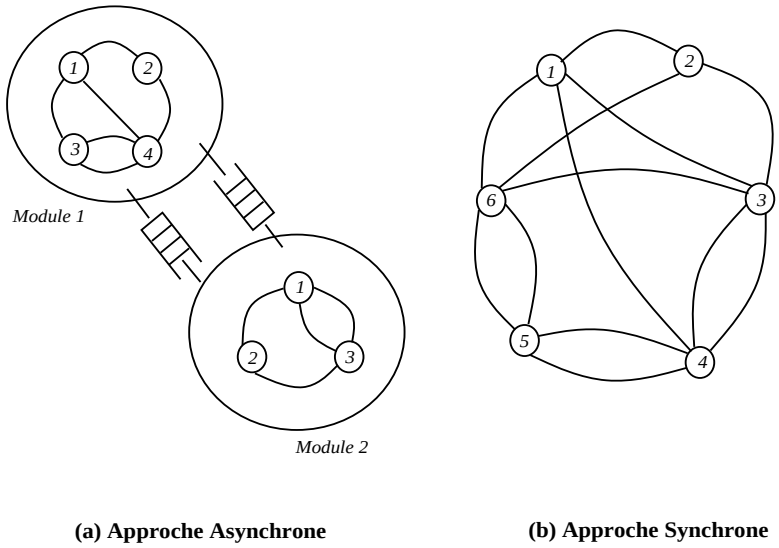


Figure 5.2 : Les approches synchrone et asynchrone

La spécification en Esterel

Une spécification en Esterel est composée de plusieurs modules, séquentiels et/ou parallèles, qui communiquent et se synchronisent par l'intermédiaire de signaux. Les signaux, émis par un module, activent tout module en attente sur ces signaux-là. Ce mécanisme de communication entre les modules apporte beaucoup de flexibilité, car des modules peuvent être ajoutés, supprimés ou échangés sans perturber l'intégralité du système. Une spécification est donc réalisée en combinant plusieurs modules élémentaires par leurs signaux d'entrée et de sortie.

Le compilateur Esterel

A la compilation, la spécification est transformée en un automate séquentiel à états finis. Les opérations des différents modules sont alors organisées en séquence en utilisant les signaux de communication. La figure 5.3 présente un exemple d'une spécification en Esterel et de l'automate généré. Cet exemple est très simple. Il est constitué d'une application qui envoie une donnée et attend son accusé de réception. La spécification de cette application est composée de deux modules en parallèle. Le premier s'occupe de l'envoi de la donnée. Il est représenté par un automate à 2 états. Dans le premier état, l'automate est en attente d'une donnée de l'application. Lorsqu'une donnée est reçue, un signal *paquet* est émis. Ce signal est en fait implémenté par une fonction qui construit un paquet à partir de la donnée reçue et la transmet sur le réseau (en utilisant l'interface « socket », par exemple). Dans le second état, l'automate est en attente d'un signal *OK*. Lorsque ce signal est reçu, le signal *Confirmation* est émis. Ce signal est implanté par une fonction, qui avertit l'application que la transmission est terminée. Le second module s'occupe de la réception de l'accusé de réception. Il est représenté par un automate à un seul état. Dans cet état, l'automate attend un accusé de réception. Lorsqu'un accusé de réception est reçu, celui-ci est testé. S'il est valide un signal *OK* est émis. Lorsque ces deux modules sont associés en parallèle et compilés, le résultat est un automate à deux états. Dans le premier état, l'automate attend une donnée de l'application.

Lorsqu'une donnée est reçue, un paquet est émis sur le réseau. Dans le second état, l'automate est en attente d'un accusé de réception. Lorsqu'un accusé de réception est reçu, celui-ci est testé. S'il est valide l'automate se termine.

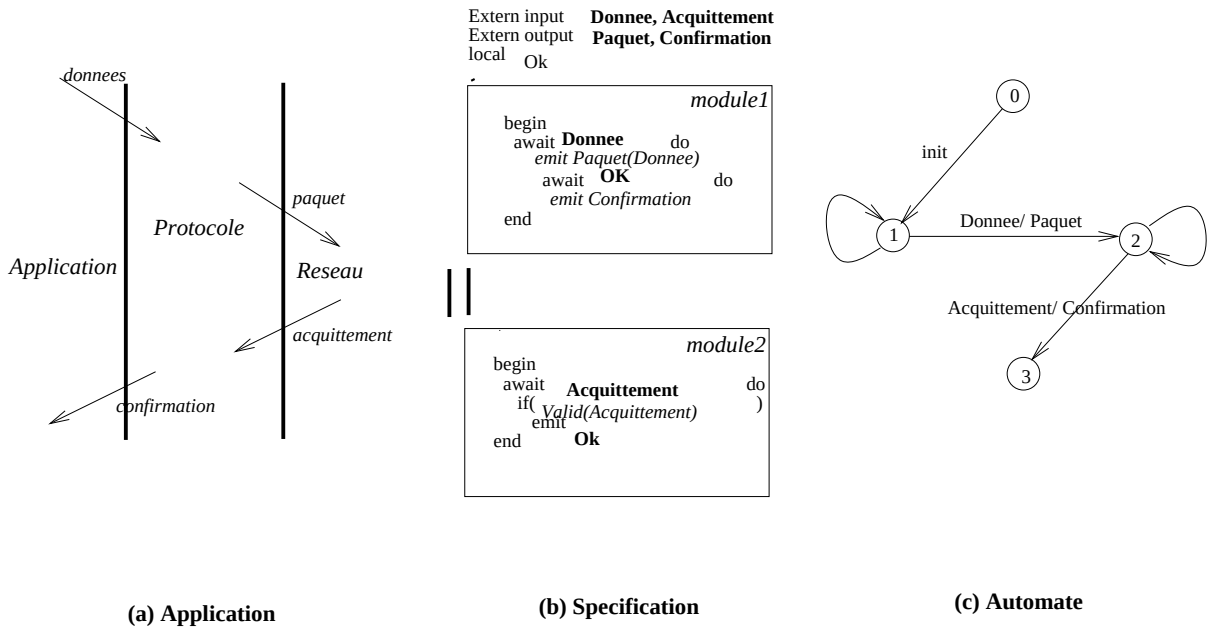


Figure 5.3 : Exemple de programmation en Esterel

L'automate généré par Esterel implante la partie contrôle du programme spécifié. Les fonctions référencées sont implémentées manuellement dans un autre langage (C dans notre cas) et reliées avec le code de l'automate généré. Dans l'exemple 5.3 ci-dessus, les fonctions *Paquet()*, *Confirmation()* et *Valid()* sont implémentées en C.

Esterel fait l'hypothèse de synchronisme parfait : l'automate ne peut être interrompu lorsqu'il traite un évènement. Cette hypothèse rend les programmes Esterel déterministes, car leurs comportements sont reproductibles. Les automates générés peuvent donc être vérifiés à l'aide d'outils de validation [5.20].

L'approche synchrone d'Esterel ne peut pas être considérée comme une solution à part entière, essentiellement parce que l'hypothèse de synchronisme n'est pas valide dans le monde réel. Une *machine d'exécution* est nécessaire. Cette machine interface l'environnement asynchrone à l'automate synchrone. Il recueille les évènements d'entrée et de sortie et active l'automate seulement lorsqu'il n'est pas ou plus actif. L'hypothèse de synchronisme est alors respectée.

5.3.2 L'optimiseur et générateur de code (HIPPCO)

HIPPCO est un nouvel optimiseur et générateur de codes. Il a été spécialement conçu pour la génération automatique d'implémentations optimisées de protocoles. Il prend en entrée l'automate généré par la première passe d'Esterel et génère une implémentation C performante. La motivation pour HIPPCO est double. Premièrement, la qualité du code

généré par Esterel est médiocre. Ce code est soit interprété, et par conséquent lent, soit inséré en ligne (compilé), et par conséquent grand. Deuxièmement, les optimisations appliquées par la plupart des compilateurs C sont limitées. Alors que les compilateurs C exécutent beaucoup d'optimisations, ces optimisations sont souvent locales et basées sur une analyse statique du code.

5.3.2.1 L'identification du chemin commun

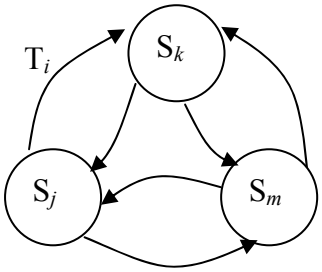
L'efficacité des optimisations dépend directement de la phase d'identification des chemins communs et rares. Une mauvaise identification du chemin commun peut générer des performances qui sont pires que celles de l'implantation initiale non-optimisée. La phase d'identification du chemin commun est donc essentielle.

L'identification du chemin commun nécessite généralement une très bonne connaissance du protocole développé. Le chemin commun est classiquement identifié manuellement par le concepteur du protocole après une analyse détaillée du code et de ses fonctionnalités.

Dans HIPPCO, cette identification est réalisée automatiquement à partir d'informations de prédictions annotées dans la spécification Esterel. Ces prédictions sont effectuées par le concepteur des modules et peuvent être modifiées par le concepteur de l'application pour une personnalisation de cette identification. Cette personnalisation permet généralement d'obtenir une identification plus fine, améliorant ainsi les effets des optimisations.

Dans les systèmes de communication, le chemin commun et le chemin rare sont distincts et complémentaires. Par conséquent, l'identification d'un seul de ces chemins est nécessaire. Dans HIPPCO, le chemin commun est d'abord identifié. Le chemin rare en est déduit. Un chemin est constitué d'un ou plusieurs flots de contrôle du programme. Un flot de contrôle d'un programme est défini par un groupe d'instructions qui traitent l'intégralité d'un évènement d'entrée. Dans la terminologie propre aux automates, un chemin est défini par une transition ou un groupe de transitions.

Le chemin commun est donc, conformément à la définition dans [5.5], composé de l'ensemble des transitions représentant 90% du temps d'exécution total de l'automate. L'identification de ces transitions nécessite donc le calcul des probabilités d'exécution des transitions de l'automate généré.



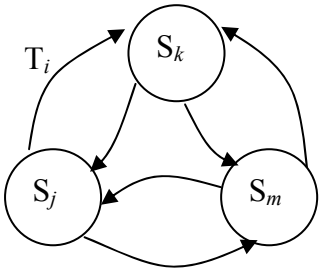


Figure 5.4 : Calcul des probabilités de transitions

Soit l'automate de la figure 5.4, la probabilité d'exécution de la transition $T_i^{j,k}$, qui dénote la $i^{\text{ème}}$ transition de l'état S_j menant à l'état S_k est égale à:

$$P(T_i^{j,k}) = P(S_j) \times P(T_i^{j,k} \mid S_j)$$

Il est montré dans [5.6] que $P(S_j)$ peut être calculée à partir de $P(T_i^{j,k} \mid S_j)$. Par conséquent, $P(T_i^{j,k}) = f(P(T_i^{j,k} \mid S_j))$ et le problème se ramène au calcul des probabilités conditionnelles des transitions.

Dans HIPPCO, chaque état de l'automate est décrit par un arbre de transitions. Les différents chemins qui vont de la racine de cet arbre aux feuilles constituent les transitions de l'état. Les feuilles désignent le prochain état dans lequel l'automate passera après l'exécution des transitions menant à ces feuilles (la feuille est désignée par <numéro prochain état>). La probabilité conditionnelle de chacune des transitions est donc égale au produit des probabilités des branches que la composent. La probabilité d'exécution d'une branche est égale à la probabilité de l'occurrence du nœud `test` dont elle dépend. Par exemple, la probabilité de la transition T_i de la figure 5.5 est égale à la probabilité que le test (e_1?) soit vrai multipliée par la probabilité que le test ($|e_1|$>Size)? soit faux.

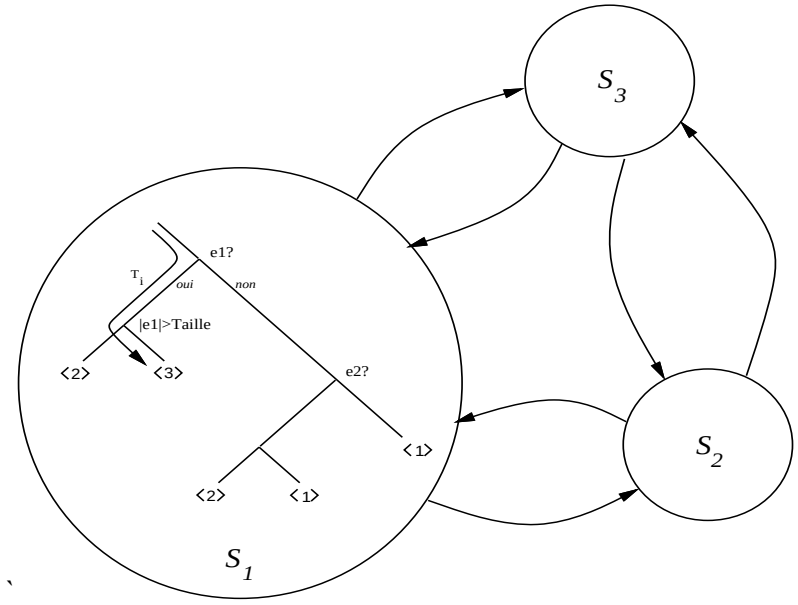


Figure 5.5 : Calcul des probabilités conditionnelles de transitions

Les nœuds `test` des arbres d'état proviennent directement de la spécification ou sont utilisés pour tester la présence éventuelle des différents évènements extérieurs. Les probabilités d'occurrence de branches des nœuds `test`, et par conséquent les probabilités des transitions de l'automate, peuvent donc être calculées à partir :

- des probabilités d'occurrence des différents évènements extérieurs
- des probabilités d'exécution des branches « vrai » et « faux » de chaque test de la spécification

Ces informations sont d'un ordre sémantique. Elles ne peuvent donc être fournies statiquement que par le concepteur du protocole, qui seul connaît le comportement « normal » du système qu'il spécifie. Nous avons donc modifié le langage Esterel en y ajoutant des annotations qui permettent de spécifier ces probabilités.

Le concepteur du protocole ou de l'application peut fournir pour chaque test de la spécification une prédiction sur la probabilité que ce test soit vrai et prédire pour chaque évènement externe sa probabilité d'occurrence. Ces prédictions sont conservées lors de la compilation de la spécification en automate et sont ensuite utilisées par HIPPCO pour l'identification des chemins.

5.3.3 L'environnement d'exécution d'HIPPCO

Tout environnement de développement de protocoles nécessite un système d'exécution performant, qui fournit des interfaces aux services des machines hôtes : mécanismes de gestion de la mémoire, temporisations et cartes réseaux. Les performances du système global dépendent directement des performances du système d'exécution. Etant donné que la conception de systèmes d'exécution est un problème bien connu [5.14], nous avons décidé de ne pas dépenser beaucoup d'efforts pour la conception d'un environnement d'exécution performant. Nous avons développé, à la place, un environnement de test minimal. Cet environnement est décrit dans la section 5.6.2.

5.4 Les optimisations de performance d'HIPPCO

L'optimisation de la vitesse d'exécution d'un programme consiste à réduire le nombre de cycles nécessaires à son exécution. Cette tâche est difficile, car cette vitesse d'exécution dépend de plusieurs composants qui interagissent.

Le temps d'exécution d'un programme peut être estimé par la formule suivante [5.11] :

$$Cycles_{total} = IC.CPI + nb_memory_accesses.miss_rate.miss_penalty$$

où $Cycles_{total}$ est le nombre total de cycles exécutées par le programme, *IC* le nombre d'instructions exécutées, *CPI* le nombre moyen de cycles par instruction, *nb_memory_accesses* le nombre des accès mémoire, *miss_rate* le pourcentage des accès mémoires qui ne sont pas dans le cache et *miss_penalty* la pénalité, exprimée en cycles, rencontrée pour chaque accès à la mémoire principale. En d'autres termes, le nombre total de cycles est composé de la somme des cycles nécessaires à l'exécution des instructions et des cycles d'attentes provoqués par les accès mémoires.

Dans cette évaluation, deux hypothèses sont faites :

- toutes les attentes de mémoire sont provoquées par les caches. Bien que ce ne soit pas exact pour toutes les machines, les attentes de mémoire générées par les caches dominent toujours les effets des autres sources d'attente.
- les cycles dus aux accès au cache sont compris dans ceux utilisés pour l'exécution des instructions et sont par conséquent inclus dans $CPI_{Exécution}$.

Selon cette formule, la vitesse d'exécution d'un programme dépend de trois composantes:

- le nombre d'instructions à exécuter (*IC*)
- le coût des attentes mémoires (*nb_memory_accesses.miss_rate.miss_penalty*)
- le nombre de cycles par instruction (*CPI*)

L'optimisation du temps d'exécution total d'un programme est donc réalisée par l'optimisation de ses composantes.

5.4.1 Ré-ordonnancement des arbres d'évènements (R)

Le coût moyen d'accès à un sous-arbre traitant un évènement dépend de la probabilité d'occurrence de cet évènement, de l'emplacement du sous-arbre dans l'automate (son rang), ainsi que du coût de l'exécution du prédicat de test. Il peut être facilement démontré que ce coût est minimal lorsqu'il est associé aux évènements les plus fréquents, les coûts les moins élevés.

L'optimisation de « ré-ordonnancement des arbres d'évènements » (R)[1] est basée sur cette observation. Elle permet de réduire *IC* en réordonnant les sous-arbres des évènements dans l'ordre décroissant de leurs probabilités d'occurrence.

Les évènements les plus fréquents (et donc appartenant au chemin commun) sont alors détectés plus rapidement, car leurs sous-arbres sont plus proches de la racine de l'arbre de l'état.

Cette optimisation est illustrée par l'exemple de la figure 5.6. La probabilité d'exécution du sous-arbre *A* est de 0.1, celle du sous-arbre *B* est de 0.2, celle du sous-arbre *C* est de 0.1 et celle du sous-arbre *D* est de 0.6. L'optimisation de ré-ordonnancement des arbres d'évènements réordonne ces sous-arbres dans l'ordre: sous-arbre *D*, sous-arbre *B*, sous-arbre *C* et sous-arbre *A*. Ainsi l'évènement `entrée D` (qui est le plus fréquent) sera détecté très rapidement car sa présence est testée en premier dans l'arbre optimisé.

5.4.2 Extension des branches et sous-arbres fréquents (E)

Les optimisations qui seront présentées dans la section 5.5 diminuent la taille des implantations générées en transformant les arbres d'Esterel en graphes. Ces transformations partagent les branches et les sous-arbres identiques afin de réduire les duplications de code. Dans l'implantation C générée, les parties partagées sont implantées par des fonctions. Ce

[1] La lettre entre parenthèses après le nom d'une optimisation désigne l'option à spécifier au compilateur HIPPCO afin d'activer l'optimisation considérée. Nous désignons parfois les optimisations par la lettre correspondante.

partage augmente donc le nombre d'indirections et ralentit inévitablement la vitesse d'exécution du code.

L'optimisation d'*extension des branches fréquentes* diminue l'impact de ces indirections sur les performances du code en supprimant les indirections du chemin commun. Cette optimisation est mise en œuvre en remplaçant chaque référence à une branche (ou un sous-arbre) par une copie de cette branche (ou ce de sous-arbre).

Le code du chemin commun est alors inséré en ligne. Cette optimisation réduit le nombre d'instructions à exécuter, car elle supprime les appels de fonctions sur le chemin commun. Elle améliore par la même occasion les performances des caches en augmentant la localité du code.

L'inconvénient majeur de cette optimisation est qu'elle augmente la taille des implantations en dupliquant certaines parties du code. Cependant, le chemin commun ne représente généralement qu'une petite partie du code. Cette règle est encore plus confirmée pour les codes de protocoles de communication. Le chemin commun de l'implantation BSD de TCP est de l'ordre de 9% de la taille totale du code. L'accroissement de taille générée par cette optimisation est par conséquent petit par rapport à la taille totale de l'implantation. On se retrouve ici confronté au compromis: taille/vitesse.

Cette optimisation est illustrée par l'exemple de la figure 5.6. Au lieu de coder tous les sous-arbres (qui sont similaires dans cet exemple) par une référence à un même sous-arbre X avec les paramètres appropriés, le sous-arbre D qui appartient au chemin commun (lignes foncées) est inséré en ligne. Il peut donc être accédé directement sans indirection.

5.4.3 Elagage des branches et sous-arbres morts (P)

Dans un environnement donné, la probabilité d'un évènement e_i est nulle, si cet évènement ne peut pas être présent. Dans ce cas, le sous-arbre correspondant à e_i ne sera jamais activé. Il est donc inutile et peut être supprimé.

La probabilité de succès d'un nœud `test` de la spécification est nulle lorsque le résultat ce `test` est toujours faux. La branche `else` est alors toujours exécutée. Inversement, la probabilité de succès d'un `test` est égale à 1, si ce test est toujours vrai. La branche `then` est alors toujours exécutée. Dans ce deux cas, l'exécution du prédicat du test est inutile car on connaît le résultat à l'avance.

L'optimisation d'*élagag*e se base sur ces deux observations pour améliorer le code généré par HIPPCO. Cette optimisation :

1. supprime tous les sous-arbres des évènements dont les probabilités d'occurrence sont nulles. Elle a deux effets principaux: (1) en supprimant, les sous-arbres de ces évènements improbables, elle réduit la taille du code généré, et (2) en supprimant les tests de présence de ces évènements, elle diminue le nombre moyen d'instructions à exécuter.
2. transforme les tests constants. Un test *T*, dont le prédicat est `test`, la branche « then » est le sous-arbre `then` et la branche « else » est le sous-arbre `else`, est remplacé par la branche `then` si `test` est toujours correct et par la branche `else` si `test` est toujours

faux. Elle a les mêmes effets que la transformation précédente: elle réduit la taille du code et diminue le nombre d'instructions à exécuter.

La figure 5.6 illustre cette optimisation. Cette figure montre l'arbre initial annoté des probabilités des entrées et des tests, ainsi que l'arbre résultant lorsque les tests constants ont été transformés (pas d'élagage de sous-arbre dans cet exemple). Dans cet exemple, le test T_1 est toujours vrai. Le test est alors remplacé par la branche `then`.

Une conséquence de cette optimisation est que le code est spécialisé et ne correspond plus exactement à la spécification initiale. Les implantations ainsi optimisées sont dépendantes de leur environnement d'exécution et ne peuvent être utilisées par d'autres environnements. Cette optimisation est appliquée à la fois sur les chemins commun et rare.

5.4.4 Insertion en ligne des appels de fonctions (I)

L'optimisation d'*insertion en ligne des appels de fonction* consiste à remplacer un appel de fonction par une copie du corps de la fonction. Ceci diminue le temps d'exécution pour deux raisons: d'abord le coup de l'appel de fonction est économisé (*IC* diminue). Ensuite, l'élimination de l'indirection permet l'application de plus d'optimisations par le compilateur de bas niveau. La contrepartie est une augmentation de la taille du code, qui pourrait ralentir l'exécution du programme par suite de son effet sur le cache [5.7, 5.8, 5.13, 5.17]. L'insertion d'une fonction est donc bénéfique si cette fonction est appelée une seule fois, ou très fréquemment ou bien si la taille du corps de la fonction est inférieure ou égal au nombre d'instructions nécessaires pour appeler cette fonction.

Les compilateurs traditionnels insèrent en général les fonctions qui satisfont la première et la troisième des conditions ci-dessus. Comme ils ne disposent pas d'information sur la fréquence des appels de fonctions ils ne prennent presque jamais en compte la deuxième condition. Dans HIPPCO, l'analyse fine de l'automate permet d'appliquer simplement et de façon plus efficace cette optimisation: toutes les fonctions rencontrées le long du chemin commun sont insérées en ligne.

5.4.5 Extraction du code rare (U)

La taille du chemin commun d'un protocole est souvent plus petite que celle du cache d'instructions (typiquement 8KiloOctets). Les accès manqués (cache misses) ne sont donc généralement pas dus à une taille de code trop importante, mais à un problème de synchronisation. Avec les caches à correspondance directe (direct mapping), qui sont les caches les plus fréquemment utilisés, les emplacements des blocs d'un programme dans le cache sont déterminés par leurs adresses. Deux instructions génèrent un conflit de cache, et par conséquent un accès manqué, lorsque la différence des adresses des blocs auxquels ils appartiennent est un multiple du nombre de blocs du cache. Une réduction du nombre d'accès manqués peut alors être obtenu en restructurant le code de telle façon que les instructions les plus fréquemment exécutées n'interagissent pas ensemble.

Ce problème d'ordonnancement des blocs de base est très difficile à résoudre au niveau du langage de spécification ou de programmation, car l'unité que manipulent les caches, le bloc de base « basic block », n'y est pas définie. Ce problème semble plus facile à résoudre au

niveau machine. Cependant à ce niveau, les fréquences d'exécution des blocs, qui dépendent de la sémantique du programme, ne peuvent être déterminées.

L'optimisation d'*extraction du code rare* propose une solution simple à ce problème d'ordonnancement. Elle *compacte* les instructions qui sont fréquemment exécutées en déplaçant le code rarement exécuté à l'extérieur du chemin commun. Le code du chemin commun ainsi optimisé est très compact et peut entièrement être contenu dans le cache. Les instructions du chemin commun ne rentrent alors plus en conflit pour l'accès au cache.

HIPPCO identifie automatiquement le chemin commun et compacte son code en implantant tous les sous-arbres appartenant au chemin rare, et qui sont rencontrés sur le chemin commun, par des appels de fonctions. Cette optimisation augmente donc le nombre des indirections sur le chemin rare, et nécessite, par conséquent, une identification précise du chemin commun. Son application manuelle est alors rendue très difficile.

Cette optimisation est illustrée par l'exemple de la figure 5.6. La branche `then` du test T_2 qui est un sous-arbre rare rencontré le long du chemin commun est remplacée par une référence.

5.4.6 Ré-ordonnancement des résultats des tests (T)

Les processeurs actuels utilisent le pipelining afin d'améliorer la vitesse d'exécution d'un programme. Cette technique consiste à exécuter des micro-instructions en parallèle. Le compilateur de bas niveau ordonnance les instructions afin de maximiser le gain du pipelining. Cependant, dans certaines situations l'exécution de la prochaine instruction nécessite la connaissance du résultat de l'instruction en cours. Par exemple, si un test est exécuté, la prochaine instruction ne peut par être lue avant de connaître le résultat du test. Afin de réduire l'impact de ce retard, les processeurs font de prédictions: ils supposent que la branche positive (branche `then`) sera exécutée. Si cette prédiction se trouve fausse, le pipeline est arrêté, et la lecture de la nouvelle instruction est entamée. Une prédiction réussie permet donc d'éviter le coût du test correspondant.

L'optimisation de *ré-ordonnancement des résultats des tests* est basée sur cette observation. Elle restructure les nœuds test dans l'arbre de façon que les résultats les plus fréquents soient dans la branche `then`, en conformité avec les prédictions des compilateurs de bas niveau.

Cette transformation est appliquée sur tout l'automate, en inversant les tests dont les résultats les plus fréquents sont sur la branche `else`. Cette optimisation améliore aussi la localité du code à cause de la structure en arbre de l'automate.

Dans la figure 5.6, cette optimisation est illustrée par l'inversion du test T_2 et de ses deux branches.

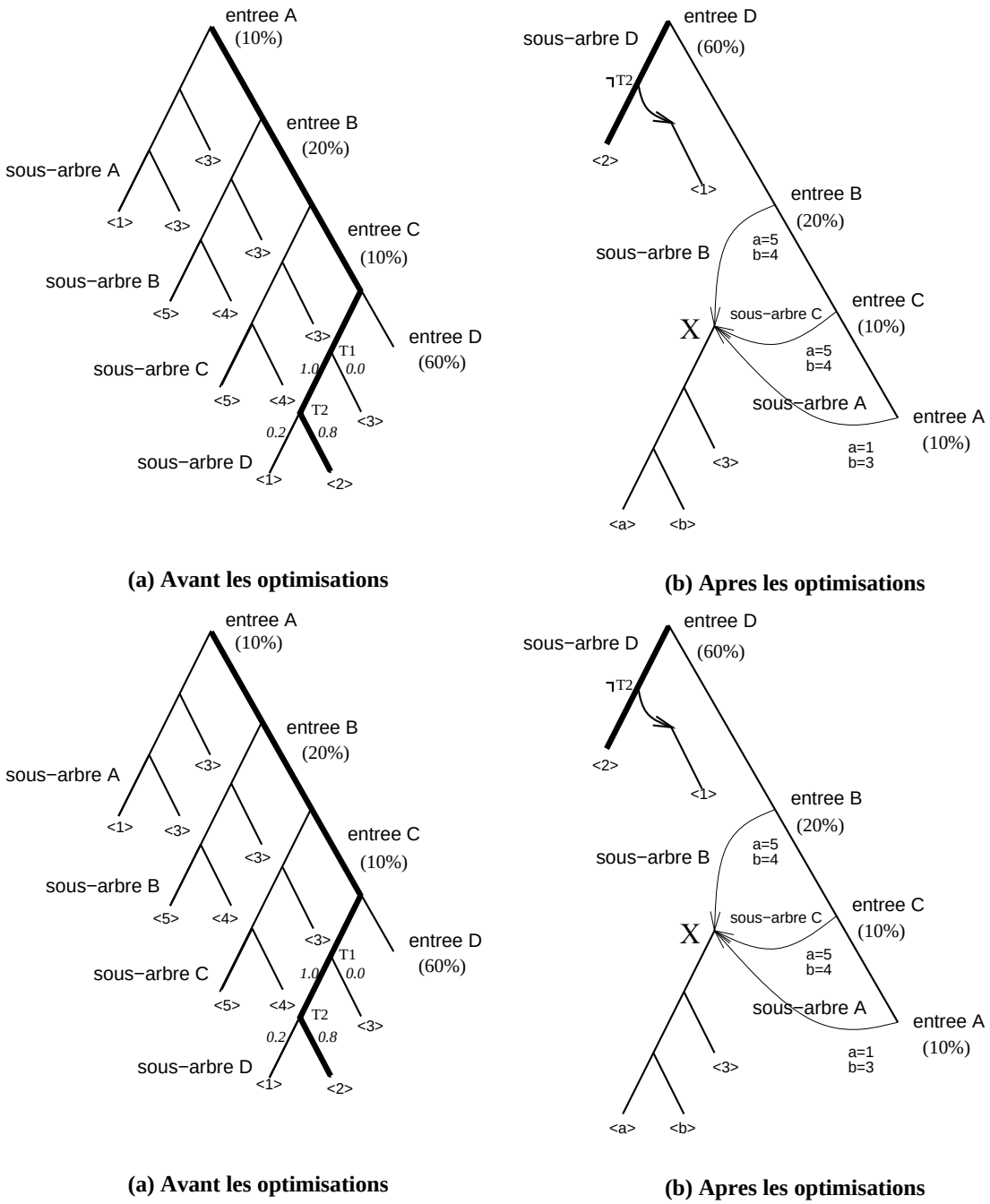


Figure 5.6 : Illustrations des optimisations de la vitesse d'exécution

5.5 Les optimisations de la taille du code

5.5.1 Elagage des états (G)

Comme nous l'avons cité dans la sous section 5.3.2.1, HIPPCO calcule les probabilités de visite de chaque état de l'automate. Dans certains environnements, il se peut que certains états aient des probabilités de visite nulles. C'est généralement le résultat d'évènements externes improbables (ayant des probabilités d'occurrence nulles). Ces états n'étant jamais visités, ils sont inutiles et peuvent être supprimés sans modifier la sémantique du protocole. C'est

précisément l'optimisation d'*élagage des états non accessibles.* Les transitions, dans les autres états, conduisant à ces états sont également supprimées par un effet de bord de l'optimisation d'*élagage des branches et sous-arbres* morts présentée dans la section 5.4.3.

5.5.2 Partage des sous-arbres similaires (N)

Cette optimisation est une extension de l'optimisation Esterel précédemment décrite. Elle consiste à partager les sous-arbres similaires. Deux sous-arbres sont dit similaires, si en supprimant certaines feuilles, ils deviennent identiques.

Cette optimisation est motivée par deux observations :

- beaucoup d'arbres ou sous-arbres ne diffèrent que par leurs feuilles. En effet, la structure de l'arbre représentée dans la figure 5.7 est très fréquente. Dans cet arbre, les deux sous-arbres[1] T_2 (a7 <1>)(a8 <3>) et T_2 (a7 <2>)(a8 <3>) sont similaires (seules les feuilles <1> et <2> diffèrent).
- l'état dans lequel l'automate bascule à partir d'un état donné est caractérisé par la combinaison des résultats d'un nombre limité de tests de l'arbre de l'état. Dans l'arbre de la figure 5.7, l'état dans lequel l'automate bascule est identifiable à partir des résultats des tests T_0 et T_2 : si T_0 est vrai alors le prochain état de l'automate est l'état S_1. Si T_0 est faux et T_2 est vrai le prochain état est l'état S_2. Si T_2 est faux le prochain état est l'état S_3.

L'optimisation proposée partage les sous-arbres similaires. Les feuilles différentes sont, dans les sous-arbres ainsi partagés, remplacés par des références à un sous-arbre, appelé *arbre des signatures*, qui permet l'identification de la valeur des feuilles partagées en fonction des résultats de certains tests antérieurs. Cette optimisation permet des réductions considérables des arbres. Le principe de cette optimisation est illustré par l'exemple simple de la figure 5.7. Cette optimisation permet de partager les deux sous-arbres T_2 (a7 <2>)(a8 <3>) ensemble ainsi qu'avec le sous-arbre T_2 (a7 <1>)(a8 <3>). L'arbre de signature T_0 (- <1>)(- <2>) permet d'identifier les feuilles partagées. Etant donné que cet arbre n'est composé que de nœuds `test`, son coût (la place qu'il occupe) est négligeable comparé au gain escompté des partages supplémentaires obtenus sur un exemple plus complexe.

5.3 Partage des sous-arbres intermédiaires (S)

Les optimisations précédentes réduisent la taille des automates en transformant les arbres de leurs états en graphes acycliques (DAGs). Ces DAGs contiennent des sous-arbres intermédiaires comme T_1 (a3 T_2)(a4 T_2) en figure 5.7.

Au même titre qu'il existe des similarités entre les sous-arbres, il existe des similarités entre les sous-arbres intermédiaires. En effet, beaucoup d'entre eux ne diffèrent que par leurs feuilles.

[1] Un sous-arbre est désigné par la syntaxe Nœud _test (action_then Nœud)(action_else Nœud)

L'optimisation proposée dans cette section partage les sous-arbres intermédiaires similaires. Ainsi deux sous-arbres intermédiaires similaires sont codés par une référence au même sous-arbre et par la liste d'arguments qui identifie la valeur des feuilles qui diffèrent.

La figure 5.7 illustre cette optimisation. Les sous-arbres T_1 (a3 T_2)(a4 T_2) sont similaires. Ils sont donc codés par une référence au sous-arbre $ST_1(x)$: T_1 (a3 [x])(a4 T2) avec x=1 ou 2 selon le sous-arbre.

