# Topologies de l'internet : des routeurs aux réseaux recouvrants

Damien Magoni

Université Louis Pasteur – Strasbourg 1
UFR de Mathématique et d'Informatique
LSIIT – UMR n° 7005 du CNRS

N° d'ordre : 718

# MÉMOIRE

Pour l'obtention d'une
Habilitation à Diriger des Recherches en Informatique

Présenté par

**Damien MAGONI**

# TOPOLOGIES DE L'INTERNET : DES ROUTEURS AUX RÉSEAUX RECOUVRANTS

Soutenu publiquement le 19 décembre 2007 devant le jury composé de :

**M**$^{me}$ **Dominique BECHMANN**, garante d'habilitation
Professeure à l'Université Louis Pasteur
**M. Jean-François DUFOURD**, rapporteur interne
Professeur à l'Université Louis Pasteur
**M. Serge FDIDA**, rapporteur externe
Professeur à l'Université Pierre et Marie Curie
**M. Olivier FESTOR**, rapporteur externe
Directeur de recherche à l'INRIA de Lorraine
**M. Mario FREIRE**, examinateur
Professeur à l'Université du Beira Intérieur
**M. Pascal LORENZ**, examinateur
Professeur à l'Université de Haute Alsace

There is nothing impossible to him who will try.
**– Alexander the Great**

*à l’Atma.*

# Remerciements

Merci infiniment à Jean-François Dufourd, Serge Fdida et Olivier Festor d’avoir accepté d’être rapporteurs de mon habilitation. Je remercie également Mario Freire et Pascal Lorenz d’avoir accepté d’être examinateurs de mon habilitation. Je remercie encore spécialement Pascal Lorenz d’avoir collaboré à mes recherches et de m’avoir donné la possibilité de co-encadrer des doctorants. Je remercie chaleureusement Dominique Bechmann de m’avoir soutenu durant mes années de maître de conférences et d’avoir accepté le rôle de garante d’habilitation. Mes remerciements vont aussi à Fabrice Heitz et Michel de Mathelin pour avoir soutenu mon projet d’habilitation. Je suis extrêmement reconnaissant de l’aide apportée par toutes ces personnes sans lesquelles je n’aurai jamais pu soutenir mon habilitation. Je tiens à exprimer ma profonde gratitude envers les doctorants et les étudiants qui ont travaillé avec moi. Enfin je remercie de tout coeur mes proches et ma famille pour leur soutien sans faille.

# Table des matières

# Résumé

L'Internet est devenu sans conteste en trois décades le support matériel majeur de l'ère de l'information. Durant cette courte période, il a subi deux mutations importantes. Tout d'abord, il a subi une mutation de taille : de quelques dizaines de systèmes interconnectés suivant une topologie fixée et bien connue, l'Internet est passé à une taille phénoménale de plus de 285 millions d'hôtes suivant une topologie désormais à dimension libre dûe au fait que son développement n'est plus centralisé. De plus, il a subi une mutation de forme : nous sommes passés des gros systèmes interconnectés par de simples lignes téléphoniques à des ordinateurs personnels mobiles interconnectés par des technologies radio en bordure de réseau et par des routeurs gigabits reliés par fibre optique en coeur de réseau.

Ces deux mutations entraînent deux défis importants pour les chercheurs qui oeuvrent à l'amélioration de l'Internet. Premièrement, il est nécessaire de connaître avec le plus de précision possible, la topologie de l'Internet. Seules des méthodes macroscopiques et statistiques permettent désormais de connaître cette topologie car plus personne n'a autorité sur l'organisation et l'expansion du réseau Internet. De plus, étant donné sa dynamicité, il est aussi souhaitable que cette topologie soit mise à jour fréquemment. Deuxièmement, l'évolution rapide des technologies a fait que les paradigmes qui sous-tendaient à la création des protocoles initiaux de l'Internet ne sont plus valables à l'heure actuelle. Les notions de mobilité, de sécurité et de diffusion quasi-inexistantes à l'origine sont désormais ardemment souhaitées par les utilisateurs mais difficiles à mettre en oeuvre dans les protocoles actuels. Il faut donc trouver les moyens d'offrir ces nouvelles fonctionnalités en conservant les propriétés initiales des protocoles qui ont fait le succès d'Internet.

Dans ce mémoire d'habilitation, nous présentons des contributions qui tentent de répondre à certains aspects de ces deux défis majeurs. Nous nous concentrons tout particulièrement sur la cartographie de l'Internet, le stockage d'états dans les routeurs exécutant des protocoles multipoints, l'implémentation de fonctionnalités réseaux dans les hôtes et la conception de réseaux recouvrants.

## Mots-clés

Adressage, attaque, cartographie, Internet, multichemin, multipoint, nommage, protocole, recouvrant, recouvrement, réseau, routage, topologie.

# Introduction

## Parcours réalisé

Ce mémoire constitue une synthèse de l'ensemble des travaux de recherche que j'ai effectué depuis l'obtention de mon doctorat en janvier 2002 soit presque six années de recherche. Le fil conducteur de tous ces travaux est assurément l'étude de la topologie de l'Internet et de ces applications. C'est pourquoi j'ai choisi ce sujet comme thème fédérateur de mes travaux. La topologie que nous étudions ici est celle définie au sens de la théorie des graphes, c'est à dire l'étude des propriétés liées à la manière dont les liens d'un graphe relient les noeuds de celui-ci. La topologie physique de l'Internet étant presque impossible à réaliser, nous présentons dans la première partie de ce mémoire des résultats sur la topologie de niveau 3 selon le modèle OSI de l'Internet, c'est à dire celle du niveau de l'Internet Protocol (IP). Puis dans la deuxième partie, nous étudions l'influence des topologies des arbres créés par les protocoles multipoints sur le stockage des états et la possibilité d'attaques par déni de service s'appuyant sur une saturation localisée de ces états. Enfin dans la dernière partie, nous présentons nos intergiciels situés au niveau applicatif. Dans le dernier chapitre en particulier, nous présentons notre intergiciel permettant de créer des recouvrements (*overlays* en anglais) au dessus de l'Internet et nous étudions l'influence de la topologie de ces réseaux virtuels sur les performances de routage de l'architecture de notre intergiciel.

Tous ces travaux ont été réalisés dans un premier temps au sein de l'équipe réseau du LSIIT de l'université Louis Pasteur durant la période 2002-2003 puis dans un deuxième temps en partenariat avec l'équipe du GRTC de l'université de Haute Alsace durant la période 2004-2007. J'ai commencé à m'intéresser à la topologie de l'Internet durant mon stage de DEA débuté en janvier 1999. Mon travail de DEA puis celui de thèse portait sur un nouveau protocole multipoint et j'avais besoin d'un modèle réaliste de l'Internet pour effectuer mes simulations. Il existait peu de données sur la topologie de ce réseau à l'époque et je me suis donc lancé dans cette étude en parallèle à mes travaux de thèse. J'étudie toujours cette problématique actuellement en partenariat avec Matthieu Latapy et Jean-Loup Guillaume du LIAFA.

Suite à ma nomination sur un poste de maître de conférences, j'ai co-encadré avec Jean-Jacques Pansiot la thèse de doctorat de Mickaël Hoerdt dont le sujet concernait le multipoint interdomaine. J'avais déjà co-encadré Mickaël lors de son stage de DEA de janvier à juin 2002 et ayant moi-même fait une thèse sur un protocole multipoint, il s'avéra naturel de co-encadrer sa thèse. Je lui ai fourni des outils permettant de faire des simulations à grande échelle de protocoles multipoints tel que mon logiciel *nem* et j'ai intégré Mickaël à mes travaux concernant la cartographie de l'Internet. Par conséquent, le chapitre premier ainsi que les chapitres trois à cinq présentent des travaux que nous avons effectué ensemble.

Les difficultés rencontrées dans la recherche concernant les protocoles multipoints ainsi que les protocoles de la couche réseau en général m'ont poussé à faire évoluer mes activités de recherche vers les protocoles de la couche application depuis juin 2002. J'ai intensifié cette recherche suite à ma nomination fin 2002. De janvier à juin 2003 j'ai co-encadré le stage de DEA de Dragos Manzateanu dans lequel nous avons étudié en détail le multipoint applicatif et en

particulier l'architecture NARADA [hCRSZ01]. Dans le même temps, j'ai jeté les bases d'une architecture nommée Dynamic Hierarchical Addressing Routing and naMing (DHARMA) destinée à fournir une interface entre le réseau et les nouvelles applications distribuées à base de recouvrements. Cette architecture a pu être approfondie grâce aux travaux de Anthony James lors de son stage de DEA que j'ai encadré de janvier à juin 2004 ainsi qu'à des stagaires de DESS (cf. dernier chapitre). Actuellement, le coeur de mes recherches reste centré sur les protocoles de niveau applicatif permettant de créer des réseaux recouvrants (*overlays*). Les possibilités offertes par ce changement de contexte sont immenses.

D'octobre 2004 à octobre 2007, j'ai co-encadré avec Pascal Lorenz et Hervé Guyennet la thèse de doctorat de Fatiha Djemili Tolba qui concerne l'économie d'énergie et la gestion de groupe dans les réseaux ad-hoc. Certaines des propriétés de ces réseaux sont en effet similaires à celles des recouvrements dont les membres sont hautement dynamiques. Depuis mars 2005, je co-encadre à nouveau avec Pascal Lorenz la thèse de doctorat de Khaldoon Shami dont le sujet est l'étude de l'architecture de communication de niveau applicatif pour réseaux recouvrants que j'ai définie début 2003 [Mag03a] et dont les fondements sont décrits dans le dernier chapitre de ce mémoire.

En janvier et février 2005, j'ai effectué un séjour de recherche au National Institute of Advanced Industrial Science and Technology (AIST) au Japon dans l'équipe du docteur Yoshio Tanaka. Cette équipe fait partie du Grid Technology Research Center (GTRC) et cette immersion m'a permis d'approfondir mes connaissances sur les grilles et en particulier de travailler sur l'intergiciel Ninf-G [TTNS05] qui est le Work Package numéro 2 du projet NAREGI financé par le MEXT. En juin et juillet 2007, j'ai effectué un séjour de recherche à la School of Electrical and Information Engineering (EIE) de l'université de Sydney en australie dans l'équipe du professeur Abbas Jamalipour. Durant cette période j'ai travaillé sur le projet Cross-Correlated Security and Service Quality in Heterogeneous Mobile Communication Networks financé par l'Australian Research Council (ARC grant number DP0771523). En 2008 je vais effectuer un séjour de recherche à l'Université du Michigan aux USA dans l'équipe du professeur Sugih Jamin pour participer au projet Secure Coordination and Communication in a Crisis Using Handheld Devices financé par le US DoHS (contrat numéro W911NF 05 1 0415).

# Contexte de recherche

Le contexte de la première partie de ce mémoire est celui de la connaissance de la topologie de l'Internet. La connaissance du réseau Internet est de première importante pour la recherche en protocoles. En effet la topologie du réseau influence le comportement des protocoles de routage, l'efficacité des protocoles multipoints, celle des protocoles de gestion de qualité de service, etc. Les premiers efforts de modélisation réaliste de réseaux étendus ont été réalisés par Doar [Doa96] dans son logiciel Tiers et par Zegura *et al.* [CDZ97, ZCD97] dans leur logiciel GT-ITM et remontent aux années 1996-1997. Ces modèles étaient hiérachisés en trois niveaux (*i.e.*, LAN-MAN-WAN) mais au coeur de chaque niveau, les réseaux étaient générés de manière aléatoire. Cela impliquait que les distributions des degrés des noeuds de ces réseaux générés suivaient des distributions normales (Gaussiennes). Ce n'est que suite aux travaux de Faloutsos *et al.* en 1999 [FFF99] que l'on s'est rendu compte que l'Internet est un réseau de dimension libre : la distribution des degrés des noeuds est une distribution dite à queue lourde et elle suit à priori une loi puissance (même si ce dernier point reste discutable). A la suite de cette découverte, beaucoup de chercheurs ont créé de nouveaux modèles de topologies réalistes de l'Internet [TPSF01, TGJ+02, BT02] ainsi que les générateurs correspondants. Nous avons participé à cette démarche mais d'une façon particulière. Nous avons créé un générateur de topologies qui fonctionne par échantillonnage de cartes IP réelles [MP02b] et non à partir d'un

algorithme théorique comme les autres méthodes. En échantillonnant les cartes suivant certaines techniques, nous avons été capable de créer des réseaux de type Internet qui conservent les propriétés les plus importantes de la topologie de l'Internet tout en étant de taille suffisamment réduite pour pouvoir être utilisés dans des simulateurs tels que *ns-2*, Opnet ou GloMoSim. La puissance des machines ne cessant de s'accroitre, il est devenu alors plus logique d'utiliser directement de vraies cartes de l'Internet dans les simulateurs. Que l'on échantillonne des cartes ou qu'on les utilise tel quel, le problème devenait désormais l'acquisition de ces cartes. Les difficultés rencontrées pour se les procurer nous poussèrent à créer fin 2002 notre propre moteur de cartographie de l'Internet au niveau IP. Il s'agit de notre logiciel *nec*, présenté en détail dans le premier chapitre de ce mémoire. De la création de modèles de réseaux étendus de type Internet, les chercheurs se sont alors tournés vers l'obtention et la validité de la cartographie de l'Internet et ils y sont encore impliqués aujourd'hui. Nous participons activement à cette étude au travers des travaux pilotés par Latapy *et al.* sur la pertinence des explorations massives de la topologie de l'Internet [LG05]. Toujours en relation avec la topologie de l'Internet, nous évaluons dans le deuxième chapitre, la robustesse de l'Internet au niveau IP en nous basant sur des cartes de routeurs et de systèmes autonomes.

Le contexte de la seconde partie de ce mémoire est celui des protocoles multipoints. Ce thème de recherche célèbre parmi la communauté des chercheurs réseaux a subi de fortes évolutions depuis sa mise en route en 1985 par Deering et Cheriton. Paradoxalement, 20 ans plus tard, seule la RFC 1112 [Dee89] datant de 1989 est devenue un RFC standard ! Toutes les autres RFCs concernant le multipoint n'ont pas dépassé le stade de "proposed standard". De plus les RFCs concernant les protocoles Protocol Independant Multicast-Dense Mode (PIM-DM) et PIM-Sparse Mode (PIM-SM), qui sont les seuls protocoles de gestion d'arbres multipoints de facto, sont toujours expérimentales ! On comprend pourquoi même actuellement très peu d'opérateurs déploient le multipoint dans leurs réseaux. Le multipoint au niveau de la couche réseau s'est révélé être un cuisant échec parce que les protocoles actuels (*i.e.*, les mêmes que ceux du début de l'Internet) se prêtent mal aux concepts du multipoint. En particulier, les adresses IP mélangent la notion de localisation à celle d'identification : le multipoint ne peut se concevoir qu'en séparant distinctement ces deux notions. Les chercheurs ont tenté de faire du multipoint sans redéfinir les autres protocoles ce qui a abouti à des solutions impraticables. Deux avancées majeures ont heureusement pu être réalisées dans ce domaine. La première est une architecture nommée EXPRESS, qui a été proposée par Holbrook et Cheriton en 1999 [HC99]. Elle a donné lieu à la création du protocole PIM-Source Specific Multicast (SSM) [Bha03] par opposition aux anciens protocoles nommés Any Source Multicast (ASM). Le protocole PIM-SSM résoud beaucoup de problèmes rencontrés dans le multipoint en autorisant uniquement, comme son nom l'indique, une diffusion de 1 vers n hôtes alors que le multipoint se définit comme une communication de m vers n hôtes. La deuxième avancée concerne le multipoint de niveau applicatif. Voyant qu'il n'était pas possible de faire du multipoint au niveau de la couche réseau, les chercheurs l'ont implémenté directement dans les hôtes, ces derniers étant connectés entre eux par des connexions point à point standard de type TCP en général. Les groupes multipoints applicatifs constituent ainsi des réseaux recouvrants. Les premiers travaux de ce type datent de 2001 [hCRSZ01, LN01] mais des solutions sont régulièrement proposées [BBK02, BKK+03, KB04]. Le multipoint au niveau réseau, qu'il soit ASM ou SSM, a toujours posé une énigme concernant son extensibilité. Les états multipoints étant stockés dans les routeurs, combien de groupes logiques pouvons nous maintenir dans l'Internet avant de saturer les routeurs ? C'est cette problématique que nous analysons en détail dans le troisième chapitre de ce mémoire. Nous étudions aussi dans le quatrième chapitre la possibilité de porter des attaques par déni de service distribué sur un routeur en saturant sa mémoire par des états multipoints à l'aide du protocole PIM-SSM.

Le contexte de la dernière partie de ce mémoire est celui des réseaux recouvrants créés par les nouvelles familles d'applications distribuées (*e.g.*, multipoint applicatif, pair à pair, grille, etc.). Une application mettant en jeu plus de deux hôtes qui établissent des connexions entre eux selon diverses topologies (*e.g.*, arbre, maillage, etc.) forme un réseau virtuel au dessus du réseau Internet qui est nommé overlay en anglais et qu'on traduit par recouvrement ou bien réseau recouvrant pour être plus précis. Les fonctionnalités réseaux d'une application de ce type sont entièrement gérées dans les hôtes. La perte de performances par rapport à une implémentation dans les équipements réseaux tels que les routeurs est compensée par la souplesse de déploiement et d'utilisation. Le multipoint par exemple a été déployé avec succès sous sa forme applicative (cf. paragraphe précédent), donc en utilisant des recouvrements. Les recouvrements multipoints [hCRSZ01, LN01, BBK02, BKK+03, KB04] sont en général constitués de connexions simultanées et sont donc synchrones et très dynamiques. Les réseaux pair à pair [ABKM01, SMK+01, RFHaSS01, RD01, RKDC01, GSGM03] permettant le partage de fichiers de manière totalement distribuée, sont par contre plutôt asynchrones (tous les membres ne sont pas connectés en même temps) et persistants. Enfin les grilles [FK99] qui sont des applications permettant le partage de ressources quelconques au dessus de l'Internet sont probablement les applications distribuées les plus avancées que l'on peut trouver actuellement. Ces grilles permettent, de par l'utilisation d'intergiciels [FK97] déployant entre autres des recouvrements, d'unifier des ressources processeurs, mémoires et stockage situées sur des machines réparties partout dans le monde. L'ensemble de ces ressources peuvent être utilisées de façon unifiée et transparente par un utilisateur quelconque de l'Internet (authentifié en général). Nous proposons dans le cinquième chapitre, un intergiciel qui implémente des fonctions réseaux au niveau d'un hôte. Cet intergiciel ne permet pas de créer des recouvrements à proprement parler mais il permet d'interfacer des applications multipoints anciennes de type Any Source Multicast (ASM) avec des réseaux fournissant uniquement le protocole Source Specific Multicast (SSM). Il illustre parfaitement l'intérêt de placer des services réseaux dans des hôtes et non dans des équipements spécifiques. Dans le sixième chapitre nous présentons l'architecture d'un autre intergiciel qui permet de créer des recouvrements pour un grand nombre d'applications distribuées. Le but de cet intergiciel est de factoriser les besoins de création et de gestion des recouvrements des applications. Son utilisation permet à l'application de se libérer de la gestion des membres de son recouvrement. Les fonctionnalités fournies par cet intergiciel peuvent être utilisées par les familles d'applications décrites ci-dessus.

## Organisation du mémoire de synthèse

Ce mémoire d'habilitation qui récapitule les travaux de recherche dans lesquels je me suis investi, se divise en trois parties distinctes comprenant chacune deux chapitres :

- Nous nous concentrons dans la première partie du mémoire sur l'étude des aspects topologiques de l'Internet. Nous étudions la topologie de l'Internet au niveau du protocole IP donc de la couche 3 du modèle OSI aussi appelée couche réseau :
  - Nous présentons dans le premier chapitre des données concernant cette topologie ainsi que notre logiciel de cartographie nommé *network cartographer* qui permet de dresser la carte des routeurs et des liens logiques IP se trouvant au coeur de l'Internet. Nous montrons que notre logiciel est plus efficace que Mercator, seul logiciel libre de cartographie disponible à l'époque. En une période de 10 jours, *nec* a découvert presque autant de routeurs et surtout presque deux fois plus de liens que Mercator. Nous étudions les recouvrements entre systèmes autonomes (AS en anglais) et routeurs et ceux-ci nous montrent que *nec* produit une carte beaucoup moins biaisée que Mercator. Nous étudions la topologie IPv6 ainsi que les délais des chemins et des liens de l'Internet

IPv4. Nous étudions aussi l'extensibilité de notre logiciel. Les chercheurs souhaitant utiliser les cartes produites par *nec* pour faire des simulations de protocoles peuvent les trouver sur le web [MH]. Le logiciel de cartographie *nec* est le fruit d'un travail collectif. La partie IPv6 a été réalisée par Mickaël Hoerdt lors de son travail de doctorat que j'ai co-encadré avec Jean-Jacques Pansiot. Les accès aux serveurs web *traceroute* ont été développés par Anthony James et Pascal Merindol lors de leurs travaux d'études et de recherche (TER) de maîtrise. La partie analyse des données a été réalisée par Déborah Malka lors de son projet d'IUP 3ème année. Enfin la partie injection directe des paquets de traçage a été développée par Jeanne Maillard et Marc Jaeger lors de leur projet d'IUP 3ème année. Le moteur de cartographie ainsi que la partie IPv4 a été réalisée par mes soins.

  - Dans le deuxième chapitre nous utilisons diverses données de la topologie de l'Internet, en particulier des cartes des systèmes autonomes et des cartes des routeurs afin d'étudier la robustesse de la topologie de l'Internet face à des attaques ciblées. Nous proposons plusieurs nouvelles techniques pour attaquer la connectivité de niveau IP de l'Internet ainsi que des nouvelles métriques pour évaluer la quantité de fragmentation dans le réseau et dans la distribution des routeurs parmi les composantes connexes. Nous montrons par exemple que la suppression de seulement 4% des routeurs laisse des groupements (clusters en anglais) d'au plus 100 routeurs dans une carte de plus de 100k routeurs. Nous étudions aussi l'emplacement des cibles et montrons qu'elles ne sont pas concentrées dans des AS spécifiques mais qu'elles sont réparties sur des milliers d'entre eux et que la qualité de la carte de niveau de routeur est cruciale pour viser avec précision les meilleures cibles. L'étude de la résistance de l'Internet présentée dans ce chapitre a pu être menée à bien grâce aux nombreux échanges fructueux ayant eu lieu entre Hyunseok Chang et Sugih Jalin de l'University of Michigan et moi-même.

- Nous explorons dans la deuxième partie du mémoire les effets à grande échelle du stockage des états multipoints dans les routeurs en fonction de la topologie des arbres correspondants :
  - Dans le troisième chapitre nous proposons un mécanisme simple conçu pour profiter de multiples plus courts chemins dans le but d'augmenter les chances qu'un nouveau membre puisse joindre un arbre multipoint tout en distribuant les états multipoints parmi les routeurs non saturés. Pour effectuer des simulations réalistes, nous utilisons une carte des routeurs du coeur de l'Internet que nous avons créé lors de nos précédents travaux (cf le premier chapitre du mémoire). Nous utilisons notre carte pour modéliser des grandes topologies intra-domaines. Nous montrons que lorsque de tels réseaux sont très saturés, l'augmentation des messages *join* alternatifs fructueux va de 25% à 55% selon la densité de session. Ces travaux sur la répartition des états multipoints ont pu être effectués grâce aux contributions de Mickaël Hoerdt. La partie empilement d'arbres d'états multipoints a été réalisée par Mickaël Hoerdt lors de son travail de doctorat co-encadré par Jean-Jacques Pansiot et moi-même. La partie redistribution des états par redirection des adhésions multipoints a été réalisée par moi-même.
  - Nous étudions dans le quatrième chapitre les possibilités que donnent les protocoles multipoints actuels de réaliser des attaques distribuées par déni de service. Nous simulons l'impact d'attaques distribuées par déni de service basées sur le protocole PIM-SSM dans des topologies de type Internet. Les résultats montrent que ces attaques peuvent se révéler d'une efficacité redoutable car elles atteignent la cible et son périmètre proche sans quasiment surcharger le réseau dans son ensemble. Nous montrons de plus qu'il n'est pas nécessaire d'avoir un grand nombre d'attaquants pour réussir à saturer n'importe quel routeur de l'Internet. Là encore, ces travaux sont le fruit du travail collectif mené par Mickaël Hoerdt, Jean-Jacques Pansiot et moi-même.

- Enfin dans la troisième partie du mémoire nous présentons nos travaux situés au niveau applicatif. Il s'agit d'intergiciels implémentés dans des hôtes mais effectuant des fonctions liées au réseau et permettant, dans le cas de notre architecture d'adressage, de créer et de maintenir des réseaux recouvrants au dessus de l'Internet pour une grande variété d'applications :
  - Nous présentons dans le cinquième chapitre une bibliothèque de fonctions multipoints nommée Libssmsdp pour les applications multi-sources en utilisant uniquement le Source Specific Multicast (SSM) au niveau de la couche réseau. Le protocole SSM est un nouveau paradigme de distribution multipoint sur IP : en poussant la sélection de source dans les hôtes, il devient un bon candidat pour offrir un service multipoint dans l'Internet tout entier. Il est bien adapté pour les applications de flux télévisuel et radio mais il ne fournit pas la découverte automatique de source comme le paradigme multipoint initial Any Source Multicast (ASM) de Steeve Deering. En considérant des applications multi-sources avec dynamicité des sources et les vieilles applications basées sur le modèle ASM, il était clair qu'il manquait une bibliothèque multipoint implémentant un protocole de découverte de source. Ces travaux ont été conduits par Mickaël Hoerdt dans le double cadre du projet européen *6net* et de sa thèse de doctorat. En tant que co-encadrant de la thèse de Mickaël Hoerdt, j'ai été impliqué dans les aspects recherche et évaluation de ce projet aux côtés de son directeur de thèse Jean-Jacques Pansiot. L'implémentation de la bibliothèque Libssmsdp a été réalisée par Mickaël Hoerdt et Frédéric Beck.
  - Dans le sixième chapitre, nous proposons une architecture distribuée d'adressage, de routage et de nommage conçue pour les réseaux recouvrants à topologies quelconques et déployés sur l'Internet. Nos résultats de simulation obtenus sur trois cartes réalistes de l'Internet de 4k noeuds à 75k noeuds montrent que notre solution entraîne un taux d'inflation des chemins variant de 78% à 193% selon la taille du recouvrement et l'emplacement du premier noeud. Nous décrivons aussi comment faire face à la dynamicité du réseau et nos résultats de simulation ont montré qu'une certaine robustesse est réalisable : en attribuant simplement plusieurs adresses à chaque noeud on multiplie par 2 le pourcentage de succès du routage hiérarchique lorsque la dynamicité du réseau est supérieure à 10%. Les premières avancées de ce projet ont pu être réalisées grâce aux contributions de nombreuses personnes. L'évaluation des algorithmes d'adressage et de routage dans le simulateur *ns-2* a été effectuée par Anthony James dans le cadre de son stage de DEA. Le premier prototype de DHARMA écrit en langage C et implémenté sous LINUX a été réalisé par Pierre-Emmanuel Corvi et Marc Jaeger dans le cadre de leur projet de fin d'année de DESS. La création d'un module de niveau noyau sous Windows afin de pouvoir efficacement implémenter DHARMA sous Windows a été menée à bien par Michel Cieplinski, Nicolas Varlet et Nicolas Wauters lors de leurs travaux d'études et de recherche (TER) de maîtrise.

La conclusion de ce mémoire présente d'une part un bilan de nos recherches ainsi que les applications potentielles issues de celles-ci et d'autre part les futures grandes lignes des projets de recherche dans lequels je suis impliqué.

# Première partie

# Cartographie et robustesse de la topologie de l'Internet

# Chapitre 1

# Cartographie distribuée de la topologie de l'Internet

## 1.1 Introduction

Durant mon travail de doctorat, j'ai dû de nombreuses fois avoir recours à des topologies réseaux afin d'effectuer des simulations de protocoles [MP01a, MP01e, MP02c]. Dans un premier temps, j'ai développé un générateur de graphes basé sur la formule de Waxman [Wax88], mais le besoin d'avoir des topologies ressemblant à celle de l'Internet lui-même m'a poussé à utiliser des cartes créées par d'autres chercheurs tels que Ramesh Govindan, Hal Burch et Bill Cheswick. J'ai développé un outil nommé *network manipulator* ou *nem* [Mag02] en abrégé, capable d'analyser en détail les caractéristiques de ces cartes et d'en extraire des sous-graphes de taille raisonnable [MP02b]. Ces sous-graphes sont créés de manière à conserver les propriétés topologiques des cartes afin d'être utilisables dans des logiciels de simulation de protocoles tels que ns-2. Malheureusement les cartes étaient "anonymées", c'est à dire que les adresses IP originales étaient remplacées par des numéros. De plus aucune information entre niveau routeur et niveau système autonome (AS) n'était fournie et enfin ces cartes se périmaient car de nouvelles cartes n'étaient pas régulièrement créées par les chercheurs précédemment cités. Tous ces facteurs m'ont poussé fin 2002 à lancer un projet de cartographie de l'Internet [MH] qui a débouché sur la création de l'outil nommé *network cartographer* ou *nec* en abrégé. Il nous permet de créér des cartes de l'Internet contenant des informations jusqu'à présent inaccessibles. Ce chapitre présente cet outil en détail ainsi qu'une analyse des résultats qu'il nous a permis d'obtenir.

Il n'est pas inutile de rappeler que l'étude de la topologie d'Internet permet de découvrir des propriétés pouvant être exploitées par des protocoles réseaux ou applicatifs afin d'améliorer leurs performances. La cartographie de l'Internet nous fournit donc un aperçu de la topologie très complexe de celui-ci et elle est par conséquent très profitable aux logiciels de simulation (*e.g.*, ns-2 [pro], GloMoSim [Par] et OPNET [OPN]) de protocoles réseaux et en particulier de ceux destinés à être déployés dans l'Internet [CDZ97, FP01]. Elle est surtout très utile à la simulation des protocoles situés au niveau de la couche réseau (couche 3 du modèle OSI) tels que les protocoles de routage, les protocoles multipoints ou encore les protocoles de gestion de la qualité de service. Malheureusement la cartographie d'Internet souffre de deux problèmes majeurs. D'abord elle prend du temps à être réalisée et ensuite elle souffre souvent d'incomplétude. Pour pallier ces deux handicaps, nous avons développé un logiciel de cartographie dont l'objectif est de cartographier le coeur d'Internet avec un maximum de précision en un minimum de temps. Dans ce chapitre nous décrivons notre logiciel et évaluons ses performances par rapport à un logiciel de cartographie existant qui est en libre disposition pour la recherche. Nous évaluons la complétude des cartes de routeurs en effectuant des recouvrements avec une topologie de sys-

tèmes autonomes. Nous évaluons aussi son utilisation dans le cadre d'IPv6 [DH98] ainsi que l'évolution de ses performances en IPv4 [Pos81b] en fonction des paramètres d'entrées tels que le nombre de traceurs utilisés. Enfin nous proposons une étude détaillée des délais des chemins et des liens dans l'Internet.

Dans tout ce chapitre, nous définissons la cartographie d'Internet comme étant l'ensemble des techniques nous permettant d'établir le graphe d'interconnexion des noeuds de niveau 3 du modèle OSI. Le niveau 3 de ce modèle est celui dans lequel les noeuds sont identifiés au niveau global (*i.e.,* chaque identifiant est unique au monde) grâce au protocole IP. Rappelons que la topologie d'un réseau est liée au niveau étudié et que par conséquent la topologie du niveau 3 de l'Internet est différente de celle de son niveau 1 (physique). Nous nous intéressons au niveau 3 car c'est à ce niveau logique que nous pouvons étudier les chemins empruntés par les paquets de données circulant entre les noeuds de l'Internet. Au niveau 3, les noeuds sont des hôtes (équipements terminaux) ou bien des routeurs (équipements d'interconnexion). La cartographie ne s'intéresse qu'à ces derniers. Nous cherchons donc à recenser les routeurs d'Internet et les connexions qui les relient (de niveau 3 donc IP). En considérant les routeurs comme des sommets et les liens logiques IP comme des arêtes, la cartographie de l'Internet nous donne donc un graphe. Nous pouvons ensuite analyser les propriétés de ce graphe avec les outils de la théorie des graphes. A noter que l'Internet possède deux niveaux d'interconnexion au niveau de la couche OSI 3. En effet les routeurs sont regroupés dans des entités nommées systèmes autonomes ou AS [HB96] (pour Autonomous Systems en anglais) interconnectées entre elles par des liens logiques créés par le protocole BGP [RL95, RG95] (on parle alors de lien BGP). Il existe donc aussi un graphe des AS se superposant au graphe des routeurs. Faire correspondre les routeurs à leurs AS respectifs s'appelle créer un recouvrement. Notre logiciel utilise la célèbre commande *traceroute* afin de produire des chemins entre des noeuds de l'Internet afin d'en inférer leur schéma d'interconnexion. Nous décrivons dans ce chapitre ce logiciel de cartographie rapide de l'Internet ainsi que notre méthode de construction de recouvrements routeurs–systèmes autonomes afin d'évaluer la complétude des cartes de routeurs.

Le chapitre est organisé en huit sections. La section 1.2 présente les travaux similaires à notre travail de recherche ainsi que le logiciel que nous utilisons pour pouvoir effectuer une comparaison avec notre logiciel. La section 1.3 présente le fonctionnement de notre logiciel de cartographie. La section 1.4 présente les cartes collectées par les deux logiciels ainsi que la carte des systèmes autonomes utilisée pour permettre de créer les recouvrements. La section 1.5 expose les limitations liées à l'utilisation des cartes. La section 1.6 décrit la méthodologie de construction d'un recouvrement. La section 1.7 présente des résultats d'analyse concernant la précision des cartes de routeurs et donc de l'efficacité des logiciels de cartographie. La section 1.8 présente les techniques utilisées pour cartographier le réseau IPv6 ainsi que des premiers résultats comparatifs entre IPv6 et IPv4. Enfin la section 1.9 présente une étude détaillée sur les longueurs et les délais des chemins ainsi que sur les délais des liens d'une carte IPv4 étendue.

## 1.2 Contexte

La cartographie de l'Internet fait l'objet de recherches depuis plusieurs années. En 1995, Pansiot *et al.* ont créé une carte de routeurs avec des *traceroute* provenant d'une machine unique vers environ 5000 destinations [PG98]. Ils ont utilisé le routage par source (*source routing*) pour découvrir les liens transverses. En 1999, Burch *et al.* ont utilisé des tables provenant du protocole de routage inter-domaine BGP (Border Gateway Protocol) pour trouver des préfixes destinations [BC99]. Ils ont aussi effectué des *traceroute* à partir d'une seule machine mais ils ont utilisé des tunnels vers d'autres machines obtenant ainsi un effet similaire au *source rou-*

*ting*. Toujours en 1999, Govindan *et al.* ont créé un logiciel de cartographie nommé Mercator qui utilise une méthode aléatoire de sélection d'adresses destinations [GT00]. C'est ce logiciel que nous utilisons pour le comparer à *nec* car c'est le seul logiciel de cartographie disponible gratuitement pour la recherche. Il fonctionne à partir d'une machine unique en envoyant des paquets UDP (User Datagram Protocol [Pos80], envoi en mode non-connecté) avec un TTL (Time To Live, champ d'un paquet IP indiquant combien ce dernier a traversé de routeurs) croissant (donc similaire à l'utilisation d'une commande *traceroute*). Il utilise aussi le *source routing* pour découvrir les liens transverses. En 2002, Spring *et al.* ont créé un logiciel nommé Rocketfuel [SMW02]. Ce logiciel fonctionne de manière similaire à *nec* (*i.e.*, il interroge des serveurs *traceroute*) mais il ne permet de cartographier qu'un seul ISP (Internet Service Provider, fournisseur d'accès) à la fois. De plus il n'est pas disponible, par contre le logiciel de résolution des alias IP utilisé par Rocketfuel et nommé Ally est disponible et nous l'utilisons avec *nec*. Récemment une version améliorée de Rocketfuel nommée Scriptroute a été proposée dans [SWA03]. A noter que des services de mesures de delais dans l'Internet ont été définis dans NIMI [PMAM98] et IDMaps [FJP+99, FJJ+01] et apportent un éclairage supplémentaire à l'aspect dynamique de la cartographie du réseau. Des approches plus théoriques concernant la cartographie sont aussi proposées par Shavitt *et al.* dans [SSWY01] (approche algébrique), par Gkantsidis *et al.* dans [GMZ03] (analyse spectrale) et par Li *et al.* dans [LAWD04]. Les biais liés aux échantillonnages de la topologie de l'Internet sont étudiés par Lakhina *et al.* dans [LBCX03].

Concernant notre méthode de création de recouvrement, celle-ci est différente et innovante par rapport aux méthodes utilisées dans [TGSE01] et dans [CJW01] car nous faisons correspondre les routeurs découverts par Mercator ou *nec* aux systèmes autonomes trouvés dans la table BGP de *route-views* grâce à l'utilisation des interfaces IP et des préfixes BGP. Par conséquent nous n'avons pas besoin de générer le graphe des AS par un algorithme de regroupement tel que celui présenté dans [TGSE01] et nous évitons les erreurs potentielles introduites par les cas où beaucoup de groupes disjoints de routeurs appartenant au même AS doivent être réassignés. De plus notre méthode de création, contrairement aux méthodes précédentes, présente l'avantage de pouvoir évaluer le taux de recouvrement (pour les AS et les connexions BGP) de la carte de niveau AS par celle de niveau routeur. Cela nous permet d'estimer la qualité de la complétude de la carte de niveau routeur. Nous définissons la complétude comme étant un rapport variant entre 0 et 1 (ou exprimé en pourcentage) et indiquant quelle est la proportion des AS (resp. des connexions BGP) contenant au moins 1 routeur (resp. 1 lien IP) parmi tous ceux de la carte des AS.

## 1.3 *network cartographer*

Notre logiciel de cartographie de l'Internet se nomme *network cartographer* ou *nec* en abrégé. *nec* se lance depuis un point donné de l'Internet (typiquement sur la machine d'un utilisateur) et interroge simultanément un ensemble de serveurs *traceroute* vers un ensemble de destinations stockées localement. Le cardinal de ces ensembles pouvant potentiellement être très élevé, un mécanisme de contrôle de ressources au niveau local et distant est mis en oeuvre afin d'éviter que les collectes ne soient prises pour des attaques. A un instant donné, le nombre de requêtes en cours est limité. Typiquement, sur un ensemble de 76 serveurs *traceroute* et environ 6000 adresses destinations nous avons décidé de limiter le nombre local de requêtes simultanées à 20. Le nombre de requêtes distantes simultanées provenant de la machine sur laquelle s'exécute *nec* est limité à 1 pour ne pas surcharger un serveur plus que ne le ferait une requête manuelle faite via un navigateur web. Dans la suite du texte nous appellerons la limite locale $x$.

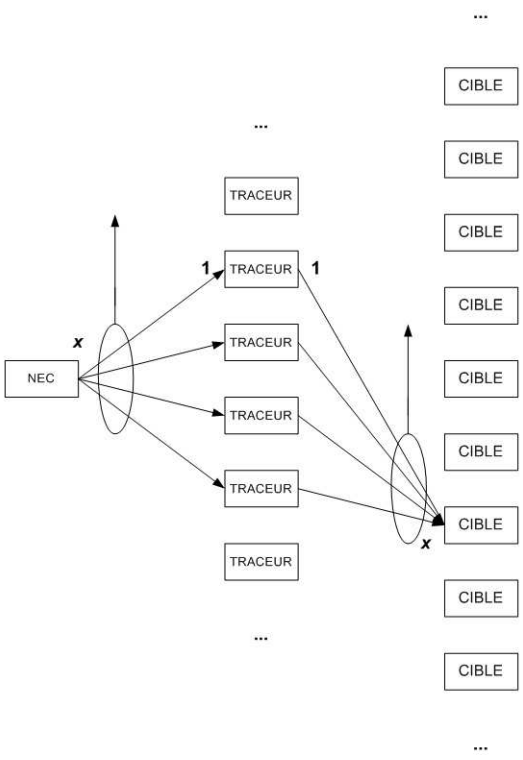


FIG. 1.1 – Fonctionnement de la collecte.

De manière plus détaillée, Le fonctionnement est le suivant : à l'initialisation, on ouvre $x$ connexions HTTP vers $x$ serveurs *traceroute* distincts auxquels on soumet une requête vers une même destination $y$ comme cela est montré sur la figure 1.1. La valeur de $x$ nous permet donc de contrôler le nombre de requêtes simultanées reçues par la destination $y$ et donc d'éviter le déclenchement de sa détection d'attaques. A noter que les $x$ requêtes reçues par $y$ proviennent de machines différentes ce qui réduit le risque qu'elles soient perçues comme des attaques. On attend ensuite les résultats, sachant qu'on ne peut pas prévoir l'ordre dans lequel ils arriveront et qu'ils peuvent être incomplets en raison des propriétés inhérentes de l'Internet (*i.e.*, filtrage des paquets ICMP [Pos81a] sur certains routeurs).

A réception d'une ou plusieurs réponses, on les enregistre pour les traiter *offline*. Si une réponse est complète, on ferme la connexion pour en ouvrir une autre et envoyer une nouvelle requête soit vers une adresse destination égale à $y$ à un serveur non encore sollicité pour cette adresse, soit vers une adresse destination différente de $y$ à un serveur ayant déjà traité $y$. Si la réponse est incomplète, on récupère les données déjà envoyées et on se remet en attente en se basant sur les mécanismes qu'offre TCP [Pos81c] pour gérer les erreurs. On ferme la connexion si une requête *traceroute* est filtrée (*i.e.*, si on reçoit une succession de symboles *).

Avant toute ouverture de nouvelle connexion, on vérifie nos deux critères limitants : d'une part qu'une requête n'est pas déjà en cours sur le serveur *traceroute* qu'on aimerait interroger et d'autre part que le nombre de requêtes en cours est inférieur ou égal à $x$. *nec* stocke des informations d'états afin de pouvoir reprendre une cartographie interrompue sans avoir à recommencer depuis le début.