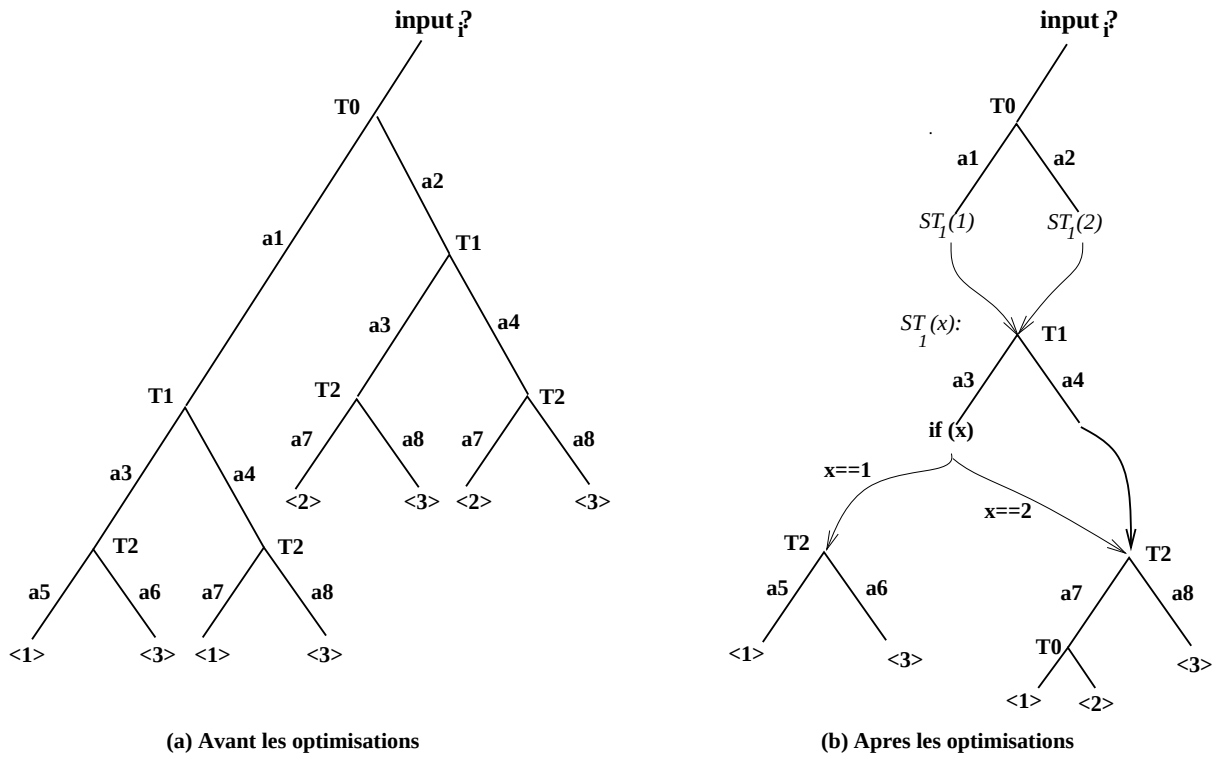


Figure 5.7 : Illustration des optimisations de la taille du code

5.6 Expérimentations

Afin d'évaluer les performances des différentes optimisations présentées dans ce chapitre, plusieurs expérimentations utilisant des implantations générées par HIPPCO ont été mises en œuvre. La partie contrôle de la phase transfert de données du protocole TCP a été complètement spécifiée en Esterel. La spécification détaillée est décrite dans [5.4]. Pour comparer les performances des implantations générées par HIPPCO, les mêmes expérimentations ont été effectuées en utilisant une implantation manuelle du protocole. Cette implantation a été directement dérivée de la version BSD de TCP.

Les expérimentations mettent en œuvre une application de transfert de fichier qui utilise différentes implantations d'un même protocole générées automatiquement par HIPPCO. L'application de transfert de fichier peut être décrite comme une application de sauvegarde: un client se connecte à un serveur et lui transmet un fichier de 1 MégaOctets (2048 paquets de 512 octets). Le transfert est unidirectionnel: le client envoie des données pures (i.e. sans

acquittements en piggy-backing[1]) et reçoit des acquittements (sans données). Inversement, le serveur reçoit des données pures et envoie des acquittements.

La spécification du protocole est similaire pour le client et le serveur, mais les prédictions sont différentes. Pour chaque expérimentation, deux implantations sont générées par HIPPCO: une est optimisée pour le serveur, l'autre est optimisée pour le client.

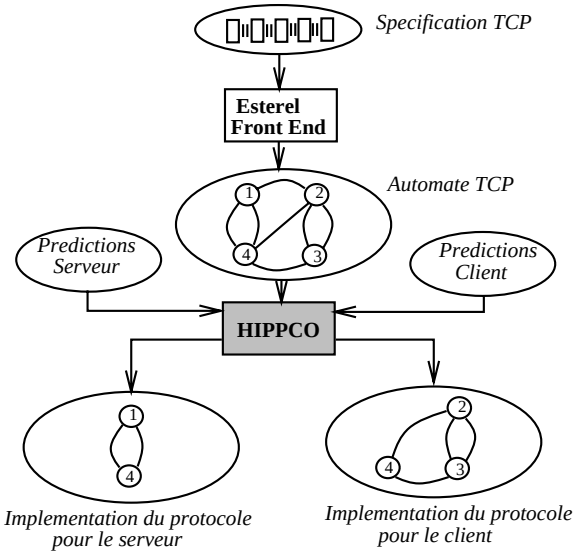


Figure 8 : Personnalisation des protocoles

Les prédictions utilisées sont de deux types:

1. les *probabilités d'occurrence des évènements* pouvant activer l'automate. Il est associé à chaque entrée de l'automate une probabilité d'occurrence.
2. les *probabilités* d'exécution des branches des tests de la spécification de l'automate. Il est associé à chaque test de la spécification la probabilité d'exécution de sa branche positive (correspondant à un résultat positif du prédicat).

Ces prédictions ont des valeurs par défaut qui sont attribuées par le concepteur du protocole. Elles sont attribuées sans connaissance a priori de leur utilisation par les applications mais selon la sémantique du protocole spécifié. Par exemple pour les spécifications de protocoles, il est très raisonnable de faire la prédiction que le total de contrôle est correct. La valeur de la prédiction par défaut du résultat du calcul du total de contrôle est alors mise à 99%. Dans le cas où des prédictions à priori sont difficiles à effectuer, on affecte à ces prédictions la valeur de 50%.

Ces prédictions peuvent être modifiées par les concepteurs des applications en utilisant des données de profilage ou les connaissances sur l'environnement d'exécution.

Cinq évènements externes peuvent activer l'automate du protocole dans nos expérimentations :

[1] Dans les connexions bidirectionnelles, les acquittements peuvent être insérés dans des paquets de données qui transitent en sens inverse. Cette technique appelée piggy-backing, qui permet de réduire le nombre de messages générés, n'est pas utilisée dans notre cas.

- *Alarm* : est le signal émis par le temporisateur lorsqu'il se termine. Ce signal indique qu'un paquet a été perdu dans le réseau. Dans nos expérimentations, nous faisons l'hypothèse que le réseau sous-jacent est assez fiable et que par conséquent la probabilité de perte d'un paquet est très petite. La probabilité de recevoir un signal *Alarm* a donc été positionnée à 2%. Cette valeur est constante pour toutes les implantations générées.
- *Input_Frame*: est l'entrée qui contient un paquet venant du réseau. Le serveur reçoit des données et envoie des acquittements, par conséquent ce type d'évènement y est plus probable. Sa probabilité a donc été positionnée à 97%. Le client envoie des données et reçoit des acquittements, la probabilité d'occurrence du signal *Input_Frame* y a été positionnée à 45%[1]. C'est également la valeur utilisée par défaut.
- *User_Input*: est l'entrée qui contient des données en provenance de l'application. Dans le serveur, aucune donnée de l'application n'est jamais reçue. La probabilité d'exécution de ce signal a alors été positionnée à 0%. Dans le client, on a fait l'hypothèse qu'il arrive autant de données du réseau, que de l'application. La probabilité du signal *User_Input* a donc été positionnée à 45%. C'est également la valeur utilisée par défaut.
- *End_of_Input*: est le signal qui indique que l'application a terminé sa transmission. Pour la même raison que le signal précédent, la probabilité d'exécution de ce signal a été positionnée à 0% dans le serveur et à 1% dans le client. La valeur par défaut est 1%.
- *Close_Connexion*: est le signal qui est envoyé par l'application pour fermer une connexion. Cet évènement n'apparaît qu'une seule fois par connexion. Sa probabilité d'exécution est alors très faible: elle a été positionnée à 1% dans le client, le serveur et par défaut.

Le nombre de tests présents dans la spécification de notre protocole est important. Les probabilités d'exécution de chacun de ces tests ne sont donc pas détaillées dans cette section. Elles apparaissent cependant dans la spécification présentée dans [5.5].

5.6.1 Description des différentes expérimentations

Les objectifs de ces expérimentations sont multiples: (1) évaluer les performances (vitesse d'exécution et des taille du code) des différentes optimisations présentées dans ce chapitre, (2) comparer les performances des implantations générées par HIPPCO à celles développées manuellement, (3) étudier l'impact des prédictions sur les performances obtenues.

Pour mener à bien ces objectifs, un ensemble d'expérimentations et un environnement de mesure ont été définis.

Pour étudier l'impact des prédictions sur la qualité des implantations, les performances d'implantations générées à partir de la même spécification mais utilisant des prédictions différentes ont été comparées. Ces implantations sont appelées:

- implantations *profilées par l'application*: il s'agit des implantations générées et optimisées à partir de prédictions attribuées par le concepteur de l'application client et

[1] On a fait ici l'hypothèse que les signaux *Input_Frame* et *User_Input* sont équiprobables au niveau du client.

serveur. Les prédictions étant différentes au client et au serveur, l'implantation du protocole au client est différente de l'implantation du protocole au serveur.

- implantations *profilées par défaut*: il s'agit des implantations générées et optimisées à partir des prédictions par défaut. L'implantation du protocole au client et identique à celle au serveur.
- implantations *spécialisées*: il s'agit des implantations profilées par l'application dans lesquelles toutes les branches ayant des probabilités d'exécution nulles ont été supprimées. L'implantation spécialisée, contrairement à une implantation optimisée, n'est plus complètement compatible avec l'implantation initiale. Elle ne peut donc pas être utilisée dans un autre environnement. Elles sont générées à partir de prédictions spécifiques au client et au serveur.
- implantations *profilées brutes*: il s'agit des implantations générées et optimisées à partir de prédictions brutes (toutes positionnées à 50%). Ces implantations sont assimilables à des implantations optimisées à partir d'une analyse purement statique du protocole.

De plus, les performances de ces implantations ont été comparées à celles d'une implantation manuelle de la version BSD du protocole TCP et de l'implantation Esterel (i.e. sans les optimisations d'HIPPCO).

Pour évaluer les performances des différentes optimisations présentées dans ce chapitre, des implantations « intermédiaires » ont été générées et testées. Ces implantations ont été générées en appliquant les optimisations successivement.

5.6.2 L'environnement d'expérimentation

Les performances des implantations de protocoles sont généralement évaluées en mesurant le débit et la latence qu'elles permettent d'obtenir. Cependant avec ces types de mesures, il est souvent très difficile de séparer dans les résultats la partie qui provient du traitement du protocole, de celle provenant de l'environnement d'exécution. Dans nos expérimentations, nous nous sommes principalement intéressés aux performances du protocole, par conséquent nous avons utilisé des métriques qui permettent une analyse plus fine des résultats. Les métriques utilisées sont le nombre d'instructions exécutées (*IC*), le taux d'échec du cache d'instructions (*miss_rate*), le nombre de cycles utilisés par instruction *CPI* qui mesure d'efficacité du pipelining, et la taille du code objet (segments texte et données en KiloOctets).

Comme il a été montré dans la section 5.4, la vitesse d'exécution d'une implantation peut être évaluée à partir des trois premières mesures. La quatrième mesure est utilisée pour comparer les tailles des implantations.

Nous avons développé notre propre environnement d'expérimentation. Cet environnement minimise les commutations de contexte, les copies et les allocations mémoire. Dans cet environnement, le protocole s'exécute au niveau utilisateur et est directement relié avec l'application. L'interface `socket` et la couche réseau ont été supprimés. Un paquet est « envoyé » en copiant son pointeur dans un buffer. Un paquet est reçu en lisant son pointeur dans ce buffer. Aucune copie ou manipulation de données n'est effectuée. Dans le but de minimiser les commutations de contexte et les communications interprocessus, le client et le serveur ont été reliés dans un seul processus et communiquent par le système de buffer, précédemment décrit.

Les expérimentations ont été effectuées (en 1995) sur une station de travail Alpha 200 de *Digital Equipment Corporation*. Cette station utilise un processeur Alpha 41466 à 166 MHz. La mémoire de cette machine est composée d'un cache primaire d'instructions et d'un cache de données de 8 KOctets chacun, d'un cache unifié secondaire de 2 MégaOctets et une mémoire principale de 64 MégaOctets. Tous ces caches sont d'accès direct et utilisent des blocs de 32 Octets.

Les résultats de ces expérimentations ont été obtenus en utilisant le logiciel ATOM de Digital. ATOM fournit une panoplie d'outils qui peuvent être utilisées pour instrumenter les implantations et obtenir entre autre des mesures sur le nombre d'instructions exécutées, le taux d'échec des caches et le nombre de cycles utilisés.

5.7 Analyse des résultats

Dans cette section, les résultats des expérimentations sont présentés et analysés. Le compilateur Esterel génère à partir de la spécification formelle de TCP un automate séquentiel intégré[1]. Le nombre théorique maximum d'états de l'automate est le produit du nombre d'états de chaque module qui le compose [5.12]. Esterel minimise le nombre d'états générés en supprimant ceux qui ne sont pas accessibles. Ainsi avec la spécification décrite dans [5.4], le nombre théorique d'états serait 4723920. Esterel le réduit à 21 états.

HIPPCO réduit le nombre d'états à 11, en partageant les états identiques[2]. Le compilateur Esterel génère quelquefois des états intermédiaires qui sont utilisés pour résoudre des problèmes de causalité (ils sont le résultat de l'utilisation de l'instruction `await tick`). Ces états sont quelquefois identiques. Par exemple dans notre spécification, ils sont utilisés comme états tampons juste avant de fermer une connexion. HIPPCO détecte ces états identiques et les partage.

Analyse des performances des optimisations d'HIPPCO

Le tableau 5.1 présente les résultats de performances des différentes implantations. Les chiffres du nombre d'instruction correspondent au nombre moyen nécessaire pour le traitement d'un paquet *entrant* par les différentes implantations. Les résultats de l'implantation BSD sont montrés pour référence.

En ce qui concerne les implantations d'Esterel, les performances diffèrent selon que le code est interprété ou compilé (le contrôle est inséré en ligne). Le code compilé est beaucoup plus rapide: il utilise environ 5 fois moins d'instructions que le code interprété. Ce résultat était prévisible. En effet, dans le code interprété, chaque action Esterel est implantée par une fonction. Un état est alors défini par une séquence de fonctions, qui sont appelées par l'interpréteur à l'exécution. Les codes interprétés sont alors très lents, ils exécutent un appel de fonction par action. En insérant en ligne ces fonctions (actions), le nombre d'instructions à exécuter est considérablement réduit. Cette technique permet également d'améliorer le pipelining et par conséquent le *CPI*.

[1] Cet automate est intégré dans le sens où l'abstraction de « module » de la spécification est supprimée dans l'automate.
[2] Deux états sont identiques, si leurs transitions le sont.

	Vitesse			*Taille* (*KOctets*)	
	IC	*Miss_rate*	*CPI*	*texte*	*données*
BSD	397	1.0	1.28	16.864	1.760
Esterel (code interprété)	2133	3.5	1.52	8.2	155.0
Esterel (code compilé)	-	-	-	1772.0	1.5
Esterel (code compilé + O)	421	3.86	1.30	45.7	1.2
HIPPCO (S)	465	4.07	1.32	30.480 (16.300)	3.52 (2.100)
+ (I)	423	3.7	1.33	30.816 (16.400)	3.60 (2.100)
+ (E)	206	0.0364	1.24	32.240 (19.648)	3.60 (2.100)
+ (T)	-	0.0345	1.23	33.136 (19.680)	3.60 (2.128)
+ (U)	206	0.0289	1.23	34.160 (21.072)	3.80 (2.384)
+ (P et G)	195	0.0217	1.23	6.672 (10.624)	1.00 (1.456)
+ (R)	193	-	-	-	-

Tableau 5.1 : Résultats des expérimentations

Cependant, l'application systématique de cette optimisation résulte en une explosion de la taille du code (1772 KOctets). L'application de l'optimisation de réduction de taille d'Esterel (O), qui partage les sous-arbres identique, résulte en une diminution de la taille à 45.7 KOctets.

Les optimisations d'HIPPCO sont montrées ensuite. L'utilisation de l'optimisation de taille (S) de HIPPCO augmente le nombre d'instructions de 44 instructions (10%). Cette optimisation partage les sous-arbres intermédiaires et par conséquent augmente le nombre d'indirections, ralentissant ainsi la vitesse d'exécution du code. Elle réduit la taille à 30.48 KOctets. Les performances du cache et du pipelining de cette implantation sont mauvaises. Ces résultats sont expliqués par la mauvaise localité du code du chemin commun.

La deuxième optimisation (I), qui insère en ligne les fonctions sur le chemin commun, réduit le nombre d'instructions de 42 (9%), mais résulte en un petit accroissement de la taille du code. Cette optimisation réduit les échecs du cache par un meilleur compactage du code.

La troisième optimisation (E), qui « étend » les branches partagées du chemin commun afin de supprimer les indirections diminue le nombre d'instructions exécutées de 217 (51%) et augmente la taille du code jusqu'à 32.24 KOctets. Les performances du cache sont considérablement améliorées. Le chemin commun étant plus petit que le cache (8 KOctets) et linéaire, le taux d'échec du cache passe de 3.7% à 0.0364%.

La quatrième optimisation (T) a un effet positif sur l'utilisation du cache et du pipelining.

La cinquième optimisation (U) ne modifie pas le nombre d'instructions exécutées. Cette optimisation implante les branches rares par des fonctions. Il n'affecte donc pas les performances du chemin commun. Il augmente les performances du cache comme on pourrait s'attendre.

Les optimisations de spécialisation (P et G), qui suppriment les branches et les états ayant des probabilités de visite nulles, réduit le nombre d'instructions de 11 (5.4%). Ces optimisations réduisent considérablement la taille du code (de 34.1 KOctets à 6.672 KOctets), et conduisent à une meilleure utilisation du cache.

Et finalement, l'optimisation (R), qui réorganise les arbres des entrées en fonction de leur probabilité d'occurrence réduit le nombre d'instructions de 2 (1%). Ce résultat correspond au ré-ordonnancement d'un arbre: au lieu de tester si le signal *End_Of_Connexion* est présent et ensuite si un paquet a été reçu, la réception d'un paquet (qui a une probabilité plus grande) est d'abord vérifiée.

Les résultats entre parenthèses dans les colonnes de la taille du code représentent les tailles des codes générés avec l'optimisation de partage de sous-arbres similaires (N). Les résultats obtenus avec cette optimisation sont très prometteurs. Cependant, dû à un manque de temps, nous n'avons pas pu exécuter les mesures de performance sur les codes utilisant cette optimisation. Toutefois étant donné que cette optimisation s'applique uniquement sur le chemin rare, la vitesse d'exécution du chemin commun ne devrait pas être pénalisée.

Analyse des performances d'HIPPCO et BSD

	I_c	O_c	*IC*	*miss_rate*	*CPI*	*Taille* (*KOctets*) *texte*
BSD	186	422	397	1.0	1.28	16.864
Esterel (interprété)	-	-	2113	3.5	1.52	8.2
HIPPCO profilage par l'application	143	147	204	0.0289	1.23	21.072
HIPPCO implantation spécialisée	139	142	193	0.0217	1.23	10.624
HIPPCO profilage brute	-	-	320	4.7	1.31	32
HIPPCO profilage par défaut	166	168	220	0.0298	1.25	24.84

Tableau 5.2 : Résumé des Résultats de Performance

Le tableau 5.2 est un résumé comparant les performances des implantations d'HIPPCO et de BSD. Plusieurs observations peuvent être faites à partir de ces résultats. Premièrement, l'idée préconçue que les compilateurs de description formelle génèrent des implantations de qualités médiocres semble se vérifier avec le compilateur Esterel (l'implantation Esterel est environ sept fois plus lente que celle de BSD).

Deuxièmement, en utilisant les optimisations appropriées, les implantations générées automatiquement peuvent être aussi rapides et parfois plus rapides (en terme de nombre d'instructions) que les implantations manuelles. En effet, les implantations générées par HIPPCO exécutent moins d'instructions que celle de BSD, même lorsque les prédictions par défaut sont utilisées. Notons que le nombre moyen d'instructions exécutés en entrée *IC* est égal à $(2\ I_c + O_c)/2$, où I_c (respectivement O_c) est le nombre d'instructions lors du traitement d'un paquet entrant (respectivement sortant). En effet, un accusé de réception est envoyé par le serveur après la réception de deux paquets). Pour I_c, le code d'HIPPCO optimisé en utilisant les prédictions par défaut, exécute 20 (11%) instructions en moins que l'implantation BSD. Lorsque les prédictions de l'application sont utilisées, le nombre d'instructions est réduit de 43 instructions (23%).

Le code d'HIPPCO spécialisé est encore plus rapide. Le traitement d'un paquet en entrée nécessite 47 (25%) instructions en moins que l'implantation BSD. Dans ce cas, les optimisations agissent plus agressivement et suppriment toutes les parties de code qui ont une

probabilité d'exécution nulle. Par exemple, dans nos expérimentations, le serveur ne reçoit que des données pures et aucun acquittement. Par conséquent, tous les sous-arbres qui traitent ces acquittements peuvent être supprimés. Cette technique permet de générer des implantations plus rapides et plus petites. Les implantations générées par HIPPCO utilisent moins d'instructions que celle de BSD grâce aux optimisations, mais aussi grâce à la structure des implantations: chaque état est implanté par une fonction différente. Ainsi certains tests qui sont indispensables dans l'implantation BSD sont supprimés dans celle d'HIPPCO. Par exemple, tester que l'état courant est *TCP_ESTABLISHED* ou que le protocole est dans l'état de retransmission n'est pas nécessaire dans HIPPCO, cela est fait implicitement. En comparaison dans la version BSD, l'automate est implanté par une seule fonction: à chaque activation de l'automate, il faut se repositionner sur l'état courant.

Le nombre d'instructions pour envoyer des acquittements (O_c) est également plus petit dans les implantations d'HIPPCO que dans celle de BSD. Le gain observé est même plus important, car la fonction de sortie (*tcp_output*()) de BSD n'utilise pas de prédiction (il n'existe pas l'équivalent de la technique de prédiction d'entête). Son implantation a été directement dérivée de sa spécification RFC et peu d'effort d'optimisation semble avoir été fait. Le code généré par HIPPCO est spécialisé pour l'envoi des acquittements et donc plus performant.

Les résultats, obtenus avec les implantations générées en appliquant l'intégralité des optimisations et en utilisant les prédictions par défaut, sont très voisins des résultats obtenus avec les implantations profilées par l'application. Leurs vitesses d'exécution sont très proches. La taille de la version profilée par défaut est 50% plus grande que la taille du code profilé par l'application. Ce résultat s'explique par une surestimation du chemin commun dans la version profilée par défaut (les prédictions de l'application permettent d'effectuer une identification plus fine du chemin commun). Cet accroissement de taille n'a cependant pas d'impact sur les performances du cache, car le code obtenu est très compact. Ces résultats démontrent la robustesse de notre approche aux imprécisions des prédictions.

Les résultats obtenus avec les prédictions brutes (toutes les prédictions sont égales à 50%) sont assez médiocres. Bien que le nombre d'instructions exécutées soit petit, l'utilisation du cache est mauvaise et la taille du code est importante. Ces résultats montrent les faiblesses et les limitations de l'approche purement statique. Cette approche qui consiste à donner le même « poids » à chacune des entrées et branches du programme ne permet pas une analyse du code assez précise pour y appliquer les optimisations que l'on propose de façon efficace. Ce résultat motive la conception d'un outil comme HIPPCO: les optimisations qu'il utilise ne pourraient pas être appliquées par des compilateurs de plus bas niveau (tels les compilateurs C), qui analysent uniquement les structures des programmes.

Les optimisations, qui spécialisent les codes, permettent d'obtenir de très bons compromis: *vitesse*/*taille*. Les raisons sont évidentes : les vitesses d'exécution des codes spécialisés sont grandes car toutes les instructions inutiles ont été éliminées, leurs tailles sont petites car les sous-arbres et les états inaccessibles ont été supprimés.

5.8 Conclusions

La génération automatique d'un code efficace devrait être effectuée dans le cadre d'une infrastructure haute performance pour l'exécution du protocole. En effet, le meilleur

compilateur ne peut compenser le coût d'une infrastructure qui génère des commutations de contexte et des copies inutiles de données. Une grande partie des travaux sur les compilateurs de protocoles a été effectuée avant l'avènement d'infrastructures haute performance pour les protocoles. De là, le besoin d'optimisation ne semblaient pas primordial à cette époque là.

La génération automatique d'un code efficace nécessite le développement d'un optimiseur et générateur de code. La génération automatique d'un code C de bonne qualité, à partir de la spécification Esterel est relativement une tâche facile, même en partant de la structure en arbre du compilateur Esterel.

L'un des résultats les plus importants de notre travail est l'affirmation que l'application systématique et automatique d'optimisation déjà connues permet de générer un code plus rapide que le code optimisé manuellement avec les mêmes optimisations. De même, l'application de ces optimisations devient presque triviale : par exemple dans la version BSD que nous avons utilisé la prédiction d'entête n'était pas implémentée pour le traitement des paquets sortants. Dans HIPPCO, les prédictions ont été appliquées systématiquement dans les deux cas (traitement des paquets entrants et sortants).

De même, il est facile de modifier certains algorithmes du protocole et de réappliquer les optimisations sur la spécification. Au contraire, pour les optimisations manuelles, la modification nécessite du temps et de l'expérience, vu qu'on ne dispose pas des spécifications modulaires à traiter.

Le gain en performance dû à l'utilisation de prédictions précises plutôt que des estimations brutes est assez impressionnant. L'application systématique d'optimisations basées sur ces prédictions sur tout le protocole améliore sensiblement les performances. Or il est facile de prédire le chemin commun d'un protocole. Ceci justifie les améliorations de performance obtenues.

En revanche, l'élimination des parties non utilisés du code ne semble pas avoir un impact significatif sur les performances. Ceci est dû à notre avis aux optimisations appliquées avant, qui ont amélioré sensiblement les performances. L'application systématique des optimisations semble donc réduire le besoin pour la spécialisation.

Notre travail devrait être poursuivi par un test à plus grande échelle afin de vérifier que le compilateur Esterel peut traiter des spécifications suffisamment large. Ce travail nous a mené à conclure qu'un compilateur de protocoles est un outil pratique pour la production d'implantations de réels protocoles, et non seulement un exercice académique. Il y a certainement plus de travail à faire pour le développement d'un tel outil, mais il ne semble pas y avoir des difficultés majeures qui bloquent le chemin vers le développement de cet outil.

Chapitre 6

Contrôle de Congestion en milieu Hétérogène

L'hétérogénéité des réseaux (en termes de bande passante disponible sur les liens, de délai de transmission de bout en bout et de taux d'erreurs sur les différents liens) et des récepteurs (en termes de capacité de mémoire et de calcul) représente un facteur structurel pour la conception des mécanismes de contrôle de transmission. Le protocole IP est adopté en tant que protocole fédérateur pour l'interconnexion de réseaux, fournissant ainsi un service « au mieux » aux applications réseaux. Ces applications point-à-point utilisant des protocoles de contrôle de transmission tels que TCP ou UDP ou multipoints utilisant des protocoles « sur mesure » partagent donc les ressources du réseau qui devrait donc prendre en compte les besoins de ces applications. La répartition des ressources entre ces multiples flots hétérogènes et la coopération de ces flots sur différents réseaux est un problème important à traiter. Une question fondamentale se pose : est ce que la couche réseau devrait jouer un rôle pour faciliter la cohabitation de ces flots ? Et si oui lequel ? Y aurait-il des solutions qui passent à l'échelle ? Ce chapitre propose une méthode originale permettant de faire cohabiter des flots en milieu hétérogène, de façon à garantir à chaque flot sa part de bande passante, tout en maintenant la complexité et les états à la bordure du réseau. C'est au prix d'une bonne modélisation de ces flots dans ces environnements que l'on peut envisager un déploiement harmonieux de ces flux hétérogènes dans ces milieux hétérogènes.

6.1 Introduction : Limites des modèles actuels

Un des défis des applications réseaux « modernes » est l'utilisation efficace et coopérative de ses ressources. Les exigences croissantes des applications dites multimédia, font apparaître le besoin de s'adapter non seulement aux capacités du réseau (contrôle de congestion) mais également aux capacités des récepteurs (contrôle de flux). Les applications d'aujourd'hui adoptent globalement trois comportements :

1. Transmission à un débit fixe.
2. Estimation de l'état du réseau et adaptation de bout en bout, au niveau applicatif, ou à travers des protocoles comme TCP.
3. Utilisation de mécanismes dans le réseau pour assurer l'équité.

Nous proposons une quatrième approche - hybride - dans laquelle on maintient l'intelligence du contrôle de congestion à l'extrémité du réseau, tout en exploitant des mécanismes très simples présents dans les routeurs du cœur du réseau. Nous présentons ci-dessous ces quatre approches.

6.1.1 Transmettre à un débit fixe

La première approche, malheureusement adoptée par une grande majorité des applications multimédia, n'est naturellement pas satisfaisante, étant inefficace et inéquitable. En outre, ces applications ne pourront bénéficier des améliorations des débits disponibles de bout en bout. Dans le meilleur des cas, on propose à l'utilisateur de choisir parmi une série limitée de codages ou de débits prédéfinis. C'est le cas par exemple de certaines applications d'audio sur Internet qui proposent le choix entre un codage LPC, GSM, ADPCM, etc. Dans le cas du client de Real Networks [6.1], l'utilisateur définit lui-même la bande passante dont il dispose (33Kb, 56Kb, 1Mb, etc.) en fonction du type d'accès.

6.1.2 Le contrôle de bout en bout TCP-courtois

A la fin des années 80, il a fallu gérer la croissance exponentielle d'Internet et éviter son implosion en gérant la congestion au sein du réseau. Pour partager les ressources entre des flots concurrents, on s'est reposé presque exclusivement sur des mécanismes de bout en bout, en particulier au travers de mécanismes de contrôle de congestion intégrés au protocole TCP (Transmission Control Protocol) [6.2]. En réagissant de concert à des signaux de congestion, les hôtes à chaque bout des connexions parviennent à un partage équitable de la ressource critique. Ces derniers adoptent en effet tous le même schéma d'adaptation, à croissance linéaire et décroissance géométrique en cas de congestion (AIMD – Additive Increase, Multiplicative Decrease). Des données générées par des sources TCP et empruntant un même routeur congestionné subissent alors le même taux de perte et perçoivent donc un même état de congestion du réseau. Ceci permet une convergence des sources [6.3] vers un état efficace et équitable, comme présenté sur la figure 6.1.

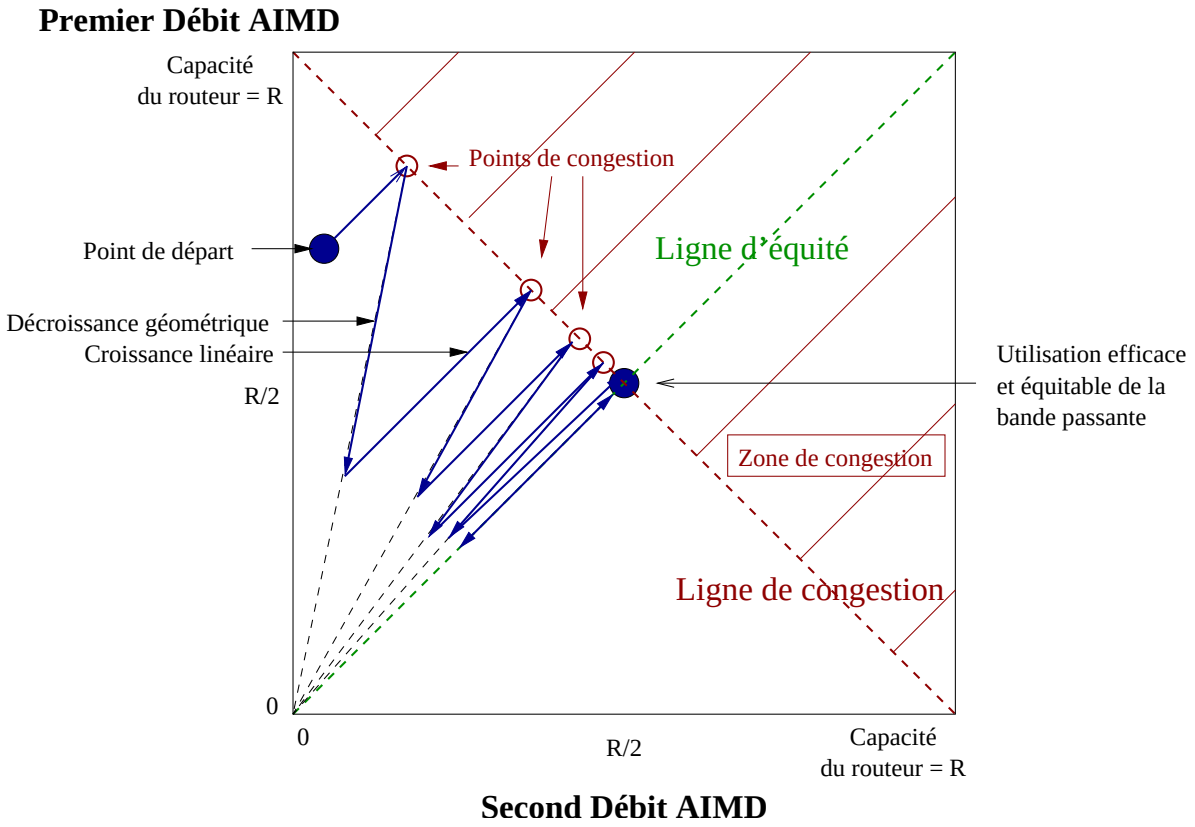


Figure 6.1 : Deux sources AIMD partageant une même ressource critique

Le succès de cette approche est incontestable mais repose tout de même sur un certain nombre d'hypothèses que l'arrivée de nouveaux médias et modes de transmission remettent en cause.

6.1.2.1 Un unique schéma d'adaptation TCP-courtois

L'équité entre les flots n'est possible que si l'ensemble des applications s'adaptent de manière unique, c'est à dire TCP-courtoise, à la congestion du réseau [6.4]. Même en suivant le schéma d'adaptation AIMD de TCP, les sources ne convergent vers une situation équitable que si les paramètres de croissance et de décroissance sont identiques dans toutes les sources. Il pourrait être intéressant de décorréler « la garantie de fiabilité » de « la régulation du

débit », ce qui nécessite d'adopter des mécanismes d'adaptation AIMD orientés débit (par opposition à des mécanismes orientés fenêtre comme pour TCP). De plus, le mode d'adaptation AIMD n'est pas adapté à tous les types de flots : c'est le cas de nombreux flots multimédias, qui ont des contraintes temps réel ou de bande passante. Pour ces dernières, les grandes variations de débit du mode AIMD entrainent une dégradation notable de la qualité perçue [6.5]. C'est pour cette raison qu'ont été proposé des protocoles d'adaptation de débit « orienté équation », comme le TCP-Friendly Rate Control (TFRC) [6.6], qui ajustent leur débit de manière plus lisse, en se basant sur des modèles de flots TCP. Toutefois, ces protocoles sont plus lents à réagir suite à des changements d'état du réseau. Enfin, l'implémentation de protocoles de bout en bout TCP-courtois pour tout type de flot est souvent difficile à mettre en œuvre : il faut en effet que l'application sache s'adapter aux variations du flot reçu. Nous croyons qu'il est plus réaliste de considérer que nous serons toujours confrontés à la présence de flots plus ou moins agressifs ou réactifs dans l'Internet.

6.1.2.2 Une perception unique de l'état du réseau permet la convergence des flots TCP-courtois

La deuxième hypothèse sur laquelle repose la validité du modèle actuel est que des flots en compétition ont une même vision de l'état du réseau. Différentes sources TCP-courtoises en compétition pour une même ressource perçoivent un même taux de perte et peuvent alors converger vers un état stable et équitable. Ceci implique que ces sources partagent un unique point de congestion. On constate ainsi que la multiplicité des points de congestion sur le chemin d'un flot TCP a un impact catastrophique sur ce dernier [6.7]. En pratique, le goulot d'étranglement sur le chemin d'une connexion TCP est souvent l'unique point de congestion. Il existe toutefois d'autres configurations dans lesquelles l'hypothèse de point de congestion unique n'est pas valable.

Dans le cas d'un *flot multipoint*, une source peut partager simultanément de multiples points de congestion sur l'arbre multipoint, avec des flots différents, ce qui complique considérablement le processus d'adaptation. Nous avons déjà évoqué les aspects inhérents au multipoint qui rendent l'adaptation TCP-courtoise de ces flots délicate : Doit-on adapter son débit au point le plus congestionné de l'arbre multipoint, et donc s'adapter au récepteur le plus lent ? De plus, trouver un débit « équivalent » à celui de TCP ne peut se limiter à trouver le plus grand taux de perte et délai de bout en bout parmi les récepteurs, ni à maintenir l'équivalent d'une « taille de fenêtre maximale » avec l'ensemble des récepteurs. Même TCP-courtois, un flot multipoint partage de nombreux points de congestion avec d'autres flots, et les acteurs intervenant dans les estimations de l'état du réseau, et donc dans l'adaptation de débit, sont multiples. Qu'ils soient orientés émetteurs ou orientés récepteurs, les interactions avec d'autres flots multipoints ou point à point sont donc non seulement difficiles, mais hasardeuses.

Dans le cas des *réseaux dynamiques*, tels que les réseaux ad-hoc ou les constellations de satellites, le point de congestion n'est pas fixe, et les flots en compétition changent en raison de l'instabilité des routes. Ce ne sont alors non seulement les estimations de l'état du réseau qui sont instables, c'est le réseau lui même.

Proposer des mécanismes à l'intérieur du réseau et assouplir le principe stipulant que « toute l'intelligence devrait être à la périphérie » s'avérera de plus en plus utile. Ce sera même probablement nécessaire, avec l'hétérogénéité croissante du réseau, l'arrivée de nouveaux types de liens et nouveaux modes de transmission.

6.1.3 Les approches réseaux

Des mécanismes « dans le réseau » d'ordonnancement des paquets et/ou de gestion de file d'attente ont été proposés pour protéger certains flots d'autres flots agressifs. Une première catégorie d'algorithmes ayant pour but de répartir équitablement la bande passante au sein d'un routeur, maintiennent un file d'attente par flot [6.8, 6.9, 6.10]. C'est le cas par exemple de Stochastic Fair Queuing (SFQ)[1][6.28] mais également de Deficit Round Robin (DRR) [6.11], avec lequel on obtient une équité presque optimale avec une complexité limitée. Une autre catégorie d'algorithmes cherche à obtenir cette équité à partir d'une seule file d'attente. Les paquets devant être éliminés sont choisis à partir des états par flots maintenus dans le routeur. C'est le cas de FRED [6.12] qui ne fournit pas vraiment d'équité entre les flots mais qui améliore cette dernière pour les flots TCP face aux flots CBR plus agressifs.

Toutefois, maintenir ces états par flot a un coût non négligeable, et pose des problèmes de passage à l'échelle, en particulier au cœur des réseaux à très haut débit ou transportant un grand nombre de flots, mais également dans les routeurs satellites tels que ceux décrits dans le chapitre 3 (vers la fin de la section 3.1). La mise à jour de ces états, leur transfert éventuel d'un routeur à l'autre, dans le cas des constellations de satellite que nous avons décrites, peuvent être très coûteux, voire mener à des instabilités dans le réseau. La simplicité au cœur du réseau est d'ailleurs une des clés du succès de l'Internet aujourd'hui.

Mais surtout, cela suppose de plus qu'un routeur au cœur du réseau puisse identifier ce qu'est un flot. On considère habituellement qu'un flot est constitué des données échangées de bout en bout par deux *applications*. Toutefois, lorsque les flots sont sécurisés et transportés via IPsec [6.13], les en-têtes TCP et UDP, cryptées, ne sont pas accessible à un routeur intermédiaire. Doit on alors considérer le flot au niveau réseau et non plus applicatif ? Un flot serait alors l'ensemble des données échangées par deux *hôtes*. A nouveau, l'utilisation de mécanismes tels que NAT (Network Address Translation), mis en œuvre de plus en plus fréquemment pour pallier la pénurie d'adresse IPv4, met cette approche en péril. Deux nombreux hôtes d'un même réseau local peuvent en effet se voir attribuer la même adresse IP vu du cœur du réseau. Enfin, comment identifier et gérer les flots manipulés de manière agrégée, dans des tunnels IP par exemple, où les paquets sont encapsulés ? Ces difficultés traduisent en fait la contradiction entre la gestion individuelle de *flots* dans les routeurs du réseau avec la philosophie d'un réseau orienté *datagramme*, dans laquelle la notion de « connexion » ou de « circuit virtuel » n'existe pas.

Stoica et al. ont proposé Core Stateless Fair Queueing (CSFQ) [6.14] dans le but d'approximer le comportement de Fair Queueing en maintenant les états par flots au bord du réseau et en les supprimant du cœur du réseau. Cao et al. ont également proposé Rainbow Fair Queuing (RFQ) [6.15], une approche similaire qui évite le calcul du débit limite dans les routeurs et qui est plus adaptée aux applications utilisant du codage en couches. Ces deux protocoles suppriment l'état par flot dans les routeurs de cœur de réseau mais gardent tout de même certaines procédures complexes et coûteuses en calcul, telles que l'estimation d'un seuil au delà duquel il faut jeter les paquets. C'est pourquoi CHOKe (CHOose and Keep) [6.16] propose d'introduire dans les routeurs des mécanismes aussi simples que ceux définis

[1] SFQ maintient en fait un nombre grand mais borné de files d'attente, et associe à chaque flot une file d'attente, à l'aide d'une fonction de hachage.

dans Random Early Discard (RED) [6.29]. Adoptant une approche intermédiaire entre les algorithmes plus complexes de Fair Queueing, et les mécanismes simples de RED, il propose ainsi d'améliorer l'équité entre les flots TCP et les flots plus agressifs, mais ne prétend pas proposer une solution garantissant l'équité. Dans tous les cas, ces protocoles sans état par flot ne proposent pas une solution parfaite au problème de l'équité dans des contextes réalistes. L'ensemble de ces algorithmes sont classifiés sur la figure 6.2.

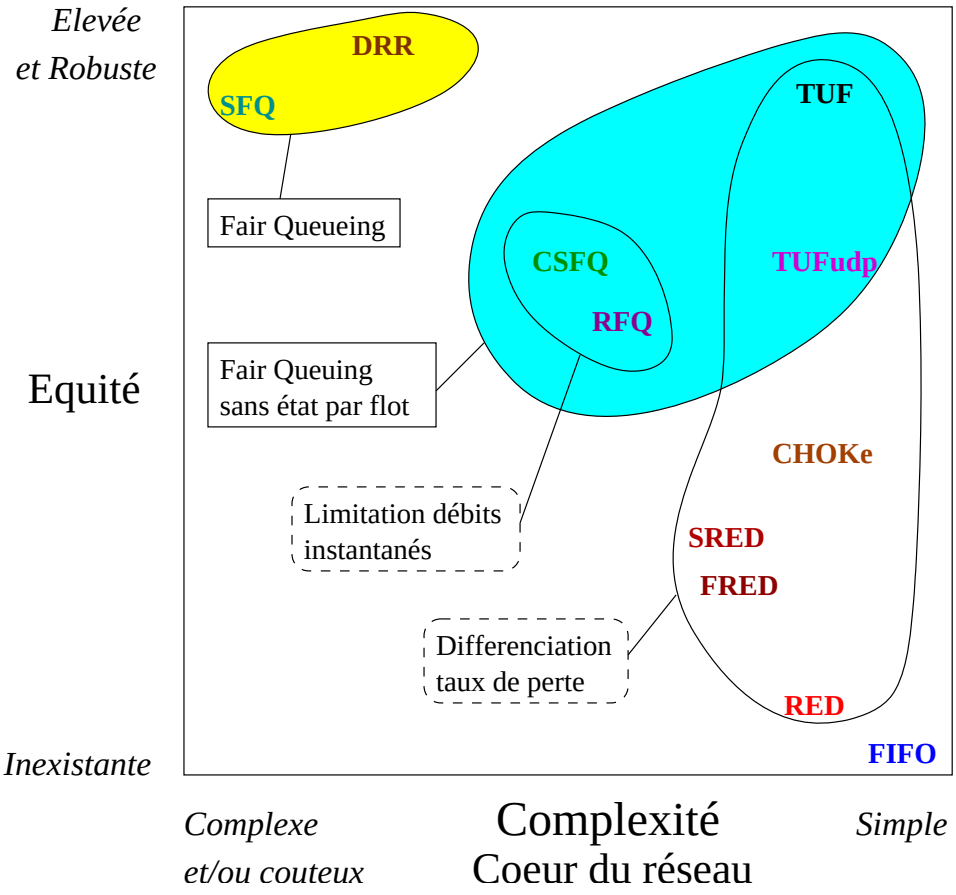


Figure 6.2 : Complexité et Equité pour différentes approches « réseau » pour fournir l'équité entre les flux

6.1.4 Adopter une approche hybride

L'utilisation de ces seuls mécanismes au sein du réseau pour réguler l'ensemble des flots ne fournit pas une solution optimale. En effet le réseau consomme des ressources pour transporter des paquets qu'il va ensuite éliminer. La figure 6.3 en présente un exemple. Le flot principal A partage un premier lien de capacité R avec un autre flot B, et un deuxième lien avec deux autres flots C et D. En utilisant exclusivement les mécanismes de Fair Queueing, les flots partagent localement la bande passante et obtiennent les débits $R/2$ pour le flot B et $R/3$ pour les 3 autres flots A, C, D.

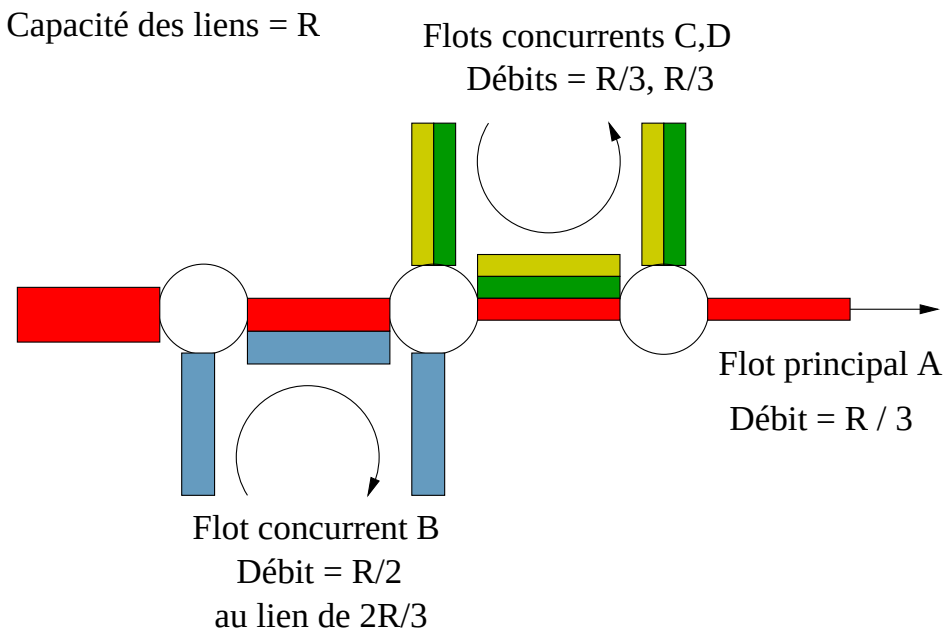


Figure 6.3 : Utilisation inefficace du réseau

Supposons maintenant que notre flot principal *A* fait du contrôle de congestion *de bout en bout*. Il réduit son débit d'émission jusqu'à obtenir le débit du goulot d'étranglement, à savoir $R/3$. Le flot *B* obtient ainsi un débit de $2R/3$ et les flots *C* et *D* obtiennent toujours le même débit $R/3$. Le contrôle de bout en bout du flot principal A permet donc d'augmenter les ressources obtenues par le flot *B* sans pénaliser d'autres flots. L'objectif à atteindre n'est donc pas simplement le partage équitable des ressources dans chaque routeur, qui conduit à l'inefficacité que nous venons de montrer. C'est pourquoi il est important coupler des mécanismes dans le réseau avec un contrôle de bout en bout.

Les mécanismes réseaux existants ignorent le comportement de bout en bout des flots qu'ils essayent de réguler. Une première classe d'algorithmes (voir figure 6.2), en particulier les algorithmes sans état dans les routeurs, cherchent à réguler le débit instantané des flots. Ils sont alors inéquitables avec les flots réactifs, qui, suite à une congestion, utilisent moins de bande passante que ce qui leur revient (voir section 6.2.2). C'est le cas par exemple de CSFQ ou de RFQ. Une deuxième classe d'algorithme, dont font partie DRR et SFQ, se concentrent sur les débits moyens, mais maintiennent alors un état par flot.

Pour conclure, dans un contexte de flots multipoints et d'un Internet avec liens satellites - géostationnaires ou constellations de satellites - il semble peu réaliste de défendre un contrôle exclusif de bout en bout. Il apparaît utile de les coupler avec des mécanismes au sein du réseau pour permettre un partage des ressources. Ce chapitre propose un tel mécanisme, qui couplé à un contrôle de bout en bout réalise un partage équitable des ressources. La force de notre approche réside dans les points suivants :

- Les mécanismes introduits dans les routeurs sont *simples* et ne nécessitent pas d'état par flot. Comparé aux autres algorithmes proposant une équité raisonnable, notre coût d'implémentation nous semble être un des plus bas.
- Contrairement aux autres algorithmes d'ordonnancement ou de gestion de file d'attente sans état par flot, nous ne cherchons pas à maintenir les débits instantanés

égaux. Nous prenons en compte *la nature réactive du flot*, et ajustons les taux de perte de façon à obtenir des débits « moyens » égaux. Cette nouvelle approche nous permet d'obtenir un bon niveau d'équité et de le maintenir dans des environnements hétérogènes où cohabitent des flots de nature différente.

6.2 TUF : Tag-based Unified Fairness

6.2.1 Objectifs d'équité : Max-Min équité

Comme nous venons de le voir, notre objectif d'équité ne peut pas se résumer en un partage « local » de la bande passante dans chaque routeur. La notion d'équité entre les flots est une propriété plus globale. Elle est d'autant moins intuitive dans un environnement hétérogène avec des flots multipoints.

Commençons par définir un ensemble d'acteurs, de ressources à partager entre ces derniers, ainsi qu'une mesure de la satisfaction de chaque acteur. Dans le cas des réseaux, cette dernière peut être le délai de bout en bout, la puissance[1], ou dans le cas qui nous intéresse ici le débit moyen reçu. On peut ensuite définir un indice d'équité [6.17], qui mesure l'équité entre les acteurs, ou définir des propriétés qualitatives comme l'équité max-min [6.18] ou l'équité proportionnelle [6.19, 6.20]. Nous nous intéresserons par la suite au critère d'équité, appelé la max-min équité : une allocation des ressources est dite *max-min équitable* s'il n'est pas possible d'augmenter la satisfaction d'un acteur sans réduire celle d'un acteur plus défavorisé. En somme, on cherche en priorité à améliorer la satisfaction des acteurs les plus défavorisés.

Il est possible d'associer des poids à chaque flot. On définit alors la satisfaction du flot F_i par $w_i \times R_i$ où R_i est le débit reçu et w_i le poids associé au flot i.

6.2.1.1 Max-Min équité pour le multipoint

Dans un réseau hétérogène multipoint, on peut distinguer trois types de flots :

- Les flots multipoints mono-débit pour lesquels tous les récepteurs reçoivent les données transmises à un débit unique.
- Les flots multipoints multi-débits pour lesquels chacun des récepteurs d'un flot peut recevoir les données à débit différent correspondant à sa capacité et aux caractéristiques du chemin entre la source et ce récepteur.
- Les flots point-à-point, que l'on peut considérer comme un cas particulier des deux premiers.

Pour les flots multipoints mono-débit, la satisfaction d'une session multipoint mono-débit est mesurée par le débit choisi par la source et qui correspond au minimum[2] des débits souhaités par tous les récepteurs. La plupart des travaux sur l'équité entre flots multipoints ne considèrent d'ailleurs que ce cas, et mettent l'accent sur l'équité *inter-flots*, oubliant ainsi

[1] La puissance est le rapport entre le débit reçu et le délai de bout en bout.
[2] On peut adopter d'autres politiques, comme par exemple choisir le 90ème-percentile des débits souhaités ce qui revient à ne pas considérer les 10% de récepteurs les plus lents.

l'équité *intra-flots*, entre les récepteurs d'un même flot. En effet, pour les flots multi-débits, il est plus logique de considérer indépendamment chaque récepteur [6.21]. Ainsi, un membre d'un groupe multipoint ne doit pas être pénalisé par la présence de membres plus lents, et doit obtenir le débit qu'il aurait eu s'il avait été seul dans le groupe, appelé parfois appelé débit « isolé » (isolated rate) [6.22, 6.23]. La définition de la max-min équité rentre parfaitement dans ce cadre. On considère alors le débit reçu par chaque récepteur comme une mesure de sa satisfaction.

6.2.1.2 Max-Min équité et partage équitable des ressources

Nous avons évoqué les mécanismes de partage équitable de bande passante dans les routeurs. En voici une définition plus formelle :

Soit C la capacité du routeur sur l'interface de sortie considérée. Soient n flots $F_1, \ldots, F_n$. Soient ρ_i le débit entrant du flot i, en amont, et R_i le débit alloué au flot i, en aval. Soit R_{fair} le *débit limite* défini comme l'unique solution de $C = \sum R_i = \sum \min(\rho_i, R_{fair})$

Dans un routeur implémentant un partage équitable des ressources, le débit limite R_{fair} correspond à la bande passante maximale que peut obtenir un flot, c'est à dire pour tout i, $R_i = \min(\rho_i, R_{fair})$.

Un flot F_i est alors :

- Contraint : le débit obtenu est, comme pour tous les autres flots contraints, égal au débit limite R_{fair} et supérieur à celui des flots non contraints : pour tout i, j, F_i, F_j contraints, $R_i = R_j = R_{fair}$
- Non contraint : Le débit obtenu est égal au débit demandé et inférieur au débit limite : pour tout i, j, F_i non contraint, F_j contraint, $R_i = \rho_i \leq R_j = R_{fair}$

Supposons que tous les flots du réseau adoptent un protocole de contrôle de congestion de bout en bout, que l'on peut définir comme un mécanisme permettant d'occuper le maximum de bande passante sans perte. On peut montrer que couplé à des mécanismes de partage équitable des ressources dans les routeurs, ce protocole de contrôle de congestion permet d'obtenir une allocation max-min équitable des ressources du réseau. Ce résultat reste vrai avec des flots multipoint multi-débits où chaque récepteur ajuste son débit avec des mécanismes comme RLM [6.21].

6.2.2 Equité et débits instantanés

Nous nous intéressons à l'équité entre flots élastiques[1] pour lesquels on mesure la satisfaction en termes de débit moyen reçu. Contraindre le débit instantané de chaque flot au débit limite ne permettra pas d'obtenir une distribution équitable de la bande passante entre les flots. En effet, suite à un signal de congestion, les flots réactifs réduisent leur débit, sous-utilisant ainsi la bande passante qui leur est allouée. On appelle cette première cause d'inéquité *l'effet variation*. On peut en voir un exemple sur la figure 6.4, où la bande passante est partagée entre un flot réactif comme TCP-Reno, et un flot CBR très gourmand.

[1] Un flot élastique est un flot dont le débit s'adapte, en particulier aux conditions du réseau.

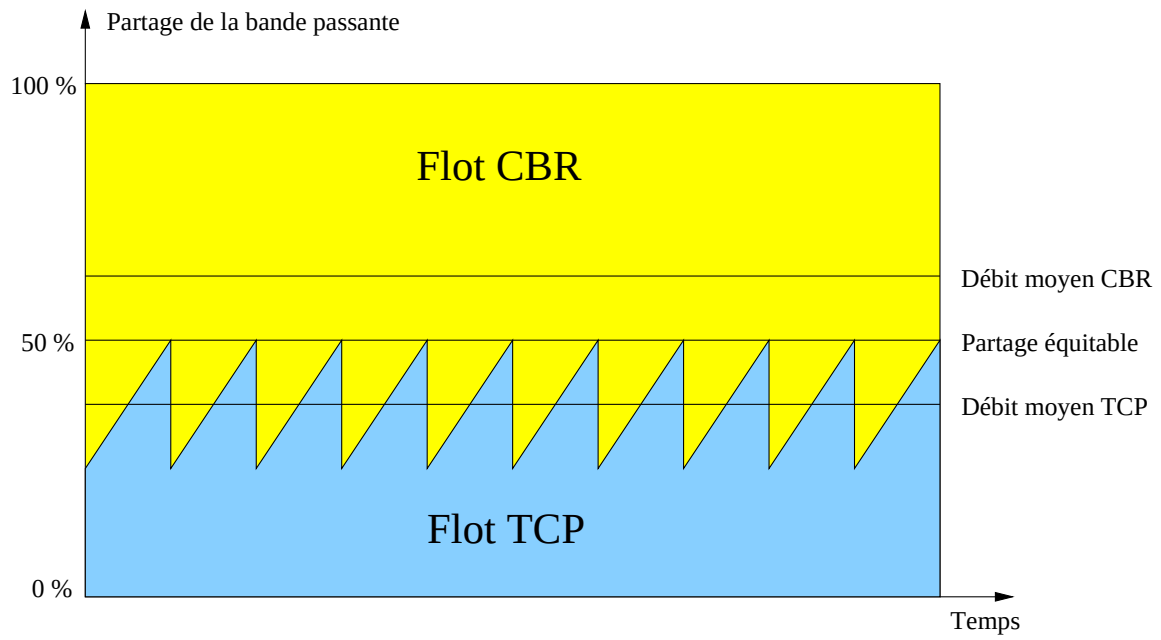


Figure 6.4 : L'effet variation

De plus, les performances des flots réactifs comme TCP peuvent être sérieusement compromises par une augmentation soudaine du taux de perte, par exemple lorsque le débit du flot dépasse le débit limite. Nous appelons cette deuxième cause d'inéquité *l'effet rafale*. Lorsque le délai de bout en bout d'une connexion TCP devient non négligeable, cette dernière réagit tardivement aux signaux de congestion, et subira donc une rafale de perte. Un deuxième exemple de situation ou peut se produire l'effet rafale est la diminution brusque du débit limite, suite à l'arrivée soudaine de nouveaux flots. Cet *effet rafale* aura un impact important et durable sur le débit moyen de la connexion TCP, comme montré sur la figure 6.5.

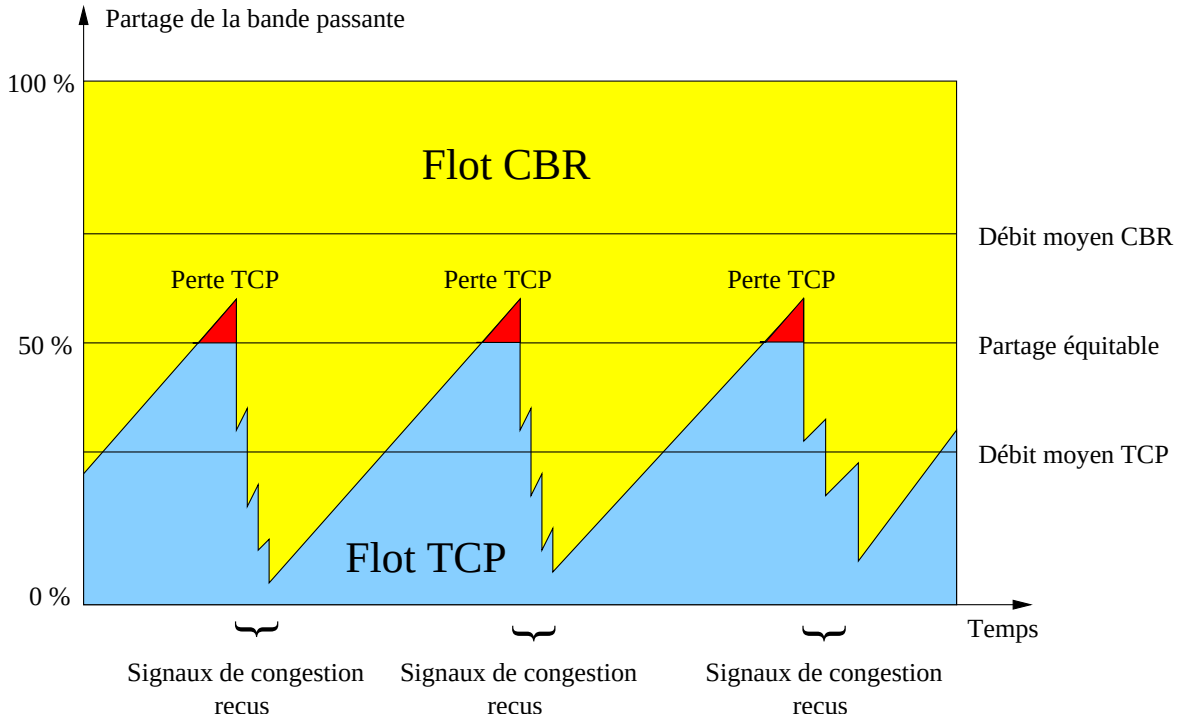


Figure 6.5 : L'effet rafale

Dans l'algorithme que nous proposons dans ce chapitre – appelé TUF ouTag-based Unified Fairness [6.24] – nous évitons d'ajuster les débits instantanés en les maintenant en-deçà d'un débit limite. Nous ajustons plutôt les taux de perte en les adaptant aux comportements de bout en bout de façon à obtenir des débits moyens équitables.

6.2.3 Description générale

C'est en différenciant les taux de perte entre les flots que l'on peut obtenir un partage équitable de la bande passante. Comme on ne veut pas maintenir un état par flot dans les routeurs, il faut ajouter de l'état dans les paquets pour obtenir des taux de perte différents au niveau des routeurs. Cet état prend la forme d'une « étiquette » dans l'en-tête du paquet. On désigne par **Tag** la valeur numérique de cette étiquette. Les routeurs du cœur du réseau se basent exclusivement sur ces étiquettes pour prendre la décision d'éliminer ou d'accepter un paquet. L'étiquetage est fait à la frontière du réseau, (ou au niveau de la source elle même), dans un nœud (le « tagger ») qui maintient un état par flot. On peut comparer cette étiquette au DS-codepoint de Diffserv. L'architecture globale est décrite sur la figure 6.6. L'étiquetage ne doit pas dépendre de l'état du réseau vu que l'on souhaite précisément s'affranchir des estimations sur l'état du réseau.

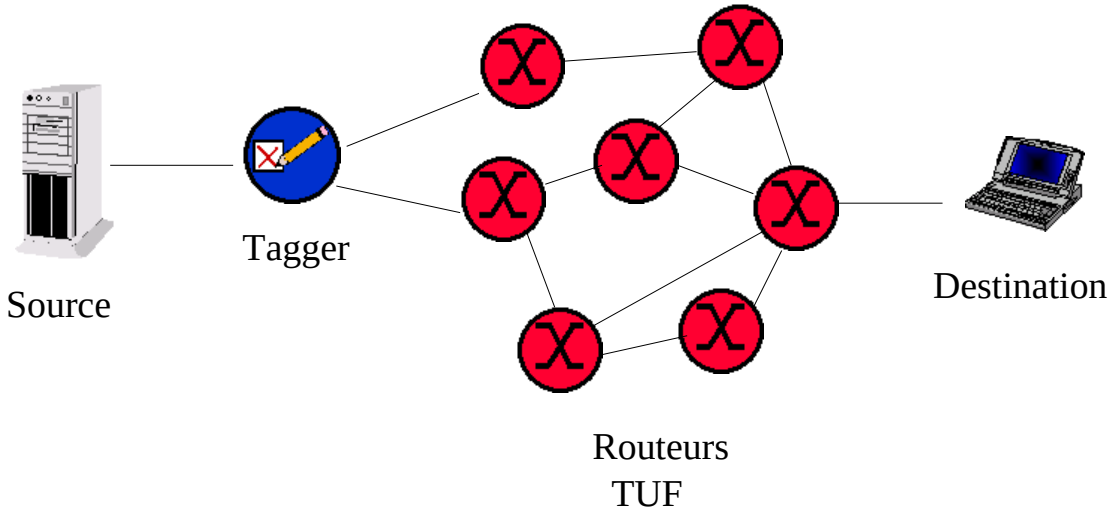


Figure 6.6 : Architecture d'un réseau TUF

Nous allons maintenant décrire le fonctionnement des files d'attente TUF du cœur du réseau puis l'algorithme d'étiquetage.

6.2.4 Gestion de file d'attente dans les routeurs TUF

C'est en se basant sur la valeur numérique des étiquettes (**Tag**) présentes dans les paquets qu'un routeur va prendre la décision d'accepter ou de rejeter un paquet. Nous avons choisi d'associer à la valeur de l'étiquette une propension au rejet : plus le **Tag** est élevé, plus le paquet sera susceptible d'être rejeté. Comme les files FIFO, la file d'attente TUF doit préserver l'ordre des paquets pour éviter les conséquences néfastes sur des protocoles tels que TCP.

Lorsque la file d'attente est pleine, notre routeur TUF va éliminer de sa file le paquet dont le **Tag** est le plus élevé, tel que présenté sur la figure 6.7. On remarquera qu'une telle file d'attente présente exactement le même profil de perte qu'une file FIFO soumis au même trafic - seuls les paquets rejetés changent.

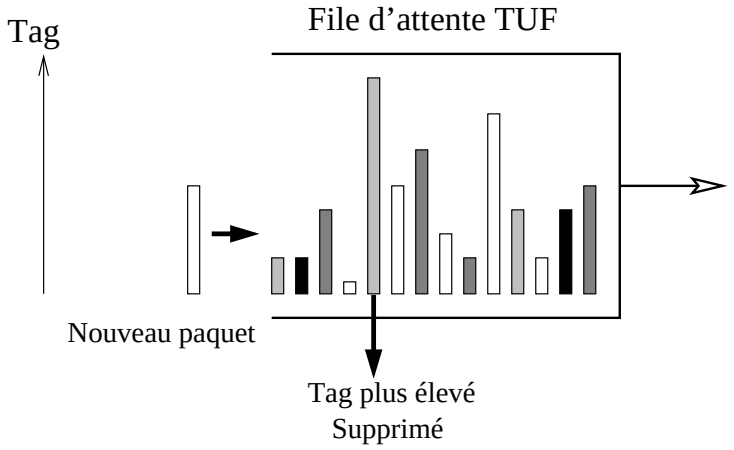


Figure 6.7 : Gestion de file d'attente dans un routeur TUF

Avant de décrire comment nous étiquetons les paquets pour obtenir un partage équitable des ressources, nous commençons dans la prochaine partie par modéliser le comportement de notre file d'attente.

6.2.4.1 Comportement d'une file d'attente TUF

Nous avons simulé une file d'attente TUF, alimentée par un trafic poissonnien. Dans cette simulation, la charge offerte est $\rho = 2$, les Tags sont uniformément répartis entre 0 et 1. La figure 6.8 montre les résultats des simulations.

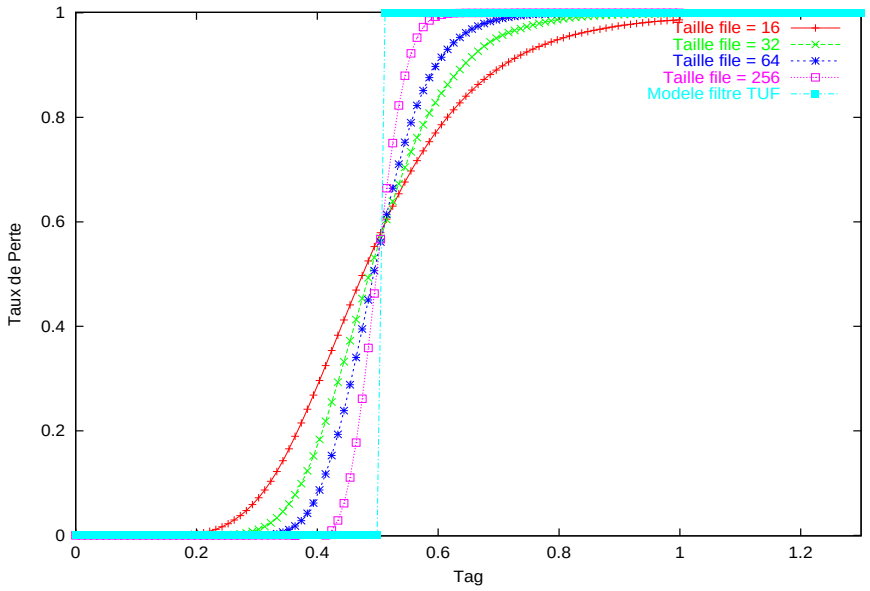


Figure 6.8 : Probabilité de perte des différents Tags dans une file TUF M/M/1/N

Lorsque la taille de la file d'attente croit, le comportement du système tend vers celui d'un filtre passe-bas, dans lesquels les paquets ayant les Tags les plus élevés sont éliminés (voir figure 6.9 pour une représentation schématique de ce comportement). Les paquets dont les Tags sont inférieurs à un seuil K sont éliminés. Ce résultat est démontré dans [6.25].

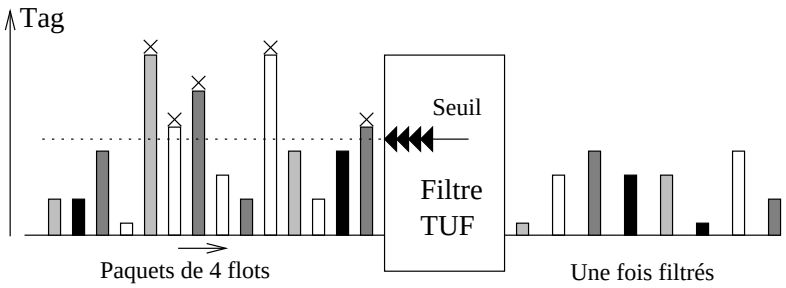


Figure 6.9 : Un routeur TUF, modélisé comme un filtre passe-bas

Nous considérerons pour la suite que la taille des files d'attentes de nos routeurs TUF est suffisante pour que l'on puisse les modéliser par des filtres passe-bas, typiquement une taille de 64 paquets par exemple.

Les tags sont en fait des valeurs discrètes et non des valeurs continues. On peut montrer qu'en cas d'égalité de Tag, la file d'attente TUF qui rejette le paquet le plus en tête de file aura la meilleure efficacité en termes de filtrage.

6.2.5 Algorithme d'étiquetage

Nous décrivons ici l'algorithme d'étiquetage qui permet d'obtenir un partage équitable de la bande passante dans les routeurs TUF. Nous souhaitons pouvoir faire cohabiter différents mode d'adaptation de bout en bout. Pour cela, décrivons de manière générique les algorithmes de contrôle de congestion que nous allons considérer.

6.2.5.1 Modélisation des algorithmes de contrôle de congestion

Nous n'étudions ici que les algorithmes de contrôle de congestion de bout en bout qui, comme TCP, n'utilisent que le taux de perte comme indicateur de congestion. On pourra par la suite étendre cette classe à ceux utilisant la notification explicite de congestion (ECN) [6.30]. Supposons ainsi que nous connaissons le débit moyen réalisé par nos sources S_1, …, S_n lorsque soumises à un taux de perte p : $B_1(p)$, …, $B_n(p)$. Les fonctions $B_i(p)$ sont strictement décroissantes, $B_i(0) = R_i$, $B_i(1) = 0$, où R_i est le débit maximum de la source S_i.

Lorsque soumis à un taux de perte p, le débit moyen reçu converge vers $B(p)$, et le débit moyen émis vers $B(p)/(1-p)$. Ainsi, les sources TCP-courtoises peuvent être modélisées par $B(p) = (1-p)\times\min(W_{\max}/RTT, C/RTT.\mathrm{sqrt}(p))$.