Le choix des adresses de destination $y$ (*i.e.*, cibles des *traceroute*) est basé sur le système autonome auquel $y$ appartient. Les adresses destinations appartiennent toutes à un AS distinct afin d'obtenir une distribution topologique optimale des cibles et par conséquent une carte la plus précise possible en utilisant le moins de requêtes possible.

*nec* est écrit en C et en C++. Son code source est disponible gratuitement pour toute utilisation non commerciale. Il peut être téléchargé à l'adresse [MH]. Cependant il n'est pas encore terminé et nous souhaitons y implémenter de nombreuses fonctionnalités supplémentaires. Il est mis à disposition de la communauté de recherche en réseaux essentiellement à des fins d'évaluation.

## 1.4 Collecte des cartes d'Internet

Notre objectif est de cartographier rapidement le coeur de l'Internet. Les campagnes de cartographie des travaux précédents [PG98, BC99, GT00] s'effectuent en général sur plusieurs semaines voire plusieurs mois. Ici nous effectuons une campagne de mesures dont la durée est de 10 jours. Bien que cela paraisse déjà important, il ne faut pas oublier qu'une campagne plus rapide pourrait porter atteinte au service fourni par les serveurs *traceroute* (dans le cas de *nec*) et/ou pourrait être interprétée comme une attaque de déni de service distribuée vis-à-vis des adresses de destination (Mercator et *nec*) ou encore comme une tentative d'espionnage de réseau (Mercator).

De plus nous parlons d'une cartographie du *coeur* de l'Internet parce que nous sommes limités aux adresses IP publiques [HKC+96] (on ne peut pas cartographier les adressages privés [RMK+96] accessibles à travers des boîtes NAT (Network Address Translation [SE01])) et nous sommes dépendant du traitement des paquets ICMP par les routeurs. Certains, suite à une sollication, ne répondent pas par renvoi de paquets ICMP, d'autres filtrent carrément tout paquet ICMP. Ce dernier comportement est très fréquent dans les réseaux des fournisseurs d'accès Internet grand public, mais très rare chez les gros opérateurs. Il en résulte que l'on cartographie plutôt le coeur que les extrémités du réseau (*i.e.*, les *traceroute* n'atteignent pas toujours leurs cibles). Nous avons donc fait fonctionner Mercator et *nec* pendant 10 jours et avons comparé les cartes de routeurs ainsi obtenues.

Pour créer les recouvrements nécessaires à l'analyse de la complétude, nous avons besoin des cartes des routeurs de L'Internet et d'une carte des systèmes autonomes. Nous avons collecté les 2 cartes de niveau routeur à partir d'une machine localisée dans notre laboratoire (le LSIIT, situé à Illkirch, France) en utilisant le logiciel Mercator développé par Govindan *et al.* et décrit dans [GT00] d'une part et notre logiciel *nec* d'autre part. Ces cartes contiennent les adresses IP des interfaces des routeurs. La résolution des alias IP (*i.e.*, l'identification des interfaces appartenant à un même routeur) est une étape importante de la cartographie.

Dans Mercator, celle-ci est réalisée parallèlement à la cartographie par une technique initialement proposée par Pansiot *et al.* dans [PG98]. Le logiciel envoie un paquet UDP vers un port fermé à chacune des 2 interfaces suspectées d'appartenir au même routeur. Si les adresses sources des paquets ICMP de réponse sont identiques, alors il y a une forte chance pour que ces 2 interfaces appartiennent au même routeur.

Dans *nec*, la résolution des alias s'effectue un peu en décalé par un logiciel nommé Ally. Une liste des interfaces susceptibles d'être des alias est créée par *nec*. Sont suspectes, toutes les adresses IP ayant un suffixe DNS identique (lorsqu'il existe), se trouvant dans le même AS et découvertes par des traces différentes. Ally est décrit dans [SMW02] et fonctionne de la façon suivante. Il envoie un paquet UDP vers un port fermé à chacune des 2 interfaces suspectées d'appartenir au même routeur. Si les valeurs du champ identificateur IP (IP identifier en anglais) des paquets ICMP de réponse sont très proches, alors il envoie un troisième paquet à la première interface. Si la troisième valeur est proche des deux premières et que les trois respectent un ordre croissant alors il y a une forte chance pour que ces 2 interfaces appartiennent au même routeur (*i.e.*, l'identificateur IP est produit à partir du même compteur donc très vraisemblablement du même routeur).

TAB. 1.1 – Cartes Internet utilisées pour l'analyse.

| Logiciel ou origine | Nb d'interfaces ou de préfixes | Nb de routeurs ou d'AS | Nb de liens | Période |
|---|---|---|---|---|
| Mercator | 58119 | 55425 | 65237 | 7-17/4/2003 |
| *nec* | 50123 | 47055 | 119909 | 19-29/4/2003 |
| *route-views* | 130815 | 15129 | 31378 | 1/4/2003 |

Nous avons aussi récupéré une carte de niveau AS produite à partir d'une image de la table de routage du routeur BGP observateur *route-views* [Uni] au début d'avril 2003. Le tableau 1.1 contient des informations générales sur ces cartes. On remarque que Mercator a détecté 4,6% d'alias et Ally 6,1%. Sans préjuger de l'efficacité de chacune des deux méthodes de résolution d'alias, on peut noter que le taux d'alias détectés se situe autour de 5% ce qui est presque le tiers de la valeur observée dans [BBBC01].

## 1.5 Limitations inhérentes aux cartes

Avant de présenter la construction et l'analyse des recouvrements routeurs–AS, deux remarques doivent être formulées concernant les limitations majeures de notre étude :

- Les cartes Internet que nous utilisons sont incomplètes. La carte de niveau routeur créée par Mercator est incomplète pour les raisons présentées dans [GT00] et dans la section 1.7. Celle créée par *nec* est limitée par les défauts inhérents à la commande *traceroute*. La carte de niveau AS provenant des données du routeur BGP *route-views* est aussi incomplète particulièrement en ce qui concerne le nombre de connexions BGP. En effet, Chang *et al.* ont montré dans [CGJ+02] que l'utilisation de sources additionnelles d'information de routage de niveau AS permet d'augmenter le nombre de connexions BGP détectées jusqu'à 47,7%. Cela a un impact direct sur la connectivité du réseau d'AS. Notre but n'est pas d'obtenir une précision absolue dans la cartographie mais plutôt et surtout une précision relative la plus élevée possible par rapport aux outils et aux données dont nous disposons pour la réaliser.
- Les cartes de niveau routeur produites par Mercator ou *nec* (ou n'importe quel autre outil de mesure topologique de niveau IP) ne doivent pas être vues comme contenant uniquement des routeurs réels et des liens point-à-point physiques mais simplement comme des cartes contenant des noeuds IP et des liens IP. D'ailleurs beaucoup de ces noeuds et liens IP sont purement virtuels et donc n'existent pas physiquement. Cela s'explique par le fait que, particulièrement dans les coeurs de réseaux, les technologies de commutation de circuits (virtuels) telles que Frame Relay, ATM ou MPLS [RVC01], peuvent non seulement masquer leur propre topologie à la couche IP mais peuvent aussi émuler des routeurs ou des liens IP selon une topologie complétement différente de la leur. Bien que nous utilisions le terme "routeur" dans la suite de ce chapitre pour des raisons de simplification, cette caractéristique ne doit pas être oubliée.

## 1.6 Construction des recouvrements

Nous proposons dans cette section une méthode de construction d'un recouvrement (*overlay* en anglais) routeurs–systèmes autonomes. Nous avons implémenté notre méthode dans notre

logiciel *nem* [Mag] dont le code est disponible sur le Web. Nous espérons que de tels recouvrements seront utilisés par la communauté de recherche concernée par les protocoles de l'Internet. En effet, ces informations peuvent être utilisées pour optimiser le placement intra ou inter domaine de serveurs mandataires (*proxy* en anglais), de membres d'arbres multipoints applicatifs, etc. Cependant dans ce chapitre, nous souhaitons utiliser les recouvrements pour évaluer la complétude des cartes de routeurs.

Nous construisons un recouvrement de manière à lier la topologie des routeurs à celle des AS. Nous utilisons le contenu d'une table de routage BGP créée le 1 avril 2003 par *route-views* pour fabriquer le recouvrement. Cette table contient 15129 AS. Pour la construction du recouvrement, nous associons chaque préfixe réseau trouvé dans la table à l'AS qui le déclare (*i.e.*, l'AS situé le plus à droite du chemin d'AS). Cet AS n'est pas nécessairement l'AS d'origne du préfixe car ce dernier peut être masqué par l'aggrégation des chemins d'AS [RL95]. Des erreurs peuvent donc être introduites à ce niveau. Lorsqu'un préfixe peut être associé à plus d'un AS (suite à une erreur de protocole ou de base de données), nous conservons le premier AS ayant le drapeau "i" (*i.e.*, interne) positionné si il existe, sinon nous conservons le premier AS trouvé (quelques cas dans notre table, aussi sources d'erreurs). La table contient 130815 préfixes réseaux (ce qui est consistant avec les résultats trouvés par Bu *et al.* dans [BGT02]).

Puis nous utilisons nos informations sur la topologie IP collectées en utilisant le logiciel Mercator ou notre logiciel *nec* pour construire une carte de niveau routeur de l'Internet. Pour chaque interface, nous recherchons le plus long préfixe réseau correspondant et nous associons l'AS source (ou celui qui le déclare) possédant ce préfixe réseau à l'interface.

Contrairement à la méthode utilisée dans [TGSE01], nous n'avons pas utilisé les IRR (Internet Routing Registries, enregistrement des informations de routage) comme sources additionnelles d'information car elles ne sont pas assez précises pour notre utilisation. D'ailleurs Chen *et al.* ont montré dans [CCG+02] que presque 62% des enregistrements de la base de données RIPE (Réseaux IP Européens, base de données contenant en particulier des informations sur les plans d'adressage des réseaux européens) sont égaux à néant (*void* en anglais) ou obsolètes malgré le fait que RIPE soit activement maintenue à jour. Parmi les interfaces non résolues, beaucoup d'entre elles appartiennent à des AS (comme peuvent le montrer quelques requêtes aux IRR) mais certaines d'entre elles telles que les 188.1.x.x du réseau de recherche allemand (nommé DFN) sont configurées de sorte à n'appartenir à aucun AS. Nous marquons toutes les interfaces non résolues comme appartenant à l'AS de numéro 0. Nous définissons l'AS de numéro 0 comme suit : "une adresse IP associée à l'AS de numéro 0 n'appartient à aucun AS". En dépit du nombre important d'adresses privées trouvées dans [TGSE01], nous n'avons pas trouvé d'adresses privées dans nos données de niveau IP produites par Mercator. Ces adresses sont très probablement filtrées sur le chemin de retour à notre machine.

Finalement pour chaque routeur, nous définissons son adresse IP et recherchons son numéro d'AS en utilisant les interfaces associées à chaque routeur par Mercator ou *nec*. Si le routeur n'a qu'une interface, le processus est simple : nous assignons l'adresse IP et le numéro d'AS de l'interface au routeur. Si le routeur possède plusieurs interfaces, nous recherchons l'interface ayant la plus petite adresse IP et un numéro d'AS différent de 0 et nous assignons cette adresse IP et son numéro d'AS à ce routeur (cette méthode peut produire des erreurs). A la fin du processus, chaque routeur possède une adresse IP par défaut ainsi qu'un numéro d'AS (qui peut être égal à 0). Nous lions alors les routeurs à leurs AS respectifs dans notre logiciel *nem* afin de créer la structure de données du recouvrement. Nous utilisons le recouvrement pour déterminer quels routeurs pourraient être des routeurs BGP potentiels. Pour chaque routeur, nous regardons ses routeurs voisins et si l'un ou plusieurs d'entre eux se trouve dans un AS ayant un numéro d'AS non nul et différent de celui du routeur initial alors nous marquons ce dernier comme étant un routeur BGP.

Notons que notre méthode de création de recouvrement est très différente des méthodes utilisées dans [TGSE01] et dans [CJW01] car nous faisons directement correspondre les routeurs aux AS trouvés dans la table BGP grâce aux interfaces IP et aux préfixes BGP. Ainsi nous n'avons pas besoin de générer le graphe d'AS par un algorithme d'effondrement (collapse en anglais) tel que celui présenté dans [TGSE01] et nous évitons ainsi les erreurs potentielles créées par les cas où de nombreux groupements disjoints de noeuds appartenant au même AS doivent être réassignés.

## 1.7 Analyse de la qualité des cartes

Dans cette section, nous étudions les propriétés des cartes collectées par Mercator et *nec* afin d'évaluer leurs qualités.

### 1.7.1 Propriétés générales

Le tableau 1.2 contient les valeurs des principales propriétés des cartes qui, nous le rappelons, ont été collectés en 10 jours. Premier point, *nec* a découvert environ 15% moins de routeurs que Mercator. Une augmentation de la fréquence des requêtes et/ou du nombre de serveurs *traceroute* nous permettrait sans doute de combler ce retard. Après construction des recouvrements, nous trouvons que 12% des routeurs présents dans la topologie IP de Mercator peuvent être définis comme des routeurs BGP selon notre méthode de résolution présentée à la section 1.6, alors que ce taux est de presque 60% pour la topologie recueillie par *nec*. Nous ne possédons pas actuellement d'informations permettant de vérifier ce chiffre par rapport à sa valeur réelle dans l'Internet, mais cela signifie que notre méthode de distribution des cibles (*i.e.*, adresses de destination des *traceroute*) qui consistent à en choisir une par AS, fait découvrir beaucoup de liens IP inter-domaines. Deuxième point et certainement le plus important de tous, *nec* a découvert presque 84% de liens de plus que Mercator ! Ce résultat recherché par notre logiciel est très positif. Il illustre l'avantage de l'utilisation de points de mesures multiples et distribués par rapport aux techniques de *source-routing* qui sont très insuffisantes pour découvrir des liens transverses en peu de temps. Il s'ensuit un degré moyen de la carte *nec* plus de 2 fois supérieur à celui de Mercator (donc une beaucoup plus forte densité de liens redondants). Le pourcentage de la partie maillée (*i.e.*, le graphe amputé de tous ses arbres) des cartes est éloquent à cet égard. L'Internet est beaucoup plus maillé que ne le faisait apparaître les travaux précédents (85% des routeurs de la carte *nec* sont dans le maillage). En tout cas la valeur du proucentage de la partie maillée ne doit pas être utilisé comme indicateur (*i.e.*, invariant d'une propriété du graphe). Toujours grâce aux recouvrements avec la carte d'AS, nous pouvons calculer la complétude des cartes de routeurs. La complétude des AS donne le pourcentage des AS qui contiennent au moins 1 routeur (idéalement, il devrait être de 100%). La complétude des connexions BGP donne le pourcentage des connexions BGP qui contiennent au moins 1 lien IP (il devrait être de 100% comme pour les AS). Ces indicateurs sont relativement grossiers mais suffisants pour montrer les limites des cartes de routeurs. On peut voir dans ces valeurs le fort biais de Mercator qui ne remplit que 12% des AS et 6% des connexions. *nec* par contre atteint des valeurs très supérieures avec des taux de remplissage respectifs de 35% et 28%. Bien que cela soit loin d'être exhaustif, le progrès apporté par *nec* par rapport aux techniques de cartographie antérieures est indéniable.

TAB. 1.2 – Propriétés des cartes collectées.

| Propriété | carte Mercator | Carte *nec* | Ecart |
|---|---|---|---|
| Routeurs | 55425 | 47055 | -15.1% |
| Routeurs BGP | 12.0% | 59.8% | +398.3% |
| Liens IP | 65237 | 119909 | +83.8% |
| Degré moyen | 2.35 | 5.09 | +116.6% |
| Partie maillée | 24.8% | 85.7% | +245.5% |
| Complétude AS | 11.7% | 35.0 % | +199.1% |
| Complétude connexion BGP | 6.1% | 27.8% | +355.7% |

TAB. 1.3 – Valeurs des principaux indicateurs.

| Indicateur | carte Mercator | Carte *nec* |
|---|---|---|
| CC du degré | 0.98 | 0.96 |
| Exposant du degré | -2.80 | -2.36 |
| CC du rang | 0.96 | 0.95 |
| Exposant du rang | -0.65 | -0.73 |
| CC de la taille des arbres | 0.98 | 0.97 |
| Exposant de la taille des arbres | -2.12 | -3.92 |
| CC du rang des arbres | 0.98 | 0.96 |
| Exposant du rang des arbres | -0.83 | -0.35 |
| CC du nb de plus courts chemins | 0.67 | 0.95 |
| Exposant du nb de plus courts chemins | -1.38 | -1.51 |
| CC du rang des paires | 0.99 | 0.99 |
| Exposant du rang des paires | -0.58 | -0.58 |
| Distance moyenne | 16.3 | 9.0 |
| Excentricité | 30.9 | 18.9 |
| Rayon | 23 | 15 |
| Diamètre | 44 | 28 |

### 1.7.2 Lois puissances et distances

Les indicateurs topologiques sont des outils intéressants pour évaluer la similitude d'un graphe avec la topologie de l'Internet. Les indicateurs sont définis dans nos travaux antérieurs [MP02a]. Les lois puissances ont été étudiées originellement par Faloutsos *et al.* dans [FFF99] et Medina *et al.* dans [MMB00]. Le tableau 1.3 contient les valeurs des principaux indicateurs de nos 2 cartes de routeurs. Ils sont proches en général (des variations plus fortes sur les exposants sont usuelles et difficiles à interpreter), excepté ceux liés au nombre de plus courts chemins et ceux concernant les distances.

On peut voir dans le tableau 1.3 que le coefficient de corrélation (CC) du nombre de plus courts chemins (illustrant la présence de la loi puissance décrite ci-dessous) n'est pas vérifié par la carte de Mercator (*i.e.*, le CC est inférieur à la valeur seuil de 0.95). En effet dans [MP01b] nous étudions la distribution du nombre de plus courts chemins distincts ($pccd$) et nous définissons la fréquence d'un nombre de $pccd$, $f_n$, comme étant le nombre de paires ayant une valeur de $pccd$ de $n$ (*i.e.*, les paires ont exactement $n$ plus courts chemins distincts).

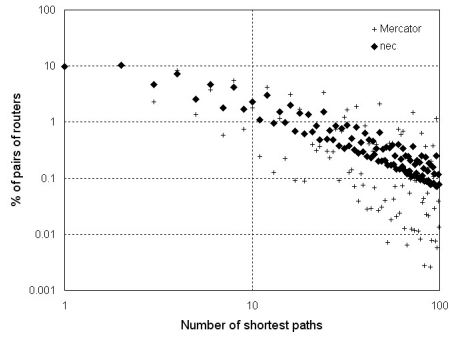


FIG. 1.2 – Distribution du nombre de plus courts chemins.

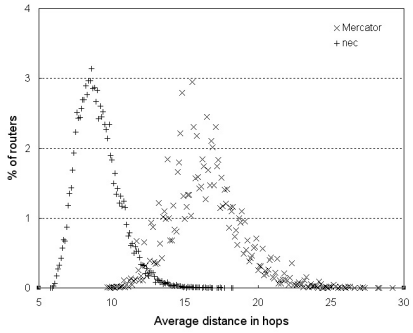


FIG. 1.3 – Distribution de la distance moyenne des routeurs entre eux.

**Loi 1** ***5ème loi puissance (exposant du nombre de plus courts chemins distincts)*** *La fréquence, $f_n$, d'un nombre donné de plus courts chemins distincts entre une paire de sommets, $n$, est proportionnelle au nombre de plus courts chemins distincts $n$ à la puissance d'une constante, $\mathcal{N}$ :*

$$f_n \propto n^{\mathcal{N}} \tag{1.1}$$

**Définition 1** *Nous définissons l'exposant du nombre de plus courts chemins distincts, $\mathcal{N}$, comme étant égal à la pente du tracé de la fréquence du nombre de plus courts chemins distincts en fonction du nombre de plus courts chemins distincts sur une échelle log-log.*

La 5ème loi puissance dérivée de la distribution du nombre de plus courts chemins distincts est liée à la quantité d'arêtes redondantes impliquées dans les plus courts chemins. Elle est vérifiée dans les cartes de routeurs et d'AS de l'Internet. Cette propriété non validée par la carte collectée par Mercator illustre le manque de précision de Mercator. La carte de *nec* par contre vérifie cette propriété cruciale grâce au nombre important de liens découverts. La figure 1.2 illustre la distribution du pourcentage de paires de routeurs ayant de 1 à 100 plus courts chemins. Les points doivent être alignés. On remarque la forte dissémination des points de Mercator alors que ceux de *nec* sont nettement plus rapprochés.

Concernant les distances, exprimées en nombre de sauts (*i.e.*, en nombre de liens traversés), la carte de Mercator à des propriétés très différentes des grandes cartes de routeurs alors que celle de *nec* possède pratiquement les mêmes valeurs que celles mesurées dans des cartes de plus de 200k routeurs [MP02b]. Le manque de liens de la carte de Mercator entraîne un étirement des distances entre les routeurs ce qui donne des valeurs augmentéees de plus de 50% par rapport aux valeurs usuelles pour tous les indicateurs de distance. La figure 1.3 illustre la distribution du pourcentage de routeurs ayant une distance moyenne donnée par rapport aux autres (arrondie au dixième). Les points de *nec* forment une distribution normale très nette et bien centrée autour de 9 sauts contrairement à ceux de Mercator qui sont éparpillés autour de 17 sauts.

### 1.7.3 Présence de distributions à queue lourde

Pour vérifier la précision de nos recouvrements, nous étudions la distribution de la taille des AS. La taille d'un AS est égal au nombre de routeurs contenus dans cet AS. Nous avons montré précédemment que le taux de complétude est de 12% pour Mercator et de 35% pour *nec*. Cela montre que nos cartes de niveau routeur sont encore loin de la complétude totale. Nous cherchons les distributions à queue lourde trouvées dans les recouvrements d'Internet par Tangmunarunkit *et al.* dans [TDG+01]. Une méthode éprouvée et utilisée dans [TDG+01] et [BT02, BDPT02] pour réaliser cela consiste à tracer la fonction de distribution cumulative

complementaire de la taille des AS (CCDF en anglais) sur une double échelle logarithmique. Les figures 1.4 et 1.5 présentent les CCDF des distributions de la taille des AS pour les cartes de Mercator et de *nec* et celles-ci présentent clairement le caractère d'une distribution à queue lourde. On peut noter que la CCDF de Mercator est légèrement bombée alors que celle de *nec* est parfaitement droite ce qui dénote une parfaite corrélation. Par contre la queue de la distribution de *nec* est brisée à partir d'une taille de 1000. Cet effet de bord est probablement dû à un manque de données causé par la courte période de mesure. Ce phénomène apparait aussi pour Mercator mais il est un peu masqué par la courbure du tracé.

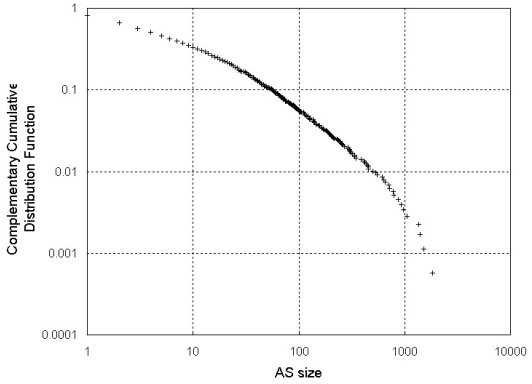


FIG. 1.4 – Distr. cum. comp. de la taille des AS vus par Mercator.

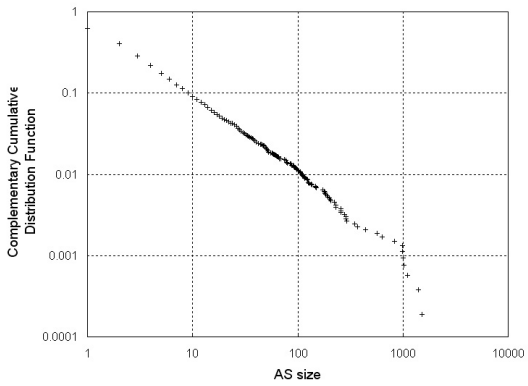


FIG. 1.5 – Distr. cum. comp. de la taille des AS vus par nec.

Grâce à notre technique de création de recouvrement, nous pouvons étudier les tailles des connexions BGP. La taille d'une connexion BGP est égale au nombre de liens IP qui vont de n'importe quel routeur d'un AS donné vers n'importe quel routeur d'un AS donné différent du précédent. De façon similaire aux AS, seulement une partie des connexions contiennent un ou plusieurs liens IP. Ceci nous donne un ratio du nombre de connexions remplies (*i.e.*, ayant une taille $\geq 1$) *vs* le nombre total de connexions de 6% pour Mercator et 28% pour *nec* ce qui est plus faible que les taux de remplissage des AS. Une fois de plus cela tend à montrer que la carte de Mercator manque un nombre significatif de liens IP. Les figures 1.6 et 1.7 présentent sur une double échelle logarithmique les CCDF des distributions de la taille des connexions BGP et celles-ci révèlent des queues lourdes. Nous sommes, à notre connaissance, les premiers à présenter ce type de résultats (intuitivement prévisible).

Nous pouvons en conclure que bien que nos recouvrements routeur–AS soient incomplets, ils contiennent beaucoup de distributions à queue lourde typiquement trouvées au niveau macroscopique de la topologie de l'Internet [TDG+01]. Sur ce point, les cartes de Mercator et de *nec* exhibent des distributions similaires, bien que celles de *nec* soient plus précises comme le montrent les coefficients de corrélation des régressions linéaires.

## 1.8 Cartographie de l'Internet IPv6

Dans cette section, nous expliquons comment nous avons collecté une carte globale de l'Internet IPv6 actuel à l'aide de notre logiciel de cartographie et nous décrivons quelques propriétés topologiques de cette carte en comparaison avec celles de l'Internet IPv4. A ce jour nous sommes à notre connaissance les premiers à présenter un tel résultat. Celui-ci permet de se faire une idée du déploiement d'IPv6 dans l'Internet actuel.

La sélection des adresses IPv6 (*i.e.*, des cibles pour les requêtes *traceroute*) est basée sur leur préfixe. La campagne de collecte a été effectuée en deux étapes. Dans la première étape,

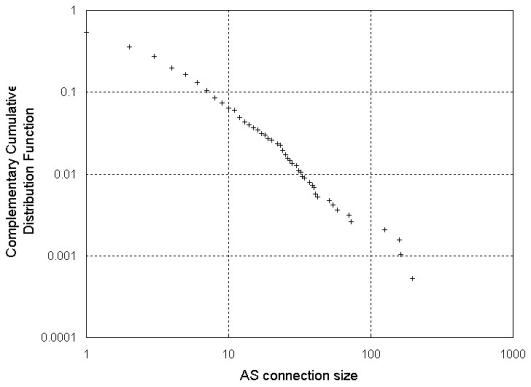


FIG. 1.6 – Distr. cum. comp. de la taille des connexions vues par Mercator.

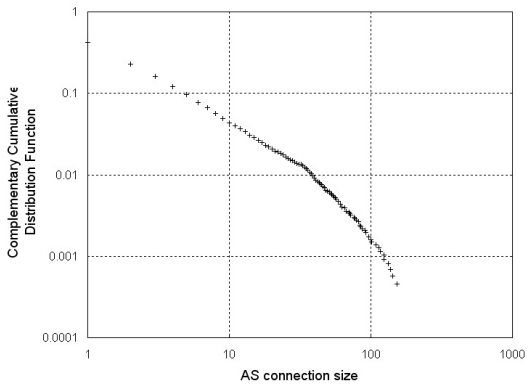


FIG. 1.7 – Distr. cum. comp. de la taille des connexions vues par nec.

toutes les adresses IPv6 appartiennent à un préfixe IPv6 distinct récupéré dans les bases de données [IAN]. Ces adresses de destinations ont été générées en concaténant le préfixe IPv6 et un identifiant Ethernet EUI-64 fictif. Pour améliorer la précision de la première carte obtenue à l'issue de la première étape et parce que les addresses IPv6 générées ont une forte probabilité de ne pas exister sur le réseau, nous avons effectué une deuxième étape de collecte en utilisant cette fois les adresses IPv6 réelles obtenues lors de la première étape de collecte *traceroute*.

### 1.8.1 Collecte de la carte Internet IPv6

Le fonctionnement de *nec* pour cartographier l'Internet IPv6 et l'Internet IPv4 est très similaire. Il existe pour cela des serveurs web *traceroute* pouvant effectuer des traces IPv6. De même nous voulons cartographier l'Internet IPv6 de la manière la plus extensive possible. A notre connaissance nous somme les premiers à produire une carte de l'Internet IPv6 aussi récente en utilisant la technique des serveurs *traceroute* distribués. Pour collecter cette carte, nous avons lancé *nec* pendant 15 jours. Comme l'indique la table 1.4, nous avons effectué deux collectes pendant ces 15 jours car la liste des premières cibles était trop petite et pas assez réaliste (*i.e.*, dû à la création d'adresses fictives). La première collecte a utilisé 628 cibles et 40 traceurs, et nous a permis de découvrir 2652 addresses IPv6 supplémentaires. La deuxième collecte a utilisé les nouvelles addresses découvertes en plus des addresses fictives. En totalité, nous avons découvert 5909 noms et/ou adresses IPv6.

De la même manière que pour la campagne de cartographie Internet IPv4, la durée de 15 jours est dû au fait que nous limitons le nombre de requêtes simultanées sur un serveur *traceroute* pour éviter de surcharger leur lien. Cette durée ne peut pas être comparée à une durée de campagne de cartographie IPv4 mais est déjà très intéressante compte tenu du fait que nous produisons une carte la plus exhaustive possible.

Il y a très peu de filtrage ICMP sur le réseau IPv6 Internet. Ceci est dû à l'âge relativement récent de ce réseau et parce qu'il est encore considéré comme un réseau expérimental par beaucoup d'opérateurs. Cette propriété nous permet d'effectuer une résolution d'alias (*i.e.*, désambiguation des interfaces IP) plus facilement, plus particulièrement en utilisant les options de "source-routing" ce qui s'avère impossible à faire dans le réseau IPv4.

La résolution des alias est tout aussi important en IPv6 qu'en IPv4 et peut se réveler plus difficile compte tenu de la taille de l'espace d'adressage de la nouvelle version du protocole IP. Pour résoudre les aliases nous avons utilisés plusieurs méthodes qui se répartissent en méthodes négatives et positives. Les méthodes négatives sont utilisées pour trouver si deux adresses n'appartiennent pas au même noeud IPv6 afin de diminuer le nombre d'alias a résoudre par les

TAB. 1.4 – Caractéristiques des cartes Internet IPv4 et IPv6.

| Source | Nombre d'interfaces | Nombre de routeurs | Degre moyen des noeuds | nombre de liens | Date |
|---|---|---|---|---|---|
| *nec* IPv4 | 50123 | 47055 | 5.1 | 119909 | 19-29/4/2003 |
| *nec* IPv6 | 5909 | 3834 | 6.5 | 12401 | 27/5-10/6/2003 |

méthodes positives qui sont généralement plus coûteuses. Nous avons utilisé trois méthodes négatives et une méthode positive de résolution des alias. Un nombre important d'interfaces découvertes contenaient seulement le nom DNS et pas l'adresse IPv6 car la commande *traceroute* IPv6 ne fournit pas systématiquement l'adresse IP lorsqu'elle arrive à résoudre le nom de domaine. A cause de cela nous avons défini une méthode négative basée sur le DNS. Celle-ci est la même que celle décrite en section 1.4.

La deuxième méthode négative est basée sur le fait que chaque adresse découverte par le même serveur *traceroute* est sur un arbre et qu'a fortiori toutes celles d'un même arbre ne peuvent pas être des alias potentiels. La dernière méthode négative fait l'hypothèse que chaque noeud IPv6 dispose d'une seule pile IPv6 et d'une seule table de routage. Nous effectuons un "ping" des deux adresses IPv6 et si le TTL n'est pas le même pour les deux paquets ICMP retournés, il ne sont plus considérés comme des alias potentiels.

En utilisant les méthodes négatives, nous somme passés de $n^2$ couples d'alias potentiels avec $n = 5909$ à $n = 250$. Nous avons testé ces couples potentiels avec une méthode de résolution des alias positive. Cette méthode est basée sur le fait que si un message ICMP est envoyé vers une destination, il est envoyé avec l'adresse source de l'interface utilisé pour atteindre la destination. Cette méthode a été initialement proposée par Pansiot *et al.* dans [PG98] et a été décrite dans la section 1.4.

### 1.8.2 Comparaison de l'Internet IPv4 avec l'Internet IPv6

Dans cette section, nous comparons les résultats obtenus dans les sections précédentes dans lesquelles nous avons utilisé *nec* pour obtenir une carte de l'Internet IPv4 avec la carte IPv6 obtenue lors de ce travail. Nous montrons que même si le réseau IPv6 actuel est très petit comparé au réseau IPv4, il contient des propriétés très similaires. La table 1.4 donne une idée de la taille des deux réseaux. Le réseau Internet IPv6 est environ dix fois plus petit que celui IPv4 et contient un nombre important d'alias. Ceci peut être expliqué par le fait que le réseau IPv6 actuel contient proportionnellement plus de tunnels que le réseau IPv4.

La figure 1.8 nous montre que la distance moyenne sur le réseau IPv6 est plus petite que la distance moyennne en IPv4. Les deux réseaux sont fortement maillés. La figure 1.9 nous montre que la distribution du nombre de plus courts chemins suit une loi puissance tout comme en IPv4. La carte IPv6 contient globalement plus de liens, ce qui augmente le nombre de plus courts chemins. Cette proprieté peut être exploitée pour diminuer le nombre d'états dans les routeurs comme montré dans [HPMG03].

## 1.9 Inflation et délais des chemins et des liens

Cette section présente de nombreux résultats concernant l'inflation des chemins liée aux protocoles de routage ainsi que l'analyse des délais des liens inférés par les traces.

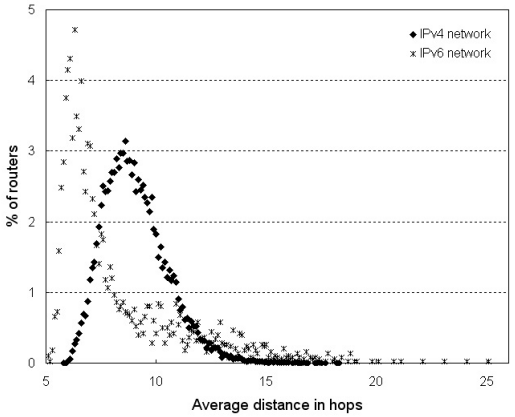


FIG. 1.8 – Distribution de la distance moyenne.

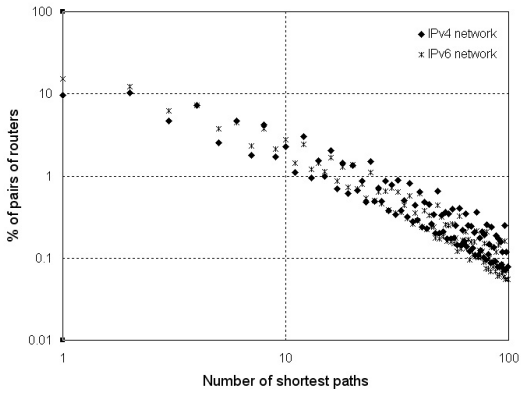


FIG. 1.9 – Distribution du nombre de plus courts chemins.

### 1.9.1 Inflation des chemins dans l'Internet

Nous avons étudié l'inflation des chemins (routes) de l'Internet à l'aide des traces et de la carte produites par une campagne effectuée en septembre 2003. Tout *traceroute* définit un chemin réellement emprunté par un paquet depuis le serveur *traceroute* (source) jusqu'à l'adresse cible (destination). Après avoir créé la carte, nous pouvons calculer les plus courts chemins entre toutes les paires de routeurs en utilisant l'algorithme de Dijkstra pour chaque routeur de la carte. Nous obtenons parmi tous les plus courts chemins, tous ceux entre les sources et les destinations de nos chemins *traceroute*. Nous mesurons la longueur des chemins *traceroute* et des plus courts chemins en utilisant pour métrique le nombre de sauts. L'inflation d'un chemin est égal à la longueur d'un chemin *traceroute* d'une paire source-destination divisée par la longueur du (d'un) plus court chemin de cette même paire. Le résultat de la division est un nombre réel supérieur ou égal à 1. Nous l'appelons le ratio de la longueur du chemin ou aussi l'inflation du chemin. Avec les traces de la deuxième campagne, nous avons pu calculer le ratio de 2196960 chemins (*i.e.*, paires source-destination).

La distribution cumulative des ratios des 2M de paires est tracée sur la figure 1.10. Les ratios varient de 1 à 10. Nous avons fixé 10 comme limite supérieure. Nous avons mesuré quelques valeurs supérieures à 10 mais nous les considérons anormales. Ces valeurs anormales peuvent être expliquées par le fait que les chemins *traceroute* proviennent des données brutes alors que les plus courts chemins proviennent de la carte des routeurs après désambiguation des interfaces et élimination des bouts de graphes non connexes. Donc certaines interfaces IP et liens IP ont disparu de la carte et peuvent être la cause de chemins ayant des propriétés étranges telles qu'une inflation très élevée. L'un des faits les plus interessants de ces données est que seulement 2.72% des chemins ont un ratio de 1 (*i.e.*, cas où le chemin *traceroute* a la même longueur que le plus court chemin correspondant). De plus, 43.4% des chemins ont un ratio supérieur à 2 ce qui est clairement non négligeable ! Cela correspond a une inflation du chemin d'au moins 100% par rapport au plus court chemin correspondant.

Ces résultats confirment que l'inflation des chemins n'est pas un phénomène marginal mais qu'il a un impact très important sur la majorité des chemins. Les causes peuvent être nombreuses :

– Cela est dû principalement aux politiques de routage dans l'Internet surtout en inter-domaine. Les liens et les routes peuvent avoir des poids/coûts de routage fixés administrativement (surtout en inter-domaine) et donc les routes peuvent ne pas être optimales. De plus dans le routage inter-domaine, il est techniquement impossible de determiner le

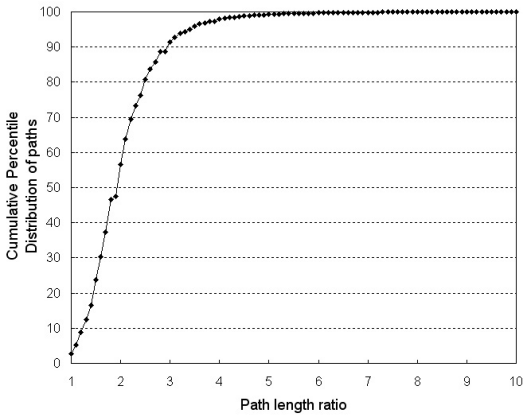


FIG. 1.10 – Distribution cumulative de l'inflation des chemins.

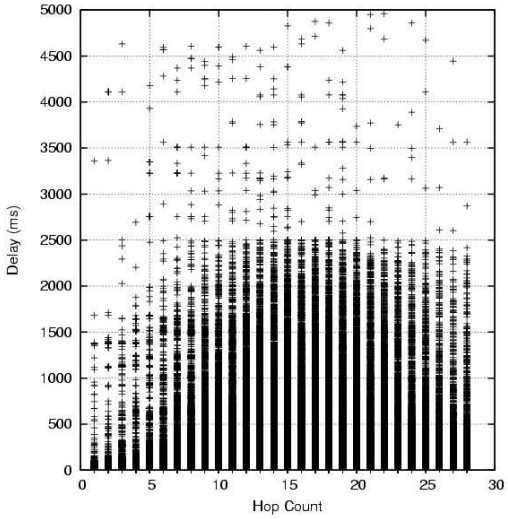


FIG. 1.11 – Délais des chemins *vs* nombre de sauts des chemins.

plus court chemin au niveau routeur d'un routeur vers un autre routeur situé dans un AS différent car seul un sous-ensemble de la topologie intra-domaine est annoncée par les protocoles de routage intra-domaines au protocole inter-domaine BGP.

– De nombreux chemins sont coupés dans les traces (*i.e.*, ils s'arrêtent avant d'atteindre la destination désirée) à cause du filtrage ICMP. Nous utilisons la vraie destination (*i.e.*, le dernier routeur qui répond) et non le routeur cible original pour déterminer le ratio mais si nous avions des traces complètes, peut être que les ratios seraient plus faibles surtout si il n'y a pas d'inflation dans la partie manquante du chemin.
– Cela peut aussi être causé par notre sélection de destinations comme cibles *traceroute*. Nous ne choisissons peut-être pas des chemins qui reflètent des chemins typiques (*i.e.*, entre sources et destinations plus fréquentes que d'autres).

Nous avons aussi étudié dans les traces de notre deuxième campagne, les valeurs des temps aller-retour (*i.e.*, Round-Trip Time en anglais) et des délais. Les mesures des délais que produisent les traces sont toujours des RTT. En effet pour pouvoir mesurer des aller simples, il faudrait disposer de deux horloges parfaitement synchronisées : une horloge au départ et une à la destination. Cependant les horloges des machines actuelles ne sont pas synchronisées, et celles qui le sont (via le protocole NTP par exemple) n'ont pas une précision suffisante. En effet les délais sont de l'ordre de 10 à 1000 ms ce qui est très petit par rapport à la précision que fournit NTP. Par conséquent on utilise uniquement l'horloge de départ et on ne peut donc mesurer que des RTT. L'heuristique de base pour obtenir le délai correspondant au RTT d'un chemin ou d'un lien (si il n'y a qu'un seul saut) consiste donc à diviser son RTT par 2. C'est que nous faisons ici. Le délai réel peut être différent car les chemins et les liens asymétriques peuvent faire varier la valeur du délai d'une valeur très faible jusqu'à une valeur proche du RTT (surtout pour un lien unique). L'erreur maximale commise peut donc atteindre le RTT divisé par 2 soit une erreur de 100%. En ce qui concerne les liens IP, comme la majorité des media sont symétriques en particulier dans les coeurs de réseau, notre heuristique est acceptable et l'erreur moyenne est probablement très inférieure à l'erreur maximale. Pour les chemins, une étude plus approfondie devra être effectuée pour évaluer l'impact des chemins asymétriques.