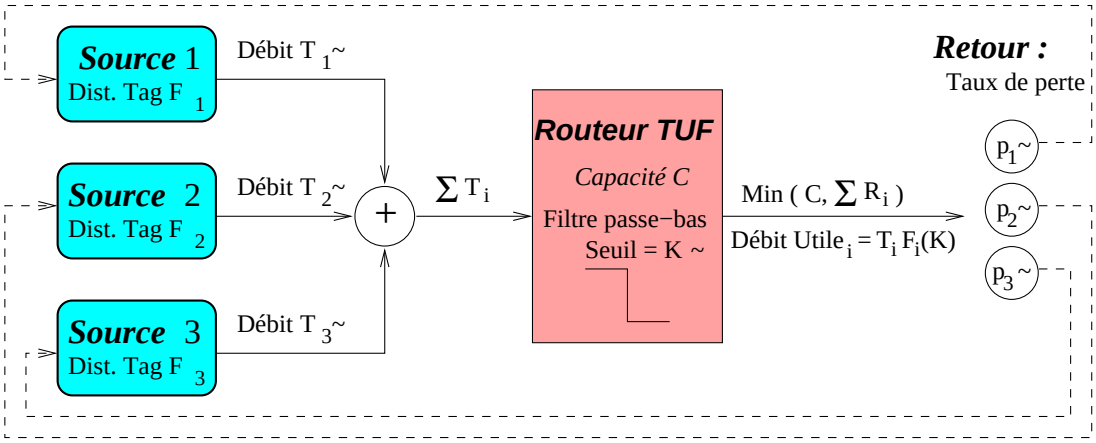


Figure 6.10 : Modélisation des contrôles de congestion de bout en bout

La dynamique du système, présentée sur la figure 6.10 est la suivante :

- Les sources émettent à un débit $T_i(t)$, borné par R_i.
- Soit F_i la loi de distribution cumulative des Tags : $Pr(\text{Tag} \leq K) = F_i(K)$.
- En cas de congestion, les paquets dont les Tag dépassent le seuil $K(t)$ sont éliminés. Le débit sur l'interface de sortie du routeur est alors égal à sa capacité.
- Les sources observent un taux de perte égal à $p_i = 1 - F_i(K(t))$.
- Les sources ajustent alors leur débit d'émission en conséquence. On suppose qu'elles cherchent à converger exponentiellement vers le débit cible $T_i^* = B_i(p_i)/(1 - p_i)$: $(\alpha > 0)$ $dT_i/dt = \alpha.(T_i^* - T_i)$

6.2.5.2 Stratégie d'étiquetage

Soit X une variable aléatoire uniformément distribuée sur l'intervalle [0,1]. Soit $B_i(p_i)$ les fonctions débit-perte d'un ensemble de sources S_i. Il est démontré dans [6.25] que le fait d'étiqueter les paquets de la source S_i avec $Tag_i = B_i(X)$ entraine une répartition **stable** et **équitable** de la bande passante. L'explication intuitive étant la suivante : l'étiquetage des paquets en tenant compte de la réactivité des flux (représenté par la fonction $B(p)$), prédispose les paquets des flux plus agressifs à un taux de perte moyen plus important de façon à obtenir en fin de compte un partage équitable des ressources entre flux de différentes agressivités.

L'étiquetage fonctionne ainsi :

- A l'étiquetage d'un paquet, on détermine le comportement de bout en bout du flot. Celui-ci est déterminé soit à partir de l'en-tête du paquet (e.g. numéro de protocole), soit par la source si celle-ci est responsable de l'étiquetage. On suppose qu'un modèle de ce type de flot existe, et que l'on peut ainsi connaître la fonction $B(p)$ correspondante.
- Un réel X est choisi aléatoirement entre 0 et 1.
- On calcule ensuite le Tag : Tag = $B(X)$

La figure 6.11 présente la distribution des Tags ainsi obtenus pour un flot TCP dont le délai de bout en bout (RTT) est de 50ms et pour un flot CBR (Constant Bit Rate) à 100 paquets par

secondes. Supposons que notre filtre passe-bas TUF élimine les paquets étiquetés avec des valeurs supérieures à $\underline{K}$. La figure montre alors les taux de perte X_{TCP} et X_{CBR} correspondant des deux flots TCP et CBR. On en déduit ensuite aisément que le débit des deux flots est $B_{TCP}(X_{TCP}) = B_{CBR}(X_{CBR}) = \underline{K}$. On obtient ainsi bien deux flots de débits égaux, précisément égal $\underline{K}$ .Voir [6.25] pour plus de détails sur les calculs.

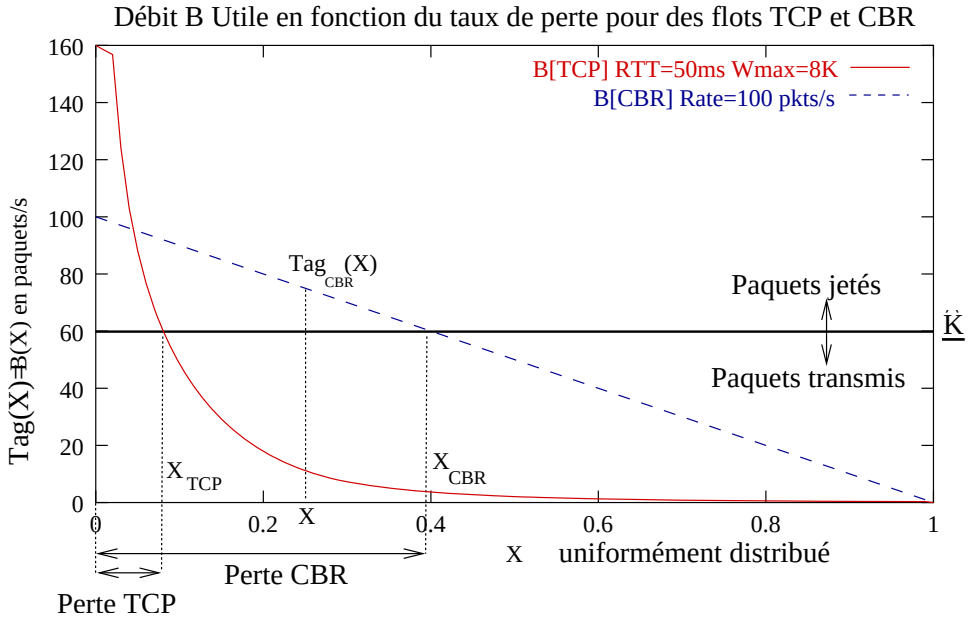


Figure 6.11 : Distribution des Tags pour des flots TCP et CBR (débit utile B en fonction du taux de perte pour des flots TCP et CBR

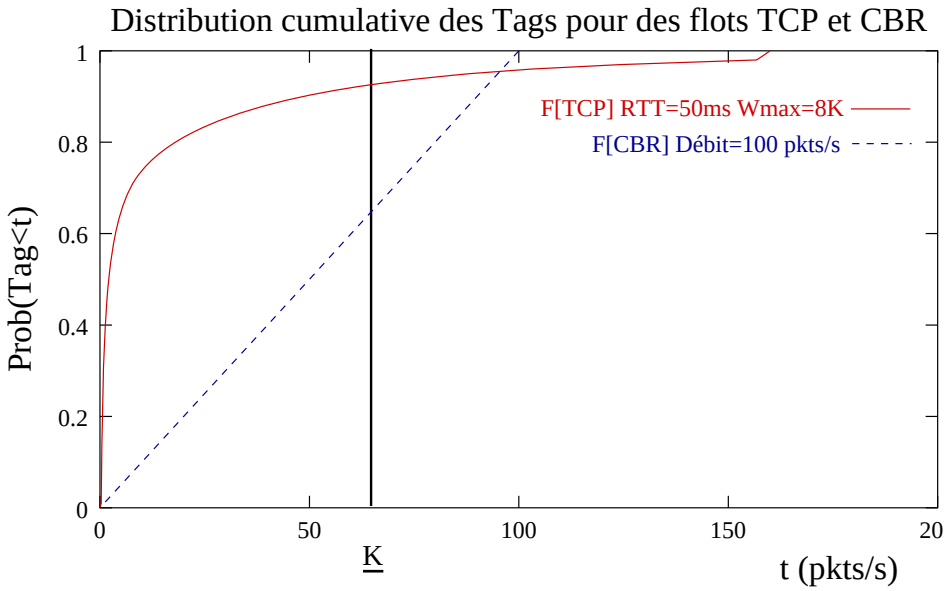


Figure 6.12 : Distribution cumulative des Tags pour des flots TCP et CBR

Nous pouvons donc montrer que :

- Le débit d'émission de tout flot converge vers une valeur **stable** $\check{T}_i$
- Le débit reçu du flot i est alors $\check{T}_i \times (1-p_i) = \check{T}_i\ (1-F_i(\underline{K})) = \min(\underline{K}, R_i)$. Les flots contraints obtiennent le débit limite $\underline{K}$, et les flots non contraints leur débit maximum R_i.

6.3 Simulations

Nous proposons ici un ensemble de simulations réalisées à l'aide du simulateur ns-2 [6.26]. Ces simulations ont pour but de montrer que TUF se comporte aussi bien que les mécanismes d'ordonnancement classiques de Fair Queueing, bien que n'introduisant pas d'état par flot. On souhaite également montrer qu'ignorer le comportement de bout en bout des flots qui réagissent à la congestion est cause d'inefficacité pour les mécanismes existants sans état par flot. Pour cela, nous introduisons le mécanisme TUF_{udp}, une version restreinte de notre mécanisme TUF, dans lequel tous les flots, en particulier les flots TCP, sont étiquetés comme des flots UDP non réactifs. On montre que TUF_{udp} se comporte à peu près comme les autres mécanismes sans état par flot et se trouve être bien moins équitable que TUF.

Ainsi, nous comparons les débits et équités réalisés par l'ensemble des protocoles suivants :

- TUF : Notre mécanisme Tag-based Unified Fairness
- TUF_{udp} : La version restreinte de TUF, dans laquelle les paquets sont étiquetés sans tenir compte du comportement de bout en bout. C'est un mécanisme conceptuellement comparable à RFQ ou à CSFQ.
- CSFQ : Core Stateless Fair Queuing. Nous utilisons l'implémentation de CSFQ pour le simulateur ns-2 fournie par les auteurs [6.27].
- FIFO : Le routeur FIFO classique - premier entré, premier servi.
- RED : Le routeur Random Early Discard [6.29].
- DRR : Le routeur Deficit Round Robin. DRR fait autorité en matière d'équité pour nos simulations. C'est un mécanisme apprécié, ses performances en termes d'équité étant bonnes pour une complexité raisonnable [6.11].
- SFQ : Le routeur Stochastic Fair Queueing. Il s'agit d'une implémentation simplifiée de Fair Queueing [6.28].

Dans les deux premières séries de simulations, nous avons reproduit la plupart des scénarios présentés dans [6.14] et utilisés dans d'autres articles [6.15] afin d'utiliser une même base de comparaison. L'objectif est de montrer le comportement équitable de TUF dans des environnements non-hostiles, où le temps aller-retour (RTT) est faible (2ms), et le trafic est lisse.

Dans la troisième série de simulation, nous mettons en évidence l'effet du temps aller-retour (RTT) et du trafic en rafale sur les performances des flots réactifs dans la plupart des algorithmes. TUF est le seul algorithme sans état par flot à maintenir l'équité dans ces contextes.

Par défaut, les liens ont une capacité de 10 Mbps, et un délai de propagation de 1ms. La longueur des buffers est de 64 Ko, chacun des paquets faisant 1000 octets (valeurs par défaut de ns). Pour DRR et SFQ, le nombre de files potentielles a été limité à 1024 - ce qui est déjà

assez important. La taille totale de la file d'attente pour SFQ est de 2Mo. Les simulations durent, sauf précision contraire, 60 secondes.

6.3.1 Un seul lien congestionné

La topologie dans la première série de simulation est présentée sur la figure 6.13. Nous évaluons l'équité des différents mécanismes lorsque des flots TCP et UDP partagent un même lien.

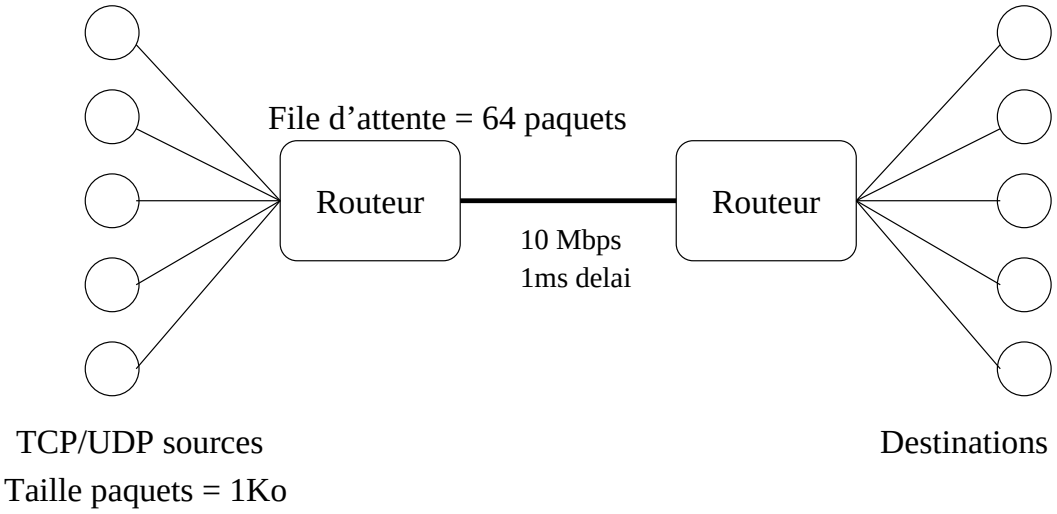


Figure 6.13 : Topologie avec un unique point de congestion

Dans la première simulation, 32 flots UDP avec des agressivités (i.e. débit) variables se partagent une bande passante de 10Mbps. Le débit du flot numéro i est égal à i+1 fois le débit limite, c'est à dire $(i+1)\times 10\text{Mbps}/32$. Les résultats (figure 6.14) montrent le débit normalisé[1] pour l'ensemble des flots UDP. RED et FIFO ne proposent pas d'équité satisfaisante, le débit réalisé étant proportionnel au débit d'émission. Les flots avec les identifiants les plus grands obtiennent davantage de bande passante. En revanche, l'ensemble des mécanismes proposant de l'équité, SFQ, DRR, CSFQ, et TUF, se comportent bien dans ce cas très simple et proposent un débit identique à l'ensemble des flots UDP.

[1] Il s'agit du rapport entre le débit réalisé et le débit limite, qui devrait être égal à 1 pour les flots contraints si le mécanisme est équitable

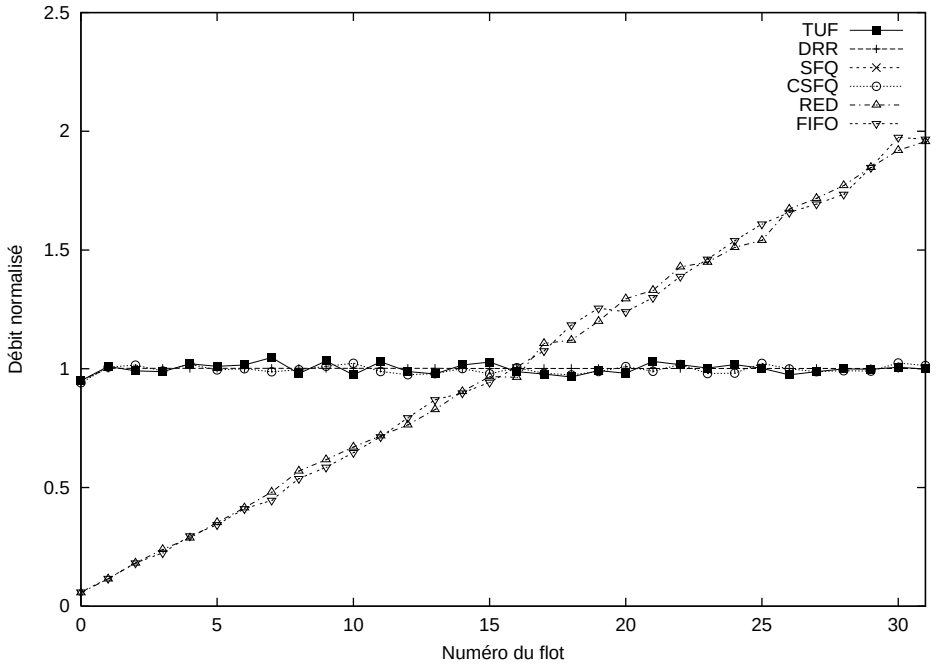


Figure 6.14 : 32 flots CBR se partageant une bande passante de 10Mbps.

Dans la seconde simulation de cette série (figure 6.15), on évalue l'impact d'un flot UDP agressif et non réactif (flot numéro 0) sur 31 autres flots TCP partageant le lien congestionné. Le flot UDP transmet à un débit égal à la capacité du lien (10 Mbps). A nouveau, les mécanismes RED et FIFO donnent la plupart de la bande passante à ce flot, étouffant l'ensemble des connections TCP - avec un léger avantage pour RED. SFQ, DRR, CSFQ and TUF restreignent le flot UDP au voisinage du débit limite, ainsi que toutes les autres connections TCP.

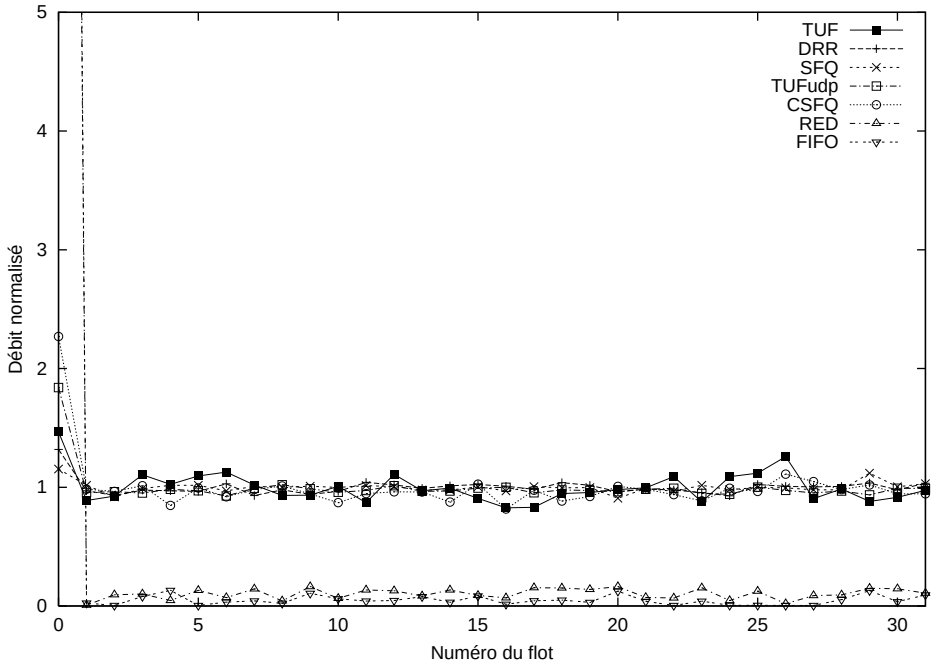


Figure 6.15 : un Flot CBR (ID=0) de 10Mbps partageant le lien avec 31 flots TCP.

Dans la troisième et dernière simulation de cette série (figure 6.16), nous évaluons le débit réalisé par une connexion TCP soumise à la pression d'un nombre *croissant* de flots UDP agressifs sur ce même lien. Chaque flot UDP transmet au double du débit limite, auquel il a droit. On remarque la chute des performances de DRR au delà de 22 flots : ceci provient de la limitation de l'espace mémoire réservé pour la connexion TCP dans le routeur congestionné [6.14]. C'est donc un problème d'implémentation. On constate que tous les algorithmes de partage équitable des ressources offrent des performances acceptables pour la connexion TCP, qui obtient dans tous les cas 60% du débit limite. TUF_{udp} et CSFQ offrent toutefois un débit sensiblement inférieur pour la connexion TCP par rapport à TUF, DRR, ou CSFQ. Ceci est typiquement un symptôme de *l'effet variation*, qui, pour TCP-Reno, limiterait le débit normalisé à 0,75.

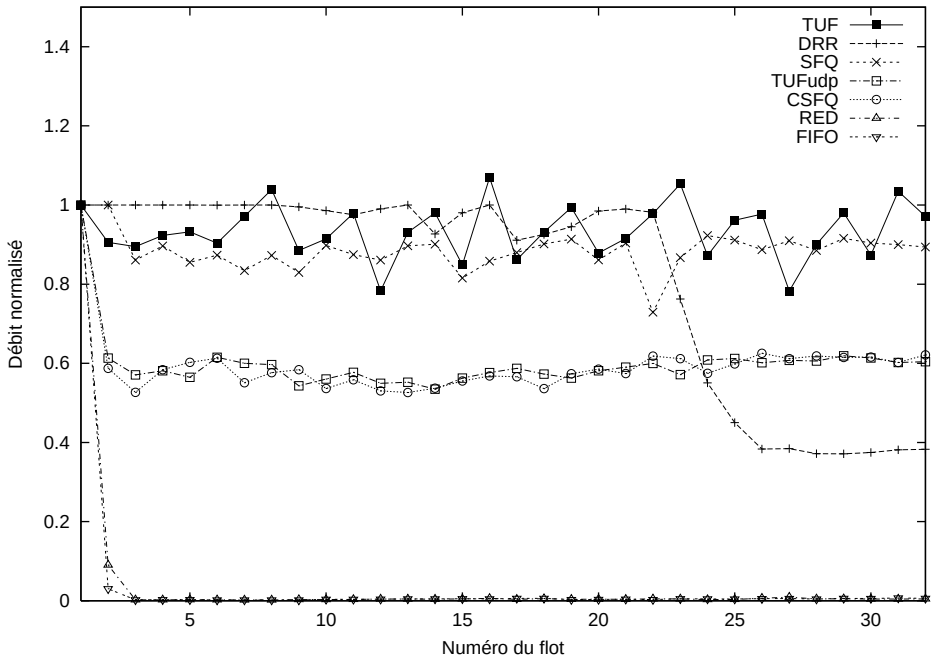


Figure 6.16 : Un flot TCP partageant le lien avec 0..31 flots UDP.

6.3.2 Multiples liens congestionnés

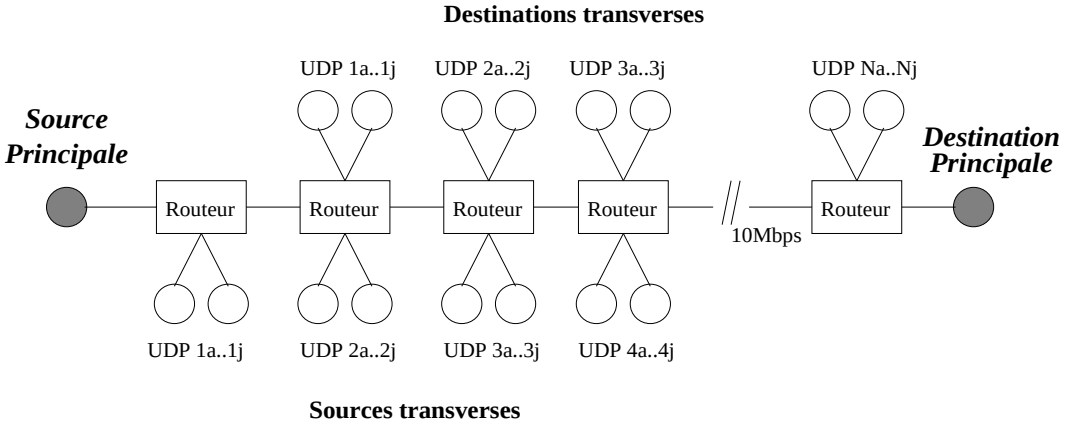


Figure 6.17 : Liens congestionnés multiples.

Cette seconde série de simulation exploite une topologie dans laquelle des liens congestionnés sont mis en série (figure 6.17). Le trafic dont on cherche à mesurer les performances traverse les routeurs de part en part et partage les liens avec du trafic transverse. L'objectif de ces simulations est d'évaluer la robustesse des algorithmes lorsque les flots traversent plus d'un lien congestionné, et lorsque les flots en compétition n'ont pas la même source et destination. 10 sources CBR émettent chacune un trafic de 2Mbps sur chacun des liens congestionné. Le trafic entre sur le chemin par un routeur et ressort par le suivant.

Dans la première simulation (figure 6.18), on démontre la robustesse de notre mécanisme TUF face à la succession de liens congestionnés. Comme dans la simulation précédente (figure 6.16), TUF_{udp} et CSFQ proposent des équités acceptables (autour de 60%), mais qui se dégradent rapidement avec le nombre de liens congestionnés dans le cas de CSFQ. Il est étonnant de constater que DRR et SFQ souffrent également d'une dégradation des performances lorsque le nombre de liens augmente. RED et FIFO ne permettent pas à la connexion TCP d'obtenir un débit significatif.

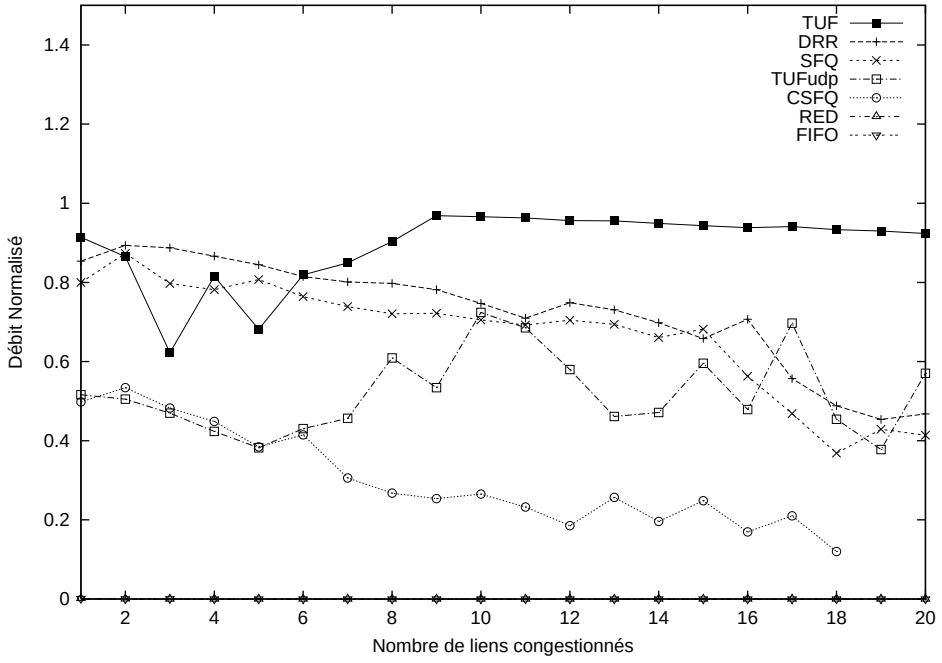


Figure 6.18 : Connexion TCP traversant N=1..20 liens congestionnés

Dans la seconde simulation (figure 6.19), la source principale est une source UDP, et n'est pas affectée par le trafic transverse. Le débit normalisé est voisin de 1 dans tous les cas (TUF, CSFQ, SFQ, et DRR), alors que RED et FIFO sont à nouveau incapables d'assurer un débit significatifs à ces flots. (Remarquons que TUF et TUF_{udp} sont ici équivalents). SFQ et DRR ont des performances légèrement supérieures, probablement en raison des approximations faites par TUF lors de l'estimation du débit du flot principal, estimations qui n'existent pas dans SFQ et DRR.

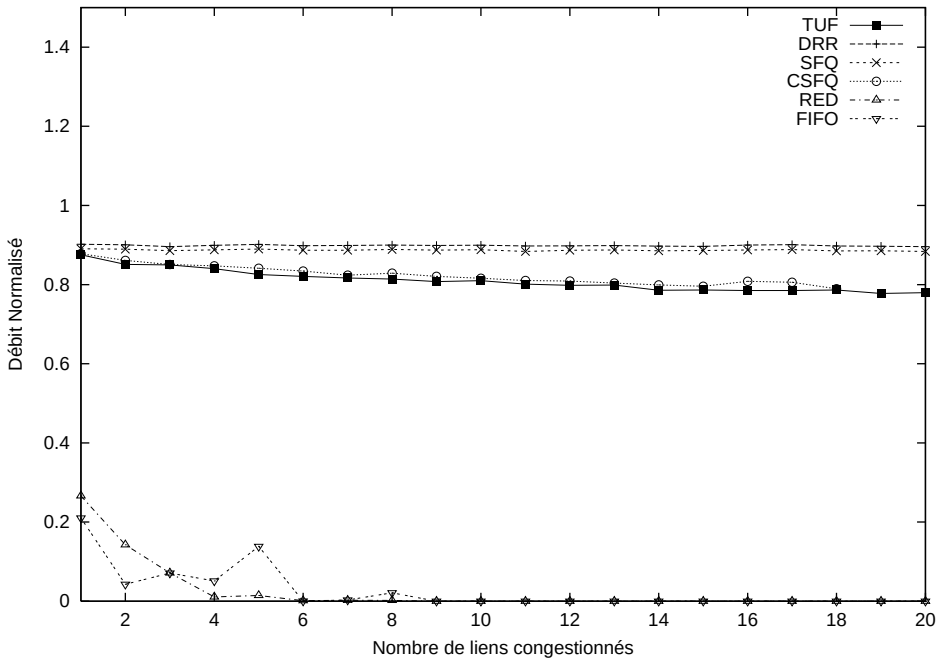


Figure 6.19 : Connexion UDP traversant N=1..20 liens congestionnés

6.3.3 Environnements hétérogènes

Cette troisième série de simulation a pour objectif de montrer la vulnérabilité des mécanismes qui cherchent à limiter le débit instantané des sources, comme c'est le cas de tous les algorithmes de Fair Queueing sans état par flot existants autres que TUF. Nous cherchons ainsi à montrer que la prise en compte du comportement de bout en bout des protocoles de contrôle de congestion est nécessaire pour obtenir de l'équité. Nous mettons aussi en évidence *l'effet rafale* présenté dans la section 6.2.2 ; dans la première simulation en introduisant des délais de bout en bout plus réaliste que 2ms, dans la seconde, en introduisant du trafic beaucoup plus irrégulier, qui provoquent de grandes variations dans le débit limite.

6.3.3.1 Délais de bout en bout

Dans cette simulation, nous reprenons la topologie avec un unique lien congestionné (figure 6.13). 8 flots CBR, émettant au double du débit limite, partagent le lien avec un flot TCP. Cette fois ci, le délai de bout en bout varie de 2ms, comme dans les simulations précédentes, à 1s. La figure 6.20 présente le débit normalisé du flot TCP. En raison des grands délais de bout en bout, et pour obtenir des statistiques satisfaisantes sur les pertes, nous avons augmenté le temps de simulation à 10000 secondes.

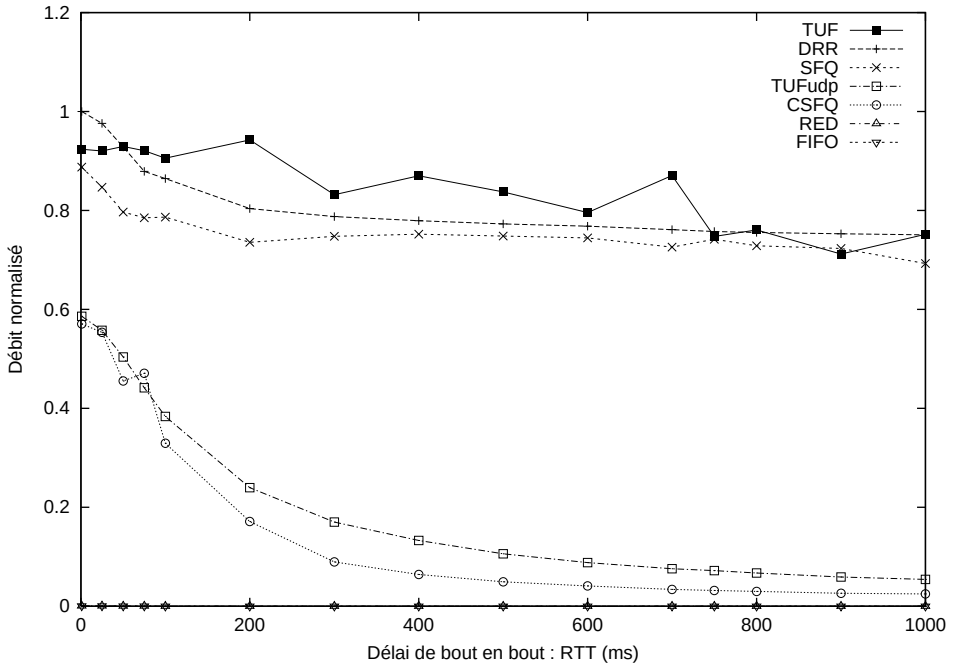


Figure 6.20 : Equité et délais de bout en bout

La figure 6.21 nous montre la distribution de l'intervalle de temps qui sépare deux pertes de paquets consécutives. L'axe des X représente l'intervalle de temps, et l'axe des Y le nombre d'évènements avec un tel écart entre deux pertes consécutives. Cette courbe confirme l'arrivée en rafale des pertes pour CSFQ et TUF_{udp}: le nombre de paquets perdus et 5 à 10 fois plus important, et le nombre de pertes très rapprochées est très important. Ce n'est pas le cas de TUF, DRR, et SFQ. Dans le cas de DRR et SFQ, la fréquence des pertes est parfaitement régulée, grâce à la mise à jour d'état par flot. En revanche, TUF a une distribution plus régulière. Il s'agit en fait d'une distribution poissonnienne, dont l'intensité dépend du comportement de bout en bout du flot et du débit limite dans ce routeur.

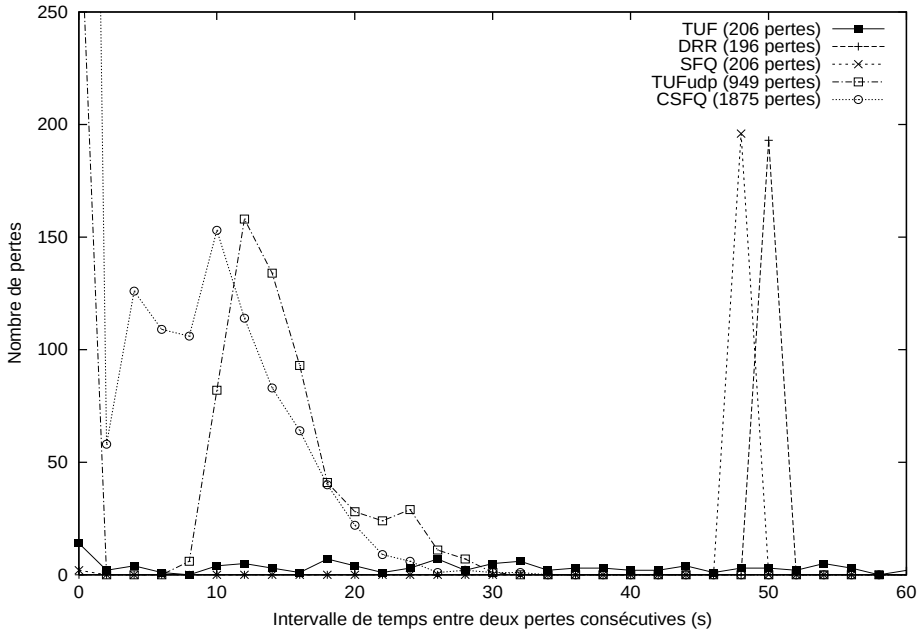


Figure 6.21 : Distribution des temps entre deux pertes consécutives

Sur la figure 6.20, nous mettons en évidence l'impact de l'effet rafale sur l'équité. Il est particulièrement intéressant de voir la différence qui existe entre TUF_{udp} et TUF, vu que ces derniers implémentent les mêmes algorithmes mais que seul le dernier prend en compte la nature TCP du flot. Les performances de TUF_{udp}, comme celle de CSFQ, se dégradent sensiblement quand le délai de bout en bout augmente. DRR et SFQ qui maintiennent, grâce à des états par flots, maintiennent des débits moyens et non des débits instantanés, obtiennent des performances comparables à celles de TUF.

6.3.3.2 Trafic en rafale

Cette dernière simulation utilise la topologie (figure 6.17) avec 5 liens congestionnés. Les sources CBR qui constituaient dans la série précédente le trafic transverse, sont maintenant remplacées par des sources ON/OFF. La durée de la rafale (ON) et du temps inter-rafale (OFF) sont distribués exponentiellement avec une même moyenne comprise entre 5ms et 1s. L'intensité moyenne du trafic transverse est maintenue identique à celle de la simulation précédente : les sources ON/OFF émettent à 4Mbps pendant leur rafale.