Nous avons cherché une corrélation entre la longueur d'un chemin exprimée en nombre de sauts et sa latence (ou délai, qui est égal au RTT donné par le dernier routeur ayant répondu et divisé par 2). Chaque point de la figure 1.11 représente un chemin avec sa longueur en abscisse

et sa latence en ordonnée. 1,34M de chemins sont représentés car toutes les traces n'ont pas forcément des informations sur le RTT qui sont exploitables. On ne peut observer sur la figure ni linéarité, ni corrélation particulière. Le seul phénomène que l'on peut voir est une borne maximum du délai du chemin en fonction de sa longueur (en écartant les points extrêmes). Sous cette borne, on trouve toutes les valeurs de délais possibles. Le tracé suivi par l'évolution de la valeur de cette borne maximum ressemble un peu à une distribution normale. On s'attendrait à ce que cette borne augmente régulièrement avec la longueur des chemins, mais nous devons prendre en compte que les valeurs RTT mesurées par les *traceroute* sont dynamiques et dépendent non seulement de la longueur du chemin (délai théorique minimum) mais aussi de la charge de certains routeurs du réseau (engorgement, files d'attentes avec diverses priorités, etc). Donc il faudrait différencier nos mesures entre les délais minimaux (les paquets des traces n'ont subi aucun ralentissement) et les délais dynamiques réels (les paquets des traces ont été ralentis à un ou plusieurs endroits) : cela est techniquement impossible à réaliser avec les seules informations fournies par les serveurs *traceroute*. Le fait que la part relative des délais minimaux (traces non freinées) soit plus importante dans les longs chemins (20 sauts ou plus) peut expliquer le fait que la valeur de la borne maximale soit plus faible que pour des chemins plus courts.

L'inflation des chemins a été étudiée par Tangmunarunkit *et al.* dans [TGSE01] en 2000. Ils ont trouvé que plus de 95% de leurs chemins ont un ratio inférieur ou égal à 2 alors que nous n'en avons trouvés que 56.6%. Cependant leur étude ne portait que sur 61k traces alors que nous en avons 2M et ils ont utilisé Mercator pour créer la carte (*i.e.*, donc cette dernière manquait un grand nombre de liens transverses, voir section 1.7). C'est pourquoi nous pensons que nos résultats sont plus précis. Notons qu'une augmentation du nombre de sauts dûe à l'inflation de chemin entre deux routeurs peut ne pas se traduire automatiquement par une forte augmentation du délai sur ce chemin. Etudier l'inflation des chemins en utilisant le délai comme métrique fait partie de nos objectifs ultérieurs. Très récemment, Spring *et al.* ont effectué une étude proche de la nôtre dans [SMA03] et ils ont trouvé que l'augmentation moyenne du délai des chemins pour les chemins inter-domaines (le plus mauvais cas) est inférieur à 14 ms. Cependant ils n'ont pas étudié la corrélation entre l'inflation du délai et celle du nombre de sauts.

### 1.9.2 Etude des délais des liens

Comme cela est précisé précédemment, nous avons aussi étudié dans les 3Go de traces de notre campagne de septembre 2003, les valeurs des RTT donnés par les *traceroute*. Ci-dessous l'exemple d'une trace typique :

```
traceroute to 24.100.0.103 (24.100.0.103) from ns2.acadia.net (142.167.1.1), 40 hops max
 1  ip142167001252.acadia.net (142.167.1.252)  0.677 ms  0.622 ms  0.481 ms
 2  63.111.121.197 (63.111.121.197)  8.676 ms  8.501 ms  9.339 ms
 3  196.at-1-0-0.CL2.BOS1.ALTER.NET (152.63.25.126)  8.682 ms  8.678 ms  9.095 ms
 4  0.so-6-2-0.XL2.NYC9.ALTER.NET (152.63.29.114)  15.170 ms  15.130 ms  15.149 ms
 5  POS7-0.BR1.NYC9.ALTER.NET (152.63.18.221)  15.228 ms  15.087 ms  15.244 ms
 6  acr2-so-5-3-0.NewYork.cw.net (206.24.193.245)  16.759 ms  16.170 ms  15.929 ms
 7  agr4-loopback.NewYork.cw.net (206.24.194.104)  16.613 ms  16.955 ms  16.538 ms
 8  dcr2-so-7-3-0.NewYork.cw.net (206.24.207.205)  15.753 ms  16.104 ms  15.798 ms
 9  dcr1-loopback.Chicago.cw.net (208.172.2.99)  38.989 ms  39.370 ms  39.279 ms
10  telus-rogers.Chicago.cw.net (208.175.10.86)  49.834 ms  49.704 ms 47.761 ms
11  gw01.esna.phub.net.cable.rogers.com (66.185.80.193)  50.105 ms  50.021 ms  50.653 ms
12  gw01.mtmc.phub.net.cable.rogers.com (66.185.81.166)  48.114 ms  47.789 ms  50.860 ms
13  gw01.ajax.phub.net.cable.rogers.com (66.185.81.170)  50.717 ms  50.375 ms  50.563 ms
14  66.185.81.198 (66.185.81.198)  50.734 ms  50.530 ms  50.718 ms
15  tlgw1.ajax.phub.net.cable.rogers.com (24.100.0.1)  50.686 ms  51.066 ms  51.040 ms
16  * * *
17  * * *
```

Pour une ligne donnée, les valeurs temporelles exprimées en millisecondes, représentent le temps aller retour d'un paquet depuis la source (le serveur *traceroute*) jusqu'au routeur dont l'adresse est donnée en début de ligne. Trois paquets sont envoyés afin d'éviter des valeurs

biaisées par un retard de traitement momentané sur le trajet d'un paquet et donc trois valeurs sont mesurées. Ces valeurs permettent de déduire les délais sur les liens de la façon suivante :

1. On calcule la moyenne $m_n$ des trois valeurs de RTT pour le routeur $n$ (on pourrait envisager un traitement plus complexe que la simple moyenne).
2. On calcule la moyenne $m_{n-1}$ des trois valeurs de RTT pour le routeur $n-1$.
3. On calcule $(m_n - m_{n-1})/2$ pour obtenir le délai du lien de $n-1$ vers $n$.

Il arrive que le résultat soit négatif (avec notre méthode, le délai entre 11 et 12 ci-dessus sera négatif). Dans ce cas, si il est négatif et peu important (proche de zéro), alors on attribue au lien un délai de 1 ms. A noter que les traces peuvent passer de très nombreuses fois sur un même lien. On refait le calcul à chaque passage sur un lien donné et on stocke toutes les valeurs dans un tableau qui lui est propre. Lors de la construction de la carte, on attribue à chaque lien un délai égal à la moyenne de tous les délais stockés dans son tableau. A noter que pour les délais, on considère les liens comme asymétriques. Donc on calcule les délais en tenant compte du sens de parcours du lien et par conséquent chaque lien a donc deux délais : le délai du routeur $n-1$ vers $n$ et celui du routeur $n$ vers $n-1$. Nous attribuons la valeur 0 a tout délai de lien que nous n'avons pas pu déterminer. En effet les traces ont un sens de parcours et comme il y a beaucoup plus de destinations que de sources, la majorité des trajets ne sont jamais fait en sens inverse. Notre technique nous a permis de déterminer les valeurs de 194431 délais, soit un taux détermination de 34,4%. Les autres délais ont pour valeur 0 (*i.e.*, indéterminé).

La figure 1.12 illustre la distribution de ces 194k délais. Celle-ci est fortement biaisée et sa partie de gauche pourrait presque être approximée par une loi puissance. La majorité des liens (considérés dans le sens de leur mesure) ont donc des délais très faibles, ce qui est logique. La distribution cumulative de ces mêmes délais illustrée sur la figure 1.13 fait apparaître un tracé plus compact, mais pas une linéarité. On ne peut donc affirmer se trouver devant une distribution à queue lourde bien que le tracé s'en rapproche, cependant n'oublions pas que ces délais sont très dynamiques (ils dépendent de l'état du réseau lors de leur mesure). On peut noter deux points intéressants sur la figure 1.13 : seulement 28,5% des liens ont un délai supérieur à 10 ms et moins de 1,8% des liens ont un délai dépassant les 100ms. On peut raisonnablement considérer que les liens ayant des délais supérieurs à 100ms sont soient de très longue connexions transcontinentales, soient des liaisons satellites, soient que leurs mesures ont été faussées par des engorgements du réseau.

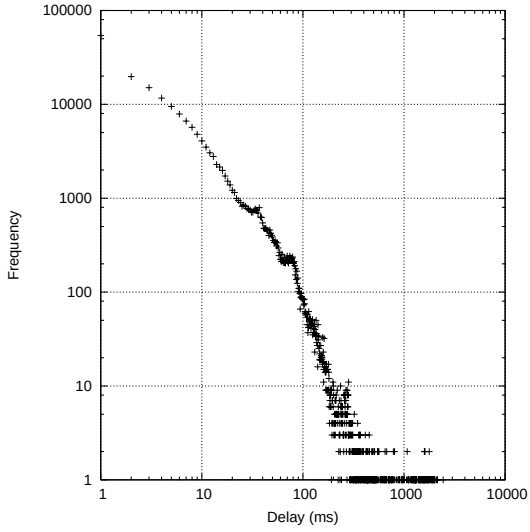


FIG. 1.12 – Distribution des délais des liens.

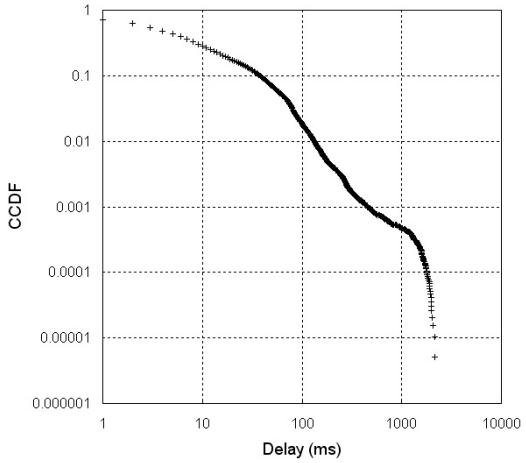


FIG. 1.13 – Distribution complémentaire cumulative des délais des liens.

La faible corrélation des délais des chemins et de leurs longueurs en nombre de sauts est un résultat nouveau à notre connaissance et très intéressant. Cela signifie que les simulations de protocoles ne doivent plus se contenter de délais identiques sur tous les liens comme cela se voit encore très fréquemment dans les topologies des simulations utilisant *ns-2* par exemple. En effet cela revient à utiliser la métrique du nombre de sauts. Nos résultats de la figure 1.11 montrent que le nombre de sauts d'un chemin n'est pas directement corrélé au délai de celui-ci. Il est donc préférable d'utiliser des délais différents sur les liens et dont les valeurs se rapprochent de nos résultats. Les topologies utilisées dans les simulations de protocoles Internet doivent donc s'inspirer de la distribution des délais des liens de la figure 1.12 afin d'accroître leur réalisme.

Nous avons cherché à découvrir une corrélation entre la taille d'un routeur exprimée par le nombre de liens IP adjacents sortants à celui-ci et le délai de ses mêmes liens IP adjacents sortants. Chaque point de la figure 1.14 représente un lien (dans son sens de mesure) avec son délai en abscisse et le degré de son routeur de départ en ordonnées. L'idée étant que plus un routeur est interconnecté (degré élevé), plus il est en coeur de réseau et donc plus il possède de connexions rapides. Le tracé nous montre que cette idée n'est pas fondée. Il n'apparait pas de corrélation évidente entre la taille d'un routeur et le délais de ses liens adjacents sortants. De même nous avons cherché à découvrir une corrélation entre la position d'un routeur exprimée par sa distance moyenne à tous les autres routeurs mesurée en nombre de sauts et le délai de ses mêmes liens IP adjacents sortants. Chaque point de la figure 1.15 représente un lien (dans son sens de mesure) avec son délai en abscisse et la distance moyenne de son routeur de départ en ordonnées. L'idée étant que plus un routeur est proche du centre du réseau (distance moyenne faible), plus il possède de connexions rapides. Encore une fois, il n'apparait pas de corrélation évidente entre la position d'un routeur et le délais de ses liens adjacents sortants.

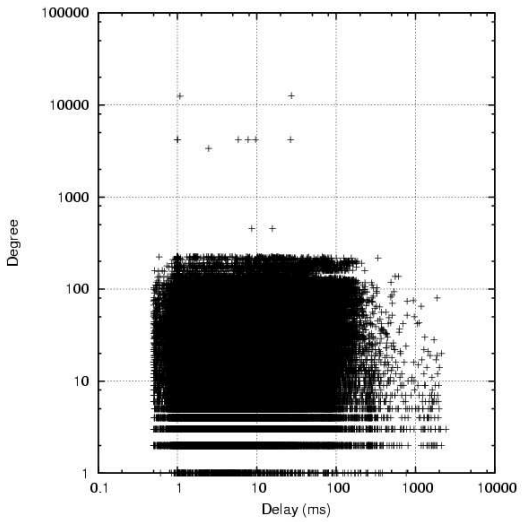


FIG. 1.14 – Degré des routeurs *vs* délais des liens.

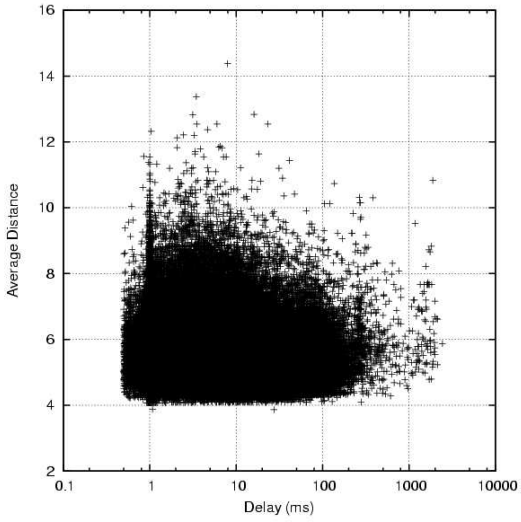


FIG. 1.15 – Distance moyenne des routeurs *vs* délais des liens.

## 1.10 Conclusion

La cartographie d'Internet est une opération complexe. Dans ce chapitre nous avons décrit notre logiciel *nec* de cartographie rapide et nous avons montré que celui-ci est nettement supérieur à Mercator, seul logiciel libre de cartographie actuellement disponible. En 10 jours, *nec* a découvert presque autant de routeurs et surtout presque deux fois plus de liens que Mercator.

La carte obtenue possède une quantité relative de liens supérieure à une carte d'AS ce qui est un résultat nouveau pour une carte de niveau routeur à l'échelle de l'Internet public. De plus les recouvrements nous montrent que *nec* produit une carte beaucoup moins biaisée que Mercator. Nous avons montré que *nec* peut aussi cartographier la topologie IPv6 et que ce dernier bien que très petit, partage certaines caractéristiques du réseau IPv4. Cela est logique étant donné le fait que ce réseau se déploye le plus souvent sur du matériel appartenant au réseau IPv4. Nous avons enfin étudié les délais des chemins et des liens de l'Internet IPv4. Nous avons observé que la longueur en nombre de sauts des chemins est faiblement corrélée au délai de ces mêmes chemins. Nous avons découvert que la distribution des liens est fortement biaisée et se rapproche d'une distribution à queue lourde ce qui est là encore un résultat inédit à notre connaissance. Plus de deux tiers des liens de l'Internet ont un délai mesuré inférieur à 10ms.

Nous travaillons actuellement à l'optimisation du nombre et de l'orientation des traces nécessaires à mener à bien les cartographies ainsi qu'à l'amélioration de la détermination des délais mesurés sur les liens. En effet, les délais ont une nature hautement dynamique liée à l'activité du réseau. Nous espérons que les résultats produits par notre logiciel (*e.g.*, cartes IPv4 et IPv6, recouvrements routeurs–AS, délais des liens) seront profitables à la communauté réseau pour les simulations faisant appel à une modélisation réaliste de l'Internet. Les personnes désireuses d'utiliser les cartes produites par *nec* (et contenant les recouvrements associés) pour faire des simulations de protocoles peuvent les trouver à l'adresse [MH].

Le projet *nec* a pu être mené à bien grâce aux contributions de nombreuses personnes. La partie IPv6 a été réalisée par Mickaël Hoerdt lors de son travail de doctorat que j'ai coencadré avec Jean-Jacques Pansiot. Les accès aux serveurs web *traceroute* ont été développés par Anthony James et Pascal Merindol lors de leurs travaux d'études et de recherche (TER) de maîtrise. La partie analyse des données a été réalisée par Déborah Malka lors de son projet d'IUP 3ème année. Enfin la partie injection directe des paquets de traçage a été développée par Jeanne Maillard et Marc Jaeger lors de leur projet d'IUP 3ème année. Les résultats que nous avons obtenus sont actuellement utilisés par d'autres laboratoires et en particulier par Matthieu Latapy et Jean-Loup Guillaume du LIAFA dans leurs travaux concernant l'optimisation des explorations des grands réseaux d'interaction [LG05].

# Travaux relatifs au chapitre

1. Jean-Loup Guillaume, Matthieu Latapy and Damien Magoni. Relevance of massively distributed explorations of the Internet topology : qualitative results. *Computer Networks*, 50(16) : 3197–3224, November 2006.
2. Mickaël Hoerdt et Damien Magoni. Cartographie distribuée de la topologie du coeur de l'Internet. *Annales des Télécommunications*, 60(5–6) : 558–587, mai 2005.
3. Damien Magoni and Mickaël Hoerdt. Internet core topology mapping and analysis. *Computer Communications*, 28(5) : 494–506, March 2005.
4. Mickaël Hoerdt and Damien Magoni. Completeness of the Internet core topology collected by a fast mapping software. In *Proceedings of the 11th International Conference on Software, Telecommunications and Computer Networks*, pages 257–261, Split, Croatia, October 2003.
5. Mickaël Hoerdt et Damien Magoni. Complétude de la topologie du coeur d'Internet par cartographie rapide. In *Actes du 10ème Colloque Francophone sur l'Ingénierie des Protocoles*, pages 409–424, Paris, France, octobre 2003.
6. *network cartographer (nec)*.
   http ://dpt-info.u-strasbg.fr/~magoni/nec/

# Chapitre 2

# Analyse de la robustesse de la topologie de l'Internet

## 2.1 Introduction

L'une des applications les plus immédiates des données collectées par *nec* et présentées au chapitre précédent, consiste à étudier la résistance de l'Internet face à des attaques réseaux. En effet la connaissance précise de la topologie du coeur de l'Internet, de celle du réseau des systèmes autonomes et des correspondances exactes entre ces deux réseaux nous permet d'étudier finement l'impact d'attaques visant à briser le fonctionnement d'Internet. Cette étude est d'autant plus motivée par le fait que les avances récentes dans l'analyse des réseaux à dimension libre d'échelle (*i.e.*, scale-free networks) ont montré que leurs topologies sont très faibles contre des attaques. La distribution non homogène de la connectivité des réseaux de transmission actuels de grande taille, dont l'Internet, pourrait être exploitée par des intrus malintentionnés afin d'endommager ces systèmes. Cependant, il n'y a pas eu beaucoup d'études sur les approches et les conséquences de telles attaques ciblées. Dans ce chapitre, nous proposons une étude appronfondie de la robustesse de la topologie de l'Internet face aux attaques à sa couche réseau (IP). Plusieurs techniques d'attaques sont présentées ainsi que leurs effets sur la connectivité de l'Internet. Nous montrons que bien que la suppression d'une petite fraction des noeuds (moins de 10%) puisse endommager la connectivité de l'Internet, une telle attaque par suppression de noeuds exigerait une grande quantité de travail pour être menée à bien. Pour réaliser ceci, nous étudions en détail les interactions entre les niveaux intra-domaines et inter-domaines de l'Internet par l'utilisation d'un recouvrement (*i.e.*, overlay en anglais).

Tôt ou tard chacun sera connecté à l'Internet. En Europe de l'ouest, 29% des ménages étaient connectés à l'Internet en 2001. Et c'était une augmentation de 33% par rapport à l'année 2000. Nous inter-agissons de plus en plus par l'Internet et par conséquent nous devenons plus dépendants de son fonctionnement. Les messages tels que "hôte inaccessible" nous deviendront de plus en plus stressants. Puisque le nombre de personnes utilisant l'Internet et l'importance de son utilisation (par exemple pour des tâches administratives ou commerciales) se développeront énormément dans le proche futur, la connectivité de l'Internet deviendra un facteur majeur de la productivité de nos systèmes économiques. N'importe quelle panne de connectivité se transformera en énormes pertes de bénéfice. Cette caractéristique transforme l'Internet en cible potentielle pour des attaques malintentionnées ou terroristes.

Naturellement il y a beaucoup de niveaux de connectivité s'étendant de la couche physique jusqu'à la couche application. Cependant une rupture à n'importe quel niveau arrêtera le fonctionnement de tous les niveaux supérieurs. Ainsi une attaque vers des niveaux plus bas fera beaucoup plus de dommages. Dans ce chapitre nous concentrons notre étude de la robustesse de la connectivité de l'Internet sur le niveau le plus bas que nous pouvons analyser (*i.e.*, compte

tenu des données à notre disposition), à savoir la couche réseau du modèle ISO donc la couche de l'Internet Protocol (IP). Le reste du chapitre est organisé comme suit. Dans la section 2.3, nous présentons les cartes de l'Internet que nous utilisons dans notre étude. Puis nous proposons plusieurs méthodes pour attaquer la topologie de l'Internet dans la section 2.4. Finalement, dans la section 2.5 nous discutons des métriques choisies pour les mesures de connectivité et nous donnons des résultats de mesure sur la connectivité de l'Internet sous attaques.

## 2.2 Contexte

La robustesse de la connectivité de l'Internet n'a pas été intensivement étudiée par la communauté de recherche. Une des premières études de cette sorte a été effectuée par Albert *et al.* dans [AJB00] pendant l'année 2000. Ils ont évalué les effets de la suppression aléatoire et ciblée de noeuds sur les réseaux exponentiels et de dimension libre. Ils ont utilisé des graphes aléatoires ainsi que des cartes réelles. Pour les réseaux à dimension libre, ils ont utilisé une carte des systèmes autonomes (Autonomous Systems ou AS en anglais) de l'Internet et une carte du World Wide Web (WWW). Leurs métriques pour évaluer les effets de la suppression des noeuds incluent le diamètre du réseau, la taille du plus grand groupement (*i.e.*, composante connexe) et la taille moyenne des groupements isolés (*i.e.*, tous les groupements à l'exclusion du plus grand). Ils ont trouvé que lors d'une attaque (*i.e.*, suppression ciblée de noeuds), les réseaux à dimension libre tels que l'Internet et le WWW se brisent en de petits fragments pour une fraction seuil de suppression des noeuds. Lors d'une panne, les réseaux à dimension libre conservent leur plus grand groupement intact même pour un taux extrêmement élevé de suppression.

Des autres travaux semblables sur la robustesse d'Internet ont été menés à bien par Tauro *et al.* dans [TPSF01] pendant l'année 2001 sur une carte d'AS. Ils ont également utilisé la taille de la plus grande composante connexe en tant que métrique et ont abouti aux mêmes conclusions que [AJB00] : le réseau des AS est sensible aux pannes ciblées de noeud tandis qu'il est robuste aux pannes aléatoires de noeuds. Pour ce qui concerne la robustesse de l'Internet au niveau des routeurs, Broido *et al.* ont étudié dans [BC01] la robustesse de la composante géante d'IP (une carte de 52505 noeuds créée à partir des données de Skitter) sous attaque et ont mesuré plusieurs propriétés comprenant le diamètre de la composante géante, la taille de la plus grande composante, etc. Ils ont tiré les mêmes conclusions pour le niveau routeur que [AJB00] pour le niveau AS.

En outre en 2001, Palmer *et al.* évaluent dans [PSF+01] la robustesse de la carte de niveau routeur collectée par les projects SCAN [GT00] et Internet Mapping de Lucent [BC99] et qui contient approximativement 285K noeuds. Comme métriques, ils utilisent le nombre de paires accessibles et l'exposant du nombre de sauts. La première métrique dérive de la fonction de voisinage. Comme cette fonction exige beaucoup de calcul, les auteurs ont proposé une fonction approximative de voisinage en utilisant l'analyse par exploitation de données (data mining en anglais). La deuxième métrique dérive d'une loi puissance définie par Faloutsos *et al.* dans [FFF99]. Avec ces deux dernières métriques, ils ont étudié des suppressions aléatoires d'arête et des suppressions aléatoires et ciblées de noeuds. Tout commes les précédentes études, ils ont constaté que la suppression des noeuds importants (soit par le degré ou bien par l'exposant du nombre de sauts), affecte sévèrement la connectivité du réseau tandis que les suppressions aléatoires d'arêtes ou de noeuds ne l'affecte pas autant et moins rapidement.

Plus récemment, Tangmunarunkit *et al.* ont étudié dans [TGJ+02] la robustesse d'une carte d'AS et d'une carte de routeurs (toutes les deux collectées en mai 2001) en utilisant une métrique de la théorie des graphes appelée résilience. Nos travaux sont une extension de ces études précédentes. Nous allons plus loin dans l'analyse de la robustesse de l'Internet afin de trouver s'il est possible de détruire la connectivité de l'Internet et à quel coût.

À un niveau plus théorique, il vaut la peine de noter que plusieurs exemples d'études théoriques de réseaux générés par modélisation, qui donnent des résultats tout à fait semblables à ceux obtenus par nous mêmes pour de vrais réseaux, sont donnés par Callaway *et al.* dans [nMEJNSW00], par Cohen *et al.* dans [CEbAH01] qui contient également une discussion sur le diamètre, et enfin par Holme *et al.* dans [HKYH02], qui présente également l'effet des corrélations. Les physiciens tracent habituellement le nombre de groupements de taille $s$, comme une fonction cumulative de $s$ sur une échelle log-log. Cette distribution a été étudiée complètement dans la littérature physique et il est usuel que la distribution suive une loi puissance avec une pente de - 2.5 au point de transition et une loi puissance avec une coupure exponentielle au-dessus et au-dessous du point de transition comme montré par les travaux de Newman *et al.* dans [NSW01] et par ceux de Cohen *et al.* qui étudient également dans [CbAH02] la généralisation de cette loi puissance lords des pannes aléatoires dans les réseaux de dimension libre.

## 2.3 Cartes de l'Internet

Étudier la robustesse d'Internet implique de connaître la topologie d'Internet. Dans cette section nous présentons les données que nous utilisons dans nos expériences et nous expliquons comment nous construisons un recouvrement afin d'associer les noeuds IP à leurs AS respectifs.

### 2.3.1 Données sources

Comme nous voulons obtenir des résultats précis et directement applicables, nous n'utilisons pas les cartes de niveau AS de l'Internet pour la base de notre étude parce qu'elles sont à grain trop grossier. Au lieu de cela, nous nous concentrons sur la connectivité IP et donc nous préférons travailler au niveau des routeurs de l'Internet. Nous utilisons trois cartes de l'Internet. La premiere est une carte anonyme de niveau routeur qui est le résultat du fusionnement d'une carte rassemblée par le projet SCAN [GT00] et d'une carte collectée par le projet Internet mapping de Lucent [BC99]. C'est la plus grande carte d'Internet de niveau routeur actuellement disponible à notre connaissance. Elle a été collectée en 1999 et utilisée dans [PSF+01]. La carte telle quelle n'est pas connexe. Nous avons retiré 33 noeuds afin de rendre cette carte connexe. C'est négligeable en comparaison de la taille de cette carte. En outre ces noeuds étaient la plupart du temps dans des composantes connexes de taille 1 ou 2 (*i.e.*, noeuds seuls ou paires de noeuds). La deuxième carte est une carte de niveau routeur collectée depuis notre laboratoire (appelé LSIIT et situé à Illkirch, France) en utilisant le logiciel Mercator écrit par Govindan *et al.* et décrit dans [GT00]. Cette carte est connexe. La collecte a durée quatre mois à partir d'avril à juillet 2002. À la différence de la carte de 1999, celle-ci contient les adresses IP des interfaces des routeurs. La troisième et derniere carte est une carte de niveau AS collectée par le routeur BGP *route-views* [Uni] au début de Juillet 2002. Nous l'employons principalement pour construire un recouvrement avec notre carte de 2002 mais également pour la comparaison avec les résultats d'analyse de niveau routeur. La table 2.1 contient quelques informations sur ces cartes.

### 2.3.2 Construction du recouvrement

Nous construisons ici un recouvrement de manière très similaire à la méthode présentée dans la section 1.6 aussi nous ne détaillerons ici que les différences par rapport à la section 1.6. Nous construisons un recouvrement de manière à lier la topologie des routeurs à celle des AS en utilisant le contenu d'une table de routage BGP créée le 1 juillet 2002 par *route-views*.

TAB. 2.1 – Cartes Internet utilisées dans les simulations

| Origine | Nb de noeuds | Nb de liens | Date |
|---|---|---|---|
| SCAN-LUCENT | 284772 | 449228 | 1999 |
| LSIIT | 188347 | 235991 | 4-7/2002 |
| *route-views* | 13529 | 28060 | 7/2002 |

Cette table contient 13529 AS. La table contient 119814 préfixes réseaux (ce qui est là encore consistant avec les résultats trouvés par Bu *et al.* dans [BGT02]).

Puis nous utilisons l'information IP collectée par Mercator pour créer une carte de niveau routeur de l'Internet. La description de Mercator ainsi que ces limitations se trouvent dans [GT00]. Mercator peut effectuer une disambiguation d'interface basique et peut donc partiellement assigner correctement de multiples interfaces à leur routeur correspondant. La carte résultante contient 203854 interfaces et 188347 noeuds. Cela nous donne un taux d'interfaces multiples de 8.2% ce qui est la moitié de la valeur observée dans [BBBC01]. Une première explication de cette différence est que notre carte avec un degré moyen de 2.5 manque probablement un nombre important de liens redondants qui constituent potentiellement des interfaces multiples vers des noeuds déjà découverts. Puis pour chaque interface/routeur, nous recherchons le plus long préfixe réseau correspondant et nous associons l'AS source (ou celui qui le déclare) possédant ce préfixe réseau, à cette interface. Durant ce processus, 1296 interfaces n'ont pas pu être mises en correspondance avec un AS. 57 de ces interfaces étaient des adresses de classe A, 405 de classe B et 834 de classe C. Les interfaces non résolues représentent 0,64% de toutes les interfaces ce qui est comparable au taux de 0,48% mesuré dans [TGSE01]. Enfin nous trouvons par la méthode décrite dans la section 1.6 que 40316 routeurs peuvent être considéré comme étant des routeurs BGP, ce qui représente 21.4% de tous les routeurs de cette carte IP. Nous n'avons actuellement aucune information pour confirmer la réalité de ce chiffre par rapport à sa valeur réelle.

Pour évaluer l'exactitude de notre recouvrement nous étudions la distribution de la taille des AS. La taille d'un AS est égale au nombre de routeurs contenu dans cet AS. Nous constatons que seulement 5277 AS contiennent 1 ou plusieurs routeurs dans notre recouvrement et cela représente seulement 39% de tous les noeuds de la carte d'AS. Ceci montre clairement que notre carte de niveau routeur est loin d'être complète. En dépit de cette imperfection, nous cherchons des distributions à queues lourdes trouvées dans les recouvrements d'Internet créés par Tangmunarunkit *et al.* dans [TDG+01]. Une méthode commode déjà utilisée dans [TDG+01] et [BT02] pour réaliser ceci est de tracer la fonction de distribution cumulative complémentaire de la taille des AS (CCDF en anglais) sur une échelle log-log. La figure 2.1 montre la CCDF de la distribution de la taille des AS et nous pouvons voir qu'elle est à queue lourde. Pour poursuivre l'analyse de [TDG+01], nous tracons également sur la figure 2.2 la corrélation entre la taille et le degré des 100 plus grands AS. La corrélation pour ces AS est forte et très semblable à celle trouvée dans [TDG+01].

La figure 2.3 montrent sur une échelle log-log la CCDF du nombre de routeurs BGP par AS et celle-ci présente une distribution à queue lourde (une régression linéaire donne un CC de 0.993). Enfin grâce à notre technique de création de recouvrement, nous sommes en mesure d'étudier la distribution de la taille des connexions BGP. La taille d'une connexion BGP est égale au nombre de liens IP allant de n'importe quel routeur appartenant à un AS donné vers tout routeur situé dans un autre AS donné différent. Comme les AS, seulement 6630 connexions sont remplis par un ou plusieurs liens IP. Ceci donne un taux de remplissage de 23.6% ce qui est beaucoup plus faible que le taux de remplissage des AS. Ceci tend encore à illustrer que

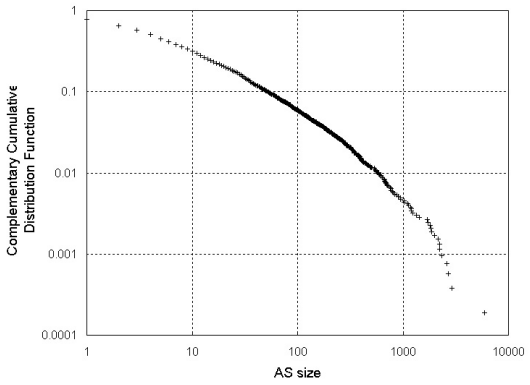


FIG. 2.1 – Distribution complémentaire cumulative de la taille des AS.

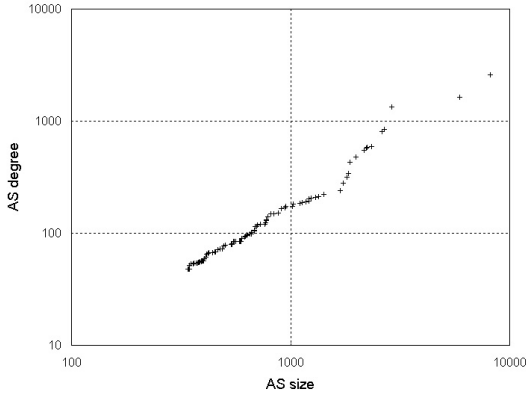


FIG. 2.2 – Corrélation entre la taille des AS et leur degré.

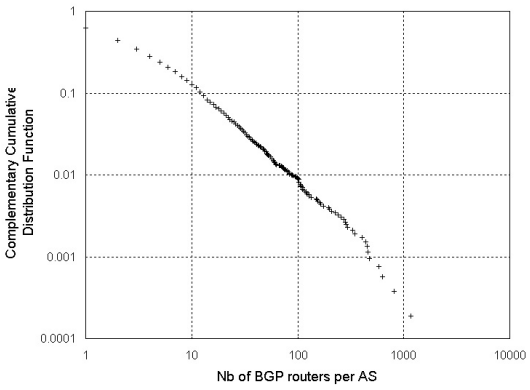


FIG. 2.3 – Distribution complémentaire cumulative du nombre de routeurs BGP par AS.

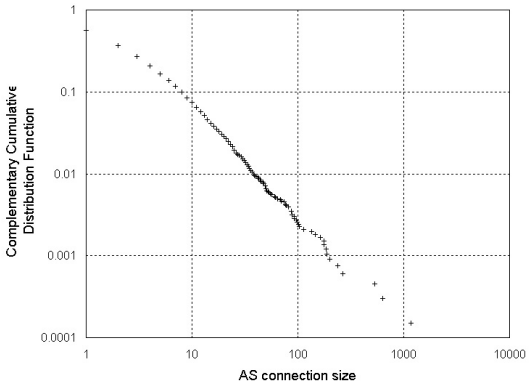


FIG. 2.4 – Distribution complémentaire cumulative de la taille des connexions BGP.

notre carte de niveau routeur manque un nombre significatif de liens IP. La figure 2.4 montrent sur une échelle log-log la CCDF de la taille des connexions et on peut voir qu'elle révèle une distribution à queue lourde (avec un CC de 0.995). Nous pouvons conclure que bien que notre recouvrement routeurs–AS soit incomplet, il contient beaucoup de distributions à queue lourde qui sont des caractéristiques typiques du niveau macroscopique de la topologie de l'Internet.

## 2.4 Techniques d'attaques

Avant d'expliquer comment nous attaquons la connectivité d'Internet, quelques éléments doivent être précisés concernant les limitations évidentes de notre étude. Les limitations les plus importantes sont :

- les cartes d'Internet que nous utilisons sont incomplètes. Voir les détails en section 1.5. Ceci a un impact direct sur la robustesse de la connectivité des réseaux de routeurs et d'AS.
- les cartes de routeurs produites par Mercator (ou tout autre outil de mesure de topologie IP) ne devraient pas être vues comme contenant seulement de vrais routeurs et de vraies connexions point-à-point mais simplement comme des cartes contenant des noeuds IP et des liens IP. Là encore, voir les détails en section 1.5.

– nous pouvons seulement mesurer la destruction du réseau en mesurant la taille de ses composantes connexes. Ceci signifie que chaque noeud a la même valeur. Ce n'est certainement pas vrai en réalité. En effet la quantité d'information valable disponible sur le réseau local d'un routeur donné ne sera probablement pas être égale à celles des autres routeurs. Par exemple, si un serveur très populaire est coupé de la plus grande partie du réseau (et aucun miroir n'existe) alors les utilisateurs se plaindront même si 90% de tous noeuds peuvent toujours communiquer. Cependant il est difficile d'assigner une valeur de qualité ou d'importance de l'information aux noeuds IP sans étude antérieure sur ce sujet.

Dans ce chapitre, nous n'étudions pas les pannes aléatoires de noeuds ou de liens. Ce sujet a déjà été étudié dans [AJB00], [TPSF01], et [PSF+01] et toutes ont montré que la connectivité globale de l'Internet est très résistante aux pannes aléatoires. Nous sommes plutôt intéressés par la suppression ciblée de noeuds. Ceci signifie supprimer des noeuds qui sont importants pour la connectivité du réseau afin de l'éclater. Nous n'étudions pas la suppression ciblée de liens pour le moment parce qu'il est plus difficile de caractériser l'importance topologique d'un lien et l'impact de sa suppression est habituellement moins efficace. En effet la suppression d'un noeud supprime simultanément tous ses liens adjacents. En réalité, la suppression d'un noeud ne serait sûrement pas quelqu'un venant dans une salle machine et détruisant les équipements. La destruction de l'Internet, comme nous le montrons plus tard, exige la suppression de centaines voire de milliers de routeurs pour être réalisable et ceci ne peut pas être fait physiquement. Nous envisageons qu'une attaque serait possible seulement par l'utilisation d'outils logiciels spécifiques comme dans le cas d'attaques par déni de service distribué (cf. chapitre quatre). Les routeurs ne seraient pas vraiment détruits mais rendus inutilisables par l'injection de paquets forgés qui causeraient des dégats dans leurs bases d'information de routage ou dans le code de leurs protocoles de routage. Cette menace n'a pas été négligée par l'IETF qui a produit plusieurs Request For Comments il y a trois ans sur l'authentification des messages des protocole de routage RIP-2 [BA97], OSPFv2 [MBW97] et BGP [Hef98].

Notre but n'est pas actuellement comment supprimer un noeud spécifique mais comment supprimer les noeuds de la façon la plus efficace possible afin de réduire au minimum la quantité de suppression de noeuds nécessaire pour atteindre un niveau donné de fragmentation. Les techniques d'attaque sur des noeuds peuvent être classifiées en deux grands groupes : statique et dynamique. Dans le groupe statique, chaque noeud se voit assigné une fois pour toutes une valeur d'importance basée sur un ou plusieurs critères. Les critères pour chaque attaque statique sont énumérés dans la table 2.2. Par exemple dans l'attaque de degré-distance, les noeuds sont évalués d'abord selon leur degré et ensuite selon leur distance moyenne. Plus le degré est haut, plus l'importance du noeud est haute. Si plusieurs noeuds ont le même degré, le noeud ayant la distance moyenne la plus faible se voit assigné la plus haute valeur, etc. Nous précisons que, comme nous l'avons dit plus tôt dans le paragraphe sur des limitations, l'importance d'un noeud est seulement évaluée en utilisant des critères topologiques (*e.g.*, le degré, le noeud est une racine, le noeud est dans le maillage). Les noeuds sont alors retirés du réseau un par un par ordre décroissant d'importance. L'avantage de cette méthode est qu'elle est très rapide à calculer parce que les valeurs sont calculées seulement une fois au début. Toutes les techniques du groupe statique suivent un algorithme commun donné ci-dessous :

```
Groupe des attaques statiques
1. Déterminer la valeur de l'importance de chaque noeud
2. Placer les noms de noeud dans une pile en ordre
   décroissant de leurs valeurs
3. Tant que (le réseau n'est pas vide)
4.     Extraire le nom du noeud supérieur de la pile
5.     Supprimer le noeud correspondant du réseau
6.     Analyser la connectivité du réseau
```

TAB. 2.2 – Critères de sélection des noeuds dans les attaques statiques

| Nom de l'attaque | Critère utilisé par ordre décroissant d'importance |
|---|---|
| Degré | Degré |
| Degré-topologie | Degré, racine, maillage, point d'articulation |
| Degré-distance | Degré, distance moyenne |

TAB. 2.3 – Critères de sélection des noeuds dans les attaques dynamiques

| Nom de l'attaque | Critère utilisé par ordre décroissant d'importance |
|---|---|
| Adaptative | Plus grande composante connexe, degré |
| Adaptative-topologie | Plus grande composante connexe, point d'articulation, degré |

Dans le groupe dynamique, chaque noeud se voit assigné une valeur d'importance, au début et après chaque suppression d'un noeud, basé sur un ou plusieurs critères. Les critères pour chaque attaque dynamique sont énumérés dans la table 2.3. Par exemple dans l'attaque adaptative, les noeuds qui sont dans la plus grande composante connexe sont choisis d'abord et sont ensuite évalués selon leur degré. Ainsi cette attaque supprime le noeud de degré le plus élevé de la plus grande composante connexe (qui n'est pas nécessairement le noeud de degré le plus élevé du réseau). Le noeud le plus important est supprimé du réseau, la connectivité est évaluée et puis les valeurs d'importance sont calculées à nouveau pour trouver le noeud le plus important parmi les noeuds restants. L'avantage de cette méthode est qu'elle est très précise en supprimant des noeuds importants dans une situation topologique donnée (car cette derniere évolue continuellement avec la suppression des noeuds). Cependant le temps de calcul est plus long. Toutes les techniques du groupe dynamique suivent un algorithme commun donné ci-dessous :

```
Groupe des attaques dynamiques
1. Déterminer la valeur de l'importance de chaque noeud
2. Choisir le noeud qui a la valeur la plus élevée
3. Tant que (le réseau n'est pas vide)
4.      Supprimer le noeud choisi du réseau
5.      Analyser la connectivité du réseau
6.      Recalculer la valeur de l'importance de chaque
        noeud restant en fonction de la nouvelle connectivité
7.      Choisir le noeud avec la valeur la plus élevée
```

Bien que les auteurs des articles précédents n'aient pas détaillé exactement comment ils ont effectué les attaques, ils ont indiqué avoir utilisé les suivantes :

- [AJB00], [TPSF01] et [BC01] ont tous utilisé l'attaque des noeuds de plus haut degré d'abord.
- [BC01] a également utilisé l'attaque des noeuds de plus faible distance moyenne d'abord.
- [PSF+01] a également utilisé l'attaque des noeuds ayant l'exposant du nombre de sauts le plus élevé d'abord.

Pour conclure, nous employons cinq techniques d'attaques : les attaques statiques du degré, de degré-topologie et de degré-distance ; et les attaques dynamiques adaptatives et adaptative-topologie. L'attaque du degré a été déjà utilisée dans des travaux précédents tandis que les quatre autres sont nouvelles (l'attaque de degré-distance n'est pas identique à l'attaque de distance).

## 2.5 Expérimentations

Les expériences consistent à tester nos cinq techniques d'attaques sur les trois cartes d'Internet à notre disposition avec une attention plus forte sur les deux cartes de niveau de routeur. Les expériences prouvent que toutes les attaques statiques donnent des résultats très semblables. De la même manière, toutes les attaques dynamiques donnent également les résultats très semblables. Par conséquent, nous présentons seulement dans cette section les résultats concernant l'attaque du degré (représentant le groupe statique) et l'attaque adaptative (représentant le groupe dynamique). En outre dans certains cas les résultats produits avec la carte SCAN-LUCENT'99 sont très proches de ceux produits avec la carte LSIIT'02. Dans ces cas-là, nous montrons seulement les résultats produits avec la carte SCAN-LUCENT'99 pour éviter la répétition.

### 2.5.1 Métriques utilisées

Trouver la métrique appropriée pour quantifier la connectivité n'est pas une tâche facile. Dans cette section nous présentons les métriques existantes, définissont les métriques que nous utilisons dans nos expériences et expliquons pourquoi nous choisissons celles-ci. Dans [AJB00], les auteurs utilisent le diamètre en tant que métrique d'interconnexité d'un réseau. Ils le définissent comme étant la « longueur moyenne des plus courts chemins entre deux noeuds quelconques". Ceci est habituellement appelé la longueur moyenne des chemins ou la distance moyenne d'un graphe, le diamètre étant habituellement défini dans la théorie des graphes [Har69] comme étant la longueur maximum des plus courts chemins entre toutes les paires de noeuds possibles. Néanmoins cette métrique peut être intéressante car l'inflation de la distance moyenne montre comment la connectivité du réseau est endommagée, mais même si un chemin est rallongé de 100% en raison des suppressions d'éléments du réseau, on peut encore communiquer. En outre la distance utilisée ici est habituellement la métrique du compte du nombre de sauts et elle n'est pas nécessairement proportionnelle à la métrique du délai ou à celle de la largeur de bande passante. L'utilisation d'un service d'évaluation des distances comme IDMaps [FJJ$^+$01] ont pu rendre possible l'utilisation du délai au lieu du nombre de sauts. Plusieurs métriques de distances telles que le diamètre et la distance moyenne sont également utilisées dans [BC01] cependant ceci ne nous informe toujours pas sur la vraie perte de connectivité parmi des noeuds. Nous voulons étudier ce que nécessite le blocage de n'importe quelle transmission entre le plus grand nombre de noeuds. Les auteurs de [AJB00] surveillent également la taille moyenne des groupements isolés (*i.e.*, tous les groupements excepté le plus grand). Cependant ce n'est pas vraiment une métrique intéressante parce que la distribution des grandeurs des groupements est très erratique (*i.e.*, elle semble être à queue lourde). Ceci signifie que les valeurs moyennes ou médianes de cette distribution ne la reflètent pas exactement. En fait, mesurer cette métrique produit toujours des valeurs comprises entre 1 et 2 pour la taille moyenne et 1 pour la taille médiane alors que des groupements de cent ou mille noeuds peuvent être présents.

Une autre métrique utilisée dans [AJB00], dans [TPSF01] et dans [BC01] est la taille de la plus grande composante connexe. C'est une métrique très intéressante parce qu'elle donne une limite supérieure sur le nombre de noeuds qui peuvent communiquer, ainsi nous utilisons cette métrique dans nos expériences. Un réseau à dimension libre tel que l'Internet est habituellement composé d'une forêt (au sens de la théorie des graphes, donc un ensemble d'arbres) et d'un maillage (*i.e.*, une partie contenant des cycles et des isthmes). La répartition est en général de 2/3 de forêt pour 1/3 de maillage. D'ailleurs la plus grande partie du maillage se compose typiquement d'une composante biconnexe géante (*i.e.*, dans laquelle il existe au moins deux chemins disjoints entre n'importe quelle paire de noeuds). Dans notre carte de 2002, la composante biconnexe représente 82.9% du maillage. Tout ceci signifie que briser la connectivité de

TAB. 2.4 – Définition des classes

| Numéro de classe | Taille du groupement |
|---|---|
| classe 0 | 1 |
| classe 1 | 2-10 |
| classe 2 | 11-100 |
| classe 3 | 101-1000 |
| classe 4 | 1001-10000 |
| classe 5 | 10001-100000 |
| classe 6 | 100001 and above |

cette partie du réseau est probablement difficile et étudier son rétrécissement durant une attaque peut fournir des informations intéressantes.

La métrique la plus intéressante à regarder serait la distribution des composantes connexes elle-même. Cependant cette distribution contient habituellement trop de valeurs à manipuler facilement. C'est pourquoi nous avons défini des groupes de valeurs de taille de composantes connexes afin d'avoir une représentation plus concise de la distribution des composantes connexes. Nous appelons ces groupes des *classes* et elles sont définies dans la table 2.4. Par exemple, une composante connexe dans la classe 2 contient au moins 11 noeuds et tout au plus 100 noeuds. Comme rappel, le numéro de classe est le logarithme décimal de la taille de la limite supérieure de cette classe. Nous choisissons des seuils absolus et non des seuils relatifs (*i.e.*, exprimés en % de la taille du réseau) parce qu'ils contiennent beaucoup plus d'information. Si une composante connexe est de la classe 0 ou 1, nous avons une idée précise de son état particulièrement au sujet des distances. Deux noeuds quelconques dans une composante connexe de classe 1 seront tout au plus à 9 sauts l'un de l'autre. Si nous définissons une classe avec une taille relative maximum telle que 0.1%, ceci représente une limite supérieure de 284 noeuds dans la carte de SCAN-LUCENT. Une composante connexe de cette taille peut être très répandue et comme le diamètre mesuré dans des cartes de niveau routeur de l'Internet est habituellement autour de 30 sauts, nous ne pouvons pas déduire comment les longueurs des chemins sont limitées dans cette composante. Nous calculons les composantes connexes en utilisant une version modifiée de l'algorithme de Tarjan [AHU83] valide pour les graphes non orientés [Baa88] et ayant la même complexité.

Classifier les composantes connexes dans des classes donne une vision intéressante sur le niveau de fragmentation du réseau mais nous proposons une métrique qui renseigne encore mieux sur la situation d'un noeud donné dans le réseau. Cette métrique est la distribution du nombre relatif de noeuds par classes (*i.e.*, le nombre des noeuds par classes exprimé comme une fraction de la taille totale du réseau). Cette métrique nous permet de répondre à la question : "quel est le % de probabilité qu'un noeud puisse communiquer avec au plus un nombre donné (*i.e.*, une limite supérieure) de noeuds pour un % donné de noeuds supprimés ? ". Nous calculons la distribution en étiquetant chaque noeud appartenant à la même composante connexe avec un identifiant commun. Nous pouvons alors calculer la taille relative de chaque classe en ajoutant les tailles (*i.e.*, nombre de noeuds) de chaque composante connexe appartenant à la même classe.

Pour conclure nous employons trois métriques : la taille de la plus grande composante connexe (qui a été déjà utilisée dans des travaux précédents), la distribution de la fréquence des classes de composantes connexes et la distribution du nombre relatif de noeuds selon leur classe, ce qui forme deux nouvelles métriques.

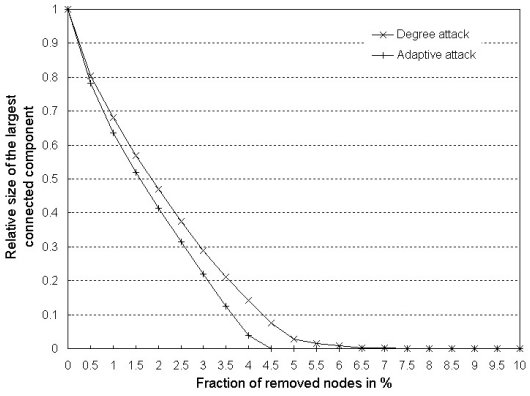


FIG. 2.5 – Evolution de la plus grande composante connexe dans SCAN'99.

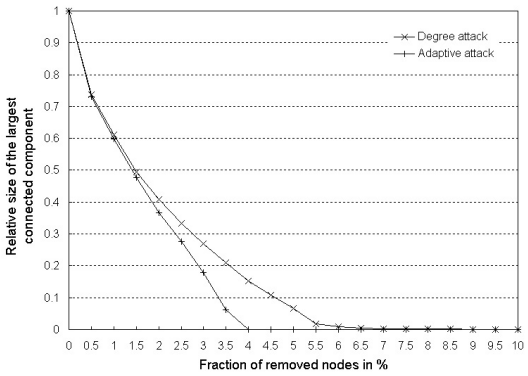


FIG. 2.6 – Evolution de la plus grande composante connexe dans LSIIT'02.

### 2.5.2 Robustesse de la plus grande composante

Nous étudions l'évolution de la taille relative de la plus grande composante connexe. La taille est exprimée comme fraction du nombre total initial de noeuds dans le réseau. La figure 2.5 montre cette évolution pour les attaques du degré et adaptative. Les deux tracés ont grosso modo la même forme. La taille diminue abruptement jusqu'à un seuil où la taille est proche de zéro. On a déjà observé ce phénomène dans [AJB00]. Savoir que le réseau peut être détruit en supprimant approximativement 5% de ses noeuds peut sembler effrayant. L'attaque adaptative diffère seulement de manière significative de l'attaque du degré dans la zone moyenne du praphe. Il réduit le seuil de 6.5% à 4.5% ce qui ne semble pas représenter un gain important par rapport à l'attaque du degré. Les attaques sur la carte LSIIT'02 illustrées dans la figure 2.6 sont plus efficaces, en particulier l'adaptative. Ceci peut être provoqué par le manque de liens redondants dans cette carte qui a un degré moyen de 2.5 comparé au degré moyen de 3.15 de la carte SCAN-LUCENT'99.

La figure 2.7 et la figure 2.8 présentent l'évolution de la taille relative de la plus grande composante connexe tracée sur une échelle logarithmique décimale. Cette représentation est très intéressante parce qu'elle met en valeur le fait que l'attaque adaptative est beaucoup plus efficace que l'attaque du degré. En effet pour la carte SCAN-LUCENT'99 à 4% de suppression de noeuds, la taille de la plus grande composante connexe est déjà un ordre de grandeur inférieur sous attaque adaptative que sous l'attaque du degré. À 5.5% cet intervalle atteint 3 ordres de grandeur. Même à 10%, il y a toujours un intervalle d'un ordre de grandeur. Les résultats avec la carte LSIIT'02 présentent un comportement semblable bien que le tracé de l'attaque du degré soit plus erratique.

### 2.5.3 Fragmentation des composantes connexes

La fraction du nombre de composantes connexes pour une classe donnée en fonction du nombre total de composantes connexes est donné dans la figure 2.9 pour la carte SCAN-LUCENT'99 sous une attaque du degré. Après suppression de 0.5% de noeuds, la plupart des classes ne contiennent pas une quantité significative de composantes connexes. Seulement la classe 0 et la classe 1 composent la majeure partie de toutes les composantes connexes.

Ceci montre clairement que l'attaque du degré produit principalement des noeuds isolés ou des composantes connexes très petites (ayant moins de 10 noeuds). Ceci signifie que même une attaque ciblée telle que supprimer le noeud de degré le plus élevé ne réussit pas à briser le réseau en plusieurs parties de taille égale. La figure 2.10 présente un comportement semblable

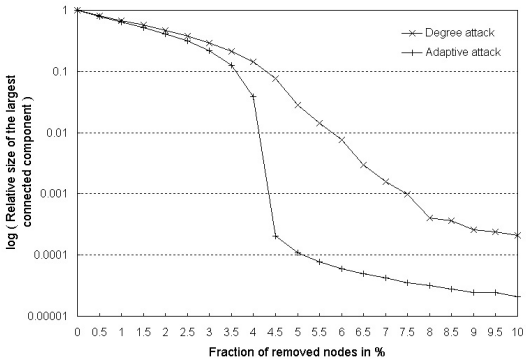


FIG. 2.7 – Evolution de la plus grande composante connexe dans SCAN'99 (log).

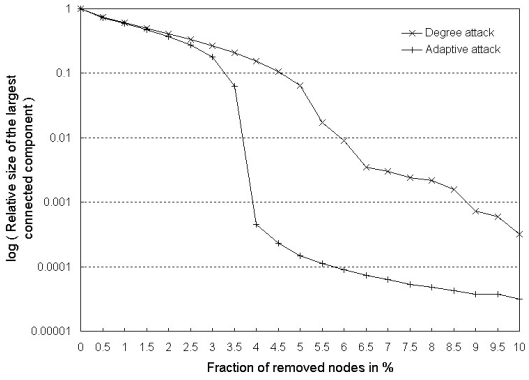


FIG. 2.8 – Evolution de la plus grande composante connexe dans LSIIT'02 (log).

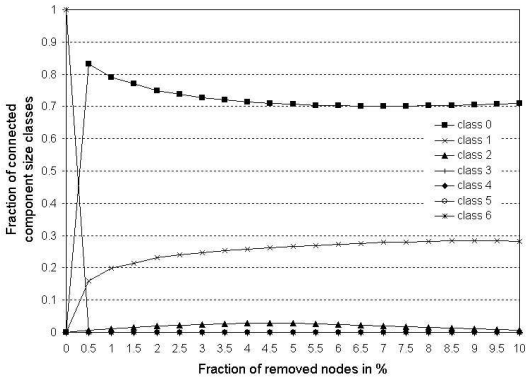


FIG. 2.9 – Fraction des classes de composantes (attaque du degré).

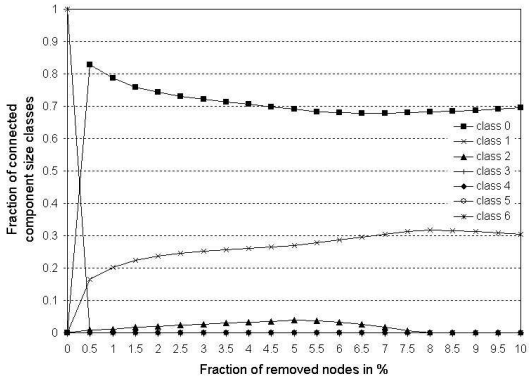


FIG. 2.10 – Fraction des classes de composantes (attaque adaptative).

pour l'attaque adaptative. Nous trouvons des résultats semblables pour la carte LSIIT'02.

Il aurait été très intéressant d'analyser les diamètres des composantes connexes pour avoir une idée de quelle distance nous pouvons nous déplacer à l'intérieur d'une composante connexe (en fonction de sa taille par exemple). Comme le diamètre du niveau routeur de l'Internet est petit (autour de 30 sauts), les composantes connexes de la classe 2 (jusqu'à 100 noeuds) ou plus pourraient encore couvrir le réseau entier même si ils ne peuvent pas atteindre beaucoup d'autres noeuds. Les résultats cités dans [PSF+01] au sujet de l'exposant du saut tendent à indiquer fortement que ce n'est pas le cas. La taille du voisinage en fonction du nombre de sauts diminue beaucoup sous attaque ce qui en même temps que nos résultats tend à montrer que les composantes connexes des classes 2 à 4 (classes moyennes) ont probablement un petit diamètre. En outre, en raison du coût de calcul des diamètres pour chaque composante connexe, nous choisissons de laisser cette étude à nos travaux futurs.

La fraction du nombre de composantes connexes pour une classe donnée en fonction du nombre total de composantes connexes est présenté dans les figures 2.11 et 2.12 et tracé sur une échelle logarithmique décimale pour la carte SCAN-LUCENT'99. Encore une fois cette représentation met en valeur le fait que l'attaque adaptative est beaucoup plus efficace que l'attaque du degré.

Dans l'attaque adaptative, les composantes sont cassées beaucoup plus rapidement et en beaucoup plus petites tailles que sous l'attaque du degré. Par exemple après suppression de 4.5% de noeuds, aucune composante de la classe 3 ou plus ne subsiste dans le réseau alors que

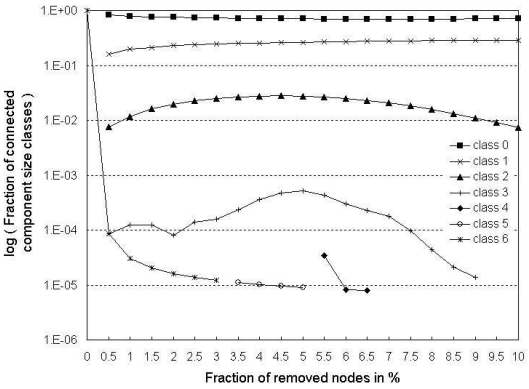


FIG. 2.11 – Fraction des classes de composantes (attaque du degré, échelle log).

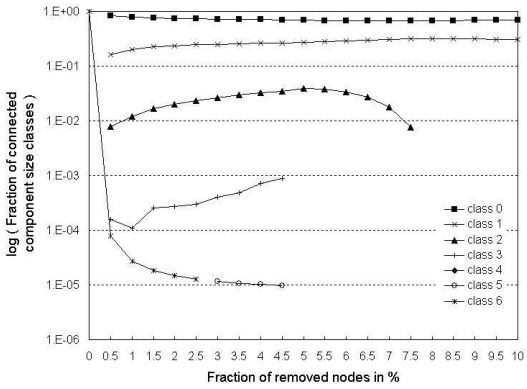


FIG. 2.12 – Fraction des classes de composantes (attaque adaptative, échelle log).

les composantes de la classe 3 existent jusqu'à 9% de suppression de noeuds sous une attaque du degré. Nous trouvons les résultats semblables pour la carte LSIIT'02.

### 2.5.4 Distribution des noeuds

Nous présentons ici comment les noeuds eux-mêmes sont distribués parmi les classes. La figure 2.13 présente cette distribution pour la carte SCAN-LUCENT'99 sous une attaque du degré. Cette figure présente comment la réduction de la plus grande composante connexe produit des noeuds dans les classes 0 à 2. Les classes 3 à 5 semblent vraiment capturer l'état transitoire de la plus grande composante vers sa destruction. La distribution sous une attaque adaptative est présentée sur la figure 2.14 et présente quelques différences avec une distribution sous une attaque du degré. En particulier les tracés pour les composantes des classes 1 et 2 ne fléchissent pas autant que pour une attaque du degré et ils semblent même être plutôt linéaires quand la fraction de suppression de noeuds augmente au delà de 5%. En conséquence, à 8% de suppression de noeuds, seules des composantes de classes 0 et 1 demeurent dans le réseau sous une attaque adaptative tandis que des composantes de classes 0 à 3 peuvent encore être trouvées sous une attaque du degré. En outre l'existence de classes intermédiaires 3 à 5, en raison de la décomposition de la plus grande composante, est beaucoup plus brève.

Ces figures sont très intéressantes dans le sens où elles fournissent le % de probabilité d'être dans une situation donnée de fragmentation. Par exemple sous une attaque adaptative qui a supprimé 3% de noeuds, vous avez 25% de probabilité d'être encore connexe à au plus 9 autres noeuds, tandis qu'à 6.5% cette probabilité atteint 50% ! Nous trouvons des résultats très semblables pour la carte LSIIT'02.

## 2.6 Conclusion

Dans ce chapitre, nous avons proposé plusieurs nouvelles techniques pour attaquer la connectivité de niveau IP de l'Internet ainsi que des nouvelles métriques pour évaluer la quantité de fragmentation dans le réseau et dans la distribution des routeurs parmi les composantes connexes. Nous avons montré que sous attaque adaptative, la suppression de seulement 4% des routeurs laisse des groupements de tout au plus 100 routeurs dans une carte étant 3 ordres de grandeur plus grande. De la même manière, la suppression de 1.5% des routeurs fait que le plus grand groupement se réduit à moins que la moitié de la taille initiale du réseau.

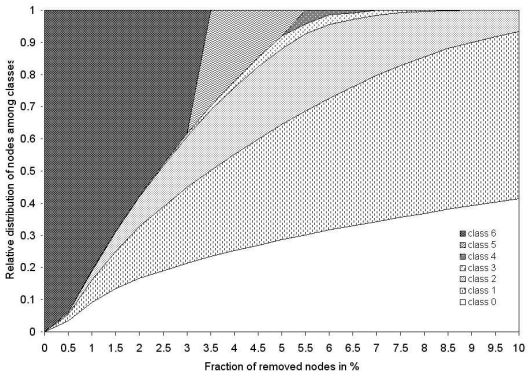


FIG. 2.13 – Distribution des noeuds parmi les classes (attaque du degré).

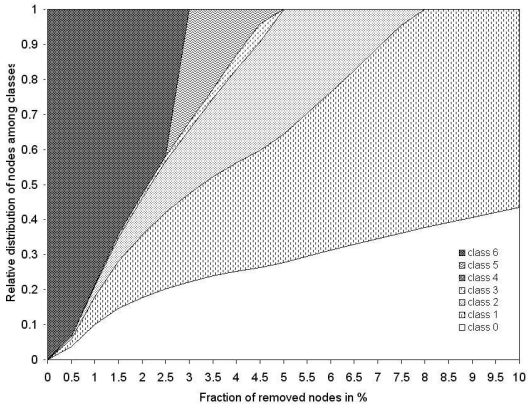


FIG. 2.14 – Distribution des noeuds parmi les classes (attaque adaptative).

En dépit de ces résultats inquiétants, nous avons également montré dans [Mag03b] que ces seuils dépendent de la taille du réseau ce qui signifie que détruire l'Internet lui-même nécéssiterait des attaques simultanées sur des centaines de milliers de routeurs. En outre, nous avons découvert toujours dans [Mag03b] que les cibles ne sont pas concentrées dans des AS spécifiques mais qu'elles sont réparties sur des milliers d'entre eux et que la qualité de la carte de niveau de routeur est cruciale pour viser avec précision les meilleures cibles. Nous précisons que la topologie de niveau routeur de l'Internet est très difficile à obtenir et reste toujours le coeur de beaucoup de projets actuels de recherches. De plus, grâce à notre méthode de recouvrement, nous avons montré que notre carte de niveau routeur de l'Internet était incomplète malgré le fait qu'elle contenait plus de 180k noeuds collectés sur plusieurs mois. Nous avons montré que les attaques dépendent fortement de la vue partielle qu'elles possèdent de la topologie. Ceci permet une attaque focalisée sur une zone efficace, mais rend une attaque globale presque impossible.

Tous ces facteurs nous amènent à penser qu'entreprendre une attaque massive sur la connectivité de l'Internet peut ne pas être faisable actuellement. D'ailleurs les résultats de notre étude pourraient être employés pour augmenter la robustesse du réseau en déterminant comment placer des liens ou des routeurs redondants afin d'élever les seuils de destruction. Ils pourraient également être utilisés dans le contexte du placement des miroirs [JJK+01, CJJ+02] pour définir où localiser des miroirs accessibles par un nombre maximum d'hôtes quand le réseau est à un niveau donné de fragmentation. Bien que l'étude de la robustesse de la connectivité de l'Internet soit une problématique récente, l'intérêt croissant concernant la sécurité donnera probablement à ce sujet un rôle important dans un avenir proche.

Comme précisé à plusieurs endroits dans ces deux premiers chapitres, la création des recouvrements routeurs–AS et les simulations des différents scénarios d'attaques ont été implémentés dans le logiciel *nem* [Mag], car cet outil contenait déjà un bon nombre de fonctions d'analyse topologique des cartes (*e.g.*, degrés, distances, connexité). *nem* est à la fois un analyseur, un générateur [MP01d] et un simulateur [Mag02, Mag05] que j'ai commencé à créer durant mon travail de doctorat et que j'ai utilisé à maintes reprises pour faire des simulations. Il est toujours utilisé actuellement par Mickaël Hoerdt et moi-même pour nos travaux de recherche. L'étude de la résistance de l'Internet présentée dans ce chapitre a pu être menée à bien grâce aux nombreux échanges fructueux ayant eu lieu entre Hyunseok Chang et Sugih Jamin de l'University of Michigan et moi-même.

# Travaux relatifs au chapitre


1. Damien Magoni. Network topology analysis and Internet modelling with *nem*. *International Journal of Computers and Applications*, 27(4) : 252–259, October 2005.
2. Damien Magoni. Tearing down the Internet. *IEEE Journal on Selected Areas in Communications*, 21(6) : 949–960, August 2003.
3. Damien Magoni. Robustness performance of the Internet connectivity under targeted attacks. In *Proceedings of the 4th International Conference on Communication Technology*, pages 1440–1443, Beijing, China, April 2003.
4. Damien Magoni. *nem* : a software for network topology analysis and modeling. In *Proceedings of the 10th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 364–371, Fort Worth, Texas, USA, October 2002.
5. *network manipulator (nem)*.
   http ://dpt-info.u-strasbg.fr/~magoni/nem/

# Deuxième partie

# Stockage des états dans les topologies des groupes multipoints

# Chapitre 3

# Répartition des états multipoints par utilisation des plus courts chemins alternatifs

## 3.1 Introduction

L'étude des protocoles de diffusion multipoint est un sujet majeur de l'équipe réseaux du LSIIT dirigée par le professeur Jean-Jacques Pansiot. En effet, mon travail de doctorat a consisté à définir un nouveau protocole de diffusion multipoint destiné à la recherche de fonctionnalités dans le réseau (OMP-NASP) et il s'est donc trouvé au coeur de cette problématique. De même, le travail de doctorat de Mickaël Hoerdt que j'ai co-encadré avec Jean-Jacques Pansiot consiste à améliorer l'extensibilité sur le nombre d'états des protocoles de routage multipoint en développant une extension de PIM-SSM [EFH+98] qui utilise les chemins multiples lors de la construction de l'arbre de routage interdomaine. Désormais armés d'une solide connaissance de la topologie de l'Internet, acquise dans la première partie de ce mémoire, nous pouvons étudier de manière approfondie la problématique du stockage des états dans les topologies des groupes multipoints. Cette étude constitue le point de départ des travaux de Mickaël Hoerdt.

Le manque de ressources dans les routeurs va devenir un problème majeur avec le déploiement de protocoles stockant des états tels que les protocoles multipoints (*e.g.*, PIM [EFH+98], CBT [Bal97a], MOSPF [Moy94], DVMRP [WPD88b]) et ceux de qualité de service (*e.g.*, RSVP [BZB+97]). En particulier, les protocoles multipoints à sources multiples ou à source unique vont atteindre à plus ou moins brève échéance, les limites des capacités des routeurs pour pouvoir maintenir les informations de topologie des arbres multipoints (*e.g.*, PIM [EFH+98], CBT [Bal97b]). L'objectif de ce chapitre est d'étudier la possibilité et le bénéfice de l'utilisation des plus courts chemins multiples lors de l'adhésion d'un nouveau membre à un arbre multipoint. Un tel mécanisme ne réduira pas la quantité totale des informations d'état dans le réseau mais il distribuera cette quantité de manière plus équilibrée sur les routeurs. L'idée est d'utiliser un plus court chemin alternatif fourni par le système de routage sous-jacent pour éviter un routeur qui ne peut pas ou ne souhaite pas stocker d'informations d'états multipoints supplémentaires. Nous avons simulé notre mécanisme sur des topologies de type Internet et nous avons trouvé que selon la taille des arbres, il peut augmenter les chances d'adhésion d'un nouveau membre jusqu'à 55% lorsque le réseau est à moitié saturé.

En effet les protocoles multipoints qui sont à stockage d'états (*i.e.*, tous ceux qui se trouvent dans la couche réseau), posent un problème de passage à l'échelle dans le réseau car ils ont besoin de stocker des informations dans des noeuds intermédiaires du réseau, les routeurs, qui n'étaient pas étudiés à l'origine pour réaliser de telles tâches. Cependant, Chalmers *et al.* ont

montré dans [CA02] que le multipoint [Dee89] est plus performant que le point à point au niveau de la bande passante à partir de quelques récepteurs seulement. Entre 20 et 40 récepteurs, le multipoint est 55% plus efficace que le point à point, 75% avec 150 récepteurs et jusqu'à 90% pour des groupes de 1000 récepteurs. Donc il paraît intéressant de fournir le support du multipoint dans le réseau en dépit du coût engendré par le stockage d'états. C'est pourquoi la communauté de recherche essaye encore de déployer des protocoles multipoints au niveau de l'Internet tout entier tout en essayant simultanément de réduire le plus possible la quantité d'états stockés dans le réseau ou d'optimiser l'emplacement de ces états. Notre solution permet d'optimiser l'emplacement de ces états multipoints en les localisant sur des plus courts chemins alternatifs (*i.e.*, sur des plus courts chemins autres que ceux donnés par défaut en point à point par le protocole de routage). Ce chapitre contient deux sections principales. Dans la section 3.3 nous décrivons le principe fondamental de notre solution et fournissons des éléments permettant son implémentation. Puis en section 3.4 nous présentons des résultats obtenus par simulation pour évaluer la pertinence de notre solution.

## 3.2 Contexte

En dépit de l'affirmation de Holbrook *et al.* qu'un routeur peut supporter un million de canaux actifs multipoints à source unique [HC99], le problème du passage à grande échelle du stockage des états multipoints dans les routeurs reste préoccupant pour la communauté de recherche en réseaux. Ce problème a été étudié au niveau inter-domaine par Wong *et al.* dans [WK00]. Ils ont trouvé que l'augmentation de la coopération (peering) parmi les réseaux formant le coeur d'Internet a entraîné une forte augmentation des états multipoints dans une poignée de domaines de coeur mais une faible augmentation dans les autres domaines. Ils ont aussi découverts que le nombre d'états multipoints est lié à la densité de sessions par une loi puissance (*i.e.*, la fraction des routeurs stockant des états est proportionnelle à la taille des groupes multipoints élevée à une puissance constante). Leurs résultats ont été corroborés par Chalmers *et al.* dans [CA02] qui ont trouvé que la majorité des arbres multipoints inter-domaines partagent une forme commune aux niveaux routeur et système autonome. En d'autres termes, ils ont découvert que l'éventail des formes possibles des arbres multipoints inter-domaines est contraint par la topologie sous-jacente. Plusieurs solutions ont déjà été proposées pour résoudre le problème du passage à l'échelle des états multipoints. Tian *et al.* ont défini un système de tunnel multipoint dynamique dans [TN98] pour éviter de stocker des informations multipoints dans les routeurs de transit (*i.e.*, non-branchants). Thaler *et al.* ont proposé un schéma d'agrégation des états multipoints dans [TH00] pour réduire le nombre d'entrées multipoints, tandis que Zappala *et al.* a étudié des techniques de routage multipoint par des chemins alternatifs dans [Zap00]. Plus récemment, Fei *et al.* dans [FCGF01a, FCGF01b] et Jun-Hong Cui *et al.* dans [CMK+02] ont proposé plusieurs techniques destinées à agréger les états multipoints. Cependant, aucune de ces méthodes n'est liée à notre approche d'adhésions alternatives par plus courts chemins multiples qui entraîne une distribution plus efficace des états.

Ce chapitre propose une nouvelle solution qui peut être combinée avec les solutions précédentes et qui a été conçue pour :

- augmenter les chances du message d'adhésion (appelé message *join*) d'un nouveau membre d'atteindre l'arbre multipoint désiré en évitant les routeurs bloquants,
- distribuer les états parmi les routeurs en les stockant dans des routeurs acceptant encore de créer de nouveaux états.

Nous insistons sur le fait que notre solution ne réduit pas la quantité d'états multipoints mais elle répand ces états sur un plus grand nombre de routeurs ce qui améliore leur répartition dans le réseau. Notre méthode peut être utilisée par les protocoles multipoints en mode épars

qui gèrent la construction et la maintenance d'arbres multipoints tels que [EFH+98, HC99]. Notons que tout protocole utilisant une structure d'arbre pour sa distribution des données ainsi qu'un mécanisme d'adhésion par chemin point à point orienté vers la source peut bénéficier des résultats de notre étude.

## 3.3 Utilisation des chemins multiples pour rejoindre des arbres multipoints

Dans cette section nous décrivons les hypothèses que nous faisons concernant le fonctionnement de notre solution. Nous étudions aussi les capacités de stockage d'états multipoints de routeurs populaires et définissons l'expression de routeur "saturé". Enfin, nous décrivons le fonctionnement de notre solution ainsi que des indications sur son implémentation.

### 3.3.1 Hypothèses

Nous supposons d'abord que nous utilisons l'Internet, que les hôtes utilisent la suite de protocoles TCP/IP et que les paquets sont acheminés par des routeurs. Nous supposons que des protocoles nécessitant un stockage d'états (*e.g.*, Intserv, protocoles multipoints) sont utilisés et occupent de la mémoire dans les routeurs. Un routeur saturé est un routeur qui refuse de créer de nouvelles entrées d'états multipoints peu importe la raison.

On suppose qu'un nouveau membre souhaite joindre un groupe multipoint et par conséquent son arbre multipoint correspondant. Dans nos simulations, nous construisons les arbres multipoints comme spécifié dans la norme Protocol-Independant Multicast - Single Source Multicast (PIM-SSM), car c'est le protocole le plus utilisé actuellement et celui qui a le plus de chance d'être déployé à grande échelle dans un avenir proche. La création d'un arbre multipoint se passe de la façon suivante : on choisit aléatoirement une source, puis on ajoute des membres (*i.e.*, récepteurs) un par un et nous stockons un état dans chaque routeur traversé le long du chemin point à point de chaque récepteur vers la source si il ne possède pas déjà le même état (*i.e.*, si le routeur n'appartient pas encore à cet arbre). Donc l'arbre obtenu est un arbre des plus courts chemins enraciné à la source multipoint. Quand nous construisons un arbre nous ne considérons pas les routeurs saturés, nous supposons que l'arbre a été créé sans problème. Ce n'est que lorsque nous simulons l'envoi d'un message *join* d'un nouveau récepteur (*i.e.*, son routeur désigné) jusqu'à la source et donc son arbre correspondant que nous prenons en compte les routeurs saturés.

Dans le cas d'un protocole multipoint à source quelconque (Any-Source Multicast (ASM)) tel que PIM-SM, le message *join* serait ciblé vers un Rendez-vous Point (RP) au lieu de la source multipoint. De plus dans ce dernier cas, l'arbre serait un arbre des plus courts chemins enraciné au RP. Nous n'avons pas encore étudié l'impact de ces différences dans nos simulations. Cependant les résultats de Chalmers *et al.* nous encourage à penser que les deux types d'arbres auraient toujours des topologies similaires. La densité de session est égale au nombre de récepteurs dans l'arbre exprimé en pourcentage du nombre total de routeurs. Donc la taille d'un arbre est liée à la densité de session mais pas égale à cette dernière (*i.e.*, les noeuds dans l'arbre qui ne sont pas récepteurs doivent aussi être comptés).

Pour finir, nous supposons que le protocole de routage point à point sous-jacent nous fournit toutes les informations sur les **plus courts chemins multiples**. Cela signifie que, pour une destination donnée et dans n'importe quel routeur du réseau, le système de routage est capable de nous indiquer tous les prochains routeurs (*i.e.*, *next hop routers* en anglais) qui mènent à cette destination par des plus courts chemins (*i.e.*, tous ces chemins ont donc la même longueur). Les chemins ne sont pas forcément disjoints au sens de la théorie des graphes (*i.e.*, les chemins

| Vendeur | Série | Moteur de route | Mémoire maxi. | Nb. maximum d'états multipoints |
|---|---|---|---|---|
| Cisco | 12000 | PRP-2 | 4GOctets | 256K |
| Juniper | E | SRP-40G+ | 2GOctets | 128K |

TAB. 3.1 – Capacités de stockage d'états des routeurs

peuvent avoir des parties communes). Des protocoles de routage intra-domaine tels que OSPF [Moy98], IGRP et IS-IS peuvent fournir ces informations de chemins multiples alors que RIP [Mal98] ne le peut pas.

### 3.3.2 Saturation d'un routeur

Il est très difficile d'obtenir des informations précises concernant les capacités de stockage d'états des routeurs fabriqués par les grandes companies de matériel réseau. La table 3.1 présente quelques informations collectées sur les sites web de Cisco et Juniper pour les meilleures configurations matérielles possibles disponibles début 2004. Cisco annonce qu'un routeur 12810 peut stocker 256000 groupes multipoint mais il ne précise pas si cela est valable pour la capacité de mémoire vive (SDRAM) par défaut où pour la capacité SDRAM maximale. De manière similaire, Juniper spécifie qu'un routeur E-series peut stocker 16384 groupes multipoint mais il ne précise pas la quantité de mémoire nécessaire. Si nous supposons que les états sont stockés dans une carte SRP-5G contenant 256MB de SDRAM, nous en déduisons que stocker un groupe multipoint dans un routeur Cisco ou un routeur Juniper nécessite une moyenne de 16K octets. Cela semble beaucoup mais une implémentation de PIM-SM [EFH$^+$98] dans un routeur nécessite typiquement trois tables : une Multicast Routing Information Base (MRIB) pour gérer les plus courts chemins multipoint nécessaires au Reverse Path Forwarding [DM78], une Tree Information Base (TIB) pour la gestion de tous les arbres traversant ce routeur et une Multicast Forwarding Information Base (MFIB) dérivée de la TIB pour acheminer efficacement les paquets multipoints. De plus les tailles des entrées des deux dernières tables dépendent du nombre d'interfaces impliquées. Toutes ces structures de données consomment beaucoup de mémoire. Par conséquent, actuellement le nombre maximum d'états multipoints qui peuvent être stockés dans les routeurs haut de gamme d'aujourd'hui est de l'ordre de $10^5$. La plupart des routeurs auront des capacités d'un voire deux ordres de grandeur inférieurs aux routeurs haut de gamme.

D'un autre côté, la classe D contient un peu moins de $2^{28}$ (*i.e.*, plus de 268M) adresses multipoint potentiellement utilisables qui peuvent être utilisées par les protocoles ASM et avec les canaux définis dans PIM-SSM, le nombre d'adresses de groupes peuvent augmenter du nombre d'hôtes multiplié par $2^{24}$ (*i.e.*, plus de 16M) canaux par hôte. Donc si 1% des hôtes de l'Internet, ce qui aujourd'hui représente 1 à 3 millions d'hôtes, peuvent utiliser des protocoles multipoint et en particulier PIM-SSM, et si chaque hôte crée 5 canaux SSM, cela va produire 5M à 15M d'entrées différentes dans les routeurs traversés. Les routeurs de coeur de l'Internet seront traversés par une large part de ces arbres comme l'a montré Wong *et al.* dans [WK00] et donc ils devront aisément stocker plus de 50% de tous les états possibles. Cela signifie au moins 2,5M d'entrées à stocker pour ces routeurs de coeur, largement plus que ce que les routeurs haut de gamme actuels de la table 3.1 sont capable de supporter. Donc la saturation des routeurs peut clairement devenir un phénomène fréquent si le multipoint est massivement déployé et rendu accessible aux usagers. Si ce problème n'est pas très connu, cela est surtout dû au fait que la majorité des opérateurs et des fournisseurs d'accès n'offrent pas de services multipoint aux utilisateurs.

Nous définissons un routeur saturé comme étant un routeur qui ne peut plus (ou ne veut plus) stocker d'entrées d'états multipoints supplémentaires à un moment donné. Bien sûr, cet état de saturation varie dans le temps car les états multipoints sont créés et détruits dynamiquement. Dans nos expérimentations, nous adoptons une vue macroscopique et supposons que $x$ % des routeurs sont saturés à un instant donné $t$ (instantané). De plus les routeurs saturés sont choisis de façon purement aléatoire. Nous sommes conscients que si nous empilons un grand nombre d'arbres multipoints, d'après les résultats de Wong *et al.* [WK00], seuls quelques routeurs vont stocker un état pour presque chaque arbre (*i.e.*, surtout ceux situé dans le coeur du réseau) alors que les autres ne stockeront que quelques états au total. Cependant, ces routeurs fortement chargés seront très probablement bien équipés en ressources matérielles par leurs opérateurs alors que les autres routeurs plus éloignés du coeur de réseau seront plutôt sous équipés et donc plus prompts à saturer rapidement. C'est pourquoi nous pensons qu'il est satisfaisant de les choisir aléatoirement pour l'instant.