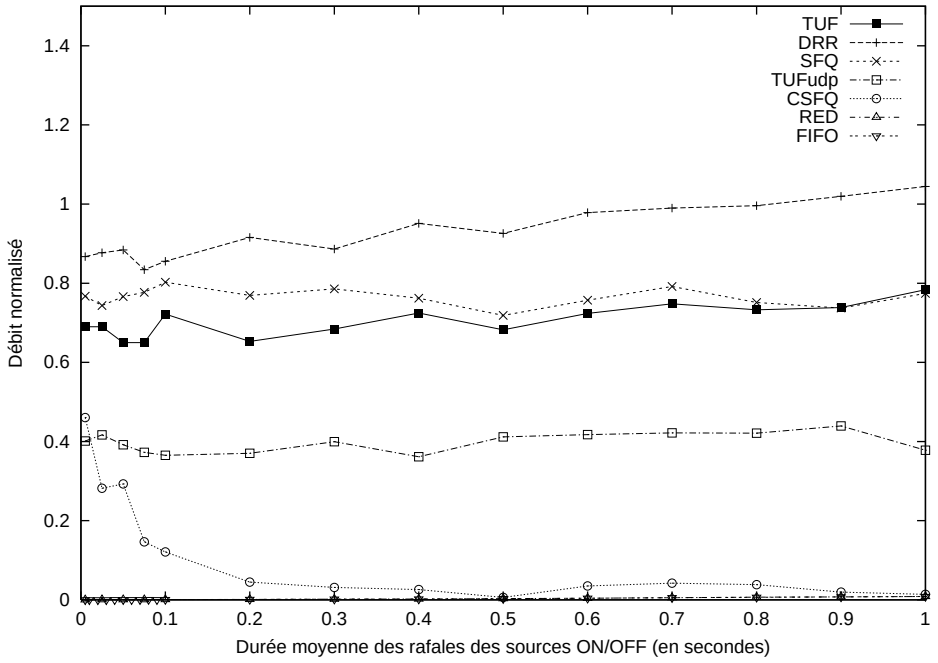


Figure 6.21 : Flot TCP en compétition avec du trafic ON/OFF}

La figure 6.21 présente à nouveau les faibles performances de TUF_{udp} et CSFQ, comparées à celles de TUF, DRR et SFQ. Les performances de CSFQ se dégradent particulièrement lorsque l'on atteint 100ms comme durée moyenne d'une rafale, ce qui correspond précisément à la constante de durée permettant l'évaluation du débit « instantané » des flots.

6.4 Implémentation

6.4.1 Le module d'étiquetage

TUF a été implémenté dans la version 2.2 de Linux. Dans la version actuelle, l'étiquetage est réalisé par la source, au niveau de la couche IP, comme présenté sur la figure 6.22, pour tous les paquets dont le champ TOS vaut 0x28. En choisissant 0x28 comme valeur par défaut du champ TOS, toutes les applications existantes ont leur paquets étiquetés de manière transparente, à la fois pour les flots TCP et les flots UDP, ces derniers étant étiquetés comme des flots non-réactifs. Pour les flots UDP, on maintient un état par socket pour évaluer le débit moyenné du flot. Pour TCP, les valeurs de délai de bout en bout, de la temporisation de retransmission, et la taille maximale de la fenêtre d'émission sont déjà évaluées par le noyau. Le calcul d'un Tag TCP ou UDP ne requiert pas plus de 90 opérations élémentaires (addition ou multiplication) et ne représente donc pas un facteur limitant dans la capacité d'émission de la machine.

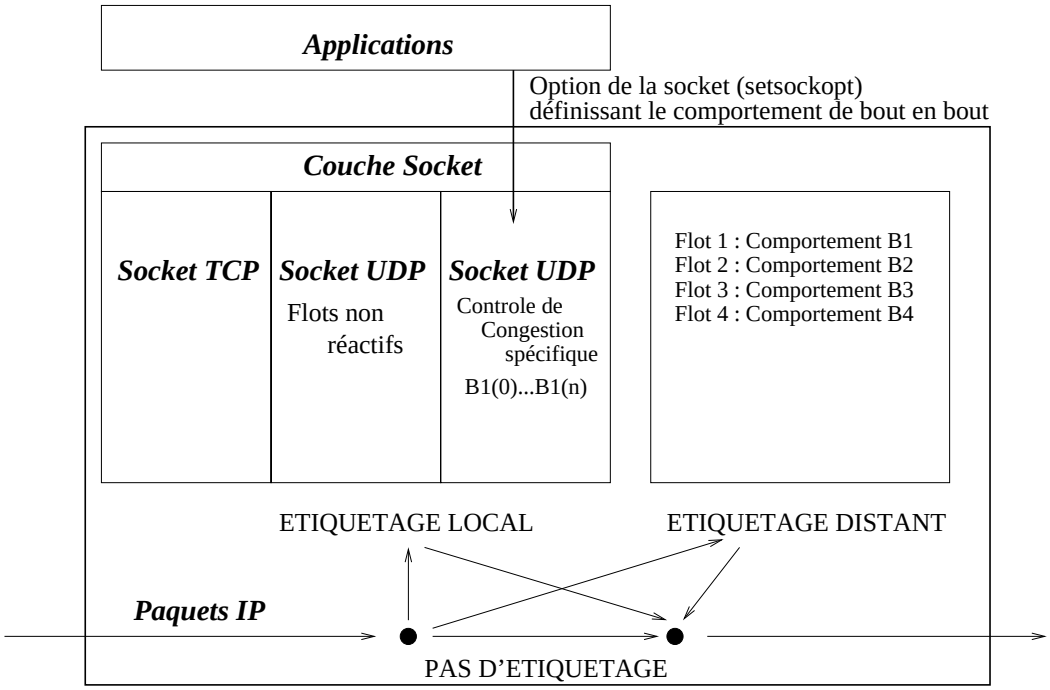
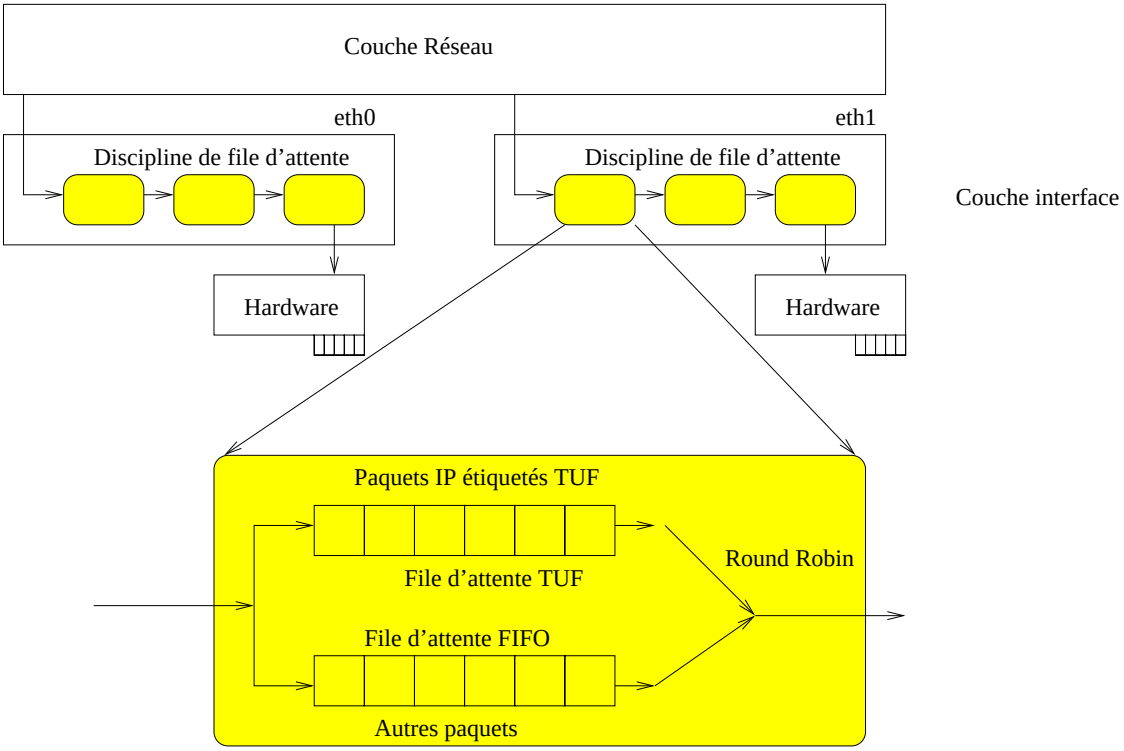


Figure 6.22 : Le module d'étiquetage TUF

Il est également possible de définir une nouvelle option socket pour les applications qui souhaite définir elles-mêmes leur comportement de bout en bout.

- L'application calcule les Tags correspondant à n valeurs discrètes de X comprises entre 0 et 1 : $V(B(0))$, $V(B(1/n))$, $V(B(2/n))$, …, $V(B(1))$. Pour $n=2^{32}$, et un tag sur 16 bits, ceci représente un buffer de 64Ko.
- L'application charge ces valeurs dans le module d'étiquetage du noyau (ou à distance à l'aide d'un paquet propriétaire si l'étiquetage n'est pas fait localement). Ceci se fait à l'aide d'une commande *setsockopt* qui manipule les options socket.
- Le module d'étiquetage pourra ensuite obtenir les Tags $V(B(X))$ par interpolation entre les valeurs connues.

6.4.2 Les files d'attentes TUF



Gestion de la file d'attente TUF

Figure 6.23 : Insertion de TUF dans le noyau linux

Dans le noyau Linux (v2.2), on peut attacher à chaque interface de sortie une liste de files d'attente. Vue du noyau, une file d'attente est considérée comme un objet à laquelle sont attachées des méthodes d'insertion et d'extraction de paquets. Nous avons intégré la gestion de file d'attente TUF décrit précédemment et présenté sur la figure 6.23. Il est alors possible d'activer TUF sélectivement sur telle ou telle interface, et de combiner les mécanismes d'équité assurés par TUF avec d'autres mécanismes et files d'attentes comme une file d'attente RED ou un mécanisme proposé dans diffserv.

6.4.3 Expérimentation

Nous avons expérimenté notre algorithme TUF sur une petite topologie en Y. Nous avons introduit des flots CBR agressifs, des flots TCP, et des flots beaucoup moins agressifs comme des flots audio générés par l'application audio RAT[1]. Lorsque l'on active TUF sur l'ensemble des interfaces de sortie du routeur central, le débit réalisé par le flot TCP et la qualité de la session audio changent radicalement, et on obtient une bonne équité avec les autres flots CBR.

6.5 Conclusion

[1] Voir http://www-mice.cs.ucl.ac.uk/multimedia/software/rat.

Nous pensons que le déploiement de nouveaux services dans l'Internet, multipoints ou non, ira de pair avec une hétérogénéité croissante. Pour assurer un partage équitable des ressources entre les différents flots, il y a plusieurs approches possibles :

- Exiger de chaque flot qu'il adopte un comportement « social », le comportement de référence étant celui de TCP.
- Assurer dans chaque routeur du réseau un partage équitable des ressources disponibles.

Nous pensons qu'il est irréaliste de considérer que l'on parviendra à imposer la première solution, tant sa mise en œuvre peut être complexe pour certains types de flots. La seconde méthode, conjuguée avec des mécanismes d'adaptation de bout en bout, peut aboutir à un partage équitable et efficace des ressources du réseau. Parmi l'ensemble des algorithmes de fair queueing existants, TUF propose une technique simple, robuste et sans état dans les routeurs. TUF maintient la complexité aux frontières du réseau tout en assurant une efficacité comparable aux algorithmes plus lourds, gérant les flots individuellement. TUF est par conséquent un atout pour l'accomplissement des futurs réseaux hétérogènes.

Chapitre 7

La Sécurité des Systèmes de Coordonnées Internet

Nous avons assisté ces dernières années à une énorme croissance d'applications Internet (comme Pastry [7.1], Oceanstore [7.2], ESM [7.3], Skype [7.4], etc.), basées et/ou bénéficiant des réseaux de recouvrement (ou overlay). Ces applications tiennent compte de la topologie du réseau physique pour la construction du réseau de recouvrement. En particulier, la plupart des ces applications (si ce n'est toutes) et leurs réseaux de recouvrement associés, se basent sur la notion de proximité réseau, typiquement définie en termes de délais d'aller-retour (RTT), pour l'optimisation de la sélection de voisins. Cependant, les mesures de proximité peuvent s'avérer extrêmement coûteuses en termes de consommation en bande passante. En effet, l'existence simultanée de plusieurs réseaux de recouvrement, peut entraîner un surcoût de communications élevé, dû aux mesures de proximité individuelles menées par chaque nœud de ces réseaux.

De plus, traquer la proximité au sein d'un groupe dynamique, nécessite une fréquence de mesure très grande. Cela induit encore un surcoût de mesures plus important. Afin de pallier ce problème, des systèmes de positionnement « Internet », comme [7.5, 7.6, 7.7, 7.8, 7.9, 7.10], ont été introduits. Dans ces systèmes, l'idée principale est que si chaque nœud peut être associé à une coordonnée virtuelle dans un espace approprié, la distance entre les nœuds est trivialement calculée sans pour autant avoir recours à des mesures directes. En d'autres termes, ces systèmes plongent les mesures des temps de latence (délais) entre une population de nœuds dans un espace géométrique et associent un vecteur de coordonnées (ou coordonnées) dans cet espace à chaque nœud, dans le but de permettre des prédictions de distances précises et peu onéreuses parmi n'importe quelle paire de nœud dans le réseau. L'avantage premier de ces systèmes est que si les distances réseaux (dans le sens de délais ou de temps de latence) sont plongées dans un espace de coordonnées, où une position raisonnablement précise pour chaque nœud est établie, le surcoût de mesures produit par le positionnement, est ainsi amorti sur plusieurs prédictions de distances. Ceci réduit énormément le coût en termes de mesures de distances du système entier.

Des mesures et analyses approfondies de déploiements réels de ces systèmes ont montré qu'ils atteignaient pleinement leurs objectifs [7.11], faisant d'eux un outil précieux pour supporter les applications distribuées et réseaux de recouvrement (tels que [7.4, 7.12, 7.13]), qui bénéficient de la notion de proximité réseau. En effet, les systèmes de positionnement à base de coordonnées jouissent de plusieurs propriétés souhaitables, telles que la précision, la robustesse, la stabilité, ainsi que le passage à l'échelle et les surcoûts de communications peu élevés. Cependant, il est important de souligner que ces propriétés sont souvent réalisées aux dépends de temps de convergence assez longs. Dans de tels systèmes, les nouveaux nœuds, n'atteignent une bonne estimation de leurs coordonnées, qu'après un laps de temps à l'échelle de dizaines de seconde, voire de minutes. Ceci est beaucoup trop lent, comparé aux temps de convergence réalisés avec des mesures directes entre les nœuds, et peut même être inacceptable pour certaines applications, qui ont pour but de rapidement identifier les « meilleurs nœuds » (ou voisins potentiels). Les systèmes de positionnement Internet, en

particulier ceux basés sur les coordonnées, semblent être une proposition attractive s'ils sont déployés comme un service permanent : chaque nœud faisant tourner un système de coordonnées au démarrage de son système d'exploitation. Cela permet ainsi au nœud de fournir des estimations de coordonnées précises à la demande des applications et réseaux de recouvrement. Un système de coordonnées est alors perçu comme une composante d'une « infrastructure virtuelle » qui supporte une large variété d'applications et réseaux de recouvrement. Un tel système, en revanche, pourrait être une cible parfaite pour les attaquants, puisque sa perturbation résulterait dans le disfonctionnement ou même l'écroulement de toutes les applications se basant sur ce service.

Les propositions actuelles de systèmes de coordonnées supposent par contre que les nœuds participants dans le système coopèrent pleinement et honnêtement entre eux. En d'autres termes, ces systèmes supposent que les informations échangées sont toujours correctes, ce qui les rend vulnérables à différentes attaques.

La suite de ce chapitre est organisée comme suit : dans la section 7.1, nous effectuons un tour d'horizon des systèmes de positionnement de l'Internet, existant dans la littérature, ainsi que de leurs exploitations potentielles dans diverses applications. Nous décrirons notamment les différents systèmes à base de coordonnées existants, ainsi que leurs mécanismes de sécurité.

Dans une deuxième étape, nous identifions dans la section 7.2 diverses attaques contre les systèmes de positionnement à base de coordonnées, et montrons l'impact que peuvent avoir de telles attaques sur les performances de ces systèmes.

Nous proposons par la suite une méthode générale pour la détection des comportements malicieux au sein des systèmes de positionnement durant la phase de calcul des cordonnées. Nous montrons en premier lieu, dans la section 7.3, que la dynamique d'un nœud, dans un système de coordonnées sain, exempt de comportements anormaux ou malhonnêtes, peut être modélisée par un modèle d'états linéaires, et traquée par un filtre de Kalman. De plus, les paramètres d'un filtre calibré au niveau d'un nœud donné, peuvent être utilisés pour modéliser et prédire le comportement dynamique d'un autre nœud, tant que ces deux nœuds sont proches l'un de l'autre dans le réseau. Cela a entraîné la proposition d'une infrastructure de nœuds, appelés « nœuds experts » : il s'agit de nœuds de confiance, se positionnant dans l'espace des coordonnées, en utilisant exclusivement d'autres nœuds experts. Ils sont ainsi immunisés contre des comportements malicieux dans le système. Pendant le calcul de leurs propres coordonnées, les autres nœuds peuvent ainsi utiliser les paramètres du filtre d'un nœud expert proche, comme étant une représentation d'un comportement normal, non assujetti à un comportement malicieux, pour détecter et filtrer toute activité malicieuse ou anormale. Une combinaison de simulations et d'expérimentations PlanetLab a été utilisée pour démontrer la validité, la généralité, et l'efficacité de l'approche proposée pour chacun des deux systèmes Vivaldi et NPS.

Dans la section 7.4, nous sécurisons la phase d'utilisation des distances calculées par les systèmes de coordonnées. La méthode proposée se divise en deux étapes : 1) établir l'exactitude des coordonnées annoncées en utilisant l'infrastructure des nœuds experts et la méthode de détection des nœuds malicieux, et 2) délivrer un certificat à validité limitée pour chaque coordonnée vérifiée. Les périodes de validité sont calculées à partir d'une analyse des temps d'inter-changement observés par les nœuds experts. En faisant cela, chaque nœud expert, peut estimer le temps jusqu'au prochain changement de coordonnées, et ainsi, peut

limiter le temps de validité du certificat qu'il délivrerait aux nœuds normaux. Notre méthode est illustrée par une trace recueillie à partir d'un système Vivaldi déployé sur PlanetLab, où les distributions de temps d'inter-changements suivent des distributions longue traine (distribution log-normale dans la plupart des cas, et distribution Weilbull sinon). Nous montrons l'efficacité de notre méthode en mesurant l'impact de plusieurs attaques sur les estimations de distance, expérimentées sur PlanetLab.

7.1 Les systèmes de positionnement Internet

Dans cette section, nous présentons une vue d'ensemble des différentes propositions dans le domaine des systèmes de positionnement Internet. Nous commençons par lister très brièvement quelques travaux destinés à fournir une estimation de la localisation et de la proximité dans le réseau. Ces systèmes ne reposent pas cependant sur le calcul de coordonnées virtuelles. Nous les appellerons « Services d'estimation de proximité par mesures directes ». Nous nous concentrons par la suite sur les systèmes à base de coordonnées, que nous classifions en deux classes principales: Les systèmes basés sur les balises (landmarks), et les systèmes distribués. En particulier, nous étudierons les systèmes munis de mécanismes de sécurité destinés à filtrer les nœuds malicieux. Nous montrerons cependant par la suite, que ces mécanismes sont encore primitifs, et ne peuvent en aucun cas défendre le système contre tous les types d'attaques.

7.1.1 Services d'estimation de proximité par mesures directes

Plusieurs approches dans la littérature fournissent des estimations de distances ou de proximité réseau en utilisant des mesures directes entre les paires de nœuds. Une première approche basée sur la géolocalisation, s'inscrit dans une tentative de fournir la localisation géographique des nœuds du réseau, plutôt que leurs positions Internet (e.g. Constraint-Based Geolocation [7.14] et le système IP2Geo). Une autre approche permet de fournir un classement sur la base de mesures explicites vers des balises ("Binning"). Il y a aussi des méthodes basées sur des « traceroute », ou sur l'établissement de positionnement relatif entre les nœuds (l'approche Meridian [7.15]), ainsi que la méthode basée sur la localisation et le regroupement hiérarchique que nous avons proposée dans [R.16].

7.1.2 Systèmes de coordonnées à base de Balises Fixes

Ces systèmes incorporent une composante centrale (un ensemble d'entités fixes considérées comme des balises), dont les nœuds se servent pour calculer leurs coordonnées en fonction de mesures vers ces entités balises. Le concept de positionnement par rapport à des entités balises a été introduit dans GNP (Global Network Positioning) [7.5]. Dans ce genre de systèmes, les coordonnées des balises sont en premier lieu calculées en minimisant l'erreur entre les distances mesurées et les distances estimées entre les nœuds balises. De la même manière, un nœud ordinaire dérive ses coordonnées en minimisant l'erreur entre les distances mesurées et les distances estimées vers les nœuds balises. Plusieurs approches ont par la suite repris le concept des balises, et ont pour vocation de mieux passer à l'échelle (e.g. Lighthouse et l'approche de balises virtuelles). Nous nous intéressons plus particulièrement au système NPS.

Le système NPS

Ce système [7.8] étend le système GNP à un système de coordonnées hiérarchique, où les nœuds peuvent servir comme balises (appelés « points référence ») pour d'autres nœuds. Le but principal est de pallier les problèmes de failles des balises centralisées et fixes, notamment lorsque ces entités deviennent des goulots d'étranglement dans le réseau. La différence majeure par rapport à GNP est que n'importe quel nœud qui s'est bien positionné dans le réseau peut être choisi comme un point référence par un serveur pour d'autres nœuds. Cependant, pour assurer une cohérence dans le système, NPS impose un positionnement hiérarchique entre les nœuds. Étant donné un ensemble de nœuds, NPS les partitionne dans différents niveaux. Un ensemble de 20 balises fixes sont placées dans le niveau 0, la couche supérieure du système, et qui définit la base de l'espace géométrique choisi. Chaque nœud dans un niveau L_i, choisit au hasard quelques nœuds dans le niveau L_{i-1} comme ses points référence. L'erreur relative de la prédiction de distance entre une paire de nœuds est définie comme suit : erreur relative = | dist_réelle − dist_virtuelle| / min(dist_réelle, dist_virtuelle).

Nous nous intéressons plus particulièrement au système de sécurité inclus dans NPS, qui vise à atténuer les attaques de nœuds malicieux. En effet, les nœuds peuvent mentir sur leurs positions et influencer les mesures effectuées. L'idée principale est d'éliminer le point référence s'il mène à une erreur relative trop élevée par rapport aux autres points référence. Dans nos travaux, nous considèrerons NPS, comme le système représentatif des systèmes de coordonnées à base de balises.

7.1.3 Les Systèmes de coordonnées décentralisés

Cette classe de systèmes étend le concept de positionnement par coordonnées, en généralisant le rôle des balises à tous les nœuds du système, ou en éliminant l'infrastructure des balises. Ces systèmes peuvent être alors perçus comme des systèmes de positionnement pair-à-pair. Nous introduisons en premier lieu le système PIC (Practical Internet Coordinates) [7.7] où tous les nœuds du système peuvent jouer le rôle de balises pour les autres. Le système de sécurité proposé par PIC, se base sur l'inégalité triangulaire afin de détecter les nœuds malicieux, qui d'après PIC, sont plus susceptibles de violer cette inégalité. Dans [7.7], les auteurs montrent que ce test de sécurité peut pallier les attaques d'intensité allant jusqu´a 20% de nœuds malicieux dans le système. Néanmoins, [7.16] et [7.17] ont montré que les RTTs dans le réseau violent régulièrement et couramment les inégalités triangulaires. Un système de sécurité basé sur la supposition que l'inégalité triangulaire est toujours vérifiée, pourrait mener à une dégradation des performances du système de coordonnées, notamment, lors de l'inexistence de nœuds malicieux. Nous nous focalisons dans ce chapitre sur le système Vivaldi, et nous le considérerons, représentatif des Systèmes de coordonnées décentralisés.

Vivaldi

Vivaldi [7.9] est un système de coordonnées complètement décentralisé. Il est basé sur une simulation de ressorts, où la position du nœud correspond à celle de l'extrémité d'un ressort qui minimiserait l'énergie potentielle des ressorts, et donc minimiserait l'erreur de positionnement. Un nœud se joignant au système, calcule ses coordonnées en collectant des

informations de positions et de mesures de délai à partir de quelques autres nœuds. Spécifiquement, Vivaldi place « un ressort » entre chaque paire de nœuds (i, j) dans le système, avec une longueur « au repos » correspondant à la mesure RTT entre ces deux nœuds. La longueur réelle du ressort est considérée comme étant l'estimation de distance entre les deux positions des nœuds. L'énergie potentielle d'un tel ressort est proportionnelle au carrée de déplacement par rapport à sa longueur au repos. Cette énergie représente l'erreur entre le RTT estimé et mesuré. La somme de ces erreurs à travers tous les ressorts est la fonction d'erreur que Vivaldi tente de minimiser. Une procédure identique tourne sur tous les nœuds Vivaldi. Celle-là est basée sur des échantillons récoltés par les nœuds qui fournissent les informations pour leur positionnement. Un échantillon, utilisé par un nœud i est ainsi constitué de la mesure vers un nœud j, de la coordonnée du même nœud j, ainsi que de l'erreur locale reportée par j. L'algorithme traite les nœuds à erreur élevée, en assignant des poids à chaque échantillon collecté. L'erreur relative de cet échantillon, e_s, est alors calculée comme suit :

$$e_s = | \, || x_j - x_i || - \mathrm{RTT}_{\text{mesuré}} | \, / \, \mathrm{RTT}_{\text{mesuré}}$$

Le nœud calcule alors le poids de cet échantillon comme étant $w = e_i/(e_i + e_j)$, où e_i est l'erreur actuelle (locale) du nœud i. Ce poids est utilisé en fait pour calculer un déplacement adaptatif, δ définissant la fraction de mouvement que le nœud est autorisé à faire en direction du nœud auquel il se mesure : $\delta = C_c \times w$, où C_c est une constante < 1. Le nœud met alors à jour sa coordonnée locale comme suit :

$$x_i = x_i + \delta \times (\mathrm{RTT}_{\text{mesuré}} - || x_i - x_j ||) \times \mathbf{u}(x_i - x_j)$$

où $\mathbf{u}(x_i - x_j)$ est un vecteur unitaire donnant la direction de déplacement du nœud i. Enfin, le nœud met à jour son erreur locale comme étant $e_i = e_s \times w + e_i \times (1 - w)$.

7.1.4 Discussion

Qu'ils soient basés sur des balises fixes, ou qu'ils soient décentralisés, la plupart des systèmes de positionnement actuels réalisent de bonne performances en termes de précision de positionnement, de stabilité et de passage à l'échelle. Cependant, il faut aussi noter que cela est permis aux dépends de temps de convergence longs, allant de quelques secondes à plusieurs minutes [7.11]. Cela est très loin des performances en termes de rapidité de traitement des systèmes de mesures directes entre des paires de nœuds, et ceci est de surcroît non acceptable pour les applications tenant compte de la topologie et qui visent à déterminer les « meilleurs nœuds », le plus rapidement possible.

Les systèmes de positionnement à base de coordonnées sont une proposition attractive s'ils sont déployés comme un service : chaque nœud pourra alors faire tourner un système de coordonnées au démarrage du système d'exploitation. Cela pourra ainsi permettre au nœud de fournir des estimations de distance à la demande, aux applications et réseaux de recouvrement. Un système de coordonnées est ainsi vu comme une "infrastructure virtuelle" qui supporte une large variété d'applications et réseaux de couverture. Cela dit, un système fournissant un service à large échelle, serait aussi une principale cible pour les pirates informatique, puisque sa perturbation pourrait mener au disfonctionnement ou à l'effondrement de plusieurs applications à la fois. Le fait que les systèmes de coordonnées actuels supposent que les nœuds évoluant dans le système coopèrent entièrement et

honnêtement les uns avec les autres, et que cela peut aussi les rendre vulnérables aux attaques, a été l'une de nos motivations premières pour adresser le problème de sécurité dans ce type de systèmes. En particulier, les attaques internes lancées par des nœuds (potentiellement en collusion) infiltrant le système, pourrait s'avérer très dangereuses. De ce fait, sécuriser la base de la prédiction de distance pour plusieurs applications serait plus critique, que de détailler les artefacts d'une quelconque sécurité d'une application particulière. Dans la section 7.2, nous montrerons qu'il est assez simple de s'attaquer au service de coordonnées, et nous détaillerons les principaux moyens pour y arriver, ainsi que leurs impacts sur les systèmes à base de coordonnées.

7.2 Les attaques contre les systèmes de coordonnées et leurs impacts

Cette section a pour principal but d'identifier les différentes attaques contre les systèmes de coordonnées et de montrer leur efficacité à travers leur expérimentation sur un système décentralisé représentatif : Vivaldi[1]. Notre étude basée sur des simulations montre ainsi qu'il est assez simple de déstabiliser ces systèmes et cela même si quelques mécanismes de sécurité existent. En particulier, nous quantifions les effets de stratégies d'attaques qui visent à (i) introduire du désordre dans le système, (ii) tromper des nœuds honnêtes afin de leur imposer des coordonnées loin de leur positions correctes et (iii) isoler certains nœuds cibles à travers des collusions de nœuds malicieux.

7.2.1 Menaces et classification des attaques

Nous considérons des nœuds malicieux ayant accès aux mêmes données que les utilisateurs légitimes. Cela veut dire que tous les participants ne sont pas forcément des entités de confiance, ou alors que certains nœuds malicieux ont pu contourner un quelconque mécanisme d'authentification. Les nœuds malicieux sont capables d'envoyer des informations erronées quand les autres nœuds mesurent leurs distances vers eux, ou alors d'envoyer des informations manipulées en recevant des requêtes de positionnement. Ils peuvent aussi affecter certaines métriques observées par leur cible. Les classes principales d'attaques sur les systèmes de coordonnées sont :

- Le Désordre : le but principal d'une telle attaque est de créer le chaos sous une forme d'une attaque de déni de service. Cela entraîne des erreurs élevées dans le positionnement des nœuds, ou une non-convergence de l'algorithme de calcul des coordonnées. L'attaque consiste simplement à maximiser l'erreur relative des autres nœuds dans le système, passivement en choisissant de ne pas coopérer ou en falsifiant les coordonnées, ou alors activement en retardant les mesures effectuées par les nœuds honnêtes.
- L'Isolation : où les nœuds « cibles » seraient isolés dans l'espace de coordonnées. Cette attaque cible un nœud particulier, dans le but de le convaincre qu'il est positionné dans une zone isolée du réseau. Le but ultime est, par exemple, d'obliger ce nœud cible à se connecter à un nœud complice, considéré comme étant le plus proche dans la zone isolée, afin de pouvoir par la suite lancer des attaques d'analyse de trafic,

[1] Dans [R.12], nous présentons les expérimentations liées à NPS (système de coordonnées basés sur des balises fixes), mais nous les omettons ici par souci de concision.

de rejet de paquet ou d'interception (man in the middle). Une manière d'aboutir à la réalisation d'une telle attaque, est de retarder les mesures envoyées par la victime, et de falsifier ses propres coordonnées, de manière à ce que la victime calcule des coordonnées plus élevées que la réalité, et s'éloigne ainsi de sa position (et par la même des autres nœuds).

- La Répulsion : où un nœud malicieux essaie de convaincre ses victimes qu'il est positionné loin d'eux afin de réduire son attractivité, et ainsi, par exemple, se soulager de la charge de retransmission de paquets, en ne coopérant pas dans les processus normaux de l'application. Une manière de réussir une telle attaque, est de faire en sorte de montrer que ses conditions (en termes de performances ou de position) sont pires qu'elles ne le sont en réalité. Cela est accompli en retardant les mesures, et/ou en manipulant les coordonnées transmises aux autres nœuds.
- Le Contrôle du système : cette attaque est possible dans le cas où des nœuds « normaux » peuvent être considérés comme des balises, i.e. la plupart des systèmes existants sauf les systèmes centralisés. Dans les systèmes hiérarchiques, comme NPS, les nœuds essayent de monter dans la hiérarchie dans le but de tromper ou d'influencer le maximum de nœuds honnêtes.

7.2.2 Indicateurs de performance

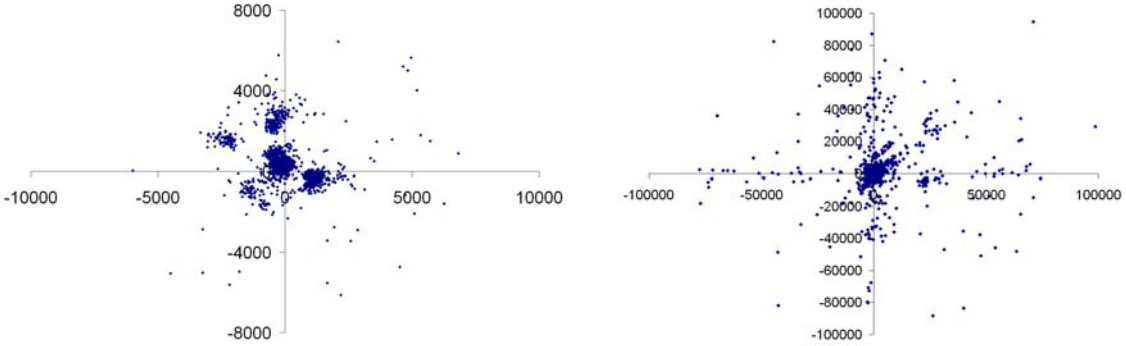
Nous avons utilisé l'erreur relative (définie dans 7.1.3 pour Vivaldi) comme indicateur principal de performance. La moyenne des erreurs relatives de tous les nœuds dans le système est utilisée pour représenter la précision globale du système. Par ailleurs, comme nous nous concentrons sur l'étude de l'impact des nœuds malicieux sur tout le système, nous avons aussi introduit le « ratio d'erreur relative » (ou ratio tout court) comme étant le rapport entre l'erreur relative mesurée en présence de nœuds malicieux normalisée et l'erreur relative du système sans tricheurs (ce scénario étant utilisé comme le meilleur des scénarios). Une valeur strictement supérieure à 1 indique manifestement une dégradation dans la précision du système. Nous utilisons aussi dans un objectif de comparaison, un scénario dans lequel les nœuds choisissent leurs coordonnées de façon aléatoire sans aucun ajustement.

7.2.3 Scénario de simulation

Nous avons utilisé le simulateur p2psim pour les scenarios de simulation pour Vivaldi. Nous avons aussi utilisé les données « King » [7.18] pour représenter les délais entre les nœuds du système. Notre réseau de recouvrement contenait ainsi au maximum 1740 nœuds, les délais entre ces nœuds étant obtenus à partir des données King. Parmi ces nœuds, un certain pourcentage de nœuds (pouvant aller jusqu'à 75%) sont des nœuds « malicieux ». Chaque nœud Vivaldi a 64 voisins, dont 32 sont choisis de façon que le RTT soit inférieur à 50 ms. La constante C_c utilisée pour le calcul du pas d'adaptation est mise à 0.25 (comme préconisé dans [7.9]). Le système est considéré comme « ayant convergé » ou « s'étant stabilisé » si toutes les erreurs relatives des nœuds convergent à des valeurs qui varient au maximum de 0.02 pour 10 ticks de simulation. Toutes les simulations Vivaldi sans nœuds malicieux ont convergé avant 1800 ticks de simulation (ce qui correspond à 8 heures à peu près, un tick représentant 17 secondes environ). Nous utilisons un espace de coordonnées à deux dimensions dans les résultats présentés dans ce document.