### 3.3.3 Principe de base et indications d'implémentation

Lorsque le nouveau membre veut joindre un groupe (*i.e.*, un arbre multipoint), il envoie un message *join* le long du chemin point à point menant à la source comme illustré sur la figure 3.1 et comme défini dans la norme de PIM-SSM. Dans notre solution, le message *join* contient un nouveau champ qui est un index définissant l'interface de sortie qui sera utilisée (parmi toutes les interfaces possibles menant à la source de l'arbre multipoint par des plus courts chemins) et que nous nommons l'*index multichemin* (ou *multipath index* en anglais). Cet index est mis à jour dans chaque routeur traversé, donc il n'est valable que pour le dernier routeur traversé avec succès. Sur la figure, le nouveau membre est à 4 sauts de la source. Le chemin point à point par défaut passe à gauche, cependant le deuxième routeur est saturé. Notre mécanisme implique que le routeur saturé renvoie un message au routeur précédent pour lui indiquer d'utiliser une route alternative. Le message *reject* contient l'adresse de la source de l'arbre multipoint ainsi que l'*index multichemin* non modifié (*i.e.*, qui conserve la même valeur que dans le message *join*). Lorsque le routeur précédent (*i.e.*, le routeur OK sur la figure) reçoit le message *reject*, il doit mettre à jour l'*index multichemin* à la prochaine interface de sortie qui sera utilisée et envoyer le message *join* sur cette interface menant à la source par un autre plus court chemin.

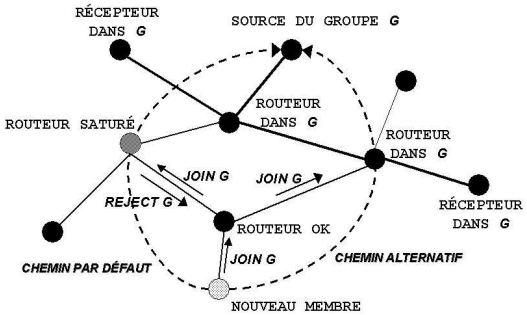


FIG. 3.1 – Nouveau membre utilisant les chemins multiples pour atteindre l'arbre.

Comme dit précédemment, pour pouvoir utiliser un autre plus court chemin que le plus court chemin par défaut, il faut que tous les plus courts chemins (si il y en a plusieurs) soient calculés et fournis par le système de routage point à point sous-jacent. Donc le système de routage doit connaître toutes les interfaces menant à une adresse IP donnée par différents chemins ayant la même longueur. Les protocoles de routage intra-domaine tels que OSPF, IGRP et IS-IS supportent la gestion des plus courts chemins multiples. Donc notre mécanisme peut être utilisé

dans tous les domaines utilisant les protocoles cités ci-dessus. Cependant RIP et le routage statique ne gèrent pas les plus courts chemins multiples donc notre mécanisme ne peut pas être utilisé dans ces deux derniers cas.

Le routeur utilise l'adresse de la source de l'arbre ainsi que l' *index multichemin* (*i.e.*, l'index de la dernière interface de sortie utilisée) pour déterminer la prochaine interface multichemin possible vers la source de l'arbre. Il envoie alors un nouveau message *join* au prochain voisin possible situé sur un plus court chemin vers la source (avec un *index multichemin* mis à jour). Dans notre exemple, à réception du message *reject*, le message *join* est envoyé au routeur de droite (*i.e.*, le routeur 2 dans l'arbre de la figure) et la tentative d'adhésion réussit (car ce routeur est déjà dans l'arbre). L'adhésion s'arrête là.

Un effet de bord intéressant de notre solution est qu'elle crée un état dans un routeur qui possède encore des ressources mémoires. En utilisant la mémoire de ces routeurs non-saturés, notre mécanisme fournit une meilleure utilisation des ressources du réseau en distribuant les états dans des routeurs qui auraient été sous-utilisés. Si sur n'importe quel routeur donné, toutes les tentatives pour joindre l'arbre échouent alors le processus s'arrête. Aucun retour en arrière n'a lieu, car cela nécessiterait de stocker plus d'information dans le paquet (*i.e.*, les index de tous les routeurs traversés) ainsi qu'un traitement plus complexe du paquet. Dans ce cas, l'adhésion échoue comme elle échouerait sans notre mécanisme lorsqu'un message *join* rencontre le premier routeur saturé sur sa route vers la source de l'arbre. Veuillez noter que sur la figure, il y a 1 routeur saturé pour 10 routeurs, donc la densité de saturation est égale à 10% et l'arbre contient 2 récepteurs donc la densité de session est égale à 20%. Comme tous les chemins alternatifs sont toujours des plus courts chemins (comme le chemin par défaut), le routage est toujours optimal ainsi que le chemin alternatif utilisé pour se connecter à l'arbre. De même si les tables de routage point à point sont cohérentes (*i.e.*, pas de boucles temporaires dues à des problèmes de routage), notre mécanisme ne créé pas non plus de boucles. Pour être efficace et réalisable, notre mécanisme doit être le plus simple possible. Nous n'avons pas besoin de créer un nouveau protocole complet pour cela. Nous pouvons implémenter notre mécanisme dans n'importe quel protocole de gestion d'arbre multipoint. Nous avons vu précédemment que nous avons besoin :

- de stocker l'*index multichemin* (*multipath index* en anglais) définissant la dernière interface utilisée) dans le message *join*,
- de définir un message *reject* contenant l'adresse de la source de l'arbre multipoint et le *multipath index*,
- un protocole de routage point à point gérant les plus courts chemins multiples.

Nous avons choisi de n'utiliser que les *plus courts* chemins multiples contrairement à Zappala *et al.* qui utilise des chemins de longueur non minimales (*e.g.*, plus court chemin plus 1 saut, plus 2 sauts) pour exploiter l'utilisation de chemins alternatifs dans le multipoint comme expliqué dans [Zap00]. En fait, il est beaucoup plus compliqué d'obtenir de l'information sur des chemins non minimaux et cela n'est pas géré par les protocoles de routage point à point, ce qui signifie qu'il faut les obtenir soi-même. Nous préférons n'utiliser que les informations fournies par le protocole de routage point à point sous-jacent pour que notre mécanisme soit pratique à mettre en oeuvre. La livraison des messages de notre mécanisme est une remise au mieux (telle que celle fournie par UDP), donc la fiabilité peut être implémentée en envoyant plusieurs messages avant de s'arrêter. Le pseudo-code de notre mécanisme est constitué de deux fonctions décrites dans la figure 3.2.

Dans une implémentation réelle tel que dans PIM-SSM par exemple nous avons seulement besoin de :

- modifier le message JOIN/PRUNE pour stocker l'*index multichemin* de la dernière interface de sortie utilisée (il peut être stocké dans le 9ème champ réservé de 8 bits ou dans un nouveau champ de 32 bits si 256 interfaces possibles sont insuffisantes),

```
A réception d'un paquet JOIN :
1. Si le routeur est "saturé"
2.   Envoyer un message REJECT au routeur précédent en utilisant
     l'interface de réception si elle est connue ou bien l'adresse du récepteur
3. Sinon
4.   Créer un état multipoint pour ce groupe/canal
5.   Déterminer l'interface de sortie qui mène à la
     source et fixer le MULTIPATH INDEX de façon adéquate
6.   Envoyer un message JOIN avec ce MULTIPATH INDEX
     par cette interface

A réception d'un paquet REJECT :
1. Trouver la prochaine interface multichemin de sortie utilisable
   en utilisant le MULTIPATH INDEX dans le paquet
2. Si toutes les interfaces menant à la source ont été essayées
3.   Arrêter la procédure JOIN
4. Sinon
5.   Mettre à jour le MULTIPATH INDEX dans le paquet à la
     prochaine interface de sortie qui mène vers la source
     et fixer le MULTIPATH INDEX de façon adéquate
6.   Envoyer un message JOIN avec ce MULTIPATH INDEX
     par cette interface
```

FIG. 3.2 – Algorithmes des plus courts chemins multiples.

– définir un message REJECT qui contient l'index de la dernière interface de sortie utilisée (le même index que ci-dessus) et l'adresse de la source de l'arbre multipoint.

Le point clé est que notre mécanisme ne nécessite pas de modification du mécanisme d'acheminement d'IP et, plus important encore, qu'il n'a pas besoin de stocker des informations d'états dans les routeurs donc il supporte le passage à grande échelle vis à vis de la capacité mémoire des routeurs.

## 3.4 Expérimentations

Dans cette section nous présentons les résultats obtenus par simulation pour évaluer l'efficacité de notre proposition. Il n'y a actuellement aucune technique fournissant un service identique ou équivalent à notre mécanisme. Donc nous ne pouvons pas comparer notre solution à d'autres approches. Nous ne pouvons que le comparer au mécanisme d'adhésion par défaut de PIM-SSM (*i.e.*, en utilisant le chemin point à point par défaut). Nous montrons que le fait de pouvoir utiliser des plus courts chemins alternatifs, à la place du plus court chemin point à point par défaut si celui-ci est bloqué, peut augmenter largement le taux de réussite des adhésions (*i.e.*, le taux de succès des messages *join*).

### 3.4.1 Paramètres

Garantir la précision des topologies utilisées pour nos simulations sur les multichemins et le multipoint est crucial. Ces topologies doivent être similaires à une carte Internet réelle concernant leurs propriétés topologiques, sinon les résultats des simulations pourraient être biaisés par la forme de la topologie sous-jacente comme nous l'avons montré dans [MP01c]. Nous avons utilisé une carte créée par notre logiciel de cartographie *nec* [HM03] comme entrée pour nos simulations. Pius nous avons utilisé notre méthode d'échantillonnage de carte sur cette carte de 47k routeurs pour produire des réseaux ayant des topologies similaires à l'Internet mais avec une taille de 4000 routeurs. Ceci est la taille typique des réseaux des gros fournisseurs d'accès à Internet en incluant les réseaux clients et les réseaux pairs comme décrit dans [SMW02]. Ces réseaux obéissent aux lois puissances des degrés trouvées dans l'Internet par Faloutsos *et al.* dans [FFF99] et surtout ils obéissent aux lois puissances concernant le nombre de plus courts

chemins distincts multiples trouvées dans l'Internet et détaillées dans nos travaux précédents [MP01b]. Ces dernières lois sont directement liées à la quantité de liens redondants dans l'Internet et sont donc d'une importante cruciale pour décrire les possibilités d'utiliser les plus courts chemins multiples.

Dans nos simulations, nous adoptons une vision macroscopique et supposons que $x$ % des routeurs sont saturés à n'importe quel instant donné comme cela est expliqué dans la section 3.3.2. Nous avons analysé des états variés de réseaux en utilisant des densités de saturation variant de 5% à 80% ainsi que des tailles variées d'arbres multipoints définies par leur densité de session (*i.e.*, le pourcentage de routeurs qui sont récepteurs dans l'arbre multipoint) variant de 2% à 40%. La table 3.2 contient la description de l'espace des paramètres de nos simulations.

| Paramètre | Valeurs |
|---|---|
| Taille des réseaux | 4000 routeurs |
| Densité de saturation | 5, 10, 15, 20, ..., 80 |
| Densité de session | 2, 4, 6, 8, ..., 40 |

TAB. 3.2 – Paramètres de simulation

Comme le processus de génération des arbres multipoints et la sélection des nouveaux membres nécessitent des sélections aléatoires (et donc des tirages aléatoires), nous avons utilisé un scénario séquentiel de simulation comme décrit dans [LK00] pour produire les résultats présentés dans la section suivante. Nous avons utilisé le générateur de nombres pseudo-aléatoires nommé Mersenne Twister [MN98] pour produire les nombres aléatoires utilisés dans nos simulations. Comme les tirages aléatoires sont la seule source d'aléas dans nos simulations, nous pouvons raisonnablement assumer que les données produites par les simulations obéissent au théorème de la limite centrale. Nous avons effectué des simulations finies dans lesquelles chaque instance consiste à sélectionner un nouveau membre puis à déterminer le nombre de plus courts chemins entre celui-ci et la source multipoint en s'arrêtant aux routeurs appartenant déjà à l'arbre multipoint, puis à compter combien de ces plus courts chemins multiples sont bloqués par un routeur saturé. Donc dans nos simulations, nous avons mesuré toutes les métriques citées dans la table 3.3.

Idéalement nous aurions dû créer un arbre multipoint, sélectionner un nouveau membre, mesurer les métriques, effectuer un contrôle séquentiel puis répéter cet enchaînement jusqu'à convergence des valeurs de sortie mais cela aurait été trop coûteux en termes de puissance de calcul. Donc nous avons effectué un contrôle séquentiel après avoir crée un arbre et sélectionné et mesuré les métriques pour 500 nouveaux membres. Tous les résultats des simulations ont été obtenus en utilisant un niveau de confiance de 0.95 ainsi qu'un seuil d'erreur statistique relative de 5% pour toutes les métriques mesurées.

| Métriques |
|---|
| Nombre de plus courts chemins du nouveau membre vers l'arbre |
| Nombre de plus courts chemins valides |
| Nombre de plus courts chemins bloqués |
| Le plus court chemin par défaut est-il valide ? |

TAB. 3.3 – Métriques mesurées

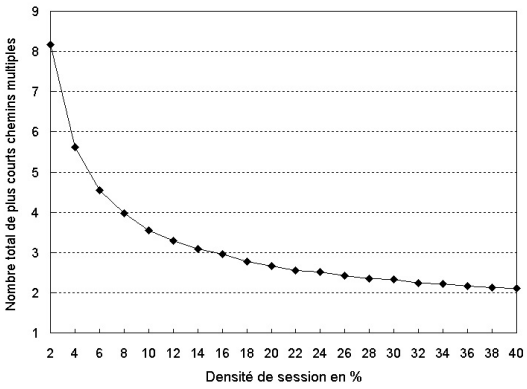


FIG. 3.3 – Nombre total moyen de plus courts chemins pour un nouveau membre joignant un arbre *vs* densité de récepteurs.

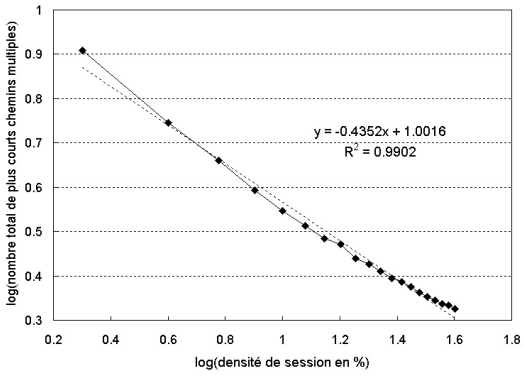


FIG. 3.4 – Log du nombre total moyen de plus courts chemins pour un nouveau membre joignant un arbre *vs* log de la densité de récepteurs.

### 3.4.2 Résultats

Pour pouvoir se faire une idée du nombre de plus courts chemins multiples entre un nouveau membre et l'arbre qu'il souhaite joindre, nous traçons la valeur moyenne (de toutes les instances) de ce nombre dans la figure 3.3 en fonction de la densité de session. Ce nombre est relativement élevé, toujours situé entre 2,2 et 8,2. Lorsque la densité de session augmente et par conséquent que la taille de l'arbre augmente, le nombre de plus courts chemins multiples diminue. En effet, lorsque l'arbre grossit, la distance entre le nouveau membre et l'arbre diminue ainsi que la probabilité d'avoir des plus courts chemins alternatifs.

Cette courbe nous indique déjà qu'il y aura beaucoup de plus courts chemins alternatifs disponibles. De plus, cette courbe possède l'allure typique d'une distribution à queue lourde. Nous traçons cette distribution sur une échelle log-log dans la figure 3.4. Nous pouvons voir qu'une régression linéaire effectuée sur ces données nous donne un coefficient de corrélation de 0.99 qui atteste de la présence d'une loi puissance dans cette distribution. Ceci n'est pas étonnant car beaucoup de propriétés topologiques des graphes de type Internet (*i.e.*, cartes de l'Internet partielles ou échantillonnées) obéissent à des lois puissances. Cette courbe est très probablement influencée par les lois puissances concernant le nombre de plus courts chemins distincts entre paires et présentées dans nos précédents travaux [MP01b].

Dans la figure 3.5 nous traçons le nombre de messages *join* qui ont échoué proportionnellement au nombre total de chemins possibles (exprimé en pourcentage), en fonction de la densité de saturation. Selon qu'il ait échoué ou non, le chemin point à point par défaut est ou n'est pas compté dans les messages *join* ayant échoué. Nous pouvons voir que ce pourcentage est très dépendant de la densité de saturation. Lorsque la densité de session est élevée, la corrélation est presque linéaire, alors que lorsque la densité de session est faible, la corrélation devient non linéaire. C'est à dire que le pourcentage de chemins bloqués augmente plus rapidement dans les niveaux faibles saturations et plus lentement dans les niveaux forts de saturation. Cette courbe montre que, très probablement parce que les chemins alternatifs sont rares dans les parties peu maillées d'une topologie de type Internet, les tentatives d'adhésion vont devenir quasi-impossible au delà d'un certain niveau de saturation car la majorité des chemins possibles seront bloqués. Comme précédemment, afin de voir l'influence de la densité de session, nous traçons le pourcentage de messages *join* ayant échoué en fonction de la densité de session dans la figure 3.6. Le pourcentage de chemins bloqués diminue lentement lorsque la densité de session augmente. La diminution est plus forte lorsque les densités de session sont faibles (*i.e.*, sous

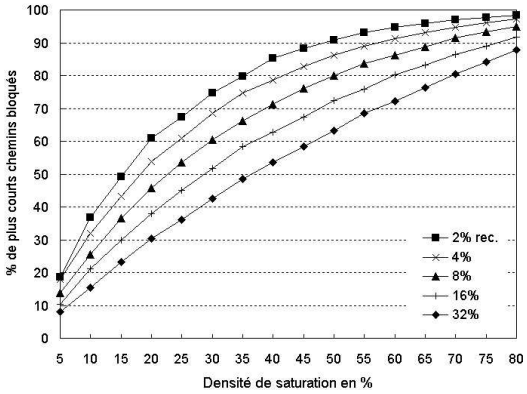


FIG. 3.5 – % de chemins bloqués parmi tous les plus courts chemins possibles pour un nouveau membre joignant un arbre *vs* densité des routeurs saturés.

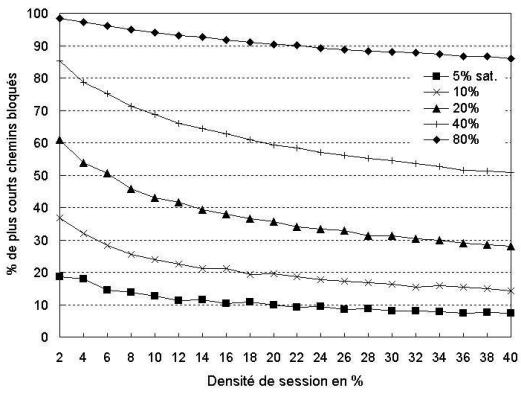


FIG. 3.6 – % de chemins bloqués parmi tous les plus courts chemins possibles pour un nouveau membre joignant un arbre *vs* densité des récepteurs.

10%). Cette courbe montre que les chances d'adhérer à un grand arbre multipoint par rapport à celles d'adhérer à un arbre plus petit sont légèrement meilleures pour un niveau de saturation donné.

Dans la figure 3.7 nous traçons le pourcentage de messages *join* ayant abouti *en utilisant le plus court chemin point à point par défaut* (par rapport à tous les messages *join*, fructueux ou non) en fonction de la densité de saturation. Ces résultats peuvent être vus comme ceux que l'on obtiendrait en utilisant la méthode d'adhésion habituelle définie dans les protocoles multipoints à source quelconque ou unique. C'est à dire sans utiliser notre mécanisme. Nous pouvons voir can ce pourcentage est très dépendant de la densité de saturation. Lorsque la densité de session est élevée (*i.e.*, grands arbres), la relation est presque linéaire, alors que lorsque la densité de session est faible (*i.e.*, petits arbres), la relation tend à être non linéaire. Le pourcentage de réussite diminue plus rapidement dans les niveaux de faible saturation et plus lentement dans les niveaux de forte saturation. Pour expliquer ce phénomène, nous pouvons voir que lorsque le réseau est moyennement saturé (*i.e.*, entre 30% et 70%) et que l'arbre est grand, le nouveau membre sera plus proche de ce dernier et donc il aura une chance plus faible d'être bloqué par un routeur saturé. Cette différence n'est plus significative lorsque le réseau est très saturé auquel cas la plupart des messages *join* utilisant le chemin par défaut échouent ou lorsque le réseau n'est pratiquement saturé auquel cas la plupart des messages *join* utilisant le chemin par défaut aboutissent. Pour illustrer l'influence de la densité de session, nous traçons le pourcentage de messages *join* qui ont abouti *en utilisant le plus court chemin point à point par défaut* en fonction de la densité de session dans la figure 3.8. Nous pouvons voir que le pourcentage de réussite ne dépend pas beaucoup de la densité de session et donc de la taille de l'arbre. Ce pourcentage n'est apparemment qu'influencé fortement lorsque la densité est faible (*i.e.*, sous 8%). Cette courbe est intéressante car elle montre que les chances d'adhérer à un arbre multipoint populaire dépendent peu de sa taille mais surtout de l'état de saturation du réseau.

Les deux dernières courbes illustrent le bénéfice pouvant être obtenu en utilisant les plus courts chemins alternatifs à l'aide de notre mécanisme au lieu de n'utiliser que le plus court chemin par défaut. La figure 3.9 montre le pourcentage de messages *join* additionnels ayant abouti en utilisant un plus court chemin alternatif non bloqué, lorsque le plus court chemin point à point par défaut est bloqué par un routeur saturé, en fonction de la densité de saturation. Nous pouvons voir que le gain est au moins linéaire par rapport à la densité de saturation density et lorsque la taille de l'arbre est petite il se réduit un peu lorsque le réseau est saturé à plus de

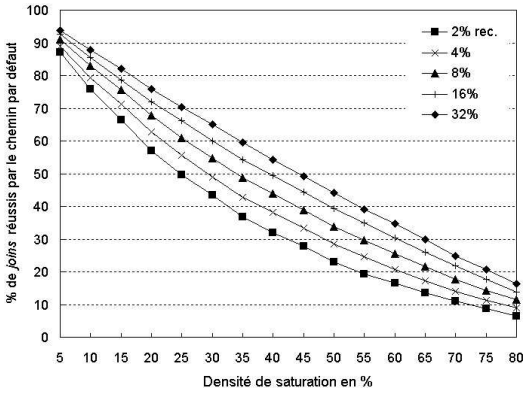


FIG. 3.7 – % de JOIN fructueux obtenus en utilisant le chemin par défaut *vs* densité des routeurs saturés.

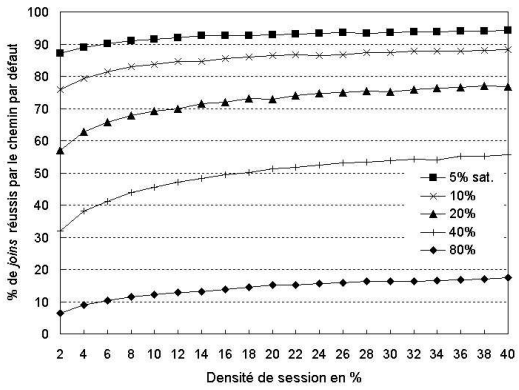


FIG. 3.8 – % de JOIN fructueux obtenus en utilisant le chemin par défaut *vs* densité des récepteurs.

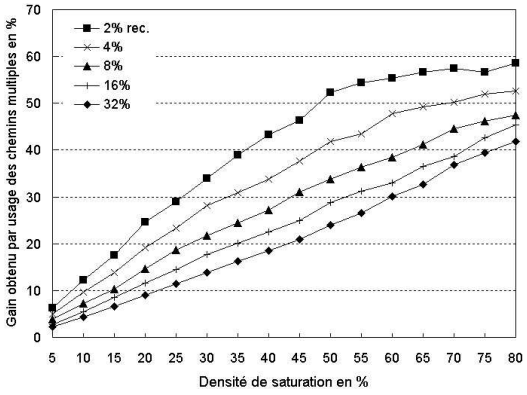


FIG. 3.9 – Gain obtenu en utilisant n'importe quel plus court chemin alternatif non bloqué *vs* densité des routeurs saturés.

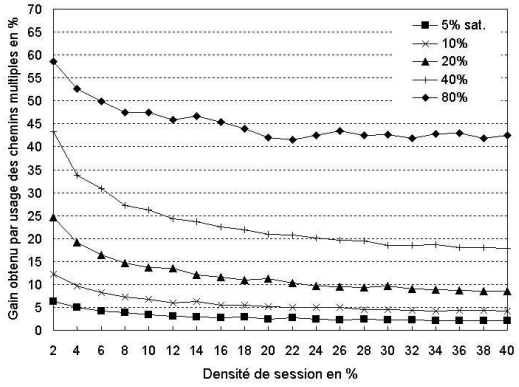


FIG. 3.10 – Gain obtenu en utilisant n'importe quel plus court chemin alternatif non bloqué *vs* densité des récepteurs.

50%. Ces résultats sont très prometteurs et nous encouragent à implémenter notre mécanisme. La figure 3.10 montre le pourcentage de messages *join* additionnels ayant abouti en utilisant un plus court chemin alternatif non bloqué, lorsque le plus court chemin point à point par défaut est bloqué par un routeur saturé, en fonction de la densité de session. Nous pouvons voir que le gain diminue lorsque la densité de session augmente, surtout dans les niveaux de faible densité. Nous pensons, comme ci-dessus, que lorsque les arbres sont petits, les chances qu'un nouveau membre soit loin de l'arbre sont plus grandes. Donc le nombre de plus courts chemins entre le nouveau membre et l'arbre augmente, fournissant plus de chemins alternatifs pour contourner les routeurs saturés. Notons que les petites fluctuations observables sur certaines parties de ces courbes sont probablement des artefacts de nos simulations. Notre technique consistant à exécuter un grand nombre (*i.e.*, 500) d'instances avant d'effectuer un contrôle séquentiel peut légèrement affecter les résultats statistiques. Si nous avions choisi des valeurs plus faibles pour le nombre d'instances, nous aurions pu obtenir des courbes plus lisses (mais la durée des simulations aurait été étendue). De même nous avons utilisé un seuil d'erreur statistique relative de 5% qui n'est peut être pas suffisamment précis pour éviter ce phénomène de fluctuations.

## 3.5 Conclusion

Améliorer l'efficacité des protocoles multipoints est important pour promouvoir leur déploiement dans l'Internet. Dans ce chapitre, nous avons proposé un mécanisme simple conçu pour profiter de multiples plus courts chemins dans le but d'augmenter les chances qu'un nouveau membre puisse joindre un arbre multipoint tout en distribuant les états multipoints parmi les routeurs non saturés. Pour effectuer des simulations réalistes, nous avons utilisé une carte des routeurs du coeur de l'Internet que nous avons créé lors de nos précédents travaux. Nous avons utilisé notre carte pour modéliser des grandes topologies intra-domaines. Nous avons trouvé que lorsque de tels réseaux sont très saturés (*i.e.*, plus de 50%), l'augmentation des messages *join* alternatifs fructueux va de 25% à 55% selon la densité de session.

Donc il apparaît très profitable d'implémenter notre mécanisme dans les routeurs, et cela d'autant plus qu'il ne nécessite que des modifications mineures des protocoles multipoints de gestion d'arbres multipoints (tels que PIM-SSM) et il ne stocke aucune information dans les routeurs. Nous sommes toutefois conscients que le support des plus courts chemins multiples par les routeurs n'est pas systématiquement disponible en intra-domaine et que notre solution n'est actuellement pas utilisable en inter-domaine car BGP ne prend pas en charge la gestion des plus courts chemins multiples.

Les travaux sur la répartition des états multipoints ont pu être effectués grâce à de multiples contributions. La partie empilement d'arbres d'états multipoints a été réalisée par Mickaël Hoerdt lors de son travail de doctorat. La partie redistribution des états par redirection des adhésions multipoints a été réalisée par moi-même. De nombreuses idées et remarques pertinentes ainsi que des relectures ont été fournies durant ces travaux par Dominique Grad, Jean-Jacques Pansiot et Pascal Lorenz.

## Travaux relatifs au chapitre

1. Damien Magoni and Pascal Lorenz. Multicast State Distribution by Joins Using Multiple Shortest Paths. *Journal Of Interconnection Networks*, 6(2) : 115–129, June 2005.
2. Damien Magoni. Multicast state balancing by using alternate shortest paths. In *Proceedings of the 13th International Conference on Computer Communications and Networks*, pages 295–300, Chicago, Illinois, USA, October 2004.
3. Mickaël Hoerdt and Damien Magoni. Distribution of multicast tree states over the IPv6 network topology. In *Proceedings of the IEEE International Conference on Communications*, pages 1991–1995, Paris, France, June 2004.
4. Mickaël Hoerdt, Damien Magoni et Jean-Jacques Pansiot. Arbres des plus courts chemins multiples dans l'Internet IPv6. In *Actes des 6èmes rencontres francophones sur les aspects Algorithmiques des Télécommunications*, pages 75–80, Batz-sur-mer, France, Mai 2004.
5. Mickaël Hoerdt, Jean-Jacques Pansiot, Damien Magoni et Dominique Grad. Répartition des états de routage multipoint par les plus courts chemins multiples. In *Actes des 5èmes rencontres francophones sur les aspects Algorithmiques des Télécommunications*, pages 139–145, Banyuls-sur-mer, France, Mai 2003.

# Chapitre 4

# Attaques distribuées par déni de service utilisant des états multipoints

## 4.1 Introduction

L'architecture de routage multipoint IP la plus déployée actuellement est dirigée par les récepteurs. Celle-ci ne fournit aucun retour aux membres du groupe sur l'accessibilité des sources de diffusion. Cette architecture implémente deux modèles (multipoint à source quelconque (ASM) ou multipoint à source unique (SSM)) ayant chacun différentes propriétés. Celles-ci peuvent avoir de multiples conséquences sur le réseau et les applications. Nous montrons ici qu'un mécanisme offrant un retour aux récepteurs d'une diffusion est manquant et qu'il peut avoir des conséquences graves en particulier dans le cas du modèle multipoint à source unique. Par simulation d'une attaque distribuée par déni de service sur un réseau réaliste, nous en analysons les conséquences.

Dans ce chapitre nous analysons en détail le problème des branches de diffusion inutiles créées sur un réseau dans le contexte du modèle de routage multipoint SSM. Dans la section 4.2 nous exposons en détail le contexte architectural dans lequel nous nous plaçons. Puis dans la section 4.3 nous décrivons les mécanismes mis en jeu dans la création d'états multipoints. Ensuite dans la section 4.4 nous expliquons plus précisement le problème des branches de diffusion inutiles dans l'architecture exposée précédemment. Enfin, dans la section 4.5, nous simulons et analysons l'efficacité d'attaques distribuées par déni de service basées sur SSM.

## 4.2 Contexte

Nos travaux se placent dans le contexte de l'extensibilité des protocoles de routage multicast en termes d'états stockés. Celle-ci a déjà été largement étudiée par la communauté [WK00], mais pas dans un contexte de sécurité. A notre connaissance, nous sommes les premiers à étudier l'impact d'une attaque par déni de service basée sur SSM. Nous rappellons dans cette partie les composantes de l'architecture de routage multipoint. Le modèle originel de groupe à source quelconque nommé Any Source Multicast (ASM) et défini par Deering [Dee91] implique bien plus qu'une simple réplication de l'information par le réseau. Deering définit un groupe comme ayant les mêmes propriétés au niveau IP que celles d'un lien Ethernet : ouvert et sans mécanisme de synchronisation explicite. Défini plus tard par Holbrook [Hol01], le modèle SSM offre moins de possibilités et est un sous-ensemble d'ASM, ce qui facilite grandement son implémentation et son déploiement. Aujourd'hui, le protocole de routage multicast IP le plus déployé pour ces deux modèles est PIM [EFH$^+$98]. Afin d'éviter un conflit de sémantique dans le protocole, deux préfixes réseaux disjoints ont été alloués par l'IANA pour chaque modèle de diffusion.

De multiples architectures de routage IP multipoint ont été définies au moment de la définition du modèle et il a été décidé d'intégrer aux réseau deux niveaux de signalisation. Ceux-ci sont encore utilisés dans l'architecture de routage de SSM actuelle pour des raisons de rapidité d'implémentation et de déploiement.

Au premier niveau, un protocole gère la signalisation entre les membres d'un groupe/canal et les routeurs IP multipoint : il s'agit de l'Internet Group Management Protocol (IGMP) défini dans [CDF+02] pour l'IPv4 ou bien du Multicast Listener Discovery (MLD) défini dans [VCF+02] pour l'IPv6. Au deuxième niveau, on trouve les protocoles de routage multipoint comme DVMRP [WPD88a], PIM [EFH+98] et CBT [BFC93]. Ils sont responsables de la signalisation entre les routeurs multipoints.

En inter-domaine, plusieurs protocoles additionnels peuvent être nécessaires selon qu'on se trouve dans le cas SSM ou ASM : MBGP [BCKR98] pour différencier les liens transportant du trafic multipoint de ceux transportant du trafic point-à-point et MSDP [FM03] pour rajouter un service de découverte automatique de nouvelles sources dans le cas ASM.

Concernant les problèmes de sécurité des protocoles multipoints, il est intéressant de souligner que l'IETF poursuit actuellement des travaux de sécurisation des protocoles de routage multipoint au sein du groupe de travail Multicast Security (MSEC), notamment autour de l'architecture pour la sécurisation de groupes.

## 4.3 Fonctionnement du multipoint lors de la création d'états

Dans cette section nous décrivons plus en détail la création d'états multipoints et la façon dont coopèrent les récepteurs, les sources et les routeurs multipoints entre eux.

### 4.3.1 Création d'états multipoints dans le réseau

Le modèle de routage multipoint actuellement déployé dans l'Internet pose très peu d'hypothèses sur la notion de groupe de communication. En réalité on peut dire qu'un groupe IP correspond à une addresse de diffusion ou d'abonnement dont la sélection est totalement libre. N'importe quelle machine de l'Internet peut se déclarer membre d'un groupe à tout moment, qu'il y ait du trafic ou non dans celui-ci. Pour cela elle a simplement à le signaler au réseau. De manière symétrique, une source peut envoyer des données vers un groupe même si il n'existe pas de membres dans celui-ci. Bien que ce modèle soit simple à utiliser pour les applications réseau multipoint (notamment parce qu'il ne demande pas de synchronisation explicite entre les participants du groupe), il peut devenir coûteux pour le réseau dans certaines situations.

Par exemple si on considère les protocoles de routage en mode dense tels que le protocole DVMRP [WPD88a] ou le protocole PIM-DM [ANS02], une source sans récepteurs inonde périodiquement le réseau avec des données. Des états liés au protocole sont créés dans l'ensemble des routeurs touchés par l'inondation. En effet ceux-ci se doivent de mémoriser les interfaces qui ont été élaguées de l'arbre de diffusion. Dans ce cas, les récepteurs sans source ne posent pas de problème. Aucun état n'est créé dans le réseau sauf sur les routeurs désignés.

Dans les protocoles de routage en mode épars comme PIM-SM [EFH+98], la diffusion ne peut se faire que si il y a des récepteurs. Elle est en quelque sorte dirigée par les abonnés du groupe. Ainsi, si un client rejoint un groupe sans qu'il soit actif, une branche de diffusion inutile sera créée avec les états protocolaires correspondants dans les routeurs. Cette branche de diffusion peut potentiellement être maintenue pendant une très longue durée. En effet le propriétaire de la branche est en quelque sorte l'application et tant que celle-ci n'a pas quitté le groupe ou qu'une source devient active dans le groupe elle sera maintenue inutilement par le réseau.

La gravité du problème s'amplifie lorsqu'on considère le routage multipoint au niveau inter-domaine dans un réseau de la taille d'Internet. Les possibilités d'attaques de déni de service ainsi que les mauvaises configurations y sont nombreuses et un nombre important de branches de diffusions inutiles peuvent être créées sur les domaines de transit.

La dernière extension du modèle de routage multipoint est dite à source unique et est nommée Source Specific Multicast (SSM) [Hol01]. Elle est incluse dans PIM-SM [EFH+98] et pousse le modèle de routage multipoint dirigé par les récepteurs encore plus loin. En effet alors qu'avant ceux-ci ne pouvaient que choisir les groupes de diffusion IP auxquels ils s'abonnaient, il peuvent maintenant sélectionner une source de diffusion en particulier. Dans ce modèle un groupe, un canal est identifié par un tuple (Adresse Source, Identifiant de Groupe).

### 4.3.2 Création de branches multipoints

Nous avons vu précédemment que les récepteurs, les sources et les routeurs multipoints sont assez indépendants les uns des autres et partagent très peu d'information. Typiquement les événements suivant se produisent :

1. Une application source choisit une addresse de groupe G, en espérant qu'elle ne soit pas déjà utilisée dans l'Internet et y envoie des données.
2. Un utilisateur est informé (par un moyen externe comme le mél, une page web ou des news) de l'existence de G et envoie un message d'abonnement multipoint sur l'interface sur laquelle il aimerait recevoir les données envoyées vers G.

**L'ordre des opérations 1) et 2) n'est pas fixé et l'opération 2) peut être executée depuis un nombre arbitraire de clients sans forcément que 1) ait été executée.**

Dans les deux modèles de groupe multipoint, un paquet IPv4 ou IPv6 destiné au groupe se différencie d'un paquet point-àpoint uniquement par son addresse de destination. Les adresses multipoint IPv4 ou IPv6 sont caractérisées par un préfixe réseau et peuvent être allouées par différents moyens, y compris manuellement.

En ASM, deux paquets multipoint ayant la même IP destination sont censés être envoyés vers le même groupe de terminaux, si l'on ne tient pas compte des contraintes de portée. En SSM, deux paquets sont envoyés vers le même groupe de terminaux seulement si ils ont la même IP source et la même IP destination. On identifie un groupe ASM par son addresse de destination alors qu'en SSM le groupe est identifié par le tuple $< IP_{source}, IP_{destination} >$. PIM-SM [EFH+98] permet déjà la création de branches spécifiques à une source en plus des branches partagées et la version 3 d'IGMP [CDF+02] permet aux terminaux de spécifier quelles sont les sources qui les intéressent. Ainsi l'architecture modifiée pour SSM est la suivante :

- Une plage d'adresse est réservée au routage multipoint SSM.
- Un terminal peut rejoindre un canal en utilisant le protocole IGMPv3 (ou MLDv2 son équivalent IPv6) et en spécifiant une addresse SSM associée à une source.
- Un routeur qui reçoit un rapport d'abonnement (S,G) depuis un terminal, avec G appartenant à la plage d'addresse SSM ne construira pas d'arbre partagé vers le point de rendez-vous PIM. A la place, il engendrera une branche spécifique à la source vers S.
- Les routeurs intermédiaires commutent en saut par saut les messages PIM-JOIN(S,G) vers la source, tout comme dans PIM-SM.
- La branche de diffusion ainsi créée se termine au dernier saut avant la source de diffusion.

S peut être n'importe où dans l'Internet. En conséquence des branches inter-domaines peuvent être construites. Pour permettre un routage politique sur le trafic multipoint, la branche construite utilise la topologie construite par le protocole MBGP [BCKR98]. La MRIB ou table unicast de construction d'arbre de routage multipoint est utilisée pour trouver la route vers S lors de l'envoi des messages PIM-JOIN(S,G). Le modèle SSM est considéré aujourd'hui par la communauté

comme une solution extensible au problème du routage multipoint inter-domaine. Il est plus simple à mettre en oeuvre et résoud des barrières de déploiement importantes présentes en ASM, au moins en IPv4 :

- Le problème de l'unicité des addresses multipoint devient local au terminal source du canal. En effet l'identifiant du canal est simplement un sélecteur ajouté à une addresse IP source sensée être déjà unique.
- Il n'y a plus de point de rendez-vous car les arbres de diffusion construits ne sont plus partagés. Ainsi le problème de l'élection ou de la robustesse du point de rendez-vous disparaît.
- Le protocole MSDP n'a plus de raison d'être comme le modèle est complêtement dirigé par les récepteurs.

Ce modèle SSM est plus simple que le précédent et convient bien aux applications de type 1 à n comme la télévision ou la radio sur Internet. La communauté espère donc le voir déployé assez rapidement d'autant plus qu'il ne nécessite pas de modifications majeures dans l'architecture actuelle, si ce n'est dans les terminaux. Néanmoins nous soulevons un problème de sécurité important dans cette architecture : l'existence potentielle de branches de diffusions inutiles.

## 4.4 Branches de diffusion IP multipoint inutiles

Les branches de diffusion IP peuvent être construites aussi bien dans l'architecture ASM que dans celle de SSM. Nous analysons chaque cas dans cette section et nous montrons que le problème peut être plus grave dans le cas SSM. Nous faisons l'hypothèse que le protocole de routage multipoint utilisé est PIM-SM [EFH+98], le plus déployé actuellement.

### 4.4.1 Le cas ASM

Lorsque le routeur désigné PIM (DR) reçoit un message IGMP report, il essaiera de se connecter à l'arbre de routage multipoint correspondant à G, si ce n'est pas déjà le cas. Pour cela, un message join(G) est envoyé vers en direction d'un routeur particulier (initialement le point de rendez-vous (RP)) pour le groupe. Ce message est commuté par les routeurs intermédiaires, ce qui crée une branche depuis le DR vers le RP. Des entrées de routage multipoint sont maintenues dans tous les routeurs intermédiaires, jusqu'à ce qu'un messages prune(G) soit reçu en aval ou qu'un certaine période de temps s'écoule sans qu'un message join(G) soit reçu d'en aval. Contrairement à d'autres protocoles comme CBT [BFC93], PIM-SM [EFH+98] ne dispose pas de moyen pour détruire une branche depuis l'amont de l'arbre. Ainsi une branche sera conservée aussi longtemps qu'elle aura des récepteurs en aval. Si le RP n'était pas joignable, par exemple suite à une partition du réseau, **Il n'existe pas de moyen permettant aux routeurs en aval d'être avertis de la panne**.

Sur la figure 4.1, nous illustrons le mécanisme dans le cas inter-domaine IPv6 utilisé avec une solution récente proposée par l'IETF [SH03]. En IPv4 et en intra-domaine, les routeurs multipoint savent que le point de rendez-vous n'est plus utilisable grace au mécanisme du BSR [EHaPH99], mais ils ne peuvent pas distinguer une panne de RP d'une panne plus sérieuse du réseau. PIM-SM permet de créer des arbres par source, aussi si une branche spécifique à la source a été créée, elle restera conservée par le réseau même si la source n'est pas joignable. Ces branches inutiles peuvent ne pas poser de problèmes dans le cas d'un simple domaine feuille de l'Internet. Ce n'est pas le cas dans un contexte inter-domaine en particulier dans le cas de SSM que nous analysons dans la section suivante.

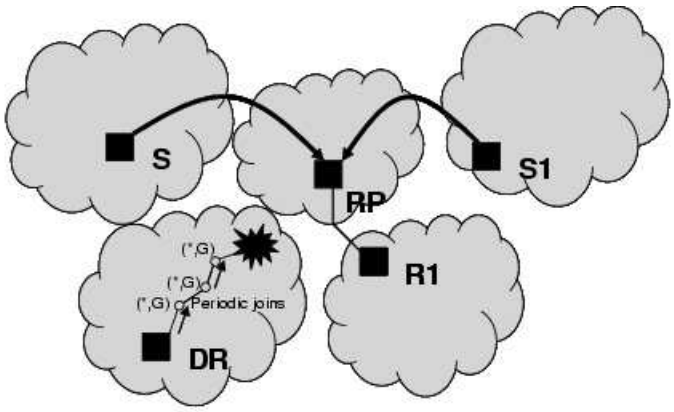


FIG. 4.1 – Problème des branches de diffusion inutiles dans l'architecture ASM

### 4.4.2 Le cas SSM

De la même manière, un terminal s'abonne au canal (S,G) et une branche est créée, puis maintenue par le réseau sans tenir compte de l'existence réelle du canal. Il est possible que le canal ne soit pas joignable pour différentes raisons, et que les branches créées soient tout à fait légitimes. Néanmoins nous pensons que cela représente un sérieux problème pour les domaines de transit où plusieurs milliers, voire millions de branches peuvent rapidement épuiser les ressources en bande passante, mémoire ou CPU des routeurs. Dans [PSM04], les auteurs confirment cette idée. La figure 4.2 nous illustre le problème. Les acteurs sont les mêmes que pour ASM, à la seule différence que la racine de l'arbre de diffusion est sensée générer des données et que les récepteurs ont précisé explicitement au réseau quelles étaient leurs sources d'intérêt.

## 4.5 Evaluation de l'impact d'une attaque DDoS par simulation

Les expérimentations effectuées par simulation sont sensibles à un nombre important de paramètres, en particulier celui des topologies utilisées. Dans cette section nous détaillons notre méthodologie expérimentale et nous présentons les résultats obtenus par nos simulations. Nous avons vu dans les sections précédentes qu'il était possible pour des récepteurs de créer des branches inutiles, destinées uniquement à utiliser les ressources des routeurs afin de saturer leur mémoire. De plus nous avons vu dans la section 3.3.2 que la saturation d'un routeur est un phénomène très plausible et qui ne peut être réfuté en l'état actuel de la technologie.

### 4.5.1 Méthodologie et paramètres

S'assurer du réalisme et de la précision de la topologie utilisée est vital pour des simulations concernant le routage multipoint, car les résultats sont très dépendants de la topologie sous-jacente. C'est pourquoi nous utilisons des topologies de type Internet pour effectuer nos simulations. Nous utilisons une topologie de l'Internet IPv6 [HM04] car sa taille reste raisonnable (*i.e.*, 4K routeurs) pour effectuer des simulations ainsi que des topologies créées par échantillonnage d'une carte IPv4 [MP02b] de tailles 2K et 3K routeurs grâce au logiciel Nem.

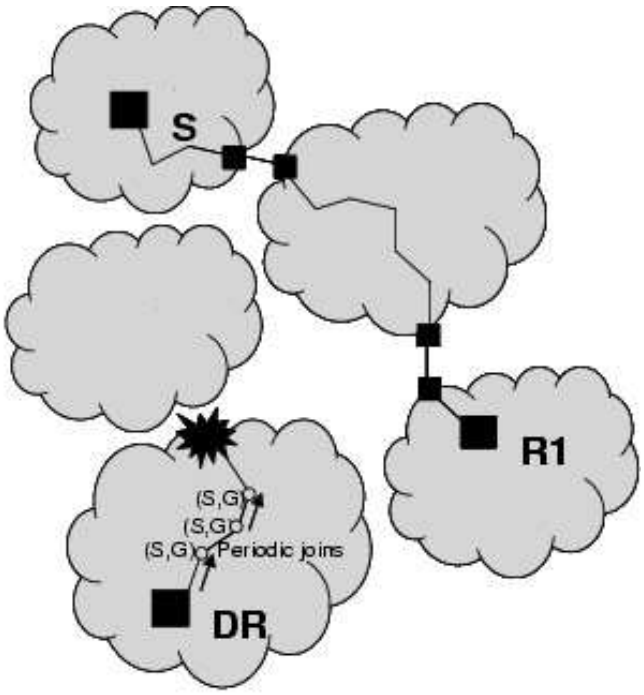


FIG. 4.2 – Problème des branches de diffusion inutiles dans l'architecture SSM

La topologie IPv6 nous garantit des résultats relativement fiables car elle a été obtenue à l'aide de mesures réelles. La carte correspondante construite par des traces distribuées est très précise sur la quantité et le placement de ses liens [HM03]. La taille encore relativement petite de ce réseau nous permet d'effectuer des simulations centralisées dans un temps raisonnable. La table 4.1 présente quelques caractéristiques importantes de cette topologie IPv6. Nous avons vérifié dans nos précédents travaux [HM04] que celle-ci posséde certaines caractéristiques typiques du réseau IPv4 notamment en ce qui concerne les distributions des degrés, des arbres et du nombre de plus courts chemins distincts. Les réseaux IPv4 de faible taille obtenus par échantillonnage d'une carte IPv4 ont des propriétés similaires.

TAB. 4.1 – Caractéristiques de la carte IPv6

| Source | Nombre de routeurs | Nombre de liens | Degre moyen des routeurs | Distance moyenne | Date de création |
|---|---|---|---|---|---|
| *nec* IPv6 | 4256 | 14326 | 6,73 | 7,92 | Juin 2003 |

Nous aimerions quantifier la répartition des états créés dans l'ensemble du réseau par des attaquants utilisant le protocole PIM-SSM. Pour cela, nous avons sélectionné aléatoirement une part variable (de 0,1% à 10%) de noeuds attaquants dans la topologie IPv6 ainsi qu'un noeud cible. Nous appelons *densité*, le pourcentage des noeuds de la carte étant attaquants. Une sélection aléatoire est justifiée car il est assez difficile de prévoir un placement spécifique des attaquants dans le réseau. Idéalement les attaquants devraient se placer de la façon la plus distribuée possible dans le réseau mais cela signifie qu'ils doivent connaître à priori avec précision la topologie du réseau et qu'ils puissent opérer de n'importe quel endroit. Les attaquants construisent

alors des branches multipoint SSM d'eux mêmes vers la cible, créant ainsi des états dans tous les routeurs rencontrés en chemin.

Comme le processus consistant à sélectionner des noeuds attaquants et cibles fait intervenir des sélections aléatoires (et donc des tirages aléatoires), nous avons utilisé un scénario séquentiel de simulation [LK00] pour obtenir les résultats présentés dans la section suivante. Nous avons utilisé le générateur de nombres pseudo-aléatoires Mersenne Twister [MN98] pour produire les nombres aléatoires nécessaires à nos simulations. Comme les tirages aléatoires sont la seule source d'aléas dans les simulations, nous pouvons assumer que les données mesurées en sortie obéissent au théorème de la limite centrale (*i.e.*, les valeurs des mesures d'une variable de sortie donnée forment une distribution normale). Nous avons effectué des simulations finies dans lesquelles chaque expérience consiste à sélectionner aléatoirement des attaquants selon une densité fixée en paramètre ainsi qu'une cible puis à créer les branches de chaque attaquant vers la cible et enfin à calculer le nombre d'états stockés dans chaque noeud du réseau à cause de ces branches. Nous avons ensuite réparti tous les noeuds en fonction du nombre d'états qu'ils stockent et en fonction de leur distance à la cible puis nous avons répété l'expérience jusqu'à la convergence de toutes les distributions des noeuds en fonction de leur nombre d'états et de leur distance à la cible. Le résultat d'une simulation est donc la répartition typique des noeuds selon leur nombre moyen d'états et leur distance à la cible pour une densité d'attaquants donnée. Tous les résultats des simulations ont été obtenus en utilisant un niveau de confiance de 0.95 avec une erreur statistique relative de 5% pour toutes les variables de sortie. Nous avons implémenté notre expérimentation en C++ dans le logiciel *network manipulator* [Mag].

### 4.5.2 Résultats

La figure 4.3 représente la répartition du nombre d'états créés dans les routeurs par une attaque de déni de service utilisant le protocole multipoint PIM-SSM. Dans ce cas précis, 0,1% des routeurs constituant le réseau sont des attaquants : ils vont donc chercher à se connecter à la cible en créant une branche d'arbre multipoint jusqu'à celle-ci et donc créer des états le long de cette branche. La figure se lit comme suit : l'axe des $x$ représente le nombre d'états exprimé en pourcentage (arrondi au dixième) du nombre maximum d'états possibles (qui lui est égal au nombre d'attaquants) et stocké dans les routeurs, l'axe des $y$ représente la distance exprimée en nombre de sauts entre la cible et les routeurs et enfin l'axe des $z$ représente le nombre de routeurs exprimé en pourcentage du nombre maximum de routeurs à une distance donnée de la cible ayant une même quantité relative d'états stockés. Par exemple, à la distance 0 de la cible, c'est à dire au niveau de la cible elle même, 100% des routeurs stockent 100% des états : la cible stocke en effet tous les états créés par les attaquants (soit 100% en relatif) et elle représente 100% des routeurs puisqu'elle est le seul routeur à distance 0 d'elle même. Autre exemple, à la distance 1 de la cible, c'est à dire au voisinage de la cible elle même, 53,2% des routeurs voisins stockent 0% des états (c'est arrondi au dixième donc stockent moins de 5% des états), 1,3% des routeurs voisins stockent 10% des états (donc entre 5 à 15%), 5,8% des routeurs voisins stockent 20% des états (donc entre 15 à 25%) et ainsi de suite. Enfin, 23,6% des routeurs voisins stockent 100% des états (donc 95% et plus). Donc, dans ce deuxième exemple, si la cible à 10 voisins et est attaquée par, disons, 200 routeurs, cela signifie que 5 de ses voisins stockent moins de 10 états et ainsi de suite. À l'extrême, 2 de ses voisins stockent plus de 190 états. L'avantage de cette représentation est de pouvoir comparer les mesures effectuées sur n'importe quelles cibles puisque les routeurs et les états sont représentés en pourcentage.

L'observation de la figure 4.3 nous révèle un résultat très intéressant. Les attaques DDOS ne saturent pas globalement le réseau ni même la large périphérie de la cible. Le routeur cible reçoit tous les états créés par les attaquants, mais le nombre de routeurs recevant plus de 70% des états diminue très rapidement de 1 à 3 sauts. A 3 sauts de la cible seuls 6% des routeurs reçoivent

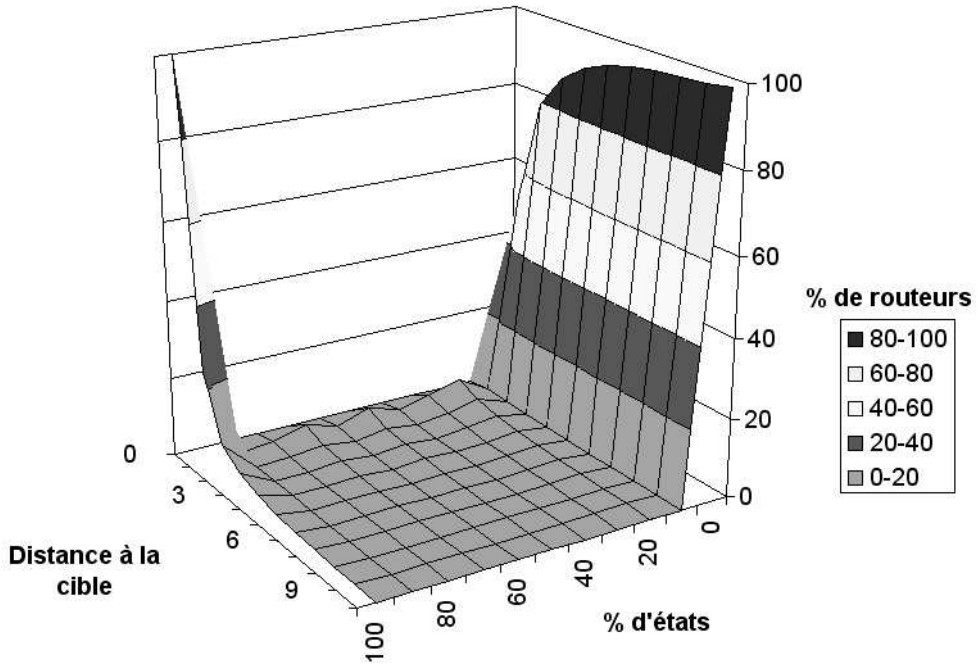


FIG. 4.3 – Répartition des états créés par 0,1% d'attaquants DDoS.

plus de 70% des états. Au delà de 3 sauts, plus de 92% des routeurs stockent moins de 5% des états (*i.e.*, 0 le plus souvent). Ces résultats sont qualitativement indépendants du pourcentage d'attaquants (cf. deuxième graphique ci-dessous). Cela signifie que des attaques DDOS basées sur le protocole multipoint PIM-SSM seront très efficaces à l'encontre de leurs cibles, mais ne seront pas très dangereuses vis à vis des réseaux (donc de leurs opérateurs respectifs) et vis à vis des attaquants eux-mêmes. En effet il serait regrettable pour les attaquants de faire tomber le réseau dans son ensemble puisque eux-mêmes ne pourraient plus en bénéficier. Ces résultats renforcent donc massivement l'intérêt des attaques DDOS multipoints. Comme chaque attaquant va tenter de créer plusieurs canaux SSM différents (ne correspondant pas aux canaux des autres attaquants afin que les branches ne se recoupent pas et qu'un maximum d'états soient créés), le nombre d'états peut atteindre $2^{24}$ (*i.e.*, plus de 16M), ce qui correspond au nombre maximum de canaux SSM gérés par une source donnée (ici la cible). Donc 256 attaquants créant chacun 1000 canaux SSM différents (ou toute autre combinaison attaquants/canaux équivalente) peuvent faire tomber n'importe quel routeur de l'Internet, même les plus gros routeurs de coeur de réseau (actuellement limités à 256K états multipoints, cf. section 3.3.2). Le routeur cible ne sera pas nécessairement hors service mais (selon sa gestion mémoire) il risque d'être très affecté dans ses performances et la création de nouvelles branches de diffusion pour des groupes légitimes sera très probablement impossible.

La figure 4.4 représente la répartition du nombre d'états créés dans les routeurs par une attaque de déni de service utilisant le protocole multipoint PIM-SSM dans le cas où 10% des routeurs constituant le réseau sont des attaquants. On peut observer que les résultats sont qualitativement très proches de ceux d'une attaque effectuée par 0,1% des routeurs. Nous avons effectué d'autres simulations avec des pourcentages d'attaquants variant entre 0,1 et 10% et nous avons trouvés des résultats similaires. De même nous avons effectué d'autres simulations sur des topologies de type Internet de tailles légèrement plus petites (*i.e.*, 2K et 3K routeurs) que celle de la topologie IPv6 utilisée dans les résultats présentés ici et nous avons là encore trouvé des résultats similaires. Nous ne rajoutons pas ici tous ces graphes afin d'éviter les redondances.

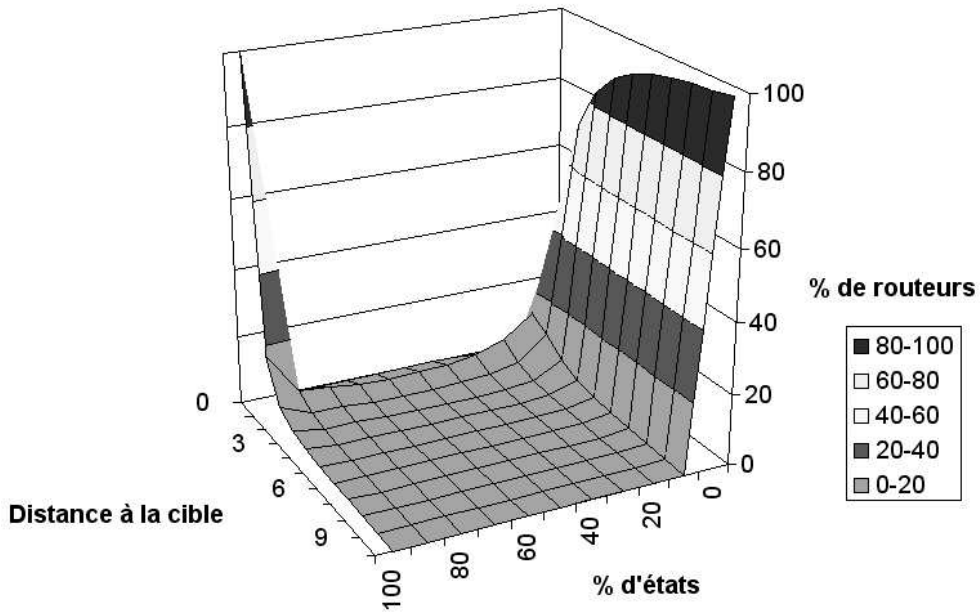


FIG. 4.4 – Répartition des états créés par 10% d'attaquants DDoS.

## 4.6 Conclusion

Nous avons étudié, tout au long de ce chapitre, les possibilités que donnent les protocoles multipoints actuels de réaliser des attaques distribuées par déni de service. Nous avons en particulier simulé l'impact d'attaques distribuées par déni de service basées sur le protocole PIM-SSM dans des topologies de type Internet. Les résultats montrent que ces attaques peuvent se révéler d'une efficacité redoutable car étonnamment elles atteignent la cible et son périmètre proche sans quasiment surcharger le réseau dans son ensemble. De plus il n'est pas nécessaire d'avoir un grand nombre d'attaquants pour réussir à saturer n'importe quel routeur de l'Internet. Ce résultat à priori inquiétant ne doit pas décourager le déployement de ce protocole. En effet, il est relativement facile de créer des mécanismes de contrôle des branches inutiles. Nous sommes actuellement en train de finaliser la définition d'un tel mécanisme, à rajouter à la spécification du protocole PIM-SM, permettant de contrôler la création des branches multipoints. Ce mécanisme rendra possible le blocage de la création des branches inutiles utilisées dans ce type d'attaque.

## Travaux relatifs au chapitre

1. Mickaël Hoerdt, Damien Magoni et Jean-Jacques Pansiot. Evaluation de l'impact d'attaques distribuées par déni de service utilisant un protocole multipoint à source unique. In *Actes des 6èmes Journées Doctorales Informatique et Réseaux*, pages 128–136, Lannion, France, Novembre 2004.

# Troisième partie

# Fonctionnalités réseaux dans les hôtes et topologies induites

# Chapitre 5

# Protocole de découverte de source au niveau session

## 5.1 Introduction

Le modèle multipoint à source spécifique (SSM en anglais) a reçu une attention considérable de la communauté de recherche car il pourrait finalement apporter une solution extensible pour certaines applications multipoints. Dans le modèle SSM il est de la responsabilité des hôtes de dire au réseau quelles sont leurs sources d'intérêt. Par conséquent, le modèle SSM est bien adapté pour les applications multipoints de un à plusieurs (telles que les flux TV et les flux radio) parce qu'il y a seulement une source à découvrir et elle est censée être bien connue. Cependant il n'existe toujours pas de protocole et d'intergiciel de couche session pour SSM dans le but d'aider les applications multipoints à source quelconque (ASM en anglais) à découvrir la présence ou le départ des sources SSM dans un environnement ASM sur SSM. Ainsi la principale chose manquant à SSM par rapport à l'ASM est un protocole de découverte automatique de source. Notre proposition consiste à mettre un tel protocole en extrémité de réseau, ou plus précisément dans la couche session de la pile réseau des hôtes.

Les applications multipoints classiques à multi-sources utilisent le modèle de transmission de groupe ASM défini par Deering [Dee91]. Dans ce modèle n'importe qui, y compris des non-membres, peut envoyer des paquets avec une opération locale vers un identificateur de groupe G. Les paquets seront transmis avec des garanties de type "meilleur effort" à tous les membres du groupe : le réseau duplique les paquets et contrôle la dynamique des sources et des récepteurs. Avec le nouveau modèle SSM de Holbrook [Hol01], il est de la responsabilité des hôtes de dire au réseau quelles sont leurs sources d'intérêt.

Dans ce chapitre, nous proposons une architecture détaillée pour un tel protocole de découverte de source placé dans la couche session. Ce protocole permettra à des applications ASM de fonctionner en utilisant les sources d'un réseau exécutant SSM. Nous avons implémenté notre protocole dans un intergiciel appelé Libssmsdp et librement disponible sous forme d'une bibliothèque à code source ouvert.

La section 5.2 donne une introduction sur les précédents travaux dans ce domaine, la section 5.3 présente une vue d'ensemble du problème et de notre solution. La section 5.4 explique plus en détail la signification d'une émulation d'un service ASM. Dans la section 5.5 nous donnons une description détaillée du protocole d'émulation ASM sur SSM. Enfin la section 5.6 donne une brève évaluation de notre proposition tandis que la section 5.7 fournit des informations sur l'implémentation de la bibliothèque Libssmsdp.

## 5.2 Contexte

Dans sa thèse [Hol01], Holbrook a considéré le cas de l'utilisation d'applications multi-sources seulement en utilisant le service réseau SSM. Il a défini deux méthodes et les a évaluées : il a proposé une méthode d'annonce par l'émetteur à l'aide d'un canal de contrôle et méthode par relais de session en utilisant un noeud central, et indique que les deux ont des avantages et des inconvénients. Il a mesuré le délai de démarrage des sessions et a simulé ses deux propositions mais il n'a pas défini une architecture pour celles-ci. Nous proposons ici une architecture détaillée basée sur ses propositions.

Dans leur papier [CS03], Chesterfield et Schooler proposent de modifier RTP/RTCP [SCFJ96] pour le faire fonctionner sur l'environnement SSM. Leur solution vise les systèmes multimédia de grande échelle et ne considérent pas le cas des conférences multipartites de taille moyenne. Leur solution pourra fonctionner pour certaines applications [MJ95, aMASK98] mais réduit l'information fournie par le protocole RTP/RTCP pour des applications de conférence et pourra seulement fonctionner qu'avec des applications multipoints utilisant RTP/RTCP. Nous essayons de définir ici une architecture pour les applications multi-sources, aussi indépendante que possible de l'application et de fournir seulement des services génériques pour un groupe applicatif.

Une architecture basée sur le relais de session a été proposée dans [ZF01]. Ils proposent de mettre des proxy multiples dans le réseau pour améliorer le déploiement de SSM. Ils ne considèrent pas problème du multi-source dans les réseaux SSM.

Des travaux plus récents proposent de modifier le modèle SSM pour permettre à des sources multiples d'émettre vers un canal [SNA03]. Ils arguent du fait que ceci ne recquiert que de petites modifications dans les entrées de routage, améliore l'extensibilité aux états multipoints, et élimine le problème du point unique de panne présent dans les solutions proposées par [Hol01]. Nous croyons que leur solution nécessitera un effort très élevé de déploiement car elle nécessite de changer les spécifications d'IGMP [Fen97] et de PIM-SSM, dans les hôtes et dans les routeurs.

Cet article décrit un protocole qui pourrait être utilisé dans l'architecture actuelle de SSM pour réaliser la découverte de source dans les réseaux SSM existants. Nous avons implémenté le protocole dans une bibliothèque gratuite disponible pour la communauté de recherche [BH].

## 5.3 Vue d'ensemble de l'architecture

Dans cette section, nous donnerons une première vue d'ensemble d'une architecture qui résout les problèmes cités, et permet l'utilisation des canaux SSM dans des applications multi-sources (ASM).

### 5.3.1 Problèmes à résoudre

Nous souhaitons utiliser des canaux dans des applications multi-sources, mais pour faire cela, nous avons plusieurs problèmes à résoudre. Parmi les problèmes de l'émulation d'un service ASM au-dessus d'un réseau SSM nous considérons différents critères qui couvrent diverses zones de l'ingénierie et de la recherche réseau. Déployabilité, extensibilité, robustesse et facilité d'implémentation pour une maintenance aisée sont une partie de ces critères. Nous donnons d'abord les critères généraux que nous voulons respecter, puis nous décrivons les problèmes spécifiques liés à l'émulation d'ASM.

#### 5.3.1.1 Critères généraux cpour cette architecture

a - Architecture rapidement déployable et utilisable.

b - Capacité à utiliser le multipoint multisource (ASM) au niveau inter-domaine.

c - Bonne extensibilité concernant le nombre de sources annoncées par groupe.

d - Tolérante aux fautes liées à la dynamicité des sources.

e - Portable, indépendant de la version de la pile IP utilisée et facile d'emploi pour un développeur.

f - L'architecture doit être transparente pour le modèle ASM.

#### 5.3.1.2 Emulation ASM

Premièrement, dans le modèle ASM décrit par Deering, un groupe multipoint est seulement identifié par son adresse IP. Mais ici, nous travaillerons avec des canaux multipoints SSM, qui sont décrits par une paire $(S, G)$ composée de l'adresse IP point-à-point d'une source $S$ et d'une adresse IP multipoint G où seulement $S$ à le droit d'envoyer des données. Mais alors, sachant que actuellement dans la plupart des applications multipoints multi-sources, un groupe multipoint est identifié par son adresse IP multipoint et le port utilisé pour envoyez des données, le problème d'identifier un groupe multipoint apparaît.

Deuxièmement, d'une part, dans l'ASM quand un récepteur joint un groupe, il découvre automatiquement les sources quand elles envoient des données au groupe. De la même manière, quand une nouvelle source apparaît, les récepteurs l'apprennent simplement en recevant les paquets envoyés. D'autre part, avec le modèle SSM, aucun tel mécanisme n'existe, parce que nous travaillons avec les canaux $(S, G)$ où $S$ est la source spécifique pouvant émettre sur ce canal.

En conséquence, les récepteurs doivent avoir un moyen de découvrir les sources **actives**et d'être au courant de l'arrivée des nouvelles sources. Pour réaliser ces objectifs, nous avons deux choix comme cela est exposé dans [Hol01] :

- Relais de session.
- Annonce de l'émetteur.

Nous expliquons maintenant une solution possible qui respecte les critères définis précédents. Cette solution est basée sur l'annonce de l'émetteur.

### 5.3.2 Solution proposée

Pour ce qui concerne les critères généraux, et avec plus de précision pour le point a) ceci signifie proposer une solution déployable sur les hôtes, et comptant seulement sur un service multipoint minimal et extensible offert par les opérateurs réseaux. Le point b) signifie utiliser le service SSM, parce qu'il est simple à maintenir pour des opérateurs et parce qu'il est considéré comme une solution viable pour le multipoint dans l'Internet IPv4 au moins. Les points c), d) et e) signifient utiliser une retransmission à états souples et UDP pour les messages de contrôle parce qu'avec ces mécanismes, nous pouvons limiter la quantité de bande passante des messages de contrôle et effectuer de la gestion d'erreur non fournie par UDP. Nous croyons que les problèmes de congestion seront résolus par le fait que notre architecture est très légère en termes de messages envoyés sur le réseau.

Pour ce qui concerne le problème de l'émulation, cette solution utilise au moins un canal par source, et un canal de contrôle, utilisé pour relayer les diverses informations de gestion et les annonces de source pour le groupe. Ce canal de contrôle identifie le groupe multipoint, et utilise UDP pour envoyer des données. Par conséquent, pour annoncer un groupe, il est seulement nécessaire d'annoncer ce canal et le port UDP employé, les récepteurs n'ayant qu'à joindre canal multipoint et le lier pour obtenir toute l'information nécessaire pour participer à la session. La source de ce canal de contrôle aura un certain contrôle sur le groupe, et exécutera diverses

exécutions pour ce groupe, comme annoncer de nouvelles sources, donner la liste de celles existantes sur demande et faire optionnellement de la gestion de groupe. Nous appelons la source du canal de contrôle le **contrôleur**.

Comme il est possible pour un même contrôleur d'avoir autorité sur plusieurs groupes, nous avons décidé d'utiliser un contrôleur unique qui prend soin des groupes multiples. Ce contrôleur utilise une socket UDP unique écoutant sur un port annoncé pour communiquer avec les sources ou les récepteurs qui veulent apprendre les sources existantes. Les messages d'annonce de source et d'autres messages de gestion sont envoyés à tous les récepteurs par l'intermédiaire du canal de contrôle. Un canal de contrôle correspond à un groupe ASM au niveau du réseau. Ce contrôleur $S$ prend soin de tous les canaux de contrôle identifiés par un canal $(S, ...)$. Ces canaux n'utilisent pas nécessairement le même port UDP pour les données à envoyer.

Ce contrôleur est exécuté au niveau application du modèle OSI, parce qu'il prend soin de l'interaction entre le réseau et les applications utilisées dans des sessions multipoints. L'avantage d' avoir ce contrôleur s'exécuter sur la couche application, est qu' il est possible de choisir son emplacement de sorte que les performances soient les meilleures, et il présente la possibilité de faire de la gestion de groupe dans la couche application. D'ailleurs manipuler des sockets UDP est très facile et permet une implémentation rapide sur les hôtes.

## 5.4 Emulation ASM au-dessus d'un service réseau SSM

D'abord nous rappelons ce qu'est exactement un groupe dans le modèle ASM. En second lieu nous considérons les différents cas qui sont possibles lorsqu'on fait de l'émulation ASM au-dessus d'un service réseau SSM.

### 5.4.1 Propriétés des groupes ASM

Un groupe dans le modèle de service ASM est un ensemble de zéro ou plus interfaces attachées pour recevoir ou envoyer des données au même identificateur de groupe G pris dans l'espace d'adresse multipoint d'IP. Dans ce groupe, nous pouvons distinguer :

- les interfaces membres uniquement réceptrices (appelées **passives** dans le reste de ce chapitre).
- les interfaces membres qui envoient seulement (appelées **actives** dans le reste de ce chapitre).
- les interfaces membres qui envoient et qui reçoivent (appelées **duales** dans le reste de ce chapitre).

Dans le modèle ASM, nous avons un identificateur $G$ de groupe qui peut se composer de membres **passifs**, de membres **actifs** et de membres **duals**, il n'y a aucune distinction et des membres **passifs** peuvent devenir **actifs** à tout moment sans règles précises et vice-versa. Un membre **dual** commute entre les deux états d'une façon non définie. En outre, il n'y a aucun engagement pour joindre le groupe (*i.e.*, être membre **passif**) pour y envoyer des données car le groupe est ouvert.

Un groupe dans le modèle de service SSM a plus de restrictions et par conséquent plus de contrôle. L'identificateur de groupe est composé du canal $(S, G)$ où $G$ est encore pris dans l'espace d'adresse multipoint IP (disjoint de l'espace d'adresse d'ASM). Il identifie un ensemble de membres **passifs** où seulement un hôte, $S$, (pas nécessairement **passif**) a été choisi par les membres **passifs** pour transmettre des paquets. Cet hôte devient **actif** du point de vue des récepteurs mais ceci n'est pas nécessairement le cas. Même avec cette hypothèse forte, le groupe est encore ouvert, selon la définition de Deering car la source n'appartient pas nécessairement au groupe de récepteurs.

Cela signifie que dans l'ASM, les flux $(S, G)$ et $(S1, G)$ seront délivrés au même ensemble de membres **passifs** identifiés par $G$ (nous ne considérons pas le filtrage IGMPv3 de l'ASM). Dans SSM, les flux $(S, G)$ et $(S1, G)$ peuvent se rapporter à un ensemble totalement différent de membres **passifs**.

Vu cette différence principale, le défi de l'émulation du modèle ASM sur un service réseau SSM peut être définie comme suit :

*à tout instant, étant donné un ensemble de canaux $(S, G), (S1, G1)...(S_n, G_n)$, chacun d'eux doit se référer au même ensemble de membres (hôtes) **passifs**. Cette émulation doit prendre en compte l'hypothèse forte que les membres **duals**, les membres **passifs** et les membres **actifs** peuvent aller et venir n'importe quand.*

### 5.4.2 Les différent cas d'émulation

Nous considérons trois degrés de dynamicité :
- la dynamicité des membres **duals**
- la dynamicité des membres **passifs**
- la dynamicité des membres **actifs**

#### 5.4.2.1 Cas 1 : Les membres passifs et actifs sont statiques

Si nous considérons l'hypothèse de simplification que le groupe est statique, (*i.e.*, il se compose toujours du même ensemble de membres **duals** pendant la durée entière de l'application), chaque membre peut se connaître par l'intermédiaire d'une simple configuration statique ou administrative. Chaque $S1, S2, ..., S_n$ membre **dual** peut se synchroniser sur un groupe G commun et joindre les canaux $(S1, G), (S2, G), ..., (S3, G)$. Nous avons alors un modèle ASM sur le service réseau SSM avec la seule différence que le groupe n'est pas ouvert, il est limité à $S1, S2, ..., S_n$ et aucune nouvelle source ne peut y apparaitre. Cette solution est efficace en termes de délais parce que, après la mise en place des différents arbres sources entre chaque membre **dual**, les émetteurs n'ont plus qu'à envoyer les données et les chemins des émetteurs aux récepteurs sont déjà établis.

#### 5.4.2.2 Cas 2 : Les membres passifs sont statiques et les membres actifs sont dynamiques

Nous considérons un ensemble de sources **actives** changeant dans le temps. Ceci signifie que l'ensemble des canaux SSM changent sur le temps. Les membres **passifs** sont statiques. Ce cas peut ne pas sembler réaliste mais on peut imaginer une application avec un ensemble fixe de membres **passifs** M1, M2, ..., Mn attendant des données de différentes sources changeant au cours du temps (qui peuvent être un sous-ensemble des membres **passifs**). Dans ce cas-ci, chaque membre **passif** pourrait écouter une autorité centrale annoncer l'arrivée/départ de nouveaux membres **actifs**pendant la durée de la session. Notons que ce cas peut concerner des applications avec des sources mobiles, car la dynamicité des sources peut être représentée comme un hôte se déplaçant entre différents réseaux IP.

#### 5.4.2.3 Cas 3 : Les membres passifs sont dynamiques et les membres actifs sont statiques

Nous considérons que l'ensemble des sources ne change pas dans le temps. Ainsi, à n'importe quel moment chaque membre **passif** peut joindre les canaux $(S1, G1)$, $(S2, G1)$, ..., $(S_n, G_n)$ de l'ensemble des membres **actifs** et les quitter lorsqu'il quitte le groupe ASM émulé. Tant que les membres **passifs** ne doivent pas communiquer les uns avec les autres (ils sont membres **passifs**), c'est une solution viable parce que les membres **actifs** peuvent encore communiquer avec les membres **passifs** par l'intermédiaire des canaux.

#### 5.4.2.4 Cas 4 : Les membres passifs et actifs sont dynamiques

C'est le cas le plus général mais le plus réaliste aussi, particulièrement quand nous considérons que beaucoup de protocoles de contrôle (comme RTCP) recquierent le feedback des deux parties. L'ensemble des sources change au cours du temps et l'ensemble des récepteurs aussi. Donc à tout moment, chaque membre **passif** doit écouter un service central dynamique donné qui annonce le départ et l'arrivée des sources. Contrairement au cas deux, un membre **passif** peut devenir un membre **actif** pour n'importe quelle raison et doit entrer en contact avec ce service d'annonce dynamique pour annoncer son arrivée. Dans ce cas nous avons un modèle ASM sur le service réseau SSM avec la seule différence que les hôtes d'extrémité gèrent la dynamicité/découverte des sources **actives** dans le groupe.

Nous avons vu les différences principales entre les modèles ASM et SSM, nous avons montré que en cas de dynamicité des membres **actifs** il y a un besoin d'un mécanisme de découverte de source placé quelque part dans les hôtes d'extrémité, car nous considérons que la couche réseau offre seulement le modèle SSM de service réseau. Ce mécanisme de découverte de source nécessitera une signalisation entre membres **actifs**et **passifs**. Nous décrivons en détail cette signalisation dans la section 5.5. Nous décrivons maintenant le service de découverte de source et comment nous pensons que ce service devrait être contrôlé par les fournisseurs d'accès.

### 5.4.3 Découverte de source en tant que service

Un fournisseur de service peut proposer un "mécanisme simple de découverte de source" basé sur cette proposition. Ceci est important parce qu'une application peut ne pas avoir assez de bande passante pour traiter le fardeau des messages de signalisation produit par cette émulation ASM sur SSM.

S'éxécutant comme service, il doit être utilisable pour plusieurs groupes et pour plusieurs types d'applications, sachant que les applications peuvent être en écoute ou en envoi vers plusieurs groupes. En tant que service, il peut être exécuté n'importe où sur le réseau (y compris sur la machine locale) et là peuvent apparaitre quelques limitations administratives ou techniques concernant nombre de groupes servis. De même, comme il est exécuté en tant que service, il doit y avoir des mécanismes d'authentification pour autoriser quelqu'un à utiliser ce service avant la création du groupe.

De plus nous devons définir la transmission entre le contrôleur applicatif et le contrôleur de réseau défini dans la section 5.3.2. Cette transmission, composée de messages administratifs, doit être aussi générique que possible car beaucoup de types d'applications pourront utiliser ce service.

### 5.4.4 Porter les applications ASM avec ce service

Nous considérons ici des applications simples UDP avec seulement une socket de données utilisée pour recevoir et envoyer des informations à un groupe ASM. Beaucoup d'applications Internet utilisent une socket de communication. Il faut noter que si l'application s'exécute sur UDP, les récepteurs et les sources doivent se synchroniser sur un port commun où les sources effectuent l'envoi et les récepteurs ont lié leurs sockets.

Pour émuler entièrement l'ASM pour un tuple $(G, P)$ où $G$ est une adresse IP multipoint ASM et $P$ un numéro de port (pour RTCP par exemple), un créateur d'application devra joindre un groupe ASM émulé au lieu d'un groupe ASM simple et s'enregistrer comme une source dans ce groupe émulé sur une socket précédemment créée. Le groupe ASM émulé identifié par le tuple $(S, G1, P)$ où $S$ est l'adresse IP du contrôleur, $G1$ une adresse IP choisie dans l'intervalle

d'adresse SSM et $P$ le même numéro de port que dans ASM. Juste avant la destruction de la socket ou après un long temps d'inactivité l'application devra radier la source.

En pratique, cela signifie trouver où la vieille application joint le groupe ASM, et substituer cette opération par une adhésion au groupe ASM émulé et trouver où l'application envoie des données au groupe et enregistrer l'application comme source avant la transmission.

La dernière partie est plus difficile, car consiste à déterminer quand l'application arrête d'envoyer des données au groupe. Par simplification, on peut considérer qu'une application cesse d'envoyer des données après que l'application se soit terminée ou que sa socket de données se soit fermée.

Mais notons que l'enregistrement devrait se produire seulement une fois dans l'application car il concerne la source IP.

Si l'application ASM est le créateur du groupe (cela signifie la première à avoir assigné une adresse IP de classe D) alors la création de groupe est contraire à l'ASM classique où la création est implicite. L'application qui a appelé cette primitive est considérée comme l'opérateur du groupe et a tous les droits sur lui, y compris celui de le clôturer même s'il y a des sources encore actives.

Notons que même si l'application fonctionne sur UDP, le service d'émulation ASM ne devrait pas modifier le comportement de la socket par rapport à son comportement IP. Même s'il y a des informations de transport contenues dans la phase d'enregistrement, elles doivent seulement être employées pour informer l'application.

Une source peut émettre vers différents canaux pour le même groupe ASM émulé. Cela signifie enregistrer une nouvelle source avec la même adresse émulée mais en indiquant un autre canal. Cela ouvre de nouvelles possibilités non présentes dans le modèle ASM.

Il peut y avoir des applications où les récepteurs connaissent les sources mais les sources ne connaissent pas les récepteurs. Il peut y avoir des protocoles qui ont besoin d'un comportement ASM pur comme le protocole RTCP. Il revient au créateur d'application de déterminer quel est le meilleur mode approprié pour l'exécution de son application.

## 5.5 Description détaillée du protocole

Dans cette section, nous décrivons comment les récepteurs apprennent l'existence des sources actives pour un groupe à un moment donné, et les nouvelles sources qui pourraient apparaitre pendant la session.

### 5.5.1 Description de l'interaction source-contrôleur

Nous décrirons l'interaction entre une source pour un groupe, et le contrôleur (ce signifie le contrôleur responsable de la source qui s'annonce) pour ce groupe.

#### 5.5.1.1 Avec une nouvelle source

Quand une nouvelle source veut s'annoncer, elle doit commencer par informer le contrôleur. Le contrôleur enregistre alors les informations sur la source dans son cache. Pour permettre la robustesse et l'extensibilité, cette annonce est envoyée périodiquement avec un message composé d'un en-tête de protocole avec une charge SDP facultatif [HPW00, HJ95]. Nous appelons ce message $ON$. Il est envoyé à la socket UDP du processus du contrôleur. Ce message doit contenir toute information nécessaire pour qu'un récepteur joigne le canal utilisé par la source pour envoyer ses données au groupe. Il est composé d'information sur le canal de la nouvelle source et optionnellement d'information de niveau transport ou IP.

Il faut noter que l'information de transport est seulement destinée aux applications et pas au protocole décrit ici. Dans l'exemple précédent, le comportement de l'application est le même que pour l'ASM. Cela signifie que les sources UDP doivent envoyer au même port UDP annoncé et que chaque récepteur doit se connecter au même port UDP pour la réception de données.

Une fois que ce message $ON$ a été traité, le contrôleur annonce la source aux récepteurs et envoie optionnellement un $ON_{RESP}$ à la nouvelle source pour l'informer au sujet de divers paramètres de session, selon la politique de gestion du groupe. Pour faire ainsi, elle achemine ce message au groupe tout entier par l'intermédiaire du canal de contrôle, si la source est autorisée à envoyer des données au groupe. Par conséquent, tout les récepteurs apprennent que cette source existe, et peuvent joindre le canal qu'elle utilise pour envoyer des données si ils le désirent. La figure 5.1 illustre ce mécanisme.