7.2.4 Impact d'une attaque de désordre

L'attaque de désordre n'a pas d'objectif spécifique concernant un utilisateur ou bien un ensemble d'utilisateurs donnés. Quand un nœud malicieux est sollicité, il répond en indiquant une position x_j choisie de façon aléatoire associée à une erreur relative e_j très faible (0.01). En plus, la réponse de la mesure est retardée d'une valeur aléatoire entre 0.1 et 1 seconde. Dans cette attaque, il n'est pas une condition de s'assurer de la crédibilité du mensonge : même si la valeur du retard n'est pas cohérente avec la position x_j, le nœud i contactant le nœud malicieux considèrera que son erreur locale est élevée et ajustera ses coordonnées avec une valeur plus élevée du pas d'adaptation (car le nœud j a envoyé une valeur e_j très faible). La figure 7.1 montre l'impact de cette attaque sur un espace de coordonnées Vivaldi.

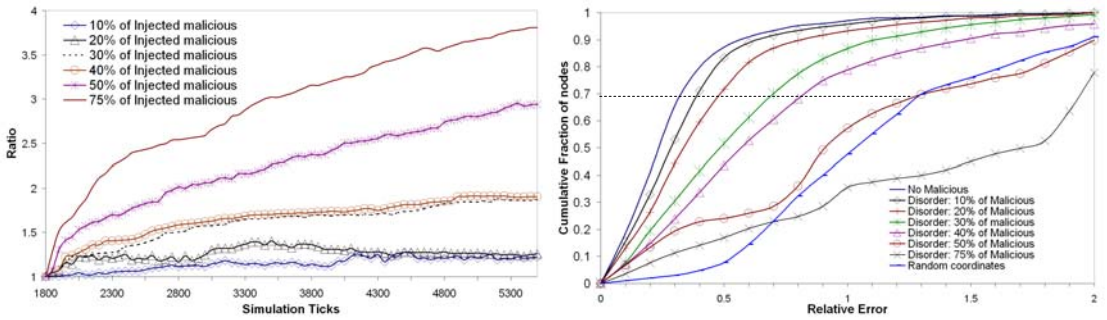


(a) sans nœuds malicieux (b) avec 10% de nœuds malicieux

Figure 7.1 : Impact de l'attaque de désordre sur un système Vivaldi avec 1740 nœuds

Nous remarquons que la topologie présente une structuration en groupe dans la figure 7.1 (a). Cette structuration est fortement modifiée avec seulement 10% de nœuds malicieux. Ceci est du au fait que les nœuds « honnêtes » vont propager dans tout le système les erreurs introduites par les nœuds malicieux.

Pour mesure l'ampleur des dégâts avec plus de nœuds malicieux, nous lançons des simulations avec une proportion croissante de nœuds qui deviennent malicieux avec la convergence de la simulation (au tick 1800). Les résultats présentés dans la figure 7.2 (a) montrent que le ratio de l'erreur relative du système a une valeur de 1.7 avec 30 à 40% de nœuds malicieux, et peut atteindre 3 avec 50% de nœuds malicieux, voire s'approcher de 4 fois pire qu'un système sain pour 75% de nœuds malicieux. En plus on remarque que, même si le système a « convergé » dans le sens que les erreurs relatives individuelles se sont stabilisées, les valeurs des erreurs sont si élevées qu'une variation importante des coordonnées n'aura presque pas d'impact sur l'erreur associée. La convergence ici signifie donc que le système ne s'améliore plus ni se détériore, même si les coordonnées des nœuds changent de façon importante.



(a) Le ratio d'erreur relative (b) La distribution cumulative de l'erreur relative

Figure 7.2 : Attaques avec une population croissante de nœuds malicieux

La figure 7.2 (b) montre la distribution cumulative de l'erreur relative des victimes pour une population croissante de nœuds malicieux. Il est clair qu'à partir de 30% de nœuds malicieux le système est fortement secoué (les erreurs augmentent). A partir de 50% de nœuds malicieux dans le système, le système s'effondre avec plus de 30% des nœuds honnêtes calculant des coordonnées avec erreurs relatives plus élevées que dans le cas d'un choix aléatoire des coordonnées. Ces résultats nous ont poussés à étudier de façon plus approfondie l'impact des autres types d'attaques.

7.2.5 Impact d'une attaque structurée

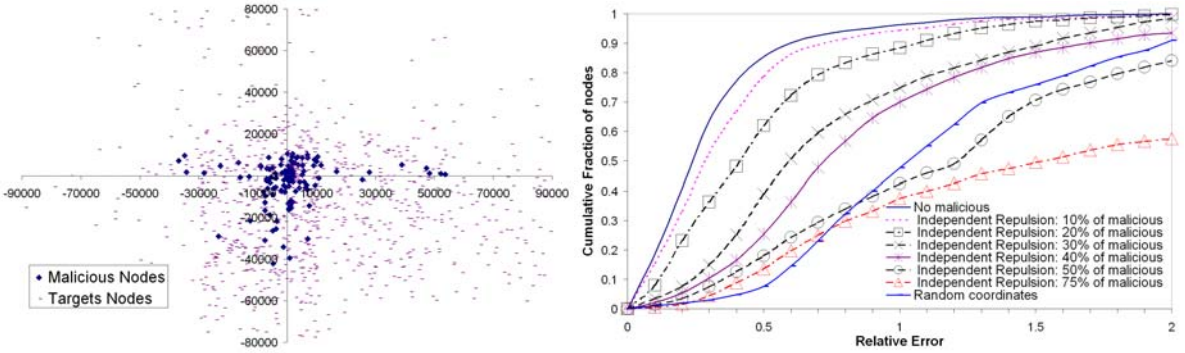
Nous avons programmé une attaque par répulsion dans laquelle un nœud malicieux essaie de faire croire à un nœud victime qu'il a une position X_{cible} (différentes de sa position réelle). Cette attaque a un objectif spécifique (isoler des nœuds, repousser des nœuds d'autres, etc.). Il est à noter que la position X_{cible} devrait être choisie de façon à assurer la crédibilité du mensonge. Cela veut dire que la distance prédite entre les deux nœuds après le mensonge devrait correspondre à la distance mesurée. Comme un nœud ne peut réduire les délais du réseau mais qu'il peut en revanche les rallonger en retardant l'envoi de la réponse à une requête, il faudrait donc de choisir la position cible en tenant compte de cela. En considérant que l'attaquant connaît la position réelle du nœud victime, il pourra calculer la valeur du RTT requise pour que le mensonge soit crédible :

$$RTT_{requis} = (\| X_{cible} - X_{réelle} \| / \delta) + \| X_{cible} - X_{réelle} \|$$

et retarder l'envoi de la réponse d'une valeur égale à :

$$RTT_{requis} - 2 \times (T_{réception} - T_{emission})$$

Il est à noter que tous les nœuds malicieux choisissent des positions cibles loin de l'origine.



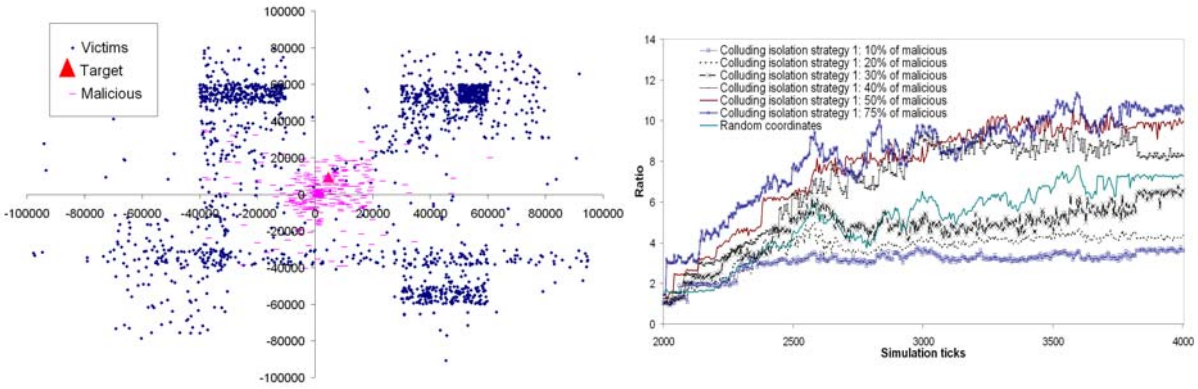
(a) 10% de nœuds malicieux (b) CDF des erreurs relatives

Figure 7.3 : Impact d'une attaque structurée

L'impact de cette attaque est montré en figure 7.3 (a). Dans ce scénario, chaque nœud malicieux choisit une position cible pour chaque nœud honnête indépendemment des autres. La position cible est donc la même pour chaque paire attaquant-victime. En comparant cette figure à 7.1 on note une disparition complète des regroupements d'origine dans le système Vivaldi sans attaque. Les distributions cumulatives des erreurs relatives montrées en figure 7.3 (b) confirment la nuisance plus grande de cette attaque, les pentes des courbes CDF sont plus faibles que celles de la figure 7.2 (b), ce qui indique une plus grande dégradation du système que dans le cas d'une attaque de désordre.

7.2.6 Impact d'une attaque avec collusion

Il s'agit ici d'une attaque structurée dans laquelle les nœuds malicieux agissent de façon concertée. Ils peuvent par exemple essayer d'éloigner des nœuds victimes d'un nœud cible spécifique. Ceci a lieu en se mettant d'accord pour une distance donnée du nœud cible pour chaque victime et en agissant de concert pour diriger les victimes vers leur position. La figure 7.4 (a) montre l'impact de cette stratégie d'attaque (dite « stratégie 1 ») sur le système Vivaldi étudié.



(a) 10% nœuds malicieux (b) ratio de l'erreur relative

Figure 7.4 : Impact d'une attaque avec collusion

La figure 7.4 (b) montre qu'au-delà de 30% de nœuds malicieux, le système devient pire qu'un système dans lequel les coordonnées sont choisies de façon aléatoire. D'autres

stratégies d'attaques peuvent être préparées : les nœuds malicieux se mettent d'accord pour annoncer des coordonnées dans une zone « retranchée » du système de coordonnées et y « amener » un nœud victime en le convainquant que sa position est dans cette zone. Nous désignons cette attaque par « stratégie 2 ». Nous présentons dans [R.15] une étude détaillée de l'impact de ces stratégies d'attaques sur le système.

7.2.7 Autres considérations

Nous avons étudié pour toutes les attaques ci-dessus, l'impact de la dimension de l'espace de coordonnées et de la taille du système. La figure 7.5 montre les résultats des simulations. Dans la figure 7.5 (a), l'erreur relative moyenne des nœuds honnêtes est mesurée après la re-convergence du système. Nous pouvons constater que plus un système Vivaldi est « précis » en l'absence d'attaque, plus il est vulnérable aux attaques (et ceci a été vérifié pour tous les types d'attaques, même si les résultats montrés ici concernent uniquement une attaque par désordre). La figure 7.5 (b) montre l'impact de l'attaque (le ratio de l'erreur relative) en fonction de la taille du système (le nombre de nœuds participant au réseau de recouvrement) longtemps après le début de l'attaque. Il apparaît qu'un système plus grand est plus difficile à déstabiliser. Dans un système plus grand les « forces du bien » arrivent à résister plus aisément aux « forces du mal ».

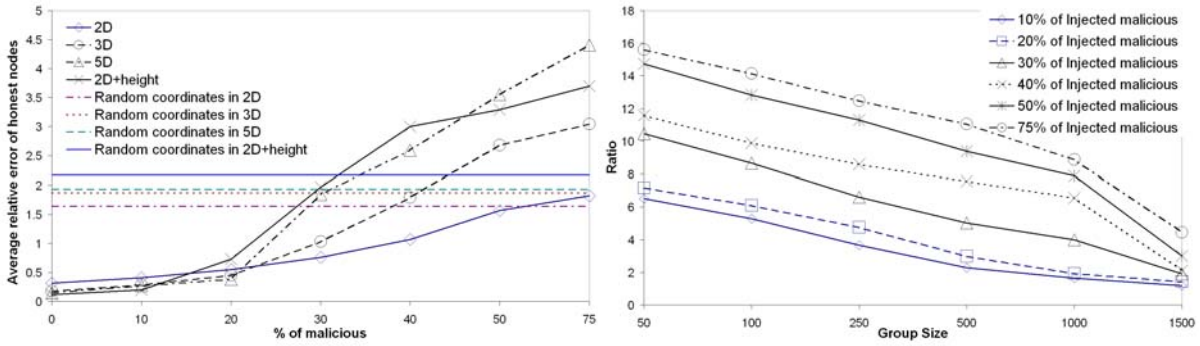


Figure 7.5 : Impact de la dimension de l'espace (a) et de la taille du système (b)

7.2.8 Conclusion

Nos simulations ont montré que les systèmes de coordonnées, décentralisées ou à base de balises[1], peuvent être déstabilisés, bien que nous avons aussi constaté que les systèmes plus larges sont plus résistants aux attaques. Nos résultats ont aussi montré qu'il y a un compromis intrinsèque au système entre précision et vulnérabilité. En effet, nous avons montré, en variant les dimensions utilisées par les systèmes, que plus le système est précis, plus il est vulnérable pour la même proportion d'attaquants dans le système.

Nous avons aussi pu observer que, quand les attaques tournent à plein régime, les performances des systèmes de coordonnées (ainsi que les applications qui les supportent), peuvent facilement se dégrader au delà de celle d'un système qui utiliserait des coordonnées

[1] Les résultats pour les systèmes à balises fixes tels que NPS ne sont pas montrés dans ce document. Voir [R.12] pour une présentation détaillée des résultats sur ce sujet.

aléatoires. Enfin, nous avons montré que les conséquences des attaques durent très longtemps à cause des résidus d'erreurs se propageant et infectant tous les nœuds dans le système.

7.3 Sécurisation de la phase de calcul de coordonnées

Dans cette section, nous nous fixons comme objectif de proposer une méthode générale et efficace pour sécuriser les systèmes de coordonnées lors de leur phase de calcul des coordonnées. Nous montrons ainsi en premier lieu, que la dynamique d'un nœud, dans un système de coordonnées exempt de comportement malicieux ou anormal, peut être modélisée par un modèle linéaire et pistée par un filtre de Kalman.

7.3.1 Modélisation du processus de calcul des coordonnées

Le but des systèmes de coordonnées, sans tenir compte de la manière de calcul, ni de l'aspect « dimension » de l'espace géométrique utilisé, est d'attribuer des coordonnées à un nœud dans le système de manière, à ce qu'à tout moment, on puisse assimiler la distance géométrique entre deux nœuds dans le système de coordonnées à la distance réseau réelle, mesurée en RTT entre ces deux nœuds. La mesure RTT en revanche, manifestement, dépend de l'état du réseau (par exemple de la charge du trafic, de l'état des files d'attentes dans les routeurs, etc.), mais aussi de l'état des systèmes d'exploitation des nœuds eux-mêmes générant du bruit dans les mesures. Ainsi la valeur exacte de RTT varie continuellement. Cela se répercute sur les calculs de coordonnées effectuées par les systèmes de positionnement, en dehors du fait qu'ils soient décentralisés ou basés sur des balises fixes. A chaque étape de calcul des coordonnées, la précision de ce calcul est déterminée en observant la déviation entre le RTT mesuré entre les deux nœuds concernés i et j (RTT_{ij}^{n}) et la distance estimée dans le système de coordonnées ($\|X_i^n - X_j^n\|$). Cette précision est définie comme étant l'*erreur relative mesurée*:

$$D_n = | \, \|X_i^n - X_j^n\| - RTT_{ij}^{n} \, | \, / \, RTT_{ij}^{n}$$

Le but de n'importe quel système de coordonnées est de minimiser un indicateur de coût (l'erreur moyenne quadratique) qui capture l'erreur relative mesurée. Les erreurs relatives mesurées sont sujettes à des fluctuations des RTTs pour les raisons mentionnées ci-dessus, en particulier, les congestions transitoires dans le réseau et les problèmes d'ordonnancement dans les systèmes d'exploitation. Afin d'isoler l'impact de ces fluctuations de RTT sur la détection d'anomalie, nous introduisons Δ_n, l'*erreur relative nominale*, que notre nœud pourrait obtenir si à l'étape n du calcul de coordonnées les RTTs ne fluctuaient pas, et si les mesures de ces mêmes RTTs étaient exactes à tout instant n. Une anomalie devient alors une simple observation de larges déviations de D_n, l'erreur relative mesurée, de sa valeur nominale Δ_n. Comme plusieurs sources contribuent à la déviation de D_n de sa valeur nominale, il est raisonnable de supposer que les deux valeurs sont fonctions l'une de l'autre comme suit :

$$D_n = \Delta_n + U_n \quad \text{(Eq. 7.1)}$$

où U_n est une variable gaussienne de moyenne 0 et de variance $\nu_{U,}$, où U_n est dû à des erreurs de mesures des RTTs, ainsi qu'aux erreurs dans le calcul des coordonnées des nœuds.

Nous nous focalisons maintenant sur la dynamique du système dans son régime nominal. Nous définissons ainsi l'*erreur du système* W_n qui représente les fluctuations des RTTs, ainsi que l'incertitude du modèle lui-même. Puisque W_n résulte de plusieurs sources contributrices, il est aussi raisonnable de supposer que c'est un processus gaussien (avec une moyenne $\hat{w}$ et une variance ν_W).

Comme première approximation, le processus Δ_n peut ainsi être modélisé comme un modèle auto régressif d'ordre 1 :

$$\Delta_{n+1} = \beta\Delta_n + W_n \quad \text{(Eq. 7.2)}$$

où β est un facteur constant strictement inférieur à 1, afin que l'erreur relative puisse converger vers un régime stationnaire indépendamment des conditions initiales. Les équations 7.2 et.7.1 définissent alors un modèle linéaire d'évolution de l'erreur relative d'un nœud.

Notre objectif est d'obtenir des prédictions de cette erreur relative à partir de ce modèle. De par ses propriétés linéaires, le filtre de Kalman est utilisé dans ce cas pour pister l'évolution de l'erreur relative nominale Δ_n et pour aussi obtenir une prédiction ($\bar{E}$) de cette erreur relative. $\bar{E}_{n|n-1}$ désigne l'estimation de Δ_n sachant les observations du délai réseau jusqu'à l'étape n-1 et $\bar{E}_{n|n}$ désigne l'estimation de Δ_n après que la mesure de D_n est effectuée. En utilisant la méthode EM (Expectation Maximization method), nous calibrons le filtre de Kalman afin de déterminer pour chaque nœud dans un système normal, les valeurs des paramètres utilisés par le filtre. Ces paramètres permettent donc de prédire le comportement d'un nœud dans un système sain et de ne pas ajuster ses propres coordonnées suite à des échanges avec des utilisateurs qui nous paraissent « louches ».

Cela nous a menés à la proposition d'une infrastructure de nœuds experts : ceux-ci sont des nœuds de confiance, se positionnant dans l'espace des coordonnées en utilisant exclusivement d'autres nœuds experts. Ils sont ainsi immunisés contre n'importe quel comportement malicieux dans le système. Pendant le calcul de leurs propres coordonnées, les autres nœuds peuvent ainsi utiliser les paramètres du filtre calibré d'un nœud expert non assujetti à un comportement malicieux, pour détecter et filtrer toute activité malicieuse ou anormale.

7.3.2 Validation du modèle

Une combinaison de simulations et d'expérimentations PlanetLab a été utilisée[1] pour démontrer la validité du modèle proposé, i.e. la pertinence du modèle pour représenter le comportement « normal » du système. Nous reportons ici les résultats obtenus pour le système Vivaldi. Nous avons ainsi pu montrer en premier lieu la validité de nos hypothèses, en particulier la supposition que le bruit du système W_n est un processus gaussien. Ceci est fondamental afin de pouvoir utiliser le cadre du filtre de Kalman. Ensuite, nous avons montré que les erreurs relatives prédites par le filtre et celles réellement calculées par les nœuds dans le système sont très semblables. Lorsque le modèle a convergé (i.e. la méthode EM a convergé et les variations pour tous les paramètres varient d'une valeur inférieure à 0.02), nous relançons le système de coordonnées, et nous observons les erreurs de prédiction comme étant la valeur absolue entre l'erreur prédite par le filtre de Kalman au niveau d'un nœud et

[1] Pour les simulations nous reprenons les mêmes scénarios que la section précédente. Pour les expérimentations PlanetLab, nous les avons réalisées sur 280 nœuds répartis dans le monde entier et implémentant une version modifiée de Vivaldi.

l'erreur relative mesurée réelle. Nous montrons dans la figure 7.6 l'évolution typique des erreurs relatives réelles et prédites pour un nœud PlanetLab implémentant le système Vivaldi.

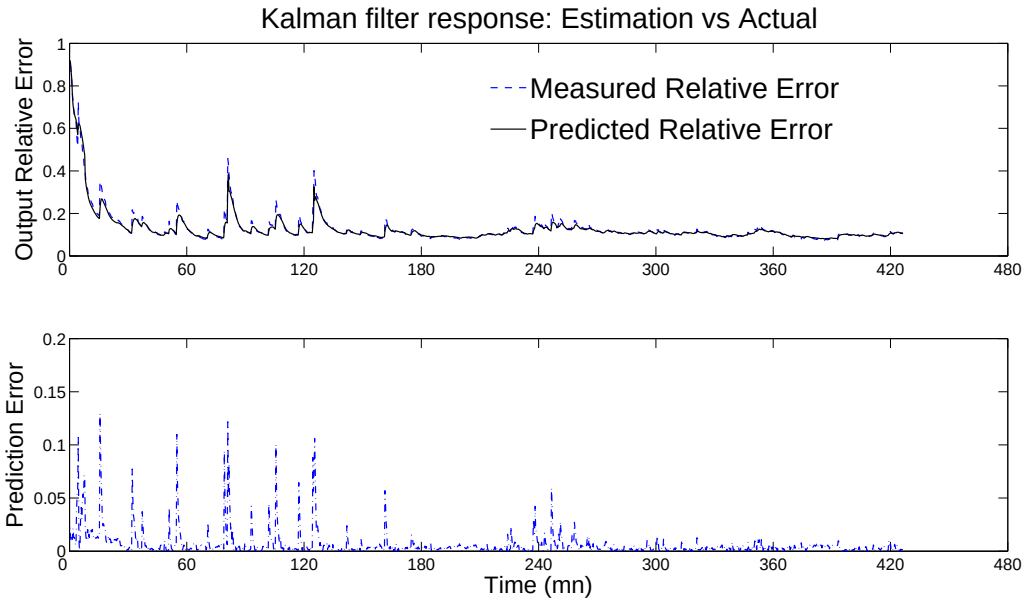


Figure 7.6 : Erreurs relatives mesurée et prédite (haut). Erreur de prédiction (bas)

On peut remarquer que les deux courbes du graphe d'en haut sont tellement proches qu'on peut à peine les distinguer. Le graphe d'en bas montre justement la différence entre ces deux courbes (l'erreur de prédiction). Nous remarquons que cette erreur est très faible (à noter l'échelle), ce qui montre que le filtre de Kalman calibré peut très bien représenter et pister le comportement d'un nœud dans le système de coordonnées.

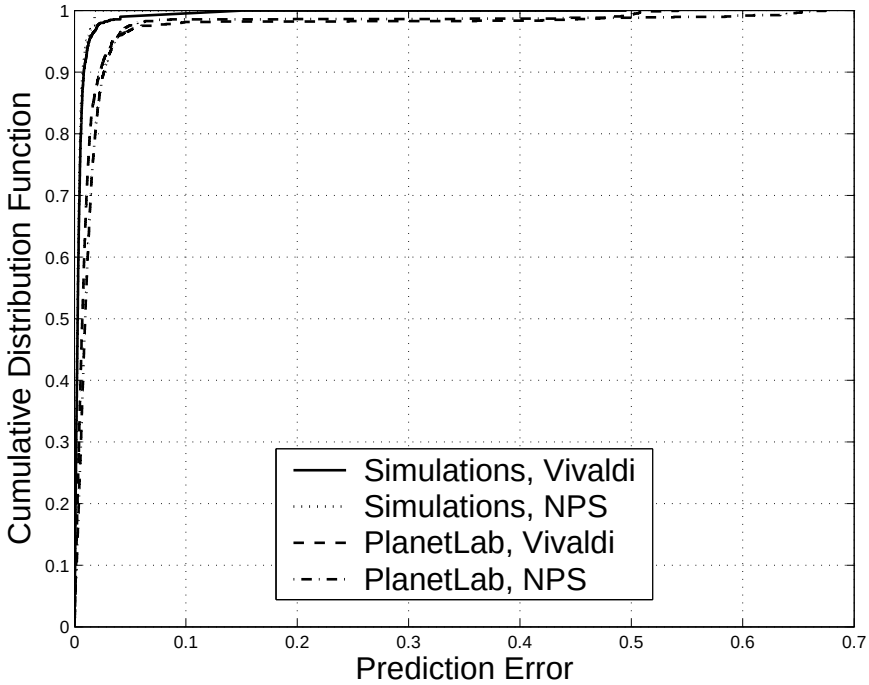


Figure 7.7 : CDF des erreurs de prédiction

La figure 7.7 montrant la CDF des erreurs de prédiction[1] confirme cela: la grande majorité des prédictions sont excellentes, avec l'existence de quelques observations aberrantes (outliers) ce qui nous a amenés à représenter l'erreur globale du système par la CDF du 95ème percentile des erreurs observées au niveau de chaque nœud normal.

Nous avons par la suite vérifié qu'un ensemble de nœuds « experts » de confiance déployés au hasard dans le réseau peut être représentatif de l'ensemble des nœuds dans le système. Une des premières questions que nous nous sommes posées était donc de savoir combien de nœuds experts fallait-il déployer pour représenter l'évolution des erreurs des autres nœuds. Nous avons ainsi pu montrer, qu'une proportion de moins de 8% de nœuds experts aléatoirement déployés dans le système de coordonnées est suffisante pour pouvoir représenter le comportement du système. La figure 7.8 montre cela : la CDF du 95ème percentile des erreurs observées au niveau des nœuds experts dans le cas où ils représentent 8% de la population totale du système, se rapprochant le plus de la courbe représentant celle des nœuds normaux.

Ayant montré que la population de nœuds experts peut représenter le comportement normal de tout le système, la question suivante que nous nous sommes posée était de savoir à quel point le comportement du système pisté par un filtre de Kalman calibré au niveau d'un nœud expert peut représenter le comportement d'un nœud normal spécifique.

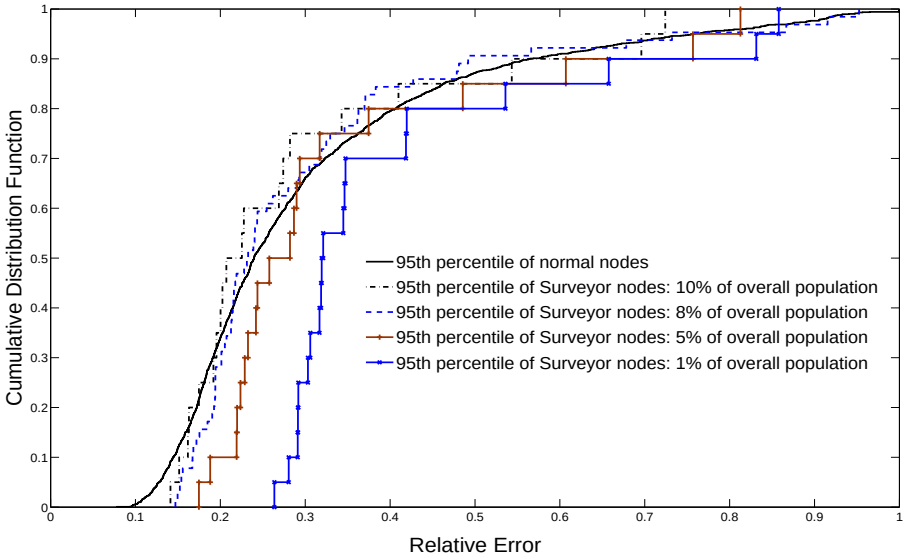
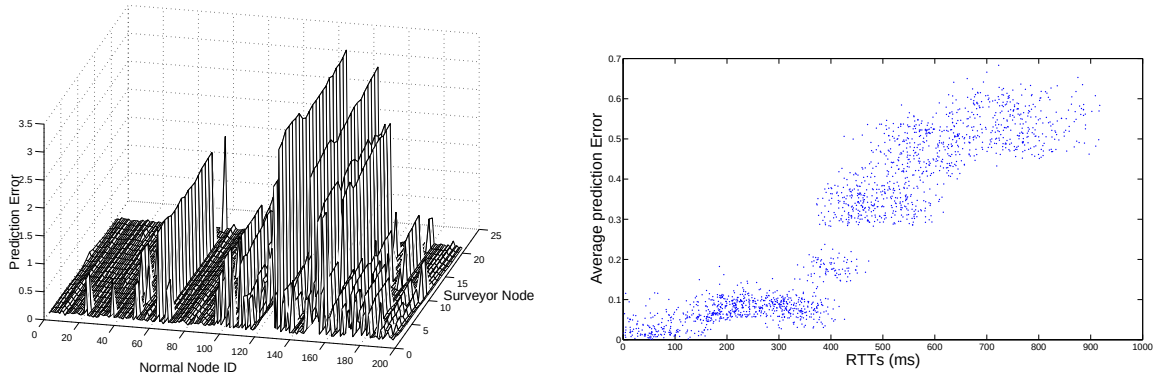


Figure 7.8 : Impact de la proportion de nœuds d'expert sur leur représentativité

Nous avons observé dans nos simulations et expérimentations PlanetLab que, même si chaque nœud normal trouve au moins un nœud expert dont le filtre de Kalman mène à de très faibles erreurs de prédiction, tous les nœuds experts ne permettent pas en revanche une bonne représentativité pour un nœud normal donné. Le choix des nœuds experts est donc primordial pour gagner en représentativité de comportement normal, en performance de prédiction et ainsi en détection de comportements malicieux. Nos expérimentations ont alors montré qu'il existe une forte corrélation entre les erreurs de prédiction et les RTTs entre le nœud normal et le nœud expert. Ceci est montré par la figure 7.9 (a) qui illustre les erreurs de prédiction pour chaque pair nœud normal – nœud expert et par la figure 7.9 (b) qui montre la corrélation entre le RTT comme distance entre le nœud expert et le nœud normal et l'erreur de prédiction.

[1] Les résultats sur NPS sont montrés mais ne seront pas détaillés.



(a) (b)

Figure 7.9 : Quel nœud expert choisir ?

En conclusion, durant leur calcul de coordonnées, les nœuds normaux obtiennent les paramètres des filtres des nœuds experts pour pouvoir prédire correctement leurs erreurs relatives, à condition que ces nœuds experts soient positionnés proches d'eux. Il est important de noter que les résultats obtenus dans cette section correspondent à une répartition aléatoire des nœuds experts. Si ces nœuds étaient placés d'une manière à mieux couvrir le réseau, les erreurs de prédiction auraient été encore plus faibles.

7.3.3 Détection des comportements malicieux

La section précédente a montré que le comportement d'un nœud normal peut être modélisé par un filtre de Kalman et que cette technique est suffisamment robuste pour que le comportement obervé d'un nœud « expert » puisse être utilisé pour représenter le comportement normal dans « le voisinage ». Si les nœuds experts sont capables de calculer leurs coordonnées dans un système exempt d'une quelconque activité malicieuse, ils peuvent alors être considérés comme des entités de confiance. Puisque ces nœuds interagissent seulement entre eux mêmes, ils sont donc immunisés contre tout comportement malicieux ou anormal, et observent donc le comportement du système dans des conditions normales. Ainsi, l'idée principale de notre méthode de détection est d'utiliser le modèle de comportement normal obtenu par les nœuds auprès des nœuds experts les plus proches pour pouvoir détecter des anomalies dans le système, et ce à chaque étape de calcul des coordonnées.

A chaque étape du calcul, chaque nœud mesure l'erreur relative D_n envers le nœud pair considéré. Nous avons dit que le filtre de Kalman fournit une prédiction de l'erreur relative $\bar{E}_{n|n-1}$ obtenue à partir des précédentes mesures. Nous obtenons donc à chaque étape, l'erreur de prédiction qui suit dans un système sain une distribution gaussienne de moyenne 0 et de variance ν_n connue et fournie aussi par le filtre. Ceci permet de détecter un comportement malicieux du nœud pair par un simple test d'hypothèse. Soit H_0 l'hypothèse que le nœud pair a un comportement normal. Le test d'hypothèse revient à vérifier si l'erreur de prédiction est suffisamment normale pour le système. Si α désigne l'agressivité du test, le problème revient à trouver la valeur du seuil t_n tel que :

$$\Pr(|D_n - \bar{E}_{n|n-1}| \geq t_n \mid H_0) = \alpha$$

Or, le filtre de Kalman permet d'établir que sous l'hypothèse H_0, (D_n - $\bar{E}_{n|n-1}$) suit une distribution normale de moyenne zéro et de variance ν_n. Ceci nous permet de calculer t_n en fonction de ν_n et de α. Si la déviation $|D_n - \bar{E}_{n|n-1}|$ dépasse le seuil t_n l'hypothèse H_0 est rejeté et le nœud pair est marqué comme étant suspect. L'étape de calcul est délaissée et la valeur D_n ignorée.

En général, lorsqu'un nœud identifie un nœud pair comme étant suspect, il le remplace par un autre voisin, sauf si c'est la première fois qu'il calcule ses coordonnées en contactant ce nœud pair. Dans ce cas, le nœud effectue une autre tentative avec un niveau d'agressivité qui tient compte de l'erreur locale du nœud. Si le test indique que le nœud pair est toujours suspect, il est écarté par le nœud. Sinon, il pourra être utilisé dans les étapes futures du calcul.

Finalement, un nœud qui rejoint le système choisit le nœud expert le plus proche de lui pour obtenir les paramètres du filtre indiquant le comportement normal du système. Si le nœud rejette la moitié des nœuds pairs lors d'une étape de calcul, il procède à un « recalibrage » en choisissant de nouveau le nœud expert le plus proche de lui et en lui demandant les paramètres du filtre. Lors de nos expérimentations, très peu de recalibrages ont été réalisés. Ceci indique que cette méthode simple pour le choix du nœud expert est convenable.

7.3.4 Évaluation des performances

Nous avons évalué en premier lieu les performances de notre protocole de détection, et nous avons prouvé que les taux de faux positifs de nos tests étaient très bas, alors que les taux de vrais positifs étaient constamment élevés et ce indépendamment des dimensions de l'espace géométrique utilisé, comme le montre la figure 7.10.

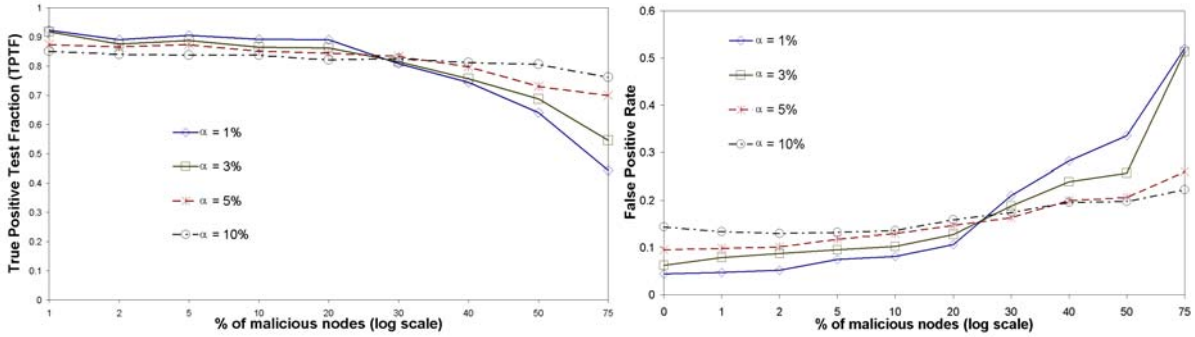


Figure 7.10 : Performance de la méthode de détection

Nous avons par ailleurs, pu démontrer à travers nos simulations et expérimentations PlanetLab, que le système Vivaldi se trouvait muni d'une protection pouvant lui permettre de fonctionner normalement et ce même lorsque un peu plus de 30% de la population était maligne. La figure 7.11 montre la CDF de l'erreur relative mesurée au niveau de tous les nœuds normaux après la re-convergence du système après une attaque. La figure montre que le système de détection utilisé protège le système presque totalement contre l'attaque (et ce pour une proportion de nœud malicieux allant jusqu'à 30%). Nous pouvons aussi constater que jusqu'à 50% de nœuds malicieux, le système de détection « arrive à tenir » réduisant ainsi l'intensité d'une attaque aussi puissante.