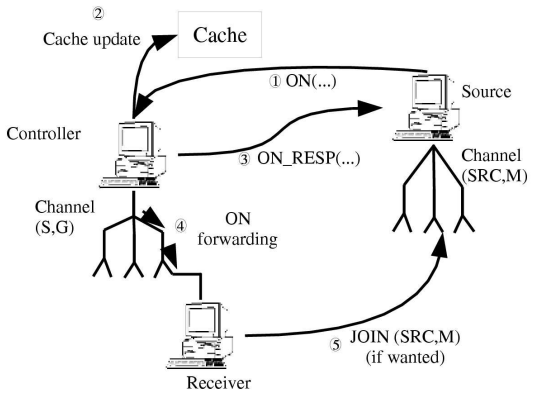


FIG. 5.1 – Interaction source-contrôleur pour un message $ON$

#### 5.5.1.2 Avec une source existante

Périodiquement, une source doit envoyer un message $ON$ au contrôleur, afin de l'informer qu'elle est encore en activité, et le contrôleur achemine ce message sur son canal de contrôle aux récepteurs. Pour chaque source, le contrôleur garde une validité du temporisateur $ON$ qui donne le temps durant lequel l'entrée de la source dans son cache est valide. Si le contrôleur ne reçoit pas un tel message avant que le temporisateur lié à la source se termine, il considère que la source n'est plus en activité, supprime l'entrée correspondante dans son cache et envoie un message $OFF$ pour informer les récepteurs que cette source n'existe plus.

Les sources et les récepteurs doivent également mettre à jour des temporisateurs afin de prendre soin de la validité des diverses informations et messages. Par exemple, une source doit exécuter un temporisateur pour la validité d'un message $ON$ afin de l'envoyer périodiquement.

Une source peut également explicitement annoncer qu'elle arrête son activité en envoyant un message $OFF$ au contrôleur, qui supprime l'entrée correspondante dans son cache et achemine ce message sur le canal de contrôle correspondant afin d'informer les récepteurs, qui quittent alors ce canal. Le mécanisme est illustré sur la figure 5.2. Notons que quelques protocoles de couches supérieurs ont déjà une fonction pour annoncer la fin d'une participation à une session (paquets BYE du protocole RTCP). Ce mécanisme peut être redondant pour quelques applications mais un point important de notre approche est que le plan de contrôle est sur UDP et les données sont dans le plan IP. Ceci signifie que toute application multipoint fonctionnant sur IP pourrait utiliser ce protocole de découverte de source contrairement à la solution proposée dans [CS03].

Le message $OFF$ est presque identique au message $ON$. Il est envoyé au port annoncé du canal de contrôle sur une socket UDP et est un message composé d'un en-tête de ce protocole

avec une charge utile contenant l'identificateur du groupe et l'identificateur du canal utilisés pour envoyer les données. Ces deux paramètres sont suffisants pour quitter le canal correspondant à la source. Si le message $OFF$ est perdu, le temporisateur de la source expirera dans le cache du contrôleur parce qu'il ne sera plus mis à jour par les messages périodiques $ON$. Par conséquent un message $OFF$ sera envoyé dans le canal de contrôle et les récepteurs se radierons de cette source.

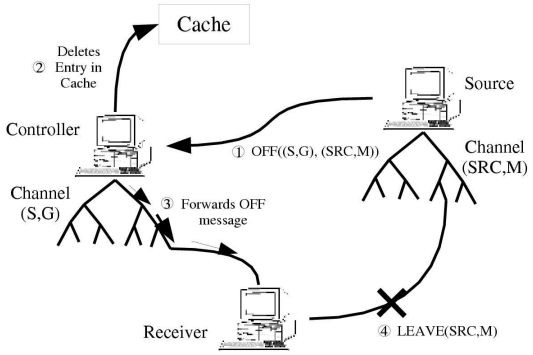


FIG. 5.2 – Interaction Source-Contrôleur pour un message $OFF$

### 5.5.2 Description de l'interaction récepteur-contrôleur

Ici nous décrivons l'interaction entre un récepteur et le contrôleur du groupe. Nous distinguons le cas quand un récepteur demande l'information au sujet des sources qui sont en activité pour le groupe, et du cas quand la source s'est annoncée elle-même.

#### 5.5.2.1 Requête d'information

Le contrôleur agit ici comme un répertoire des sources. Un récepteur qui veut apprendre la liste des sources existantes pour un groupe sur lequel le contrôleur a l'autorité, lui envoie un message en point-à-point sur un port connu du récepteur. Ce message est un paquet UDP avec une charge utile contenant l'en-tête de protocole et l'identificateur de groupe, qui est le tuple $(S, G)$. Nous appelons ce message un message $INFO_{REQ}$.

Quand le contrôleur reçoit un tel message, il regarde dans son le cache, et répond au récepteur en point-à-point avec un message $INFO_{RESP}$ donnant cette liste de sources et assez d'information pour que le récepteur puisse joindre les canaux correspondants si il le souhaite.

Si le contrôleur veut pouvoir fournir ces informations, il doit maintenir à jour un cache contenant celles-ci. Cela signifie que le cache doit contenir l'identificateur de groupe $(S, G)$. Ce cache peut contenir plus d'un contrôleur par groupe pour des raisons de robustesse par exemple et il contient les canaux de contrôle utilisés par ces contrôleurs. Nous trouvons les dates de début et de fin de validité de la session, ainsi que la liste des sources. Cela donne la table 5.1.

TAB. 5.1 – Informations dans le cache du contrôleur

| canal de contrôle | 2ème canal de contrôle | ... canal de contrôle | début | fin | sources |
|---|---|---|---|---|---|

La liste des sources se compose de zéro ou plusieurs entrées. Chaque entrée décrit une source et fournit l'information dont un récepteur peut avoir besoin pour joindre le canal utilisé pour envoyer des données au groupe. Une telle entrée est illustrée sur la table 5.2.

TAB. 5.2 – Détails de la liste des sources

| canal d'envoi | port | KICKED | MUTED | information de codage | temporisateur |
|---|---|---|---|---|---|

Les indicateurs KICKED et MUTED sont utilisés si la gestion de source est exécutée, et sont placés à 0 par défaut, et l'information de codage est facultative.

Ce cache est utilisé par le contrôleur pour stocker toute information pour un groupe, mais les récepteurs doivent mettre à jour le même cache, toutefois limité au groupe applicable joint. Ce cache est mis à jour grâce aux messages envoyés sur le canal de contrôle.

Le message $INFO_{RESP}$ est un paquet UDP avec comme la charge utile l'en-tête de ce protocole avec une description SDP décrivant la session et les sources. La partie SDP contient l'identificateur de groupe, les dates de début et de fin de la validité de la session, et une liste de zéro ou plus canaux et ports employés par les sources pour envoyer des données au groupe, et éventuellement plus d'informations sur ces sources dont les récepteurs pourraient avoir besoin.

Quand un récepteur reçoit un paquet $INFO_{RESP}$, s'il ne l'a pas encore fait, il joint le canal de contrôle, et fait la même chose avec les canaux utilisés par les sources qui l'intéressent. Voir figure 5.3.

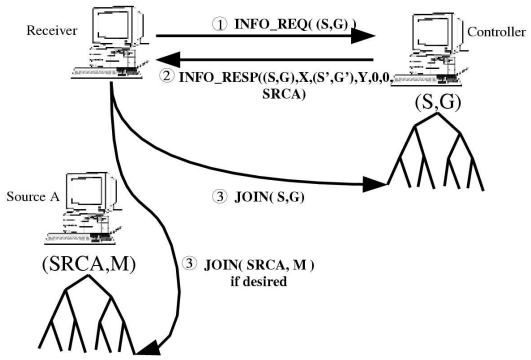


FIG. 5.3 – Interaction récepteur-contrôleur

#### 5.5.2.2 Annonce des sources

Quand une nouvelle source annonce son activité au contrôleur, le contrôleur doit informer tous les récepteurs. Pour se faire, il transfère le message $ON$ reçu de la source sur le canal de contrôle.

De la même manière, quand le contrôleur détecte le départ explicite ou implicite d'une source, il transfère ou envoie, selon le cas, un message $OFF$ pour cette source sur le canal de contrôle.

### 5.5.3 Diagrammes d'états du protocole

Dans cette section, nous décrivons les différents états dans lesquels une source peut être, selon les actions prévues dans notre protocole.

Les différents états dans lesquels peut être une source sont :

- ACTIVE : la source est considérée active, et le notifie periodiquement au contrôleur avec des messages $ON$.
- INACTIVE : la source n'a jamais été active ou a envoyé un message $OFF$.

Le diagramme d'état suivant illustré dans la figure 5.4 décrit les différents états et leurs transitions pour la partie émulation ASM sur SSM, ce qui représente en fait l'annonce des sources.

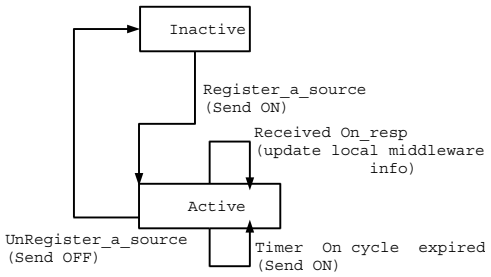


FIG. 5.4 – Diagramme d'état pour l'émulation ASM sur SSM

## 5.6 Evaluation

Dans cette section, nous donnons une brève analyse de notre proposition. Une analyse plus complète peut être trouvée dans [HBMP04]. Une question importante est l'extensibilité de cette architecture, particulièrement au niveau du contrôleur. Le contrôleur agit en tant que point de rendez-vous *ala* PIM, et un noeud de relais pour les messages de contrôle entre les sources et les récepteurs dans le groupe ASM émulé. C'est alors un point central de panne, sensible à la bande passante produite par les différentes sources annonçantes.

Nous considérons ici le problème de l'extensibilité de bande passante au contrôleur dans le cas moyen d'une périodicité de message $ON$ de 60 secondes. Cette période a la même valeur que celle des messages périodiques dans PIM-SM [EA98]. Quelle est la quantité de bande passante nécessaire au niveau du contrôleur en fonction du nombre de sources qui s'annoncent ? Si nous prenons les messages définis dans [BHP03], un message $ON$ qui contient un canal prendra 158 octets sur le câble, en considérant que la source transmet sur Ethernet et IPv6 et que ce le canal annoncé est en IPv6. Avec IPv4 il prendra 90 octets en considérant que le canal annoncé est en IPv4. Dans un cas critique, s'il y a N sources s'annonçant et se trouvant à N différents hôtes et que tous messages arrivent ensemble au contrôleur, la bande passante moyenne nécessitée est une fonction linéaire simple, N * 158 octets par mn pour IPv6 et N * 90 octets par mn pour IPv4.

La bande passante produite n'est pas trop élevée et les applications ayant plus de 1000 sources simultanées (12kb/s en IPv6) qui sont très spécifiques seront surement déployées avec d'autres solutions. D'ailleurs grâce au mécanisme d'accusé de réception, il est possible de limiter le débit des sources au niveau du contrôleur. Notons que, une fois que les messages arrivent au contrôleur, le contrôleur a le contrôle complet sur l'utilisation de la bande passante de son canal de contrôle.

## 5.7 Implémentation

L'API de notre intergiciel est définie dans un Internet draft mis à la disposition de la communauté de recherche [BHP03]. Ce draft contient la description complète des messages du protocole, le format des paquets ainsi que les définitions de tous les en-têtes et indicateurs et

la description et les valeurs typiques des temporisateurs. Nous avons implémenté notre intergiciel sous forme d'une bibliothèque appelé Libssmsdp qui peut être utilisé par des applications s'exécutant sur des systèmes d'exploitation Windows, Linux ou *BSD selon la version de la pile IP. La bibliothèque est écrite en C++ et contient plus de 8000 lignes de code. Elle est librement téléchargeable à l'adresse [BH]. Notons que notre bibliothèque se sert de la bibliothèque GNU Common C++ (librement disponible) et nécessite une implémentation hôte des protocoles multipoints IGMPv3 et MLDv2. Ces implémentations de protocole sont fournies par le projet Kame pour *BSD et peuvent être trouvées dans les dernières versions du noyau GNU/Linux (version 2.4.22) et dans les nouvelles versions 2.6.x. Actuellement, Windows fournit seulement une implémentation IGMPv3, par conséquent notre bibliothèque fonctionne seulement en IPv4.

## 5.8 Conclusion

Notre travail a consisté à fournir une bibliothèque multipoint pour les applications multi-sources en utilisant uniquement le protocole Source Specific Multicast au niveau de la couche réseau. Le protocole SSM est un nouveau paradigme de distribution multipoint sur IP : en poussant la sélection de source dans les hôtes, il devient un bon candidat pour offrir un service multipoint dans l'Internet tout entier. Il est bien adapté pour les applications de flux TV et radio mais il ne fournit pas la découverte automatique de source comme le paradigme multipoint initial ASM de Steeve Deering. En considérant des applications multi-sources avec dynamicité des sources et les vieilles applications basées sur le modèle ASM, il était clair qu'il manquait une bibliothèque multipoint (utilisable comme intergiciel) implémentant un protocole de découverte de source.

Nous avons comblé cette lacune en concevant une architecture capable de fournir un tel service de découverte. Nous avons défini un nouveau protocole et nous l'avons implémenté dans un intergiciel afin de permettre à des applications ASM, utilisant notre bibliothèque, de découvrir des sources SSM. Nous avons donné des éléments d'évaluation de cette idée et avons conclu que la fonctionnalité de découverte de source devrait être située dans les hôtes pour une majorité d'applications. Notre prochaine tâche va consister à effectuer l'évaluation des performances de notre proposition à l'aide de simulations détaillées. Nous projetons également de tester notre implémentation dans l'environnement du projet 6NET et de définir une architecture de proxy basé sur ce protocole afin d'utiliser des applications ASM non modifiées sur des réseaux SSM.

Le projet Source Specific Multicast Source Discovery Protocol Library (SSMSDP) s'inscrit dans le cadre du vaste projet européen *6net*. Mickaël Hoerdt est un membre direct du projet *6net* et le sous projet SSMSDP constitue de plus son coeur de thèse de doctorat. En tant que co-encadrant de la thèse de Mickaël Hoerdt, j'ai été impliqué dans les aspects recherche et évaluation de ce projet aux côtés de Jean-Jacques Pansiot, son directeur de thèse. L'implémentation de la bibliothèque Libssmsdp a été réalisée par Mickaël Hoerdt et Frédéric Beck. Un travail annexe à SSMSDP concernant de même le multipoint et le fonctionnement au niveau applicatif (*i.e.*, hôte) a été réalisé par Dragoş Mânzăţeanu [sMaatM03] dans le cadre de son stage de DEA que j'ai co-encadré avec Jean-Jacques Pansiot.

# Travaux relatifs au chapitre


1. Mickaël Hoerdt, Frédéric Beck, Damien Magoni and Jean-Jacques Pansiot. A source discovery protocol for ASM applications in SSM networks. In *Proceedings of the 3rd International Conference on Networking*, pages 324–330, Gosier, Guadeloupe, French Caribbean, March 2004.
2. Dragoş Mânzăţeanu and Damien Magoni. Performance evaluation of an application–layer multicast protocol. Technical Report ULP-LSIIT-RR-2003-03, Université Louis Pasteur, August 2003.
3. *Source Specific Multicast Source Discovery Protocol Library (libssmsdp)*. http ://www-r2.u-strasbg.fr/~hoerdt/libssmsdp/

# Chapitre 6

# Architecture distribuée pour réseaux recouvrants sur Internet

## 6.1 Introduction

Un nombre grandissant d'applications réseaux s'évertuent à créer des réseaux recouvrants ou plus simplement recouvrements (overlays en anglais) au dessus du réseau Internet lui-même. Plusieurs exemples de difficultés peuvent expliquer ce phénomène. Les modes de communications avancés tels quel la diffusion multipoint n'ont pas été déployés avec succès au niveau de la couche réseau, c'est pourquoi ces protocoles sont actuellement implémentés au niveau de la couche application ce qui entraîne de ce fait la création de réseaux recouvrants par des applications spécifiques. Les applications distribuées telles que les grilles (grid en anglais) et les technologies pair-à-pair (peer-to-peer en anglais) évoluent très rapidement. Cependant leur déploiement, à travers des réseaux hétérogènes tels que les réseaux globalement non adressables, pose des difficultés. Le manque d'adresses IP, par exemple, a entraîné une explosion des traducteurs d'adresses NAT (Network Address Translator) et donc créé une grande quantité d'hôtes non adressables globalement. Même si cela ne perturbe à priori pas trop le comportement d'une communication client-serveur, cela n'est plus vrai dans une application distribuée construite sur un modèle grille ou P2P dans lequel n'importe quelle entité peut jouer les deux rôles et a donc besoin d'une adresse globale. Des outils de niveau applicatif formant des réseaux recouvrants (globus) sont proposés pour résoudre ce problème, mais ils sont souvent spécifiques à des applications distribuées données. Les réseaux recouvrants créés par ces applications nécessitent en interne une forme d'adressage et de routage. Habituellement leurs topologies sont les plus simples possibles (*e.g.*, arbre, anneau) mais avec l'augmentation de leurs tailles, ces réseaux devront être capables de gérer des topologies bien plus complexes dans l'avenir.

Dans ce chapitre, nous proposons une architecture d'adressage de niveau applicatif et un mécanisme de routage ayant pour but de surmonter les limitations précédemment présentées. Notre architecture est conçue dans le but d'apporter une plus grande extensibilité et plus de flexibilité pour les communications entre membres d'une application distribuée quelconque formant une topologie non-triviale. Notre seule hypothèse est que les réseaux recouvrants soient créés au dessus de l'Internet et que leurs topologies soient par conséquent contraintes par celle de l'Internet lui-même. L'interêt de notre architecture est que les applications n'auront pas à mettre en place et à maintenir des topologies spécifiques. Ce chapitre contient trois sections. Dans la section 6.2 nous présentons des travaux antérieurs et relatifs sur l'adressage alternatif et les protocoles de routage compact. Dans la section 6.3 nous présentons les mécanismes de notre architecture d'adressage et de routage. Enfin dans la section 6.4, nous présentons quelques résultats de simulation afin d'évaluer l'efficacité de notre architecture. Nous étudions l'inflation des longueurs des chemins fournis par notre algorithme de routage, nous fournissons quelques

résultats concernant la résistance à la dynamicité du réseau et enfin nous donnons des résultats concernant l'extensibilité de notre système de nommage.

## 6.2 Contexte

### 6.2.1 Types de recouvrements

Un point important sur lequel beaucoup de chercheurs conviennent est que le nommage et l'adressage devraient être séparés [FYT91, AWSBL99, Mos01, Mag03a]. Trois niveaux de nommage ont même été proposés par Balakrishnan *et al.* dans [BLR+04]. L'adressage indirect résultant est proposé par beaucoup de protocoles de routage expérimentaux (*e.g.*, Intentional Naming System [AWSBL99], IPNL [FG01] et i3 [SAZ+02]) et particulièrement ceux conçus pour les hôtes mobiles (*e.g.*, TCP-Migrate [SB00] et Tribe [VdaFR03b, VdaFR03a]). Plusieurs architectures ont donc été proposées pour fournir différents services qui puissent être déployés sur la base des réseaux existants. Toutes ces solutions créent des recouvrements, bien que pas toujours au niveau de la couche application, afin de résoudre des problèmes tels que l'adressage uniforme et la mobilité. Des recouvrements de niveau application fournissant le support du multipoint ont été également conçus [LN01] et implémentés (*e.g.*, Yiod [Fra00], Narada [hCRZ00, hCRSZ01], SALM [BBK02], NICE [BKK+03] et ROMA [KB04]) pour être employés au-dessus des réseaux qui ne fournissent pas de protocoles multipoints. Des recouvrements peuvent également être conçus pour fournir de nouveaux services tels que la gestion de réseaux résilients (*e.g.*, RON [ABKM01]) et la recherche d'objet dans les réseaux pair-à-pair (*e.g.*, Chord [SMK+01], Pastry [CDG+02], CAN [GC01], YAPPERS [GSGM03]).

### 6.2.2 Internet Indirection Infrastructure (i3)

Afin de généraliser l'abstraction des communications point à point, et pour fournir des services tels que le multicast ou l'anycast, Ion Stoica *et al.* [SAZ+02] proposent un système d'indirection, basé sur un recouvrement (overlay, *i.e.*, une surcouche réseau). Ce système propose un point de rendez-vous pour établir les communications. Plutôt que d'envoyer explicitement un paquet vers sa destination, chaque paquet est associé à un identifiant et cet identifiant est utilisé par le récepteur pour obtenir la livraison du paquet.

Le but de i3 est de fournir une indirection. En faisant cela, émission et réception sont clairement séparés. Le principe est simple : les sources envoient leurs paquets à un *identifiant* logique, et les récepteurs expriment leur volonté de recevoir ces paquets envoyés à l'identifiant. Ce modèle est donc similaire au principe du multicast IP classique. Cependant, il existe une différence cruciale entre les deux modèles : l'équivalent i3 du JOIN effectué classiquement pour le multicast est ici beaucoup plus flexible. En effet, en multicast IP normal, les routes doivent être établies par l'infrastructure. A l'opposé, i3 permet aux récepteurs de contrôler le routage des paquets. Cela offre deux avantages. Cela permet d'abord de créer, à un niveau applicatif, des services tels que la mobilité et l'anycast à partir de ce service de base. Deuxièmement, l'infrastructure peut développer une construction en arbre efficace vers les destinataires pour faire du multipoint applicatif tout en restant simple et robuste.

Des simulations ont été effectuées pour tester l'efficacité du routage par i3. La métrique utilisée pour évaluer les performances est le rapport entre le temps de latence inter noeuds en utilisant l'overlay i3 et le temps de latence inter noeuds en utilisant le routage IP normal. On appellera ici ce rapport *overhead*. Les serveurs i3 et les identifiants qu'ils devaient prendre en compte étaient répartis aléatoirement sur les réseaux. Globalement, sur ces réseaux, les temps de latence mesurés sur un simulateur basé sur le protocole Chord [SMKK02] ont révélé un

overhead de 4 à 6 en moyenne. Cela ne prouve pas que l'architecture i3, au stade où elle a été testée est efficace, mais elle est néanmoins réalisable.

### 6.2.3 IPNL

IPNL [FG01] signifie IP Next Layer. Cette architecture a pour but de résoudre les problèmes du manque d'adresses IPv4 sur Internet. Une architecture NAT étendue est une architecture dans laquelle seuls les hôtes et les boîtes NAT (Network Address Translator) sont modifiées. En particulier, IPv4 et les protocoles sous-jacents ne sont pas modifiés. Un des plus gros changements observés sur ces dix dernières années au niveau de l'architecture IP a été l'introduction de ces NAT [EF94]. Cela a entraîné le fait que de nombreux hôtes situés derrière des boîtes NAT ne peuvent plus établir de communication point à point.

L'avantage évident d'un mécanisme tel que NAT est qu'il permet d'augmenter immédiatement l'espace d'adressage de IPv4. Un autre avantage qui ne doit pas être négligé est qu'il permet d'isoler l'espace d'adressage d'un site (c'est à dire un réseau privé) de l'espace d'adressage global utilisé jusque là sur Internet. Ainsi, un réseau privé peut parfaitement changer de fournisseur d'accès à Internet sans avoir à reconfigurer complètement son propre réseau. D'un autre côté, cela empêche l'introduction de certaines applications telles que les applications peer-to-peer. Il y a deux raisons à cela : un hôte situé derrière une boîte NAT n'est généralement pas adressable sur Internet, du moins pas d'une manière classique. De plus, certaines applications peer-to-peer ne parviennent pas à faire fonctionner correctement une translation d'adresse ou de port. Ces applications ont alors besoin qu'une passerelle de niveau applicatif soit implémentée dans les boîtes NAT. Un exemple d'une telle application est le protocole SIP (Session Initiation Protocol) [HSSR99]. Un autre aspect négatif de NAT est qu'il complique les opérations effectuées sur des réseaux à grande échelle, ainsi que la conception de nouveaux protocoles et de nouvelles applications.

IPNL a pu être testé dans un noyau Linux en implémentant les fonctionnalités d'un nl-routeur dans Click [KMC+00], et les fonctionnalités des hôtes dans la pile TCP/IP de Linux. Le prototype testé était composé de 8 machines sous Linux jouant le rôle des hôtes, divisées en deux sites de 4 hôtes chacun, chaque site comportant 2 royaumes. Il y avait en plus 8 machines jouant le rôle des nl-routeurs, avec un nl-routeur pour chaque royaume et 2 frontdoors pour chaque site. Chaque royaume a été configuré avec une zone DNS [Moc87], et les deux sites étaient reliés par des routeurs IP classiques simulant le royaume du milieu. Le benchmark TCP "netperf" a été lancé pour mesurer l'overhead entraîné par IPNL, et aucune dégradation de performances par rapport à un réseau NAT normal n'a été observée. Une panne a également été simulée sur un lien, et la connexion s'est rétablie en moyenne au bout de 3 secondes, en utilisant un protocole de routage qui rafraîchissait les routes toutes les 5 secondes. Il ne faut toutefois par perdre de vue, afin d'évaluer correctement ces résultats, que le réseau déployé pour ces tests n'était pas très étendu.

### 6.2.4 Localisation et routage décentralisés d'objets

Les avancées récentes pour la localisation décentralisée d'objets et des systèmes de routage (DOLR pour Decentralized Object Location and Routing) fournissent des mécanismes intéressants pour les réseaux à grande échelle. Ces systèmes proposent des algorithmes qui s'adaptent de façon logarithmique à la taille du réseau. La publication de Ben Y.Zhao *et al*. [YDK03] propose d'étudier le principe de *connaissance locale* et de montrer comment celui-ci peut jouer un rôle positif pour fournir une extensibilité encore plus importante sur les réseaux à grande échelle.

Le principal avantage des DOLR est d'offrir une extensibilité accrue pour des services déjà présents tels que les Domain Name Service, les moteurs de recherche sur le web, et surtout pour les technologies Peer-to-Peer (P2P) qui permettent le partage de fichiers. Ces DOLR utilisent des algorithmes de recherche pour fournir des tables de routage et un nombre de sauts entre les noeuds qui augmentent tous deux de façon logarithmique par rapport à la taille totale du réseau, tout en garantissant la bonne consistence des opérations. Lorsqu'on étudie l'extensibilité, il est important de pas considérer uniquement l'algorithme de routage mais également l'infrastructure toute entière. Pour un déploiement sur une zone étendue, les algorithmes DOLR ont besoin de supporter des mécanismes qui pourraient avoir une influence sur l'extensibilité de tout le système. Or, les effets de ces mécanismes ne sont pas encore bien compris. Ben Y.Zhao *et al.* ont donc étudié les propriétés de plusieurs travaux de recherche récents tels que CAN [RFHaSS01], PASTRY [RD01], CHORD [SMKK02], Tapestry [ZKJ01] et YAPPERS [GSGM03], qui utilisent des DOLR dans le but de construire des mécanismes adaptés à des réseaux et à des applications à grande échelle. Ces quatre propositions fournissent toutes une localisation d'objets sur des zones étendues en routant des messages vers une clé ou un identifiant unique. Il y a deux mécanismes fondamentaux pour différencier ces projets et leurs caractéristiques d'extensibilité. Le premier est le noyau de l'algorithme utilisé pour construire et gérer les tables de routage locales. Le deuxième est la mise en cache des données permettant de localiser un service particulier.

Il est clair que dans un système à grande échelle, tout point comportant un grand nombre d'informations nécessaires au bon déroulement de l'overlay peut devenir un point de congestions et de pannes. C'est pourquoi certains projets tentent de diverses manières de répartir les informations, de les distribuer au maximum. Pour effectuer du multicast, le projet Scribe [RKDC01] par exemple permet aux noeuds intermédiaires situés dans l'arbre de diffusion de gérer eux-même les messages émis par les membres. En supposant que les clients s'enregistrent en étant à proximité d'une branche de l'arbre de diffusion multicast, les messages des membres n'ont besoin de parcourir qu'une courte distance vers un parent proche pour être traités. Cela réduit ainsi la bande passante utilisée et les temps de latence. Une fois encore, on s'aperçoit que des mécanismes de routage appropriés peuvent contribuer à améliorer l'extensibilité d'un système. Ici, c'est en divisant la somme de travail à effectuer sur les différents noeuds que les mécanismes sont rendus plus flexibles.

### 6.2.5 Coût du routage distribué dans les recouvrements

Le compromis entre les tailles des tables de routage et les longueurs de chemin a été activement étudié par la communauté de recherche des algorithmes distribués. Le travail théorique sur le routage compact par Eilam *et al.* a prouvé dans [EGP98] qu'il est possible de plafonner l'étirement moyen (*i.e.*, l'inflation des longueurs de chemin) par 3 avec des tables de routage de la taille $O(n^{3/2}\log^{3/2}n)$. De même Cowen a prouvé dans [Cow99] qu'il est possible de plafonner l'étirement maximum par 3 avec des tables de routage de la taille $O(n^{2/3}\log^{4/3}n)$. Plus récemment Krioukov *et al.* ont montré que le routage compact dans l'Internet donne un étirement moyen de 1.1 [KFY04]. Cependant dans tous ces cas, ils ne décrivent pas comment implémenter le routage compact de façon distribuée.

Récemment, Xu *et al.* [XMK03] ainsi que Li *et al.* [LM04] ont montré que les topologies des recouvrements ont un impact sur les performances du routage dans ces recouvrements. Waldvogel *et al.* [WR02], Qiu [QYZS03] et Nakao *et al.* [NPB03] ont d'ailleurs proposé des architectures capables d'exploiter au mieux les topologies des recouvrements. Dans ce chapitre nous présentons une architecture distribuée où les tailles des tables de routage ne sont pas une fonction de la taille de réseau mais du degré des voisins. Bien que notre architecture ne fournisse pas une limite supérieure sur l'inflation moyenne, celle-ci est habituellement en-dessous de 3 ce qui semble acceptable suivant les indications de la section 6.4.

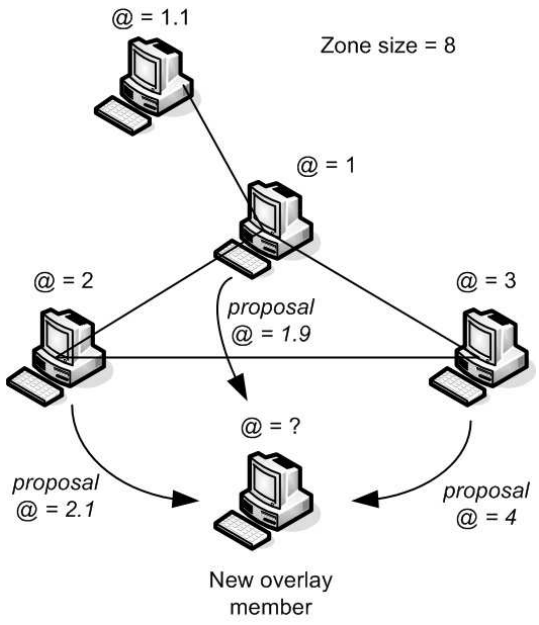


FIG. 6.1 – Nouveau noeud demandant des adresses.

## 6.3 Architecture de communication pour recouvrements

Dans cette section nous présentons les mécanismes de notre architecture qui permet de construire des recouvrements sur l'Internet. L'intérêt de notre architecture est de découpler l'adresse et le nom des hôtes ce qui permet de mettre en oeuvre de façon efficace des mécanismes de mobilité, de diffusion et de sécurité.

### 6.3.1 Adressage

Chaque noeud d'un recouvrement a au moins une adresse. Une adresse se compose d'un ou plusieurs champs contenant des nombres et séparés par des points. Un champ correspond à un niveau de la hiérarchie. Le niveau de l'adresse est égal au nombre de champs dans l'adresse. Le préfixe d'une adresse est égal à l'adresse sans son dernier champ. Le dernier champ s'appelle le champ local. Le nombre de niveaux dans la hiérarchie n'est pas limité. Le plan d'adressage contient des zones qui correspondent aux champs des adresses. La taille d'un champ définit ainsi la taille maximum de la zone. Toutes les zones ont la même taille maximale fixée à $n$ (appelé la taille de la zone). Il y a une zone de niveau 1 contenant $n$ noeuds (définis dans le premier champ de l'adresse). Puis il y a au plus $n$ zones de niveau 2 chacunes qui contiennent au plus $n$ noeuds (définis par les 2 premiers champs de l'adresse). Puis il y a au plus $n^2$ zones de niveau 3 chacunes qui contiennent au plus $n$ noeuds (définis par les 3 premiers champs de l'adresse), et ainsi de suite. Ceci signifie que tout l'espace d'adresse peut être théoriquement distribué et si nous avons $k$ niveaux et $l$ bits par niveau, nous pouvons adresser $2^{k \times l}$ noeuds.

L'adressage des noeuds de recouvrement est entièrement distribué : il se fonde seulement sur la connaissance locale de chaque noeud. La seule connaissance locale sur laquelle nous comptons pour le moment est le degré du noeud et les adresses et les degrés de ses voisins. N'importe quel noeud est censé connaître cette information. Supposons que la taille de zone est $n$. Chaque noeud qui a l'adresse $w.x.y$ est responsable pour attribuer les adresses suivantes à ses voisins :

– l'adresse $w.x.(y+1)$ (appelée l'adresse « suivante ») où $(y+1) \leq n$,
– l'adresse $w.x.y.1$ (appelée l'adresse « descendante »),

– les adresses $w.x.y.z$ (appelé une adresse « feuille ») où $z > n$.

Le premier noeud du recouvrement prend l'adresse 1. Les noeuds joignent le recouvrement successivement en se reliant à ceux déjà connectés. Quand un nouveau noeud veut faire partie du recouvrement, il demande des propositions d'adresse à tous ses voisins. Chaque voisin propose une adresse au nouveau venu (des possibilités de la liste ci-dessus) qu'il n'a pas déjà donné à ses autres noeuds voisins. Le nouveau venu choisit alors l'adresse la plus appropriée : habituellement l'adresse suivante ou descendante la plus courte qui appartient à un noeud avec un degré élevé (*i.e.*, nombre de connexions). Comme dit ci-dessus, une adresse feuille est une adresse dont l'étiquette locale est au-dessus de la valeur de la taille de zone. Les noeuds qui n'ont qu'une adresse feuille peuvent seulement envoyer des données à leur noeud père (*i.e.*, celui qui leur a donné une adresse) même si ils sont connexes à d'autres noeuds, ils sont considérés comme noeuds feuilles pour le système de routage. C'est pourquoi les adresses feuilles sont choisies par le nouveau venu avec la plus basse priorité.

### 6.3.2 Routage

Notre mécanisme de routage utilise l'information de chemin qui est stockée dans les adresses. Les noeuds ne créent pas des tables de routage qui contiennent les adresses de toutes destinations possibles. Chaque noeud doit seulement enregistrer les adresses de ses voisins afin d'acheminer correctement les paquets vers leur destination. Ainsi sa table de routage se compose uniquement des adresses de ses voisins. Le routage fonctionne de la façon suivante. Quand un paquet est routé dans un noeud, si l'adresse de destination pointe vers le bas de cette hiérarchie de noeuds, le paquet est acheminé vers le noeud de la zone actuelle qui mène encore plus loin vers la zone de destination (noeud d'entrée). Si l'adresse de destination n'est pas vers le bas de la hiérarchie actuelle des noeuds, le paquet est acheminé vers le noeud local 1 de la zone afin d'être envoyé à la zone de niveau supérieur (noeud de sortie). Quand un paquet est routé à l'intérieur d'une zone parce que la destination est dans cette zone ou pour aller au noeud d'entrée ou de sortie, il est routé par une technique que nous appelons "l'étiquette la plus proche". Si l'étiquette locale de destination est inférieure à l'étiquette locale du noeud actuel, alors le paquet est transféré au noeud voisin qui a l'étiquette locale la plus basse de celles qui sont supérieures ou égales à l'étiquette locale de destination. Si l'étiquette locale de destination est supérieure à l'étiquette locale du noeud actuel, alors le paquet est transféré au noeud voisin qui a l'étiquette locale la plus haute de celles qui sont inférieures ou égales à l'étiquette locale de destination. Comme les voisins ont une affectation d'étiquettes qui est continue, cette technique assure que le paquet atteindra sa destination bien que pas nécessairement par un plus court chemin.

Si un noeud se déplace ou tombe en panne (de ce fait rendant son adresse invalide), tous les paquets routés à une destination qui contient l'adresse du noeud mobile ou tombé ne pourront pas être routé non plus. Pour résoudre ce problème, chaque noeud récupère (et est identifié ainsi par) plusieurs adresses. Les adresses supplémentaires peuvent être choisies au moment de l'adhésion au recouvrement ou plus tard quand la connectivité du recouvrement change et qu'en conséquence, plus d'adresses deviennent disponibles pour le noeud. Toutes les adresses possédées par un noeud donné doivent provenir de différents voisins. Toutes les adresses possédées par un noeud donné doivent être différentes, cela signifie qu'elles ne doivent pas avoir un préfixe commun. Autrement si l'adresse du noeud disparu est incluse dans le préfixe commun, toutes les adresses ne fonctionneront pas. Cette solution à plusieurs adresses augmente la quantité d'information de routage à stocker par un facteur égal au nombre d'adresses par noeud mais l'avantage est que la dynamicité du réseau est gérée de manière transparente par le protocole d'adressage. Si un paquet est bloqué parce qu'un noeud a disparu, l'émetteur peut utiliser une des adresses alternatives pour réussir à atteindre la destination.

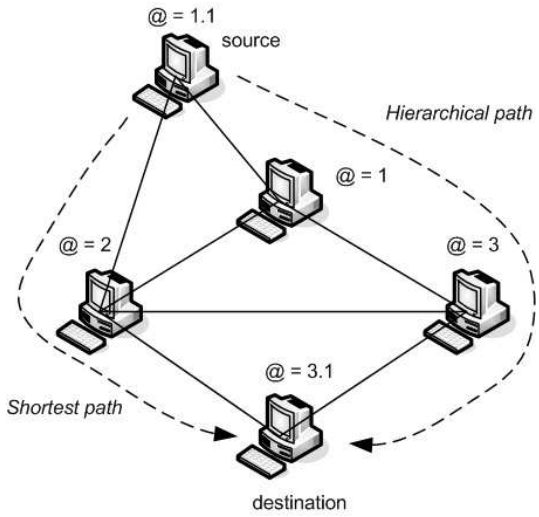


FIG. 6.2 – Inflation d'un chemin causée par le routage hiérarchique.

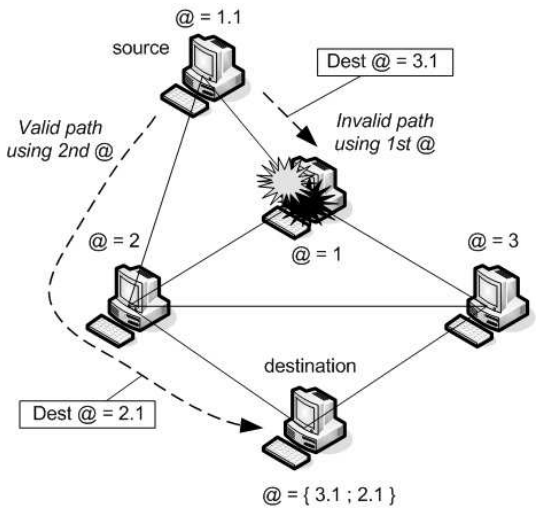


FIG. 6.3 – Routage alternatif pour résoudre les évolutions de la topologie du réseau.

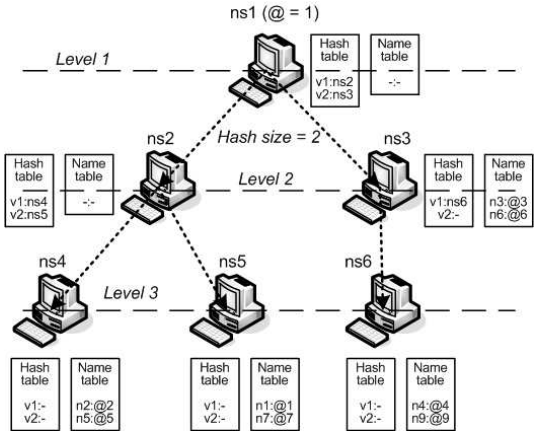


FIG. 6.4 – Organisation hiérarchique des serveurs de noms.

### 6.3.3 Nommage

Nous avons vu que chaque noeud du recouvrement possède une ou plusieurs adresses. Ces adresses dépendent de la localisation du noeud dans le recouvrement et elles sont sujettes à la dynamicité du réseau (*i.e.*, addition, suppression ou mouvement des noeuds). Afin de pouvoir communiquer de manière transparente, chaque noeud du recouvrement possède un nom unique qu'il garde durant toute son existence.

Les applications qui utilisent notre architecture, utilisent les noms des noeuds pour établir des connections entre les noeuds. Par conséquent les changements d'adresse dans les noeuds sont transparents aux applications. La correspondance entre les noms et les adresses sont faites via des serveurs de noms. Les serveurs de noms sont des noeuds du recouvrement qui acceptent d'effectuer cette tâche car ils ont des capacités de traitement supérieures et ils sont moins mobiles (le noeud d'adresse 1 est toujours un serveur de noms).

Pour être extensible, les serveurs de noms sont organisés en une strucure d'arbre. La profondeur de l'arbre est égale au niveau de la hiérarchie moins un. Chaque serveur de nom possède une table de hachage qui contient les addresses des serveurs de noms du niveau inférieur et une table de noms stockant les correspondances nom-adresse comme cela est illustré sur la figure 6.4. Un nom se compose de plusieurs parties. Chaque partie correspond à un niveau de la hierarchy des serveurs de nom. Ce n'est pas un problème si le nombre de niveaux est différent du nombre de parties.