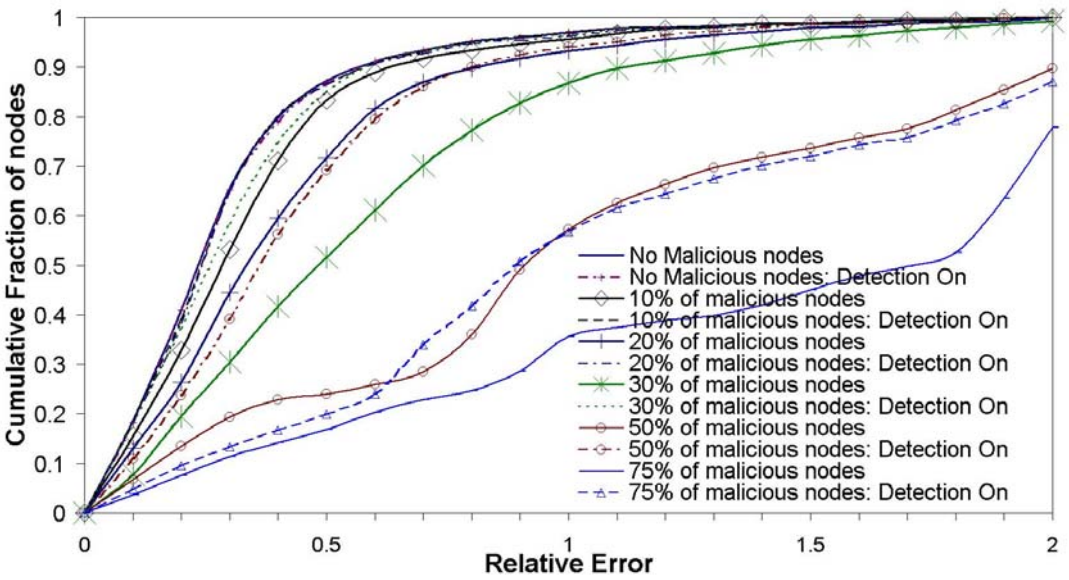


Figure 7.11 : CDF de l'erreur relative dans un système protégé

7.4 Sécurisation de la phase d'utilisation des distances

Les systèmes de coordonnées sont principalement destinés à évaluer les distances entre les nœuds, en se basant simplement sur les coordonnées échangées par les nœuds dans le système. Quelque soit le mécanisme d'échange de ces coordonnées (échanges directs, répertoire WEB, etc.), chaque nœud devrait d'une manière ou d'une autre reporter une valeur de ses coordonnées jusque là calculées. Cela permet à des nœuds malicieux d'essayer de tricher quant à leurs coordonnées transmises pour aboutir à des fins malicieuses et à des attaques sur les applications utilisant le système de coordonnées (Déni de service, Isolation, etc.). Cette section décrit très brièvement ce sujet (détaillé dans [R.11]) concernant la véracité des coordonnées transmises par les nœuds, et de la garantie de leur authenticité.

Nous avons proposé une solution basée sur l'exploitation de l'infrastructure de nœuds experts et du protocole de détections de triches décrit dans la section précédente. La méthode proposée est constituée de deux étapes. En premier lieur, il s'agit de vérifier l'exactitude des coordonnées annoncées par les nœuds lors d'estimations de distances au niveau applicatif. Nous exploiterons pour cela l'infrastructure de nœuds experts et la méthode de détection des nœuds malicieux. Dans une seconde étape, notre méthode permet de délivrer un certificat à validité limitée pour chaque coordonnée vérifiée. Les périodes de validité sont calculées à partir d'une analyse des temps d'inter changement de coordonnées observés par les nœuds experts.

En particulier, chaque nœud désirant annoncer une coordonnée à une application, doit s'adresser à un nœud expert. Celui-ci choisit un ensemble de nœuds experts entourant le nœud lui-même. Ceux-la mesurent leur distance vers le nœud afin de vérifier l'exactitude des coordonnées annoncées (en utilisant le test de détection des anomalies en erreurs relatives). Si tous les nœuds experts s'accordent à dire que les coordonnées du nœud sont valides, on lui accorde alors un *certificat*, à temps de validité limitée, incluant les coordonnées certifiées.

Un temps de validité pour chaque certificat est nécessaire parce que, le changement des conditions réseau fait en sorte que les coordonnées varient dans le temps. Au changement de ses coordonnées, un nœud honnête cesserait d'utiliser son certificat actuel, et demanderait un certificat pour ses nouvelles coordonnées. En revanche, un nœud malicieux pourrait continuer à utiliser le certificat relatif à ses anciennes coordonnées. Ainsi, un compromis est nécessaire entre passage à l'échelle et validité des certificats. Une de nos contributions a été donc d'étudier les temps d'inter-changements (i.e. le temps qui sépare les changements de coordonnées d'un nœud) dans le système Vivaldi déployé sur PlanetLab. Nous avons pu observer que ces temps d'inter-changements suivaient une distribution « lognormale » pour la plupart des cas, et une distribution Weibull dans certains cas particuliers (à noter que ces deux distributions sont des distributions longue traine). A travers des expérimentations PlanetLab, nous avons pu montrer que cette méthode est très efficace, avec un taux de détection élevé (vrai positifs) et un taux de faux positif très bas. Cette méthode permet aussi un compromis entre passage à l'échelle et sécurité. En effet, les temps de validité des certificats sont très rarement sur estimés, alors qu'ils ne déclenchent pas souvent des re-certifications.

7.5 Conclusions et Travaux Futurs

Nous avons pu montrer dans nos travaux que divers types d'attaques pouvaient être lancées contre les systèmes de coordonnées. Nous avons alors exploré dans une seconde étape, un moyen général et efficace pour sécuriser de tels systèmes. Dans la section 7.3, nous avons montré que la dynamique d'un nœud, dans un système de coordonnées sans activité malicieuse, peut être modélisée par un modèle d'états linéaires, et traquée par un filtre de Kalman. De plus, les paramètres d'un filtre calibré au niveau d'un nœud donné, peuvent être utilisés pour modéliser et prédire le comportement dynamique d'un autre nœud, tant que ces deux nœuds sont proches l'un de l'autre dans le réseau. Les prédictions effectuées par le filtre au niveau de chaque nœud lui permettent de comparer les erreurs relatives prédites et celles mesurées. Une déviation trop grande entre ces deux valeurs permet au nœud de ne pas considérer la mesure qu'il vient d'effectuer, le munissant ainsi d'un moyen de défense lors du calcul de ses coordonnées.

Cependant, même cette sécurité reste insuffisante, lorsqu'on sait que ces systèmes sont surtout utilisés par les applications pour l'estimation des distances entre nœuds. Dans la section 7.4, nous avons alors abordé le problème de l'authenticité des coordonnées transmises (ou annoncées) par les nœuds. Nous avons étudié les temps d'inter-changements des coordonnées, et pu observer qu'ils suivaient souvent une distribution lognormale (dans de rares cas une distribution Weibull), et qu'un temps de validité de ces coordonnées pouvait être déterminé par un nœud expert. Celui-ci vérifie l'exactitude de la coordonnée annoncée par le nœud normal et lui délivre un certificat à temps de validité égale à la validité estimée de sa coordonnée actuelle.

Nos résultats, notamment pour le protocole de détection des nœuds malicieux lors du calcul des coordonnées, se sont basés sur un déploiement aléatoire des nœuds experts. Bien que la représentativité de ces derniers augmente avec un déploiement plus stratégique, ces résultats ont montré l'efficacité de notre protocole de sécurisation des systèmes à base de coordonnées. On pourrait envisager d'autre stratégies de déploiement plus optimal des nœuds experts afin d'aboutir à une meilleure représentativité à large échelle.

Considérer des attaques directes sur l'infrastructure des nœuds experts est aussi une de nos directions futures. En effet, au moins deux attaques peuvent être envisagées contre les nœuds experts. La première serait tout simplement un déni de service par inondation des nœuds experts. La seconde serait de s'attaquer uniquement aux liens entre certains nœuds experts, rendant ainsi les filtres au niveau de ces nœuds mal-calibrés. Une calibration biaisée est équivalente à une faible représentativité des nœuds experts, résultant en positionnement incorrect ou même en un taux élevé de faux positifs dans notre protocole de détection. Étudier l'impact de telles attaques et intégrer des mécanismes de défense contre des attaquants ciblant les nœuds experts, est aussi un de nos travaux futurs.

Nous avons aussi montré que notre méthode pour la validation des coordonnées annoncées est efficace contre des attaques sophistiquées. Cependant, il est à noter qu'un nœud peut connaître à quel moment son prochain calcul de coordonnées va avoir lieu, et ainsi quand est ce que ses coordonnées vont probablement changer. Un nœud malicieux peut chercher à exploiter cette connaissance pour obtenir un certificat (pour ses coordonnées actuelles) juste avant l'instant de son nouveau positionnement, et cela lui permettrait d'avoir un certificat valide pour des coordonnées anciennes. Afin de pallier ce problème, on peut envisager que les nœuds experts effectuent des vérifications périodiques aléatoires des coordonnées certifiées. Si ces derniers échouent le test de vérification, le nœud concerné pourrait être pénalisé, pour avoir utilisé un certificat périmé. Nous planifions de lancer de nouvelles expérimentations intégrant ces mécanismes pour évaluer de nouveau l'efficacité de notre méthode de certification. Nous nous fixons comme objectif final de fournir un service sécurisé pratique de coordonnées. En dehors de la précision des systèmes de coordonnées, nous pensons que sécuriser de tels systèmes, est une condition nécessaire à leur déploiement. En effet, sans la sécurité de chacune des phases de calcul des coordonnées et d'estimation des distances, nous avons pu montrer que des attaques simples pouvaient réduire ces systèmes à une utilité futile. Nous gageons que la résistance aux attaques, que nous proposons dans nos travaux, pourrait agir comme un catalyseur au déploiement massif des services de coordonnées à large échelle, bloc de base qui sera très utile pour la nouvelle architecture de l'Internet.

Références bibliographiques

Les références sont listés dans le format [x. y], où y est le numéro de la référence et x est soit le numéro du chapitre dans lequel est mentionnée la référence, soit la lettre R quand il s'agit d'une publication de l'auteur. Les références en [R.y] sont listées à partir de la page 115.

[1.1] CIGALE, The packet switching machine of the CYCLADES Network, Louis Pouzin, in Proceedings of IFIP, Stockholm (August 1974), pp 155-159.

[1.2] The OSI95 Transport Service with Multimedia Support, André Danthine (Ed.), 1994, ISBN 3540583165.

[1.3] V. Jacobson, M. J. Karels. « Congestion avoidance and control », November 1988.

[1.4] M. Gerla, M. Y. Sanadidi, R. Wang, A. Zanella, C. Casetti, S. Mascolo, « *TCP Westwood: Congestion Window Control Using Bandwidth Estimation* », In Proceedings of IEEE Globecom 2001, Volume: 3, pp 1698-1702, San Antonio, Texas, USA, November 25-29, 2001

[1.5] G. Chesson, "The protocol engine project," UNIX Review, vol. 5, no. 9, pp. 70-77, Sept. 1987.

[1.6] D. D. Clark and D. L. Tennehouse, "Architectural Considerations for a New Generation of Protocols", Proceedings of ACM SIGCOMM, 1990.

[1.7] D. Tennenhouse. « Layered Multiplexing Considered Harmful », in Protocols for High-Speed Networks, Rudin and Williamson (Editors), North Holland, Amsterdam, 1989. Based on a presentation at IFIP WG 6.1/WG6.4 International Workshop on Protocols for High-Speed Networks, Zurich, May 1989.

[1.8] Voir la description du projet Hipparch : http://www-sop.inria.fr/rodeo/hipparch/

[2.1] The Rendez-Vous video conferencing application. See http://franklyonnet.free.fr/IVStng/FLYLibs/flylibs.html.

[2.2] E. Léty, T. Turletti , F. Baccelli. « SCORE: a scalable communication protocol for large-scale virtual environments », IEEE/ACM Transactions on Networking (TON), vol.12, n.2, p.247-260, April 2004

[2.3] L. Zhang, VirtualClock: a new traffic control algorithm for packet-switched networks, ACM Transactions on Computer Systems (TOCS), Volume 9, Issue 2, pp. 101–124, May 1991.

[3.1] O. Arbouche, QoS support over a satellite link, Engineering diploma report, EMI/INRIA Sophia Antipolis, 1999.

[3.2] Voir la description du projet @IRS : http://www-sop.inria.fr/rodeo/rrizo/AIRS/DescrTech-airs.html.

[3.3] T. Block, C. Barakat, W. Dabbous, F. Filali. « Reliable Multicast file transfer over Satellite Networks », Rapport interne de collaboration INRIA-Hitachi, septembre 2002.

[3.4] C. Barakat, M. Malli, N. Nonaka, "TICP: Transport Information Collection Protocol ", in Annals of Telecommunications, vol. 61, no. 1-2, pp. 167-192, January-February 2006.

[3.5] L. Rizzo, L. Vicisano, RMDP: an FEC-based Reliable Multicast protocol for wireless environments, ACM Mobile Computing and Communications Review , num. 2, vol. 2, pp. 23-, 1998.

[4.1] David D. Clark, John Wroclawski, Karen R. Sollins, Robert Braden: Tussle in cyberspace: defining tomorrow's internet. IEEE/ACM Transactions on Networking 13(3): 462-475 (2005).

[4.2] T. Anderson, L. Peterson, S. Shenker, and J. Turner. “Overcoming the Internet Impasse Through Virtualization”. IEEE Computer, April 2005

[4.3] Seminaire de Scott Shenker : http://cleanslate.stanford.edu/seminars/shenker.php

[4.4] Presentation de Van Jacobson : http://video.google.com/videoplay?docid=-6972678839686672840

[4.5] http://www.nets-find.net/ThirdPIMeeting.php

[4.6] Rapport du workshop de la NSF sur “Overcoming Barriers to Disruptive Innovation in Networking”. Janvier 2005.

[4.7] Committee on Research Horizons in Networking, Computer Science and Telecommunications Board, National Research Council. “Looking Over the Fence at Networks: A Neighbor's View of Networking Research”, National Academies Press, Washington, D.C.

[4.8] Voir http://www.planet-lab.org

[4.9] Voir http://www.one-lab.org

[5.1] G. Berry. « Real-Time Programming : special purpose or general purpose languages », in : Information processing 89, Elsevier Science Publisher B.V., editor : G. X. Ritter. 1989.

[5.2] F. Boussinot, R. de Simone. « The ESTEREL language », Technical report N° 1487, INRIA Sophia-Antipolis, July 1991.

[5.3] C. Castelluccia. « Automating header prediction », in Proceedings of the 1st Annual Workshop on Compiler Support For System Software, Tucson, Arizona, 1995.

[5.4] C. Castelluccia, W. Dabbous. « Modular communication subsystem implementation using a synchronous approach », in Usenix High-Speed Networking, Oakland, USA, August 1994.

[5.5] C. Castelluccia, W. Dabbous. « HIPPCO: an HIgh performance protocol code optimizer », Technical Report 2748, INRIA Sophia-Antipolis, December 1995.

[5.6] C. Castelluccia, P. Hoschka. « A compiler-based approach to protocol optimization », in Proceedings of High Performance Communication Subsystems Workshop, Mistic, Connecticut, August 1995.

[5.7] K. D. Cooper, M. W. Hall, L. Torczon. « An experiment with inline substitution », in Software-Practice and Experience, June 1991.

[5.8] J. W. Davidson, A. M. Holler. « Subprogram inlining: A study of its effects on program execution time », in IEEE Transactions on Software Engineering, February 1992.

[5.9] J. Favreau, M. Hobbs, B. Strausser, A. Weinstein. « User guide for the NIST prototype compiler for Estelle », Technical Report 83-3, Institute for Computer Science and Technology, National Institute of Standards and technology, September 1989.

[5.10] R. Gupta, C. Chi. « Improving instruction cacge behaviour by reducing cache pollution », in Proceedings of the Supercomputing Conference, New-York, USA, November 1990.

[5.11] J. L. Hennesy, D. D. Patterson. « Computer Architecture: A Quantitative Approach », Morgan Kaufmann, San Mateo, California, 1990.

[5.12] B. Hoffmann, W. Effelsberg. « Efficient implementation of Estelle specification », Technical report, Universitat Mannheim, March 1993.

[5.13] P. Hoschka. « Optimisation automatique dans un compilateur de talon de communication », Phd Thesis, INRIA Sophia-Antipolis, July 1995.

[5.14] N. Hutchinson, L. Peterson. « Design of the x-kernel », in Proceedings of the ACM Symposium on Communications Architectures and Protocols, pages 65--75, Stanford, California, August 1988.

[5.15] J.A. Manas, T. de Miguel. « From LOTOS to C », in FORTE, 1988.

[5.16] S. McFarling. « Program optimization for instruction caches », in Third International Conference on Architectural Support for Programming Languages and Operating Systems, April 1989.

[5.17] D. Mosberger, L. Peterson, S. O'Malley. « Protocol latency: MIPS and reality », Technical Report TR 95-02, Department of Computer Science, University of Arizona, January 1995.

[5.18] P. Oechslin. Implémentation optimisée de protocoles a haut débits. Technical report, EPFL Lausanne, 1995.

[5.19] K. Pettis, R. C. Hansen. « Profile guided code positioning », in Proceedings of the ACM SIGPLAN'90 Conference on Programming Language Design and Implementation, June 1990.

[5.20] V. Roy, R. de Simone. « Auto and autograph », in Proceedings of Workshop on Computer Aided Verification, June 1990.

[5.21] S. E. Speer, R. Kumar, C. Partridge. « Improving unix kernel performance using profile based optimization », in Winter Usenix, 1994.

[5.22] L. Svobodova. « Implementing OSI systems », in IEEE Journal on Selected Areas in Communications, September 1989.

[5.23] P. Van Eijk, C. Vissers, M. Diaz. « The Formal Description Technique LOTOS ». Elsevier Science, The Netherlands: North Holland, 1989.

[6.1] Real networks, http://www.realnetworks.com

[6.2] V. Jacobson, M. J. Karels. « Congestion avoidance and control », November 1988.

[6.3] D. M. Chiu, R. Jain. « Analysis of the increase and decrease algorithms for congestion avoidance in computer networks », Computer Networks and ISDN Systems, vol. 17, pp. 1--14, 1989.

[6.4] S. Floyd, K. Fall. « Promoting the use of end-to-end congestion control in the Internet », IEEE/ACM Transactions on Networking, August 1999.

[6.5] W. Tan and A. Zakhor. « Real-time internet video using error resilent scalable compression and tcp-friendly transport protocol », IEEE Transactions on Multimedia, vol. 1, no. 2, pp. 172--186, June 1999.

[6.6] S. Floyd, M. Handley, J. Padhye, J. Widmer. « Equation-based congestion control for unicast applications », in Proceedings of ACM SIGCOMM, May 2000.

[6.7] S. Floyd. « Connections with multiple congested gateways in packet-switched networks part 1: One-way traffic », Computer Communication Review, vol. 21, no. 5, pp. 30--47, October 1991.

[6.8] A. Demers, S. Keshav, S. Shenker. « Analysis and simulation of a fair queueing algorithm », in Proceeding of ACM SIGCOMM, 1989, pp. 3--12.

[6.9] J. C. R. Bennett, Hui Zhang. « WF2Q : Worst-case fair weighted fair queueing », in Proceedings of IEEE Infocom'96, San Francisco, CA, March 1996, pp. 120--128.

[6.10] J. C. R. Bennett, Hui Zhang. « Hierarchical packet fair queueing algorithms », in Proceedings of ACM SIGCOMM'96, October 1996, vol. 26, pp. 143--156.

[6.11] G. Varghese. « Efficient fair queuing using deficit round robin », in Proceedings of ACM SIGCOMM'95, 1995, vol. 25, pp. 231--243.

[6.12] D. Lin, R. Morris. « Dynamics of random early detection », in Proceedings of ACM SIGCOMM'97, Cannes, France, October 1997.

[6.13] S. Kent, R. Atkinson. « IP encapsulating security payload (esp) », RFC2406, November 1998.

[6.14] I. Stoica, S. Shenker, H. Zhang. « Core-stateless fair queueing: Achieving approximately fair bandwidth allocations in high speed networks », in Proceeding of ACM SIGCOMM'98, 1998.

[6.15] Z. Cao, Z. Wang, E. Zegura. « Rainbow fair queueing: Fair bandwidth sharing without per-flow state », in Proceedings of IEEE Infocom'2000, March 2000.

[6.16] R. Pan, B. Prabhakar, K. Psounis. « Choke, a stateless active queue management scheme for approximating fair bandwidth allocation », in Proceedings of IEEE Infocom'2000, March 2000.

[6.17] R. K. Jain, D-M. W. Chiu, W. R. Hawe. « A quantitative measure of fairness and discrimination for resource allocation in shared computer system », Tech. Rep. TR-301, DEC, Littleton, MA., September 1984.

[6.18] D. Bertsekas, R. Gallager. Data Network, chapter 6, pp. 524--529, Prentice-Hall, 1987.

[6.19] F. Kelly. « Charging and rate control for elastic traffic », European Transactions on Telecommunications, vol. 8, pp. 33--37, 1997.

[6.20] F.P. Kelly, A.K. Maulloo, D.K.H. Tan. « Rate control for communication networks : shadow prices, proportionnal fairness and stability », Journal of the Operationnal Research Society, vol. 49, pp. 237--252, 1998.

[6.21] D. Rubenstein, J. Kurose, D. Towsley. « The impact of multicast layering on network fairness », in Proceedings of ACM SIGCOMM'99, 1999.

[6.22] T. Jiang, E. W. Zegura, M. Ammar. « Inter-receiver fair multicast communication over the internet », in Proceedings of NOSSDAV'99, 1999.

[6.23] T. Jiang, M. H. Ammar, E. W. Zegura. « Inter-receiver fairness : A novel performance measure for multicast ABR sessions », in Proceedings of ACM Sigmetrics'98, Madison, Wisconsin, June 1998.

[6.24] A. Clerget, W. Dabbous. « Tuf: Tag-based unified fairness », in Infocom, 2001, pp. 498--507.

[6.25] A. Clerget. Hétérogénéité des réseaux IP multipoint. Thèse de doctorat de l'ENST. Décembre 2002.

[6.26] S. McCanne, S. Floyd. « Ucb/lbnl/vint network simulator (ns) version 2.1b5 », http://www-mash.cs.berkeley.edu/ns/, 1999.

[6.27] I. Stoica. « CSFQ simulation scripts for ns-2 », http://www.cs.cmu.edu//home/aclergetistoica/csfq/, 1998.

[6.28] Pawan Goyal, Harrick M. Vin, Haichen Cheng. "Start-Time Fair Queueing: A Scheduling Algorithm for Integrated Services Packet Switching Networks", IEEE/ACM Transactions on Networking, Vol. 5, No. 5, October 1997.

[6.29] Sally Floyd, Van Jacobson, "Random Early Detection gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, Vol. 1, No.4, August 1993, p. 397-413.

[6.30] K.K. Ramakrishnan, S. Floyd, D. Black. "The Addition of Explicit Congestion Notification (ECN) to IP". RFC 3168, Proposed Standard, September 2001.

[7.1] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems", In Proceedings of IFIP/ACM International Conference on Distributed Systems Platforms, Heidelberg, Germany, November, 2001.

[7.2] J. Kubiatowicz et al. "OceanStore: An Architecture for Global-Scale Persistent Storage", in Proceedings of the Ninth international Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000), November 2000.

[7.3] Y. Chu; S.G. Rao, S. Seshan, H. Zhang. "A case for end system multicast", IEEE Journal on Selected Areas in Communications, Volume 20, Issue 8, Oct 2002, pp. 1456 – 1471.

[7.4] S. A. Baset, H. G. Schulzrinne, "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol", in: INFOCOM 2006.

[7.5] T. E. Ng, and H. Zhang, Predicting internet network distance with coordinates-based approaches, in Proceedings of the IEEE INFOCOM, New York, June 2002.

[7.6] M. Pias, J. Crowcroft, S. Wilbur, S. Bhatti, and T. Harris, Lighthouses for Scalable Distributed Location, in Proceedings of International Workshop on Peer-to-Peer Systems (IPTPS), Berkeley, February 2003.

[7.7] M. Costa, M. Castro, A. Rowstron, and P. Key, Practical Internet coordinates for distance estimation, in Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS), Tokyo, March 2004.

[7.8] T. E. Ng and H. Zhang, A Network Positioning System for the Internet, in Proceedings of the USENIX annual technical conference, Boston, June 2004.

[7.9] F. Dabek, R. Cox, F. Kaashoek and R. Morris, "Vivaldi: A decentralized network coordinate system". In Proceedings of the ACM SIGCOMM, Portland, Oregon, August 2004.

[7.10] Y. Shavitt and T. Tankel, Big-bang simulation for embedding network distances in euclidean space, in Proceedings of the IEEE INFOCOM, San Francisco, April 2003.

[7.11] J. Ledlie, P. Gardner, and M. Seltzer, "Network Coordinates in the Wild", In Proceedings of NSDI, Cambridge, April 2007.

[7.12] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications", In Proceedings of SIGCOMM, San Diego, CA, August 2001.

[7.13] Azureus BitTorrent Client. http://azureus.sourceforce.net

[7.14] Cheikh ahmadou bamba Gueye, Artur Ziviani, Mark Crovella and Serge Fdida, « Constraint-Based Geolocation of Internet Hosts », IEEE/ACM Transaction on Networking, December, 2006.

[7.15] Bernard Wong Aleksandrs Slivkins Emin Gün Sirer, "Meridian: A Lightweight Network Location Service without Virtual Coordinates", in Sigcomm 2005.

[7.16] H. Zheng, E. K. Lua, M. Pias, and T. Griffin, Internet Routing Policies and Round-Trip Times, in Proceedings of the Passive Active Measurement (PAM), Boston, March 2005.

[7.17] E. K. Lua, T. griffin, M. Pias, H. Zheng, and J. Crowcroft, *On the accuracy of Embeddings for Internet Coordinate Systems*, in Proceedings of Internet Measurement Conference (IMC), Berkeley, October 2005.

[7.18] K. P. Gummadi, S. Saroiu, and S. D. Gribble, King: Estimating Latency between Arbitrary Internet End Hosts, In Proceedings of SIGCOMM Internet Mesasurement Workshop (IMW), Pittsburgh November 2002.

List of Publications by September 2007

The list contains 1 submitted journal paper, 9 journal papers, 1 submitted conference paper, 44 conference papers, 5 national conferences, 2 books (editor), 2 theses, 3 books chapters, 1 RFC, 2 Internet-Drafts, 14 INRIA reports and 11 other documents.

Journal Papers

[R.1] M.A. Kaafar, T. Turletti, W. Dabbous. *Supporting Large Scale Overlay-Multicast Applications*, submitted to Computer Networks, July 2007.

[R.2] F. Filali, W. Dabbous. *Fair Bandwidth Sharing Between Unicast and Multicast Flows in Best-Effort Networks*, in Computer Communications Special Issue on Quality of Future Internet Service, Vol. 27, Issue 4, pp. 330-344, March 2004.

[R.3] F. Filali, G. Aniba, W. Dabbous. *Efficient Support of IP Multicast in the Next-Generation of GEO Satellite*, in IEEE JSAC Special Issue on Broadband IP Networks via Satellites, Vol. 22, Issue 2, pp. 413-425, February 2004.

[R.4] M. A. Ruiz Sanchez, E. Biersack, W. Dabbous. *Survey and Taxonomy of IP Address Lookup Algorithms*, IEEE Network Magazine, Vol. 15, Issue 2, pp. 8-23, March/April 2001. (Printed version and errata).

[R.5] L. Wood, A. Clerget, I. Andrikopoulos, G. Pavlou, W. Dabbous. *IP routing issues in Satellite Constellation Networks*, International Journal of Satellite Communications, Vol. 19, Issue 1, pp. 69-92, February 2001.

[R.6] C. Barakat, E. Altman, W. Dabbous. *On TCP Performance in a Heterogeneous Network : A Survey*, IEEE Communication Magazine, Vol. 38, Issue 1, pp. 40-46, January 2000.

[R.7] W. Dabbous. *High performance protocol architecture*, in Computer Networks and ISDN Systems, Vol. 29, Issue 7, pp.735-744, August 1997.

[R.8] C. Castelluccia, W. Dabbous, S. O'Malley. *Generating Efficient Protocol Implementations from Abstract Specifications*, in Proceedings of ACM SIGCOMM'96, Stanford University, California, CCR, Vol. 26, Issue 4, pp. 60-71, August 1996. Also in IEEE/ACM Transactions on Networking, Vol. 5, Issue 4, pp. 735-744, August1997.

[R.9] C. Diot, W. Dabbous, J. Crowcroft. *Multipoint Communication: A Survey of Protocols, Functions and Mechanisms*, IEEE Journal on Selected Area in Communication, Vol. 15, Issue 3, pp. 277-290, April 1997.

[R.10] C. Castelluccia, W. Dabbous. *Automatic Protocol Code Optimizations*, in Journal of Electrical and Electronics Engineering Australia, Special Issue on Networking, Vol. 16, Issue 1, pp. 19-28, March 1996. Also in Proceeding of the second HIPPARCH Workshop, Sydney, Australia, December 1995.

International Conference and Workshop papers

[R.11] M. A. Kaafar, L. Mathy, C. Barakat, K. Salamatian, T. Turletti, W. Dabbous. *Certified Internet Coordinates*, submitted to NSDI 2008, San Francisco, April 2008.

[R.12] M. A. Kaafar, L. Mathy, C. Barakat, K. Salamatian, T. Turletti, W. Dabbous. *Securing Internet Coordinate Embedding Systems*, in Proceedings of ACM SIGCOMM'07, Kyoto, Japan, August 2007.

[R.13] D. Dujovne, T. Turletti, W. Dabbous. *Experimental Methodology for Real Overlays*, ROADS'07 Workshop, Warsaw, Poland, July 2007.

[R.14] M. A. Kaafar, L. Mathy, T. Turletti, and W. Dabbous. *Virtual Networks under Attack: Disrupting Internet Coordinate Systems*, in Proceedings of CoNext 2006, Lisboa, December, 2006.

[R.15] M. A. Kaafar, L. Mathy, T. Turletti, W. Dabbous. *Real attacks on virtual networks: Vivaldi out of tune*, in Proceedings of ACM SIGCOMM Workshop on Large Scale Attack Defense, Pisa, Italy, September 11-15 2006.

[R.16] M. A. Kaafar, T. Turletti, W. Dabbous. *A Locating-First Approach for Scalable Overlay Multicast*, in Proceedings of IEEE International Workshop on Quality of Service (IWQoS), New Haven, CT, USA, June 19-21 2006.

[R.17] M. Malli, C. Barakat, W. Dabbous. *An Enhanced Scalable Proximity Model*, in Proceedings of IEEE International Workshop on Quality of Service (IWQoS) (extended abstract), New Haven, USA, June 2006.

[R.18] M. A. Kaafar, T. Turletti, and W. Dabbous. *A Locating-First Approach for Scalable Overlay Multicast*, in Proceedings of IEEE Infocom Student Workshop (extended abstract), Barcelona, Spain, April 2006.

[R.19] M. Malli, C. Barakat, W. Dabbous. *Landmark-based End-to-End Bandwidth Inference*, in Proceedings of IEEE Infocom Student Workshop (extended abstract), Barcelona, Spain, April 2006.

[R.20] M. Malli, C. Barakat, W. Dabbous. *Application-level versus Network-level Proximity*, in Proceedings of the Asian Internet Engineering Conference (AINTEC), Bangkok, December 2005.

[R.21] H. Kim, K. G. Shin, W. Dabbous. *Improving Cross-domain Authentication over Wireless Local Area Networks*, in Proceedings of SecureComm'05, Athens, Greece, IEEE, September 2005.

[R.22] H. Kim, W. Dabbous, H. Afifi. *A Bypassing Security Model for Anonymous Bluetooth Peers*, in Proceedings of Wirelesscom 2005, Hawaii, U.S.A., IEEE, June 2005, vol. 1, pp. 310-315.

[R.23] M. Malli, C. Barakat, W. Dabbous. *An Efficient Approach for Content Delivery in Overlay Networks*, in Proceedings of IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, January 2005.

[R.24] F. Louati, C. Barakat, W. Dabbous. *Adaptive Class-based Queuing for handling Two-Way traffic in Asymmetric Networks*, in Proceedings of the conference on High Speed Networks and Multimedia Communications (HSNMC), Toulouse, July 2004.

[R.25] H. Asaeda, W. Dabbous. *Multicast Routers Cooperating with Channel Announcement System*, in Proceedings of SAINT 2004 Workshops, Tokyo, Japan, January 2004.

[R.26] M. A. Ruiz-Sánchez, W. Dabbous. *Controlling bursts in best-effort routers for flow isolation*, in Proceedings of ISCC'2003, Turkey, July 2003.

[R.27] R. Serban, C. Barakat, W. Dabbous. *A CBQ-Based Dynamic Resource Allocation Mechanism for Diffserv Routers*, in Proceedings of Setit 2003, Tunisia, March 2003.

[R.28] R. Serban, C. Barakat, W. Dabbous. *Dynamic Resource Allocation in Core Routers of a Diffserv Network*, in Proceedings of ASIAN'02, Hanoi, Vietnam, December 2002.

[R.29] F. Filali, H. Asaeda, W. Dabbous. *Counting the Number of Group Members in Multicast Communication*, in Proceedings of NGC'2002, IEEE, Boston, USA, October 2002.

[R.30] F. Filali, W. Dabbous. *A Simple and Scalable Fair Bandwidth Sharing Mechanism for Multicast Flows*, in Proceedings of ICNP'2002, IEEE, Paris, France, October 2002.

[R.31] F. Filali, W. Dabbous. *SBQ: A Simple Scheduler for Fair Bandwidth Sharing Between Unicast and Multicast Flows*, in Proceedings of QofIS'2002, IEEE, Zurich, Switzerland, October 2002.

[R.32] F. Filali, W. Dabbous. *A New Bandwidth Sharing Scheme for Non-Responsive Multicast Flows*, in Proceedings of ICC'2002, IEEE, New York, USA, April 2002.

[R.33] F. Filali, W. Dabbous. *A QoS-Aware Switching Mechanism Between the Two Modes of PIM-SM Multicast Routing Protocol*, in Proceedings of ITC Specialist Seminar on Internet Traffic Engineering and Traffic Management (IP2002), IEEE, Wuerzburg, Germany, April 2002.

[R.34] F. Filali, W. Dabbous. *MFQ: A Multicast Fairness-Independent and Fine-Grained AQM Mechanism for Multicast Flows*, in Poster Session, NGC'2001, London, UK, November 2001.

[R.35] F. Filali, W. Dabbous, F. Kamoun. *On the Planning of Multiservices GEO Satellite-Terrestrial Hybrid Networks*, in Proceedings of IEEE Softcom'2001, IEEE, Split, Dubrovnik (Croatia) Ancona, Bari (Italy), October 2001.

[R.36] F. Filali, W. Dabbous. *Issues on the IP Multicast Service Behaviour over the Next-Generation of Satellite-Terrestrial Hybrid Networks*, in Proceedings of ISCC'2001, IEEE, Hammamet, Tunisia, July 2001.

[R.37] F. Filali, W. Dabbous, F. Kamoun. *Efficient Planning of Satellite-Terrestrial Hybrid Networks for Multicast Applications*, in Proceedings of ICC'2001, IEEE, Helsinki, Finland, June 2001.

[R.38] A. Clerget, W. Dabbous. *TUF: Tag-Based Unified Fairness*, in Proceedings of IEEE Infocom 2001, Anchorage, Alaska, April 2001.

[R.39] R. Serban, S. Gara, W. Dabbous. *Internet QoS Signaling Protocols*, in Proceedings of IEEE Communications 2000, Bucharest, Romania, December 2000.

[R.40] C. Barakat, N. Chaher, W. Dabbous, E. Altman. *Improving TCP/IP over Geostationary Satellite Links*, in Proceedings of IEEE Globecom (General Conference), Rio, Brazil, December 1999.

[R.41] A. Clerget, W. Dabbous. *Organizing Data Transmission for Reliable Multicast over Satellite Links*, in Proceedings of the 5th Conference on Computer Communications, Africom CCDC'98, Internet and Global Networking, pp. 71-80, Le Palace Hotel, La Marsa, Tunis, October 1998.

[R.42] P. Narvaez, A. Clerget, W. Dabbous. *Internet Routing over LEO Satellite Constellations*, in Proceedings of ACM/IEEE Mobicom'98 Workshop on Satellite-based Information Services (WOSBIS'98), Omni Hotel, Dallas, October 1998.

[R.43] A. Clerget and W. Dabbous. *Organizing data transmission for reliable multicast over satellite links*, Satellite Networks Workshop, NASA Lewis Research Center, Cleveland, Ohio, June 1998.

[R.44] W. Dabbous, E. Duros, T. Ernst. *Dynamic Routing in Networks with Unidirectional Links*, in Proceedings of WOSBIS '97, Budapest, Hungary, September 1997.

[R.45] F. Lyonnet, W. Dabbous, P. Perrot. *Architecture Considerations for Videoconferencing in the Internet with Wireless Links*, in Proceedings of Telecom Interactive '97, Geneva, Switzerland, September 1997.

[R.46] E. Duros, W. Dabbous. *Supporting Unidirectional Links in the Internet*, in Proceedings of the First International Workshop on Satellite-based Information Services (WOSBIS), Rye, New York, November 1996.

[R.47] W. Dabbous, C. Diot. *High Performance Protocol Architecture*, in Proceedings of PCN'95, Istanbul, Turkey, October 1995.

[R.48] C. Castelluccia, W. Dabbous. *Modular Communication Subsystem Implementation using a Synchronous Approach*, in Proccedings of the Usenix Symposium on High Speed Networks, Oakland, USA, August1994.

[R.49] W. Dabbous. *High performance presentation and transport mechanisms for integrated communication subsystems*, in Proceedings of the 4th International IFIP Workshop on Protocols for High Speed Networks, Vancouver, Canada, August1994.

[R.50] W. Dabbous. *Analysis of a delay based congestion avoidance algorithm*, in Proceedings of the 4th IFIP Conference on High Performance Networking, Liège, Belgium, December 1992, also a book chapter in The OSI95 Transport Service with Multimedia Support, A. Danthine (Ed.), Springer Verlag, pp.280-296, 1994.

[R.51] C. Huitema, W. Dabbous. *Synchronization Schemes for OSI Multimedia Applications*, in Proceedings of Computer Networks'91, Wraclaw, Poland, June 1991.

[R.52] C. Huitema, W. Dabbous. *Minimal Complexity for the Simplest Protocol*, in Proceedings of INFOCOM'91, Bal Harbour, USA, April 1991.

[R.53] C. Huitema, W. Dabbous. *Routeing Protocols Development in the OSI Architecture*, in Proceedings of ISCIS V, Turkey, October 1990.

[R.54] W. Dabbous. *On High Speed Transport Protocols*, in Proceedings of the First Workshop on Protocols for High Speed Networks, PfHSN '89, Zurich, Switzerland, May 1989.

[R.55] C. Huitema, W. Dabbous. *Real Time Communications in Local Area Networks*, in Proceedings of the sixth European Fibre Optic Communications and Local Area Network (EFOCLAN'88), pp. 383-388, Amsterdam, June 1988.

National Conference Papers

[R.56] M. A. Kaafar, T. Turletti, W. Dabbous. *LCC: Un réseau de recouvrement multipoint passant à l'échelle*, in Proceedings of CFIP, Tozeur, Tunisia, October 30 - November 3, 2006.

[R.57] M. A. Ruiz Sanchez, W. Dabbous. *Un mécanisme optimisé de recherche de route IP*, in Proceedings of CFIP'2000 (Colloque Francophone sur l'Ingénierie des Protocoles), pp. 217-232, Toulouse, France, Cctober 2000.

[R.58] F. Lyonnet, W. Dabbous, *Un Ordonnanceur Applicatif pour les Systèmes de Téléconférence*, in Proceedings of CFIP'99, Nancy, France, April1999.

[R.59] C. Castelluccia, W. Dabbous. *Génération Automatique d'Implémentations Optimisées de Protocoles*, in Proceedings of CFIP'96, Rabat, Morocco, October 1996.

[R.60] A. Doghri, C. Huitema, W. Dabbous. *Implantation d'Applications Réseaux à Haut Débit*, in Proceedings of DNAC'88, pp.61-67, Paris, October 1988.

RFCs and Internet Drafts

[R.61] E. Duros, W. Dabbous. *Handling of Unidirectional Links with DVMRP*, Internet draft, UDLR working group, November 1997.

[R.62] E. Duros, W. Dabbous. *Supporting Unidirectional Paths in the Internet*, Internet draft, UDLR working group, November 1997.

[R.63] E. Duros, W. Dabbous, H. Izumiyama, N. Fujii and Y. Zhang. *A Link-Layer Tunneling Mechanism for Unidirectional Links*, RFC 3077, March 2001.

Books and monographs

[R.64] W. Dabbous. *Quelle architecture pour l'Internet du futur ?* HDR Thesis, Nice-Sophia Antipolis University, 2008.

[R.65] W. Dabbous (editor). *Systèmes Multimédias Communicants*, Hermès Science Publications, June 2001. ISBN 2-7462-0251-4.

[R.66] W. Dabbous, C. Diot (editors). *Protocols for High Speed Networks V*, Chapman and Hall, 1997, ISBN 0-412-75850-4.

[R.67] W. Dabbous. *Etude des protocoles de contrôle de transmission à haut débit pour les applications multimédias*, PhD. Thesis, Paris-Sud University, 1991.

Book chapters

[R.68] W. Dabbous, T. Turletti. *Scalable Virtual Environments*, book chapter in Multimedia Mutlicast on the Internet, A. Benslimane (editor), ISTE, 2007.

[R.69] W. Dabbous, T. Turletti. *Le Multipoint pour les Environnements Virtuels à Grande Échelle*, book chapter in Multicast Multimédia sur Internet, A. Benslimane (editor), HERMES Science Publications, March 2005.

[R.70] W. Dabbous. *Architecture de Protocoles Haute Performance*, book chapter in Réseaux de Communication et Conception de Protocoles, A. Serhrouchni (editor), Eyrolles, 1995.

INRIA Reports

[R.71] M. A. Kaafar, T. Turletti, W. Dabbous. *Locate, Cluster and Conquer: A Scalable Topology-Aware Overlay Multicast*, INRIA Technical Report, RT-0314, November 2005.

[R.72] T. Parmentelat, L. Barza, T. Turletti, W. Dabbous. *A Scalable SSM-based Multicast Communication Layer for Multimedia Networked Virtual Environments*, INRIA Research Report, RR-5389, November 2004.

[R.73] F. Louati, C. Barakat, W. Dabbous. *Handling Two-Way traffic in Asymmetric Networks*, INRIA Research Report, RR-4950, October 2003.

[R.74] F. Filali, W. Dabbous. *Optimization of GEO Satellite Links Deployment in the Internet*, INRIA Research Report, RR-3925, April 2000.

[R.75] A. Clerget, W. Dabbous. *Tag-based Fair Bandwidth Sharing for Responsive and Unresponsive flows*, INRIA Research Report, RR-3846, December 1999.

[R.76] C. Barakat, E. Altman, W. Dabbous. *On TCP Performance in a Heterogeneous Network : A Survey*, INRIA Research Report, RR-3737, July 1999.

[R.77] C. Barakat, N. Chaher, W. Dabbous, E. Altman. *Improving TCP/IP over Geostationary Satellite Links*, INRIA Research Report, RR-3573, December 1998.

[R.78] T. Ernst, W. Dabbous. *A Circuit-based Approach for Routing in Unidirectional Links Networks*, INRIA Research Report, RR-3292, November 1997.

[R.79] C. Castellucia, W. Dabbous. *HIPPCO: A HIgh Performance Protocol Code Optimizer*, INRIA Research Report, RR-2748, December 1995.

[R.80] C. Castellucia, I. Chrisment, W. Dabbous, C. Diot, C. Huitema, E. Siegel, R. de Simone. *Tailored Protocol Development Using ESTEREL*, INRIA Research Report, RR-2374, October 1994.

[R.81] W. Dabbous, C. Huitema. *XTP Implementation under Unix*, INRIA Research Report, RR-2102, November 1993.

[R.82] W. Dabbous. *High Performance Implementation of Communication Subsystems*. INRIA Research Report, RR-2101, November 1993.

[R.83] W. Dabbous, B. Kiss. *A reliable multicast protocol for a white board application*, INRIA Research Report, RR-2100, November 1993.

[R.84] W. Dabbous et al. *Applicability of the Session and the Presentation Layers for the Support of High Speed Applications*, INRIA Technical report, RT-144, October 1992.

Miscellaneous

[R.85] T. Parmentelat, W. Dabbous. *OneLab contributions to the PlanetLab software*, Technical note, INRIA, July 2007.

[R.86] F. Filali, W. Dabbous, *Efficient PIM-SM Configuration and Adaptation for GEO Bent-Pipe Satellite Systems*, Report for the Dipcast project, November 2002.

[R.87] P. Cipière, W. Dabbous, E. Duros, F. Filali, *A Dynamic Routing Mechanism for UniDirectional Communication Links*, Unpublished note, January 2001.

[R.88] J. Bolot, W. Dabbous. *L'Internet : Historique et évolution. Quel avenir prévisible ?* Revue Administration, No 175, pp. 44-51, April-June 1997.

[R.89] W. Dabbous, J. Bolot. *Applications Multimédia Sur l'Internet*, presented at the ERCIM Workshop on The Information Society in the Euro-Mediterranean context: Research and Information Technologies, April 1996, Sophia Antipolis.

[R.90] C. Diot, W. Dabbous. *Multimédia : Repenser le fonctionnement des protocoles*, Revue Normatique, AFNOR, No 54, February 1994.

[R.91] W. Dabbous and J. Bolot. *Study of Congestion Avoidance Mechanisms*, Research Report, OSI95 project, INRIA, June 1992.

[R.92] C. Huitema, W. Dabbous. *An NSAP approach to build transparent OSI transport bridges*, in Proceedings of Workshop for CL/CO internetworking, Washington, USA, July 1990.

[R.93] C. Huitema, W. Dabbous. *Extension of OSI TP4 to Support Transport Bridging*, in Proceedings of Workshop for CL/CO internetworking, Washington, USA, July 1990.

[R.94] C. Huitema, W. Dabbous. *End to end transmission control on ATM networks*, in Proceedings of the NATO advanced research workshop on Architecture and performance issues of high-capacity local and metropolitan area networks, INRIA Sophia Antipolis, June 1990.

[R.95] W. Dabbous, C. Huitema. *PROMETHEUS: Vehicle to Vehicle Communications*, Research Report, INRIA-Renault collaboration, August 1988.

Annexe 1

CV détaillé

Walid Dabbous

INRIA Sophia Antipolis
2004, route des Lucioles
BP 93, 06902, Sophia Antipolis, France
Tel: +33(0)4 92 38 77 18
Fax: +33(0)4 92 38 79 78
E-mail: Walid.Dabbous@sophia.inria.fr
Web page: http://planete.inria.fr/dabbous

Education

Habilitation à diriger des recherches, Nice-Sophia Antipolis University, 2008.
Thesis title: *Quelle architecture pour de l'Internet du futur* ?

Ph.D., Computer Science, Paris XI University, March 1991.
Thesis title: *Etude des protocoles de contrôle de transmission à haut débit pour les applications multimédias*.

DEA, Computer Science, Paris XI University, September 1987.
Thesis title: *Communications temps réel dans les réseaux locaux*.

Engineering Diploma, Electrical and Electronics Engineering, School of Engineering, Lebanese University, 1986.
Thesis title: *Onduleur 1 KVA à Thyristors*.

Research Interests

Future Internet Architecture and Protocols
Networking Experimental Platforms and Simulators
Experimental Methodology for Networking Protocols
Securing Internet coordinate systems
Large scale virtual environments
Large scale reliable multicast protocols
Group communications
Internet Satellite Networking
Quality of Service support
Buffer management mechanisms in routers
Flexible protocol architecture
High performance communication protocols
Unicast and Multicast Congestion control
Audio and video conferencing over the Internet

Professional Experience

1999 – Present	Senior Researcher at INRIA Sophia Antipolis Head of the Planète project-team on Protocols and Applications for the Internet
1996 – 1999	Head of the Rodéo project-team on High Speed and Open Networks
1991 – 1999	Researcher at INRIA Sophia Antipolis
2000 – 2001	Consultant for CS Telecom

Teaching Experience

2001 – Present	Professor at the Ecole Polytechnique, Palaiseau. Course on Networking in Majeure 2 for computer science (32h)
2007 – Present	Responsible of the Networking and Distributed Systems Master programme at Nice-Sophia Antipolis University
1995 – Present	Master Course on Networking at Nice-Sophia Antipolis University (24h)
1996 – 2002	Master Course on Group Communications at Nice-Sophia Antipolis University (15h)
1989 – 1996	Course on "Algorithms of the Lower Layers" at ESSI (24h)
1991 – 1994	Course on Networking at CERICS, then CERAM RID Master (24h)
1991 – 1994	Several Courses on Networks Interconnection in training centers for systems and networks engineers (12h).
1991 – 2000	Several courses on High Speed Networks at ISIA, Ecole des Mines, ENST, Master programme on Information Transmission and Processing and Master programme on Computer Science in Nice-Sophia Antipolis University.

Participation to European Projects

OneLab	FP6 STREP (2006-2008) that extends the current PlanetLab infrastructure to a federated testsbed models with more heterogeneity and enhancing the ability of applications that are running on PlanetLab to perceive the underlying network environment. Other partners: LIP6, Intel Corporation UK, UCarlos III Madrid, UCL, CINI, FT, UPisa, Alcatel-Lucent Italia, Telekomunikacja Polska. One of the major achievements of the project is a federated tested architecture taken as a reference by the European Commission in the FIRE initiative.
E-NEXT	FP6 Network of Excellence (2004-2005) that focused on Internet protocols and services. The general objective of E-NEXT is to reinforce European scientific and technological excellence in the networking area through a progressive and lasting integration of research capacities existing in the European Research Area (ERA). In this project we actively prepared the OneLab project on experimental testbeds.
Muse	IST Integrated Project (2004-2005) on the research and development of a future, low cost, multi-service access network. The project is leaded by Alcatel. Our contribution in the project focused on multicast support in access networks and on hybrid overlay/IP multicast for large scale virtual environment applications.
COIAS	ACTS project (1998-2000) on the study of the convergence of Internet, ATM and Satellite DVB networks. The project focused on reliable multicast, mobility and security in hybrid environment. Other partners: Dassault Electronique, Eurocontrol, Eutelsat, Secunet, British Telecom, UCL and Cisco. One of the major achievements of the project was the development and

deployment of Internet over satellite tunnelling software based on the udlr IETF proposed standard.

MECCANO — Follow up of MERCI (1998-2000) focusing on enhancing the quality (using FEC techniques) of the collaboration tools and their support over new transmission media such as satellite or wireless links. Other partners: UCL, UiO, Ubremen, Teles, New Learning, HP and Shell.

HIPPARCH II — Basic LTR project (1996-1998) on the experimental evaluation of new protocol architectures based on ALF and ILP. Other partners: Dassault Electronique, Uppsala University, SICS, University College London. One of the major achievements was the HIPPCO code generator and optimizer of the Hipparch protocol compiler for automatic generation of efficient network protocols.

MERCI — Telematics project (1995-1997) on the technological components required for the deployment of collaborative work tools. Other partners: UCL, GMD, KTH, UiO, Teles, HP and Shell. Major achievements are the RendezVous modular video conferencing tool, FreePhone (audio conferencing) and Mscrawl (shared white board).

HIPPARCH — EU-Australian collaboration project (1994-1995) on High Performance and Flexible Protocol Architecture. Other partners: UCL, SICS, University of Sydney. A major achievement was the first comprehensive study and evaluation of the ALF and ILP paradigms.

MICE I and II — Esprit Projects (1992-1995) on the definition of a "standard" Internet Videoconferencing Service platform for research networks. Other partners: UCL, SICS, GMD, University of Oslo, Nottingham, Brussels and Stuttgart. MICE I focused on the development of a common platform and MICE II on collaborative tools on high speed networks. One of the major achievements of the project was the IVS videoconferencing tool.

OSI95 — Esprit project (1990-1992) on the re-design of the OSI Layered Protocols Architecture for efficient operation in high speed networks. Other partners: BULL, Olivetti, Alcatel, INTRACOM, Liège University, Lancaster University and Madrid Polytechnic Institute. The project early identified the inadequacies of the OSI model for efficient large scale inter-networking.

Participation to National Projects

VTHD — (2000-2001) RNRT project on building a national experimental platform for new generation networking protocols and services. Other partners: FT R&D, ENST, INT, Eurecom and ENSTB. We contributed a large scale virtual environment application (V-Eye) used to test the ability of the underlying network mechanisms to support high performance applications with group communication and strict timing requirements. Its follow-up VTHD++ (2002-2004) focused on extending the scale of the platform and to provide the capability for the user to tune and monitor the underlying network test-bed.

Arcade	(2001-2003) RNRT project on the dynamic control of IP networks. Other partners: LIP6, INRIA, France Télécom, Thomson-CSF and QoSMIC. Our contribution in the project focused on the evaluation of the complexity of dynamic resource allocation mechanisms.
Intradiff	(2000-2002) RNRT project on the study of static resource management in enterprise networks with diffserv support. Other partners: CS Télécom, Thomson CSF Detexis (6wind), Cégétel. Our contribution based on a simulation study of diffserv mechanisms in large networks showed that functional complexity and parameter tuning are major problems that prevent large deployment of quality of service in the Internet.
Dipcast	(2000-2003) RNRT project on the support of IP multicast over multi-beam satellites based on next generation DVB processor with OBP. Other partners: Alcatel Space Industries, CNES, LAAS, CRIL Ingénierie, ENSICA, ENSEEIHT, ISIS and Polycom. Our major achievement was an encapsulation scheme for efficient on-board switching of IP multicast packets and related signalling in relation with PIM-SM.
@IRS	(1998-2001) RNRT project on the design and experimentation of new generation Internet protocols supporting mobility and quality of service in heterogeneous IP, ATM and Satellite environment). Other partners: Aerospatiale-Matra Lanceurs, LAAS, FT/CNET, INPG-LSR, RENATER, 6WIND, LIP6 and LSIIT. The follow-up project @IRS++ (2001-2003) focused on fixed mobile convergence, group communication and dynamic network services.
Constellations	(1998-2002) RNRT project on networking using satellite constellations, focusing on routing and end to end quality of service support. Others partners: CNES, ENST, INT, LM2S, LIRMM, LIP6, Supelec, Alcatel, France Télécom, Matra Marconi Space. We contributed on both data organization for reliable multicast and on IP routing support over GEO satellite constellations.
DIS-ATM	(1996-1998) DGA/MENERT collaboration project (also called Placebo) with Dassault Electronique, LIP6 and LAAS. The goal was to experiment quality of service mechanisms (intserv/diffserv) to provide enhanced services for Distributed Interactive Simulation applications (using FreePhone, RendezVous and MiMaze) over hybrid IP/ATM technologies. The experiments were done on Mirihade (first experimental platform for networking research in France) then on its follow-up Safir (which prepared the launching of Renater-2 in 1999).

Industrial collaboration

UDcast	(2007-2010) CIFRE contract with UDcast on protocols, applications and infrastructure for IP broadcast to mobile devices.
STM	(2003-2006) CIFRE contract with STM on Secured Large scale virtual environments.
Alcatel	(2003) Collaboration with Alcatel on TCP performance in a hybrid satellite/terrestrial network, in the context of an ESA study named Transat.
Hitachi	(2001-2002) Collaboration with Hitachi Sophia Labs on the scalability of the udlr approach to a large number of receivers for both routing and reliable transport levels. Another collaboration (2003-2004) on efficient authentication architecture focused on accelerating inter-domain roaming, keeping seamless mobile secured Internet services.
CS-Telecom	(2000-2001) Collaboration with CS Telecom on the support of quality of service in the Internet.
Eutelsat	(1996-1999) Collaboration with Eutelsat on support IP over unidirectional satellite links. Our major achievement is an encapsulation mechanism to support IP protocols transparently on unidirectional broadcast links. Standardization work with the IETF where we were co-chairing the udlr working group produced a proposed standard RFC 3077. Other participants to this work founded the UDcast company in 2000.
LEP	(1996-1997) Collaboration with Laboratoires d'Electronique Philips on supporting videoconferencing over hybrid wired and wireless networks.
NEC	(1997-1998) Collaboration with NEC on videoconferencing applications over high speed wireless LANs. Application level FEC integrated in the RendezVous application was tested on a 25 Mbps wireless LAN card provided by NEC.
SGS-Thomson	(1996-1997) SToRIA is an "Autoroutes de l'information" project with SGS-Thomson on testing collaboration tools on IP networks in an industrial context. Our contribution consisted in adapting the white board tools Mscrawl to support shared visualization application.
Bull	(1995) collaboration project in the context of INRIA-Bull GIE Dyade on IP Video Conferencing applications.
Renault	(1988) Contract with TREGIE Renault on Vehicle to vehicle communications in the context of the European Prometheus project.

Demonstration activities

Several public demonstration of Internet Videoconferencing tools in the context of: la science en fête, French prime minister visit to INRIA in 1995, INTEROP 1993 and during the G7 summit in Brussels in February 1993.

Early Participation to the set-up of the French MBone Multicast overlay (tunneling) Network in 1992-1993.

National Collaboration activities

GDR-PRS	(1992-1996) Contributed to the GDR/PRS and Networking Task force launched by CNRS mainly on High Performance Protocols and Reliable Multicast Communications with partners from LAAS, MASI and IRISA. The thematic school RHDM (Réseaux Haut Débit Multimédia) was initiated in the context of the GDR/PRS.

International Collaboration activities

NSF-INRIA	NSF project between INRIA and University of Wisconsin on Standard protocols and transport service gateways (e.g. rfc983/TCP-TP0/X.25). Major achievements were (1) the establishment of the first Internet connection between France and the NSFnet in July 1988 and (2) the development of a Telnet-triple X application level gateway providing support to NASA astronomers for remote access to astronomical data base (SIMBAD) using the NSF-INRIA link then gateway through the Transpac X.25 network to the database.
Arcadia	Participation to the Arcadia cost activity on Building the future internet: From fundamentals to experiments (2006-2007).
STIC Asia	(2004-2005) Collaboration with AIT and the WIDE project in the context of a French government funded STIC Asia “Internet Nouvelle Génération” project. The first year workshop was organized at INRIA Sophia Antipolis.
STIC Tunisia	(2007-2008) Collaboration with ENSI (Tunis) to a STIC Tunisia project on Security and Monitoring of Hybrid Wireless Mesh Networks.
STIC AmSud	(2008-2009) Collaboration with U Diego Portales (Chile) and U. de Cordoba (Argentina) in the context of STIC-AmSud project called ROSEATE on Realistic mOdels, Simulations and ExperimentATion of wirelEss protocols.

Standardization activities

UDLR	Starting and co-chairing of the UDLR IETF working group (from December 1996 to August 2001). Other major participants: Hughes Research Labs, Jsat, Sony, Hitachi, Cisco and later UDcast. The UDLR working group issued the RFC 3077 in March 2001.

Participation to conference program committees

Served in the following conferences as PC member: INFOCOM'06, CoNext'05, Med-hoc-net'2003, NGC (99-2003), SAINT'2001, Networking'2000, ISCC'2000, AFRICOM CCDC'98, ICCC'97, WOSBIS (97-99), CFIP (97-05), IDMS'96, Hipparch'95, HPDC (94-96). Also served as PC co-chair of PfHSN'96 and tutorial chair for Sigcomm'97. Member of the editorial board of the IEEE Communications Surveys & Tutorials electronic journal, and of a special issue of the TSI (Techniques et Sciences Informatiques) journal on the topic Networks and protocols (in 2004).

Conference Organization

Served in the organizing committee of Hipparch'95, PfHSN'96, ISCC'2001. Organized the first French Asian Workshop on Next Generation Internet (2004).

Reviewing activities

Reviewer of a large number of papers submitted to journals including IEEE/ACM Transactions on Networking, IEEE Journal on Selected Areas in Communications, Distributed Computing Journal, Computer Communications Journal, Computer Networks, Internetworking, TSI, Annals of Telecommunications, etc.

Conference and summer schools tutorials

EcoTel'98	Tutorial on TCP over Satellite, December 98, Antibes.
RHDM'98	Tutorial on Satellite Networking, May 98, Giens.
CFIP'97	Tutorial on End to end multicast transmission control, September 97, Liège.
ECMAST'97	Tutorial on Advanced Internet Protocols: Support of Audio and Video applications over the Internet, May 97, Milan.
CFIP'96	Tutorial on Multimedia Application over the Internet, October 96, Rabat.
RHDM'96	Tutorial on Point to Multipoint Communication, August 96, Cauterets.
PCN'95	Tutorial on Internet Architecture and Organization, October 95, Istanbul.
FORTE'95	Tutorial on High Performance Protocol Architecture, October 95, Montreal.
HPN'95	Tutorial on Point to Multipoint Communication, September 95, Palma.
RHDM'95	Tutorial on High Speed Protocols, September 95, Paris.

Selected Invited Presentations

Intimate'07	Presentation on Experimental Methodology for Networking Platforms.
CoNext'06	Panel presentation on Networking platforms and reproducible experiments.
AIT	Course on Group Communication in Wide Area Networks, Asian Institute of Technology, December 2003, Bangkok.
Fête de l'Internet	Panel presentation on Issues related to High Speed Networking in the context of "La fête de l'Internet" organized at Cité des Sciences et de l'Industrie in March 2003 in Paris.
EMSST	Two presentations on Multimedia Applications over the Internet at the "Amicale de l'Enseignement Militaire Supérieur Scientifique et Technique" in 2002 and 2003.
UTLS'2000	Large audience presentation on Satellite Internet Protocol Performance at the "Université de tous les savoirs" conference cycle, September 2000, Paris.
Networking'00	Presentation on Dynamic Routing over Unidirectional Satellite Links, Broad Band Satellite Networking Workshop, May 2000, Paris.
Cité des Sciences	Internet Transmission via Satellite, "La recherche en direct" conference cycle, March 1999, Paris.

SCW	Invited presentation at the Satellite Communications Workshop on IP integration with Satellite Networks, March 1999, Brussels.
UTC	Invited seminar on Multicast Routing at Compiègne Technology University, February 1998.
ENST Paris	Support of Audio and Video applications over the Internet, June 1997.
WWW conference	Panel presentation at the fifth WWW conference in Paris on Real Time Multiparty Applications (1996).
MIT LCS	Invited seminar on High Performance Protocol Architecture, July 1996.
ENSMSE	High Speed Protocols, at Ecole Nationale Supérieure des Mines de St Etienne in February 1995.
ENST Bretagne	High Performance Protocol Architecture, April 1994.

Expertise activities

2007	Participation to the FIRE group activities for the definition of a Roadmap for Research on Federated test-beds.
2007	Independent Expert assisting the European Commission for the evaluation of IST FP7 projects.
2003	Independent Expert assisting the European Commission for the evaluation of IST FP6 projects.
1999-2000	Independent Expert assisting the European Commission for the evaluation of IST FP5 projects.
1996	Techno-Economical Expertise for ANVAR
1995	Independent Expert participating to the preparation of FP4 Work Programme.
1997	Independent Expert assisting the European Commission for the evaluation of ACTS projects.
1998-2000	Expert acting as an evaluator in commission 3 of the French RNRT (Réseau National de Recherche en Télécommunication).

PhD Theses Supervised

Stevens Leblond thesis on Next Generation Peer-to-Peer Infrastructures, ongoing.
Amine Ismail thesis on Optimisation of IP protocols and applications over broadcast links, ongoing.
Mathieu Lacage thesis on Methodology for Networking Experimentation, ongoing.
Diego Dujovne thesis on Wireless Experimental Test-beds, ongoing.
Mohamed Ali Kaafar thesis on Securing Internet Coordinate Systems, September 2007.
Mouhammad Malli thesis on Inferring Internet topology from application point of view, September 2006.
Laurentiu Barza thesis on Communication architecture for Large Scale Virtual Environments, July 2004.
Fatma Louati thesis on Asymetry and Bidirectional trafics on the Internet, February 2004.
Antoine Clerget thesis on Heterogeneity in IP multicast Networks, December 2003.
Miguel Sánchez thesis on Optimization of Packet Forwarding in Best-effort Routers, September 2003.
Rareş Şerban thesis on Dynamic IP QoS management, September 2003.
Fethi Filali thesis on Deploying Multicast Services in Heterogeneous Environments, November 2002.
Frank Lyonnet thesis on Multimedia Applications over the Internet with wireless links, October 1998.
Claude Castelluccia thesis on High Performance Transmission Control Protocols, March 1996.

Participation examining boards / Reviewing

Laurent Bernaille, PhD thesis on Early Application Identification, UPMC, 2007.
Guillaume Urvoy Keller, HDR thesis topic: From Quality of Service to Traffic Analysis, UNSA, 2006.
Matti Siekkinen, PhD thesis on Root Cause Analysis of TCP throughput: Methodology, Techniques and Applications, 2006.
Ahmad AlHanbali, PhD thesis on Performance Evaluation of Mobile Wireless Networks, UNSA, 2006.
Joanna Moulierac, PhD thesis on Aggregation of Multicast communications, URennes1, 2006.
Gueye Cheikh Ahmadou Bamba, PhD thesis on Inferring Geographic Location of Internet Hosts based on Multilateration, UPMC, 2006.
Jocelyne Elias, PhD thesis on Dynamic Bandwidth Allocation in Networks with QoS Support, UPMC, 2006.
Mohammad Hossein Manshaei, PhD thesis on Cross Layer Interactions for Adaptive Communications in IEEE 802.11 Wireless LANs, UNSA, 2005.
Abelhamid Nafaa, PhD thesis on Error and Access Control for Video Streaming over Wireless LANs, UVSQ, 2005.
Imed Romdhani, PhD thesis on Support of multicast communications in a Mobile-IP environment, UTC, 2005.
Mickaël Hoerdt, PhD thesis on Source Specific Multicast (SSM): Towards a scalable inter-domain multicast service, 2005.
Fabrice Arnal, PhD thesis on Optimisation Reliable Multicast Communication over GEO satellites, ENST, 2004.
Julien Fasson, PhD thesis on Architecture for IP/Satellite Integration, ENST, 2004.
Idris Rai, PhD thesis on QoS support in Internet edge routers, Eurecom, 2004.
Karim Sbata, PhD thesis on Multicast Proxies NeTwork: A Multi-protocols Broadcast Architecture, INT, 2003.
Artur Ziviani, PhD thesis on Quality of Service and Location-Awareness, UPMC, 2003.
Sara Alouf, PhD thesis on Parameter estimation and performance analysis of several network applications, UNSA, 2002.
Rolland Vida, PhD thesis on Protocol design for Group and Mobility Management in a multicast environment, UPMC, 2002.
Hatem Bettahar, PhD thesis on Multicast routing with QoS support, UTC, 2001.
Thomas Ziegler, PhD thesis on Optimizing Fairness and Efficiency of Internet Congestion Control, UPMC, 2001.
Arnaud Legout, PhD thesis on Multicast congestion control in best effort networks, Eurecom, 2000.
Isabelle Hamchaoui, PhD thesis on Design and evaluation of ATM VPNs, UPMC, 1999.
Olivier Fourmaux, PhD thesis on Multicast in an IP over ATM environment, UPMC, 1998.
Heba Koraytim, PhD thesis on Multiple Access Protocols and Resource Allocation over Satellite Links, ENST Paris, 1998.
Dominique Grad, PhD thesis on Logical Routing for Group Communications, UStrasbourg, 1997.
Raymond Schneider, PhD thesis on A Reliable and Ordered Broadcast Protocol, UStrasbourg, 1995.

Master theses and Internship supervised

D. Koudriashov, Ecole Polytechnique, on Multicast Routing in Overlay Networks, 2004.
G. Constantin, Ecole Polytechnique, on Support of PIM-SM over satellite links, 2002.
R. Yerbanga, ENSIAS, on Scalability of RIP and DVMRP Routing Protocols over satellite links, 2002.
R. Guerin, DEA UTT, on Simulation of Diffserv QoS mechanisms in a Network, 2002.
G. Aniba, INPT, Support of IP multicast over Satellite with OBP, 2002.
S. Heriard-Dubreuil, ENS, on Congestion Control for Reliable Multicast, 2001.
W. ZiYu, DEA RSD UNSA, on QoS support mechanisms in the Internet, 2000.
L. Barza, DEA RSD UNSA, on Supporting Video Transmission Using SSM, 2000.
M. Ruiz-Sanchez, DEA RSD UNSA, on Fast Forwarding Mechanisms in IP routers, 1999.

N. Boutayeb, EMI, on QoS support mechanisms in the Internet, 1999.
O. Arbouche, EMI on QoS support over a satellite link, 1999.
F. Filali, Master ENSI, on Optimizing satellite networks with Unidirectional Links, 1999.
I. BenHmida, ENSI, on Support of IP audio application over Frame Relay, 1999.
P. Narvaez, MIT, on Routing in IP satellite constellations, 1998.
N. Chaher, DEA RSD UNSA, on Performance of TCP over satellite, 1998.
O. Alayli, AUB, on Multicast Routing over Satellites, 1998.
W. Soubra, ENSEIHT, on Routing in satellite constellations, 1998.
C. Jalpa, DEA RSD UNSA, on Support of Reliable Multicast over RTP, 1996.
C. Lefèvre, ISEP, on Performance Evaluation of the high Speed Myrinet Networks, 1996.
K. Okba, INPT, on Transmission Control Mechanisms for Reliable Multicast, 1995.
H. Ricardon, DEA RSD UNSA, on IP Extension to support mobility in high speed networks, 1993.
B. Kiss, Eurecom, on Shared white board application, 1993.
T. Turletti, ESSI, on TCP/IP – X.25 gateway, 1989.
H. Soubra, DEA, on Real Time Data Transmission over LANs, 1989.

Selected development activities

Implementation of the XTP protocol in user space, 1993.
Optimised Implementation of transport level OSI protocols, 1989.
Implementation of several protocol gateways for TCP, X.25 and OSI Internetworking, 1988.
Modification of the Unix kernel to support real time applications, DEA Internship, 1987.
Implementation of an Image Memory for a Robotics Application, LIMSI, 1985.

Programming skills

Unix, Windows
Java, C, C++, PASCAL, BASIC, FORTRAN
LISP, PROLOG, ADA

Awards and distinctions

Best Class Average in engineering studies (1981-1986).
Scholarship from the French Government to do PhD studies in France (1986).
Scientific Prize CS2000 from the Communications and Systems in July 2000.

Languages

Arabic, French and English.

Personal Information

Born in Beirut, June 30th 1964, Married with two Children.

Short Biography

Dr Walid Dabbous is a senior researcher at INRIA and professor at the Ecole Polytechnique. His research interests include: Future Internet Architecture and Protocols, Networking Experimental Platforms and Simulators, Experimental Methodology for Networking Protocols, Securing Internet coordinate systems, Large scale virtual environments, Large scale reliable multicast protocols, Group communications, Internet Satellite Networking, Quality of Service support, Buffer management mechanisms in routers, Flexible protocol architecture, High performance communication protocols, Unicast and Multicast Congestion control, Audio and video conferencing over the Internet. He graduated from the Faculty of Engineering of the Lebanese University in Beirut in 1986 (Electrical Engineering Department). He obtained his DEA and his Doctorat d'Université from the University of Paris XI in 1987 and 1991 respectively. He joined the RODEO Team within INRIA in 1987. He is a staff researcher at INRIA since 1991, and leader of the RODEO (then Planète) team since 1996.

List of Publications by September 2007

The list contains 1 submitted journal paper, 9 journal papers, 44 conference papers, 5 national conferences, 2 books (editor), 2 theses, 3 books chapters, 1 RFC, 2 Internet-Drafts, 14 INRIA reports and 11 other documents.