Dans la suite, nous supposons qu'il n'y a aucune dynamique du réseau (*i.e.*, pannes). Quand un nouveau noeud a joint le recouvrement, il a choisi une ou plusieurs adresses. Il choisit alors un nom et envoie un message contenant son nom et ses adresses au noeud 1 pour stocker cette information dans un serveur de noms. À la réception du message, le noeud 1 exécute un hachage sur la première partie du nouveau nom du noeud et envoie le message au serveur de noms correspondant du deuxième niveau. À réception, le serveur de noms du deuxième niveau hache la deuxième partie du nom et envoie le message comme précédemment, vers le bas de la hiérarchie des serveurs de noms. Si le nom n'a plus de partie à hacher ou s'il n'y a aucune entrée pour le résultat du hachage ou si la table de hachage est vide dans un serveur de noms, alors ce dernier serveur doit stocker le nom et les adresses dans sa table de noms si ce nom n'existe pas déjà (autrement un autre nom doit être choisi par le nouveau noeud).

Quand un noeud veut obtenir les adresses d'un noeud destinaire, connaissant son nom (le nom est censé être connu par un mécanisme externe), il envoie un message de requête au noeud 1 contenant le nom à résoudre et sa propre adresse). La demande est expédiée au serveur de nom propre par le hachage du nom exactement comme durant une opération de stockage. Le serveur

de noms stockant le nom renverra un message contenant les adresses du nom de destination au noeud demandant en employant l'adresse d'expéditeur stockée dans la demande.

Afin de fournir un équilibrage de charge et de faire face à la dynamique du réseau, nous employons une approche de réplication. Nous supposons qu'un facteur $k$ de redondance est choisi au début de la construction du recouvrement et que le nombre total de serveurs égale $s$. Les $k$ noeuds de premier niveau ayant les adresses 1 à $k$ seront des serveurs de noms de niveau 1. Ils auront une copie de la table de hachage du noeud ayant l'adresse 1 et ils exécuteront les mêmes fonctions agissant de ce fait en tant que serveurs redondants. Les clients (*i.e.*, noeuds demandant des adresses) peuvent donc envoyer des messages de stockage et de requêtes aux noeuds 1 à $k$. En outre chaque entrée dans la table de hachage dans n'importe quel serveur dans la hiérarchie stockera $k$ serveurs de noms au lieu d'un.

Le résultat d'un hachage donnera jusqu'a $k$ serveurs possibles si tous sont opérationnels et l'un d'entre eux sera sélectionné aléatoirement pour recevoir le message. Quant au premier niveau, les serveurs de noms correspondant à une entrée de la table de hachage devront maintenir les mêmes tables de hachage et tables de noms car ils agissent en tant que serveurs redondants. Ainsi il y a un compromis entre fournir un équilibrage de charge et une tolérance aux pannes et contrôler la réplication pour les $s$ cliques de $k$ serveurs de noms identiques. Nous envisageons également d'utiliser un système de cache de noms dans tous les noeuds du recouvrement afin d'augmenter les performances.

Il y a au moins deux différences principales entre notre stratégie de distribution du stockage des noms et le célèbre DNS qui relie des noms de domaine à des adresses IP. D'abord, les noms de domaine sont toujours principalement des aliases des adresses IP et si un équipement IP entre dans un réseau IP différent, il obtiendra un préfixe IP différent et il ne pourra habituellement pas ainsi garder son Fully Qualified Domain Name (quelques services DNS dynamiques suivent les changements d'IP, IP mobile peut aider aussi). En second lieu, notre solution ne se sert pas des appels itératifs comme dans le DNS. Les messages des clients sont envoyés aux serveurs de noms de niveau supérieur de façon récursive jusqu'aux serveurs en bas de la hiérarchie et finalement le serveur stockant les adresses désirées répond directement au client. Enfin notre architecture doit être autonome pour chaque recouvrement afin d'être déployé sans contraintes par des hôtes. Nous ne pouvons pas employer ou demander des modifications au service DNS actuel.

## 6.4 Expérimentations

Dans cette section nous présentons les résultats obtenus par simulation pour évaluer l'efficacité de notre architecture. Nous montrons que notre architecture d'adressage et de routage hiérarchiques a des performances acceptables concernant les longueurs de chemins et une bonne résistance à la dynamicité du réseau.

### 6.4.1 Paramètres

Assurer l'exactitude des topologies utilisées pour nos simulations sur l'adressage et le routage est crucial car les résultats dépendent fortement de celles-ci. La topologie doit être semblable à une vraie carte d'Internet en ce qui concerne ses propriétés topologiques, autrement les résultats de simulation pourraient être biaisés par la forme de la topologie sous-jacente comme nous l'avons montré dans [MP01c]. Ainsi, pour évaluer nos protocoles d'adressage et de routage, nous avons utilisé 3 cartes d'Internet recueillies par notre logiciel *nec* en 2003 et librement disponible à [MH]. Ces cartes sont plus précises en ce qui concerne leurs liens que les cartes produites par les efforts précédents comme nous le montrons dans [HM03] et cette différence a

TAB. 6.1 – Métriques mesurées.

| Métrique |
|---|
| Longueur du chemin du routage optimal (en sauts) |
| Longueur du chemin du routage hiérarchique (en sauts) |
| Taux d'inflation (longueur du chemin hiérarchique divisée par celle du plus court chemin) |
| % de routes valides |

un impact sur certaines de leurs propriétés topologiques. Leurs tailles s'étendent de 4k noeuds à 75k noeuds. Nous supposons en première approximation que les recouvrements peuvent être exactement modelisés par ces cartes (car ces cartes sont les sous-graphes de la topologie d'Internet) ou par des sous-graphes de ces cartes quand nous étudions la dynamicité du réseau.

Concernant l'inflation des chemins, nous avons analysé divers plans d'adressage en utilisant des tailles de zone s'étendant de 32 à 32768. Concernant la dynamicité du réseau, nous avons analysé des variations du pourcentage périodique de suppression aléatoire de noeuds s'étendant de 0 à 75% de la taille du recouvrement et en attribuant 1 à 8 (au plus) adresses à chaque noeud.

Comme le processus de générer des plans d'adressage et de choisir des noeuds source et destination pour le routage implique la sélection aléatoire (et par conséquent des tirages aléatoires), nous avons employé un scénario séquentiel de simulation [LK00] pour produire les résultats montrés dans la prochaine section. Nous avons utilisé le code de Mersenne Twister [MN98] pour produire les nombres aléatoires nécessaires à notre simulation. Comme les tirages aléatoires sont la seule source d'aspect aléatoire dans notre simulation, nous pouvons raisonnablement supposer que les données de sortie de simulation obéissent au théorème de la limite centrale. Nous avons exécuté une simulation dite "terminale" où chaque instance consiste à sélectionner deux noeuds, à déterminer le chemin hiérarchique et le plus court chemin entre eux et à diviser leur longueurs respectives. Nous avons aussi évalué le succès du routage hierarchique (lorsque le recouvrement n'est pas connexe). Ainsi dans nos simulations, nous avons mesuré les métriques données dans la table 6.1.

La dynamicité du réseau est une méthode macroscopique pour simuler l'ajout, la suppression, le mouvement et la panne des noeuds du recouvrement. Au début de la simulation tous les noeuds appartiennent au recouvrement. Avant que la simulation commence, un $x$% donné de noeuds sont aléatoirement choisis et supprimés du recouvrement. Après toutes les 10 instances éxecutées, tous les noeuds supprimés sont réinsérés dans le recouvrement et à nouveau un même % de noeuds sont aléatoirement choisis et supprimés du recouvrement. Bien que $x$ reste le même, les noeuds réels qui sont supprimés toutes les 10 instances seront différents la plupart du temps particulièrement lorsque $x$ est bas. Ceci simule l'ajout, la suppression, le mouvement et la panne des noeuds de recouvrement tout en gardant la taille du recouvrement égal à $100 - x$%. Quand nous avons simulé la dynamicité du réseau nous avons déterminé à chaque instance si le routage hierarchique est réussi ou pas.

Afin de réduire les besoins en puissance de calcul, le choix aléatoire du premier noeud et la création du plan d'adressage correspondant est fait toutes les 100 instances, la dynamicité du réseau décrite ci-dessus est effectuée toutes les 10 instances et les points de contrôle séquentiels vérifiant la convergence des varialbes de sortie sont effectués toutes les 5 instances. Tous les résultats de simulation ont été obtenus en supposant un niveau de confiance égal à 0.95 avec un seuil d'erreur statistique relative égal à 5% pour toutes les métriques mesurées. Les simulations ont été effectuées dans notre logiciel *nem* [Mag].

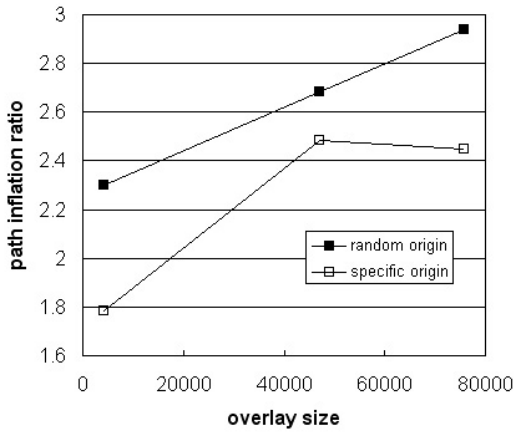


FIG. 6.5 – Inflation des longueurs des chemins *vs* taille du recouvrement.

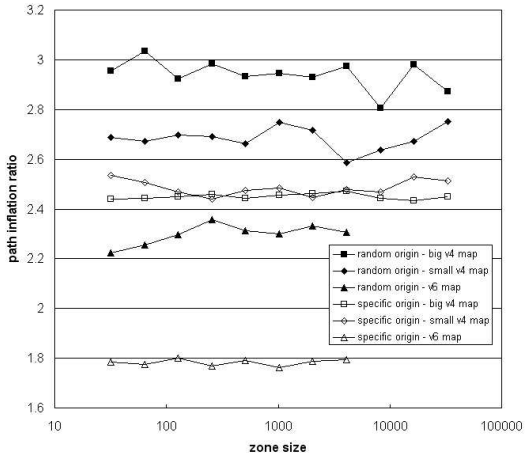


FIG. 6.6 – Inflation des longueurs des chemins *vs* taille de la zone.

### 6.4.2 Résultats

La figure 6.5 montre les valeurs du taux entre la longueur du chemin fournie par le routage hierarchique et la longueur du plus court chemin (routage plat) en fonction de la taille du recouvrement. Nous appelons ce taux le taux d'inflation des chemins. Un taux de 2 par exemple signifie que en moyenne un chemin hiérarchique est deux fois plus long que son plus court chemin correspondant (*i.e.*, entre les mêmes noeuds). Le tracé étiqueté "origine aléatoire" correspond au cas où le premier noeud du recouvrement est aléatoirement choisi. Le tracé étiqueté "origine spécifique" correspond au cas où le premier noeud du recouvrement est le noeud de degré le plus élevé de la carte. Avec un premier noeud aléatoire, le taux d'inflation est linéaire en ce qui concerne la taille du recouvrement avec des valeurs s'étendant rudement de 2.3 à 2.9 pour un recouvrement de 75k noeuds. Ceci indique une extensibilité faible mais le coefficient de pente est très et bas l'inflation demeure ainsi acceptable pour des recouvrements inférieurs à 100k noeuds. Cependant quand le premier noeud est le plus grand, le tracé fléchit : c'est une propriété intéressante. Ceci nous donne une borne inférieure sur le taux d'inflation si nous pouvons réattribuer l'adresse du premier noeud au plus grand noeud du recouvrement à un instant donné. La figure  reffig :plrzs montre les valeurs du taux d'inflation en fonction de la taille maximale de zone. L'inflation ne dépend pas de la taille de zone peu importe la carte et l'origine. Ceci signifie que nous pouvons adresser de grands recouvrements avec de grandes zones sans devoir créer trop de niveaux.

La figure 6.7 montre le pourcentage de tentatives de routage hierarchique réussies en fonction du pourcentage de dynamicité du réseau. Comme expliqué ci-dessus, un pourcentage donné de noeuds sont absents, ainsi le recouvrement peut ne pas être connexe mais composé de composantes connexes multiples. Le pourcentage est calculé comme le nombre de tentatives réussies de routage hierarchique divisé par le nombre de tentatives de routage par plus courts chemins réussies. Comme le chemin hiérarchique est plus long que le chemin le plus court, il peut sortir de la composante connexe du couple source-destination contrairement au routage le plus court et il fera ainsi échouer le routage hierarchique. Nous pouvons voir qu'avec seulement une adresse (*i.e.*, aucune route alternative), 15% de dynamicité fait chuter le taux de succès à 20%. Cependant l'ajout des adresses aux noeuds augmente fortement le taux de succès du routage hierarchique. Avec jusqu'à 4 adresses par noeud et 15% de dynamicité, le taux de succès atteint 50%. L'augmentation du nombre maximum d'adresses par noeud n'améliore pas linéairement le taux de succès parce que le nombre maximum d'adresses par noeud est aussi lié à la taille

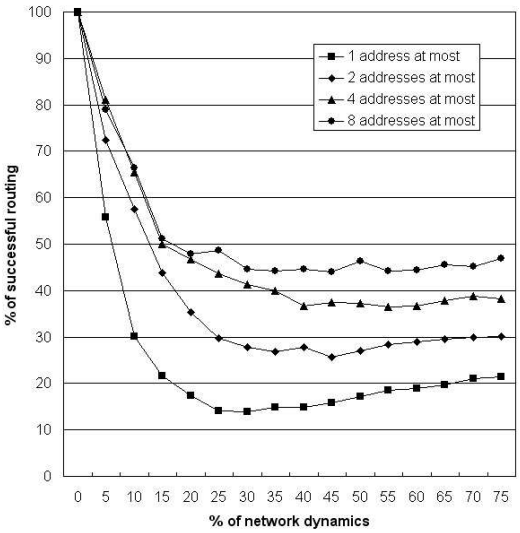


FIG. 6.7 – Pourcentage de succès *vs* dynamicité du réseau pour la carte IPv6.

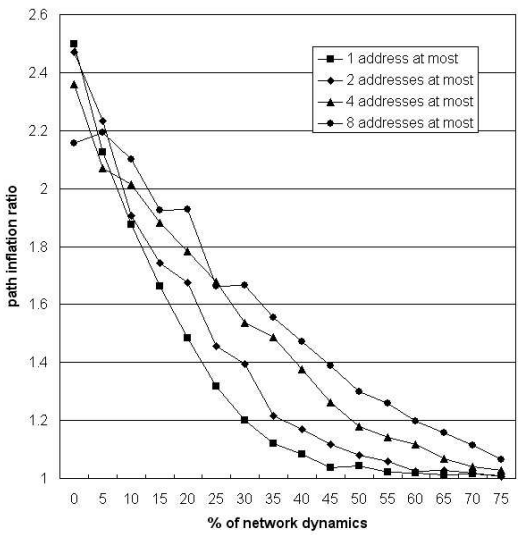


FIG. 6.8 – Inflation des longueurs des chemins *vs* dynamicité du réseau pour la carte IPv6.

de son voisinage (et celui-ci est petit pour la plupart des noeuds à cause de la topologie sous-jacente de l'Internet). La figure 6.8 montre le taux d'inflation en fonction du pourcentage de dynamicité du réseau. Le taux d'inflation diminue avec la dynamicité parce que les longueurs des chemins hiérarchiques et plus courts (*i.e.*, plats) diminuent. Alors que le réseau devient plus fragmenté (dynamicité élevée), les composantes connexes deviennent plus petites et par conséquent il en va de même pour leurs chemins internes. Ceci explique pourquoi les tracés du taux de succès avec des noeuds ayant 1 ou 2 adresses tout au plus, atteignent un minimum et augmentent à nouveau lorsque la dynamicité est élevée. Quand les composantes sont petites, les chemins hiérarchiques et plats sont généralement plus proches et le chemin hiérarchique sera moins probablement coupé (*i.e.*, il restera à l'intérieur de la composante). Il vaut la peine de rappeler que tout protocole de routage qui ne fournit pas des plus courts chemin (*e.g.*, les protocoles intra-domaines plus BGP) est sujet à des échècs de routage si le chemin calculé quitte une composante connexe.

Pour conclure cette section nous avons vu que notre architecture fournit un système de routage sans états (*i.e.*, seules les adresses des voisins sont enregistrées) avec une inflation raisonnable de 2 à 3 pour des recouvrements de tailles jusqu'à 75k (avec une borne inférieure de 2.5 pour des recouvrements de grandeur 10k). Elle fournit également une certaine robustesse à la dynamicité du réseau avec des performances raisonnables : de 80% de taux de succès lorsque 5% du recouvrement change jusqu'a 45% de taux de succès quand 50% du recouvrement change. Nous travaillons actuellement à améliorer l'attribution d'adresses : ceci abaissera les longueurs des chemins hiérarchiques, réduisant de ce fait le taux d'inflation et augmentant la robustesse à la dynamicité du réseau.

Afin d'évaluer si un taux moyen d'inflation de 2 à 3 est acceptable dans un réseau de recouvrement, nous le comparons à la distribution de l'inflation des chemins dans l'Internet IPv4 actuel que nous avons étudié dans le chapitre premier. Ces données d'inflation de chemins montrent que seulement 2.72% des chemins ont un taux de 1 (*i.e.*, le cas où le chemin *traceroute* a la même longueur que son plus court chemin correspondant). En outre, 43.4% des chemins ont un taux d'inflation supérieur à 2 ce qui est beaucoup. Ceci correspond à une inflation des chemins de 100% par rapport au plus court chemin correspondant. Ces résultats confirment que l'inflation des chemins n'est pas un phénomène marginal mais qu'elle a un impact énorme sur la plupart des chemins. Cela est principalement dû aux politiques de routage dans l'Internet et

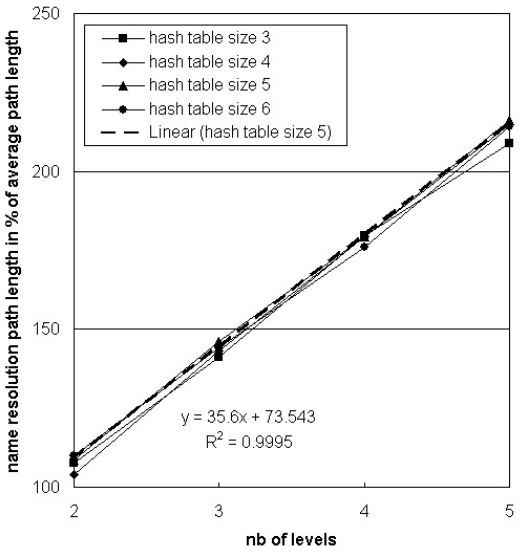


FIG. 6.9 – Longueur du chemin de résolution de nom *vs* nb de niveaux.

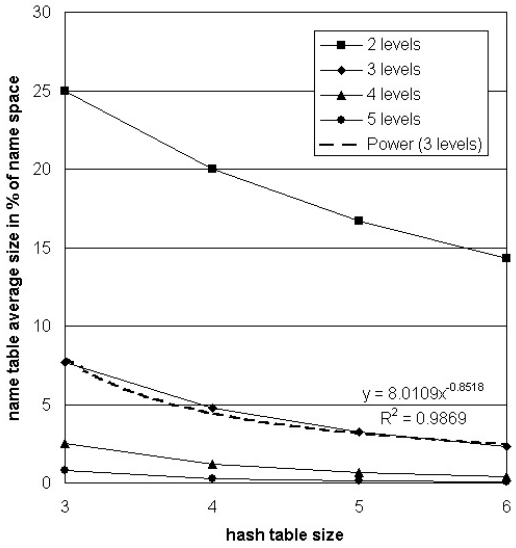


FIG. 6.10 – Taille de la table de noms *vs* taille de la table de hachage.

particulièrement dans l'inter-domaine. Nous pouvons conclure que les protocoles de routage IPv4 actuels ont un impact sur l'inflation de la longueur des chemins de plus de 100% pour presque la moitié des chemins. Notre solution, qui a une inflation moyenne de longueur de chemin s'étendant de 180% à 300%, est donc acceptable pour un protocole de routage situé dans la couche application comparée au taux d'inflation existant dans la couche réseau.

Nous évaluons maintenant les performances de notre architecture de nommage. La figure 6.9 illustre la longueur de chemin ou la distance moyenne $d$ (exprimée comme un % de l'aller-retour moyen en nombre de sauts) d'une résolution de nom (comprenant la réponse) par rapport au nombre de niveaux dans la hiérarchie $l$ et à la taille de la table de hachage $h$. Bien qu'il soit prévisible que la distance augmente à mesure que les niveaux augmentent, les tracés présentent étonnamment un comportement linéaire (comme représenté sur la figure pour une taille de table de hachage de 5). Cependant, les valeurs demeurent toujours en-dessous de 2.5 fois l'aller-retour moyen entre n'importe quelle paire de noeuds. Ces résultats prouvent que la résolution de nom implique un coût en distance (et donc en latence) raisonnable. En effet, 5 niveaux de hiérarchie peuvent gérer une quantité de noms très grande. Nous pouvons également voir sur les tracés que la distance ne change pas avec la taille de la table de hachage car tous les tracés sont très proches. Ceci est prévisible car la taille de la table de hachage aura seulement un effet sur la répartition de la charge des noms à chaque niveau. Ainsi nous pouvons écrire que $d \propto l$ (1) quand $l$ est petit. La figure 6.10 montre la taille de la table de noms $n$ (exprimée comme un % de l'espace de noms) par rapport à la taille $h$ de la table de hachage et le nombre de niveaux dans la hiérarchie $l$. Nous pouvons voir que $n$ est une fonction de $h$ à la puissance d'une constante pour $l$ fixé (comme représenté sur la figure pour une hiérarchie à 3 niveaux) et qu'elle diminue quand $h$ augmente. Ceci est prévisibles par les équations théoriques. Si nous appelons $s$ le nombre de serveurs de noms dans notre système, nous avons $s = \frac{h^l - 1}{h-1}$ (2). De plus si les noms des membres du recouvrement $m$ sont bien distribués dans les $s$ serveurs, alors les noms sont stockés dans les feuilles d'un arbre équilibré $h$-aire et nous pouvons approximer $n \approx \frac{m}{h^{l-1}}$ (3). L'explication est qu'augmenter la taille de la table de hachage augmente le nombre de serveurs et donc réduit la charge de chaque serveur. Les tracés nous montrent que le nombre de niveaux a un impact important sur la taille de la table de noms pour la même raison.

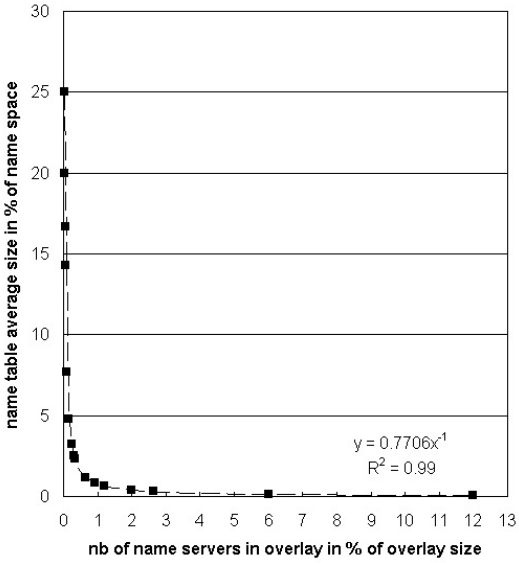


FIG. 6.11 – Taille de la table de noms *vs* nb de serveurs de noms.

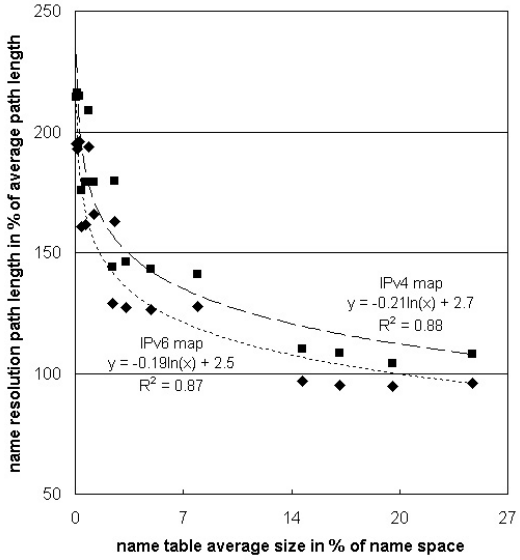


FIG. 6.12 – Longueur du chemin de résolution de nom *vs* taille de la table de noms.

Figure 6.11 illustre la taille de la table de noms $n$ (exprimée comme un % du nombre de noms) par rapport au nombre de serveurs de noms (exprimés comme un % du nombre de membres du recouvrement). Nous pouvons voir que $n$ est inversement proportionnel à $m$. Ceci est prévisible par les équations théoriques. Les équations (2) et (3) donnent $s \propto \frac{\frac{mh}{n}-1}{h-1}$ qui donne $s \propto \frac{mh}{n(h-1)}$ si $\frac{mh}{n} \gg 1$. qui pour $m$ et $h$ fixés donne $n \propto s^{-1}$. Le tracé prouve que nous pouvons réaliser un bon compromis entre le % de serveurs de noms requis et la taille des tables de noms à stocker dans chacune d'eux en choisissant des valeurs situées dans la zone inférieure gauche du tracé. La figure 6.12 illustre la distance moyenne $d$ d'une résolution de nom par rapport à la taille $n$ de la table de noms. Les équations (1) et (3) nous donnent $d \propto \frac{\log \frac{n}{m}}{\log h}$. Cependant nous pouvons voir que les tracés ne correspondent pas exactement aux données (coefficient de corrélation de 0.88 seulement). Nos multiples approximations sont peut être la cause de ces imprécisions. Toujours est-il que ces tracés nous montrent que nous pouvons obtenir un bon compromis entre le coût des résolutions de noms et le coût du stockage des noms dans chaque serveur en choisissant des valeurs dans la zone inférieure gauche du tracé.

## 6.5 Conclusion

Dans ce chapitre, nous avons proposé une architecture distribuée d'adressage, de routage et de nommage conçue pour les réseaux de recouvrement à topologies aléatoires et déployés sur l'Internet. Nos résultats de simulation obtenus sur trois cartes réalistes de l'Internet de 4k noeuds à 75k noeuds ont montré que notre solution entraîne un taux d'inflation des chemins variant de 78% à 193% selon la taille du recouvrement et l'emplacement du premier noeud. Nous avons également montré que ces résultats sont comparables à l'inflation des longueurs de chemins mesurée dans le réseau IPv4 actuel rendant de ce fait notre architecture acceptable pour des recouvrements. Nous avons décrit comment faire face à la dynamicité du réseau et nos résultats de simulation ont montré qu'une certaine robustesse est réalisable : en attribuant simplement plusieurs adresses à chaque noeud (sans n'importe quel autre mécanisme de gestion d'erreur) on multiplie par 2 le pourcentage de succès du routage hiérarchique lorsque la dynamicité du réseau est supérieure à 10%.

Nous implémentons actuellement nos mécanismes d'adressage, de routage et de nommage dans un intergiciel réseau pour hôtes. Cet intergiciel écrit en C et utilisant les *sockets* tourne sur LINUX. Il nous permettra d'évaluer l'efficacité de notre architecture en situation réelle et confirmera ou infirmera nos résultats de simulation. Nos travaux futurs vont consister à améliorer notre méthode d'allocation d'adresses, à évaluer notre méthode de réplication des serveurs de noms et à évaluer les performances des connections de niveau transport entre voisins du recouvrement.

Le projet DHARMA est particulier car c'est un projet totalement original (*i.e.*, par rapport à la cartographie) dans lequel tout est possible (car il est fondé sur les réseaux recouvrants). Il n'en est actuellement qu'à ses débuts mais il va prendre de l'ampleur. Les premiers pas de ce projet ont pu être réalisés là encore grâce aux contributions de nombreuses personnes. L'évaluation des algorithmes d'adressage et de routage dans le simulateur *ns-2* a été effectuée par Anthony James dans le cadre de son stage de DEA que j'ai encadré. Le premier prototype de DHARMA écrit en langage C et implémenté sous LINUX a été réalisé par Pierre-Emmanuel Corvi et Marc Jaeger dans le cadre de leur projet de fin d'année de DESS. La création d'un module de niveau noyau sous Windows afin de pouvoir efficacement implémenter DHARMA sous Windows a été menée à bien par Michel Cieplinski, Nicolas Varlet et Nicolas Wauters lors de leurs travaux d'études et de recherche (TER) de maîtrise. Enfin de nombreuses idées, remarques et relectures ont été apportées par Pascal Lorenz.

# Travaux relatifs au chapitre


1. Khaldoon Shami, Damien Magoni, Pascal Lorenz. A scalable middleware for creating and managing autonomous overlays. In *Proceedings of the 2nd IEEE International Conference on Communication System Software and Middleware*, Bangalore, India, January 2007.
2. Khaldoon Shami, Damien Magoni, Pascal Lorenz. Autonomous, scalable, and resilient overlay infrastructure. *KICS/IEEE Journal of Communications and Networks*, 8(4) : 378–390, December 2006.
3. Khaldoon Shami, Damien Magoni, Piotr Lipinski and Pascal Lorenz. Scalable distributed $k$-resilient name to address binding system for overlays. In *Proceedings of the 5th International Conference on Networking*, pages 46–54, Mauritius, April 2006.
4. Damien Magoni and Pascal Lorenz. Application layer addressing, routing and naming framework for overlays. In *Proceedings of the 48th IEEE Global Telecommunications Conference*, pages 932–937, Saint Louis, Missouri, USA, November 2005.
5. Damien Magoni and Pascal Lorenz. Distributed addressing and routing architecture for Internet overlays. In *Proceedings of the 4th International Conference on Networking*, pages 646–653, Saint Gilles, Reunion Island, April 2005.
6. Damien Magoni. Novel addressing and routing architecture for the Internet. Technical Report ULP-LSIIT-RR-2003-04, Université Louis Pasteur, October 2003.
7. Damien Magoni. Hierarchical addressing and routing mechanisms for distributed applications over heterogeneous networks. In *Proceedings of the 3rd International Conference on Computational Science*, pages 1093–1102, Melbourne, Australia, June 2003.
8. Damien Magoni. A scalable and unifying architecture for deploying advanced protocols in the Internet. In *Proceedings of the 10th International Conference on Telecommunications*, pages 1001–1007, Papeete, Tahiti, French Polynesia, February 2003.
9. *Dynamic Hierarchical Addressing, Routing and naMing Architecture (dharma)*. http ://dpt-info.u-strasbg.fr/~magoni/dharma/

# Programme de recherche

Mon programme de recherche s'articule autour de deux axes. Le premier axe concerne les projets locaux. Les quatre projets de recherche présentés ci-dessous ont été lancés et sont gérés par moi-même. Ils ne sont malheureusement pas directement financés par des contrats mais leurs résultats font l'objet de nombreuses publications internationales et de nombreux étudiants et doctorants y travaillent. Je compte poursuivre trois de ces projets après mon habilitation et les mener à bien dans un futur proche. Le deuxième axe concerne les projets internationaux. Les six projets de recherche présentés ci-dessous sont financés par des contrats. J'ai déjà participé à cinq de ces projets soit directement en tant que chercheur invité, soit indirectement au travers du co-encadrement d'un doctorant. Je vais participer au sixième projet durant l'année 2008.

## Pilotage de projets de recherche locaux

Les projets sont présentés dans l'ordre du plus ancien au plus récent. Le premier projet, *nem*, est arrivé à maturité et n'est plus approfondi. Le logiciel qui en est issu est par contre toujours utilisé. Je compte mener à bien dans un futur proche, les trois derniers projets de recherche qui se placent dans la continuité de mes travaux actuels. Je vais donc poursuivre mes recherches suivant trois thèmes majeurs qui présenteront une quantité importante de problématiques à résoudre. Ces thèmes sont précisément :

1. L'étude des caractéristiques topologiques et dynamiques de l'Internet.
2. L'étude d'une architecture hiérarchique et dynamique de réseaux recouvrants.
3. L'étude de l'optimisation de la connectivité dans les réseaux ad hoc sans fil.

### network manipulator

Ce premier projet débuté durant ma thèse de doctorat concerne l'analyse et la génération de réseaux. J'ai écrit le logiciel *nem* sur plusieurs années et il est désormais bien abouti. Il permet d'analyser la topologie des réseaux sous forme de graphes (diamètre, rayon, connexité, etc) et de générer des topologies réseaux réalistes à utiliser dans des simulateurs tels que *ns-2* ou Opnet. Il peut aussi analyser des cartes de l'Internet de niveau routeur ou AS et il est capable de créer des cartes de recouvrement entre routeurs et AS. Je n'implémente plus de nouvelles fonctionnalités dans ce logiciel et ce projet est donc clos.

### network cartographer

Le deuxième projet concerne la topologie de l'Internet que j'ai présenté dans la première partie de ce mémoire. Ce sujet est si vaste et si complexe que son étude est loin d'être aboutie. De plus les résultats provenant de la cartographie de l'Internet ont un intérêt capital pour le développement de nouveaux protocoles et la compréhension du comportement macroscopique de ce réseau. Je vais donc continuer à développer le logiciel *nec* et poursuivre ma collaboration avec Matthieu Latapy et Jean-Loup Guillaume du LIAFA.

### DHARMA

Le troisième projet concerne notre architecture de recouvrement présentée dans le chapitre six de ce mémoire. Ce projet n'en est qu'à ses débuts et il reste encore beaucoup à faire afin de produire un prototype logiciel distribuable. De plus, de nombreux points techniques doivent être approfondis et nos travaux futurs sur ce projet vont consister principalement à améliorer notre méthode d'allocation d'adresses, à développer les différents niveaux de nommage ainsi que leur implémentation (*e.g.*, DHT), à évaluer notre méthode de réplication des serveurs de noms et à évaluer les performances des connexions de niveau transport entre voisins du recouvrement. Ce travail important constitue mon projet principal de recherche. Je serai heureusement épaulé dans cette tâche par Pascal Lorenz du GRTC et par notre doctorant commun Khaldoon Shami.

### DIORAMA

Le quatrième projet, lié à la thèse de Fatiha Tolba, concerne l'optimisation des rayons de transmission en fonction de la connectivité dans les réseaux ad hoc sans fil et la gestion des groupements dans ces mêmes réseaux. Les réseaux ad hoc sans fil forment un domaine en pleine expansion à l'heure actuelle et les problématiques qu'ils soulèvent ont des points communs avec celles des réseaux recouvrants dont les membres sont très dynamiques. Leurs propriétés communes sont, entre autres, l'auto-organisation, le routage pair-à-pair et les communications résilientes. Les applications des approches utilisées par les recouvrements P2P pourraient fournir aux pairs mobiles de réseaux ad hoc sans fil du contrôle de flux optimisé, des mécanismes d'équilibrage de charge et du routage sensible à la proximité avec qualité de service. Il y a de plus des défis intéressants à relever en utilisant des recouvrements P2P dans des réseaux à environnement hybride filaire et sans fil (*e.g.*, correspondance entre *super-peers* et noeuds fixes). Par conséquent, je pense que l'étude de ces domaines en parallèle peut se révéler très profitable.

## Participation à des projets de recherche internationaux

Suite à des contacts noués avec des chercheurs d'autres universités, j'ai déjà participé à cinq projets extérieurs à l'ULP et je vais participer à un projet international d'envergure durant l'année 2008. Les projets sont présentés dans l'ordre du plus ancien au plus récent.

### Projet Inet

J'ai travaillé avec l'équipe de Sugih Jamin de l'Université du Michigan aux USA sur le projet Inet Topology (NSF grant no. ANI 0082287 et ONR grant no. N000140110617). Le logiciel Inet est un logiciel similaire au logiciel *nem* et nous avons mis nos recherches en commun afin d'améliorer ces logiciels de génération de topologies de type Internet. J'ai séjourné une semaine à Ann Arbor en octobre 2001 et j'ai pu échanger des informations avec les membres de l'équipe ayant développé Inet.

### Projet 6net

J'ai participé au projet européen 6net à travers le stage de DEA et la thèse de Mickaël Hoerdt co-encadrés par Jean-Jacques Pansiot et moi-même de janvier 2002 à décembre 2004. Le projet 6net (IST contract no. IST 2001 32603) vise à mettre en place un réseau expérimental académique européen de technologie IPv6 et supportant en particulier le multipoint et la mobilité. Dans ce cadre, les travaux de Mickaël Hoerdt ont permis de produire un logiciel, deux Internet draft et un délivrable D3.4.2 pour le WP 3.

## Projet MétroSec

J'ai travaillé avec l'équipe de Matthieu Latapy du Laboratoire d'Informatique Algorithmique, Fondements et Applications de l'Université Paris 7 sur le projet MétroSec (ACI Sécurité et Informatique) en fournissant des cartographies détaillées de l'Internet au niveau des routeurs et des systèmes autonomes grâce au logiciel nec de 2003 à 2005. Dans le cadre du sous-projet 3 de détection de ruptures dans la topologie, les données recueillies ont permis d'explorer la possibilité de détecter des ruptures intervenant sur le réseau (pannes et congestions, essentiellement) par l'observation de la dynamique de la topologie au niveau IP (telle que vue par traceroute). Les travaux effectués nous ont permis de publier un article de recherche dans un journal international.

## Projet Ninf-G

J'ai travaillé avec l'équipe de Yoshio Tanaka et Satoshi Sekiguchi du Grid Technology Research Center du National Institute of Advanced Industrial Science and Technology au Japon sur le projet Ninf (MEXT grant to NAREGI project WP 2) durant deux mois en janvier et février 2005. Le projet Ninf-G consiste à fournir un intergiciel aux applications distribuées souhaitant utiliser des grilles d'ordinateurs (Grid computing). Ninf-G est une implémentation de référence du système Grid Remote Procédure Call (RPC) utilisant l'intergiciel Globus. Ninf-G fournit les APIs du GridRPC qui sont proposées pour standardisation par le Grid RPC Working Group du Global Grid Forum. Depuis le 26 septembre 2005, l'API GridRPC est publiée en tant que recommendation proposée (GFD-R.P 52). L'intergiciel Ninf-G en est actuellement à sa version 4.2.2.

## Projet Cross-Correlated Security and Service Quality in Heterogeneous Mobile Communication Networks

J'ai travaillé avec l'équipe du Pr. Abbas Jamalipour dans son laboratoire situé à la School of Electrical and Information Engineering à l'université de Sydney en Australie durant deux mois en juin et juillet 2007 sur un projet intitulé Cross-Correlated Security and Service Quality in Heterogeneous Mobile Communication Networks. Ce projet est soutenu par l'Australian Research Council Discovery Grant Scheme (sous contrat numéro DP0771523). Le but de ce projet est de fournir une architecture de communication permettant de coupler la sécurité et la qualité de services dans les réseaux de communication de mobiles hautement hétérogènes. Ces réseaux mixent les technologies cellulaires (e.g. GSM, CDMA2000, UMTS, UMB, etc) et les technologies sans fil (e.g. WiFi, WiMax, WiBro, etc).

## Projet Secure Coordination and Communication in a Crisis Using Hand-held Devices

J'ai été invité par le Pr. Sugih Jamin à venir effectuer une période de recherches aux Etats-Unis dans son laboratoire situé au Department of Electrical Engineering and Computer Science à l'université du Michigan (Ann Arbor) pour collaborer à un projet intitulé Secure Coordination and Communication in a Crisis Using Hand-held Devices. Ce projet est soutenu par le US Department of Homeland Security (sous contrat numéro W911NF-05-1-0415) et financé à hauteur d'un montant de 1,35 millions de dollars sur 3 ans.

Le but de ce projet est de fournir une architecture de communication sécurisée basée sur des équipements portables de type PDA ou smartphone et faisant appel à des réseaux sans fil. Cette architecture va permettre aux équipes d'intervention des différents organismes impliqués

(police, pompiers, etc) de communiquer de manière flexible afin de coordonner les tâches à effectuer et ce de façon distribuée. Cette architecture se nomme WebBee. Actuellement, les technologies utilisées par les équipes d'urgence et de sécurité sont basées sur l'utilisation d'ordinateurs portables et de modems spécialisés. Leur coût élevé limite leur déploiement et fait que la majorité des communications s'effectue encore par radio avec des rapports établis sous forme papier. L'architecture proposée dans ce projet permet d'utiliser des équipements et des infrastructrures sans fil commerciales à coût réduit car leurs caractéristiques techniques sont suffisantes. La solution proposée se compose de trois parties :

1. Une infrastructure de communication et de traitement basée sur des équipements terminaux portables de petite taille.
2. Une infrastructure de sécurité pour réduire les attaques possibles contre l'infrastructure de traitement (serveurs web et base de données).
3. Un système de gestion de base de données pour stocker et analyser les données.

L'architecture WebBee possède un certain nombre de caractéristiques qui la rendent unique. La partie accès web utilise un serveur qui filtre et traite dynamiquement les pages html pour les adapter automatiquement aux terminaux (e.g. afin d'éviter les inconsistences entre les copies des pages html en wap/wml). Je vais participer au travail de recherche sur cet aspect étant donné l'expérience que j'ai acquise dans ce domaine en développant le logiciel network cartographer (nec) qui fait intensivement appel à du parsing web automatique et dynamique. La partie sécurité se focalise sur un scénario de communications asymétriques où le nombre d'utilisateurs téléchargeant des données du site web de coordination est beaucoup plus important que le nombre d'utilisateurs mettant à jour les données. Les techniques d'authentification et de chiffrement varieront donc selon le rôle et le type de communication (e.g. téléchargement ou mise à jour) afin de minimiser les impacts de la cryptographie sur les performances du système. La partie communication se compose d'un module d'infrastructure instantanée embarqué dans les vehicules pour établir des points de communications pour les terminaux mobiles si besoin est (e.g. suite à la destruction des stations de base ou des points d'accès des opérateurs publics situés près du centre de la catastrophe). Enfin, la partie fouille de données de l'architecture propose un serveur de base de données adapté à l'analyse de données spatiales et non-spatiales. Le module de surveillance des données va mettre en œuvre des méthodes extensibles de requêtes (e.g. indexation spatiale par quadtree). Cela permettra d'alerter l'utilisateur quasiment en temps réel lorsqu'un motif est détecté par le serveur. Un premier prototype non sécurisé de cette architecture est déjà disponible à l'adresse http ://webbee.eecs.umich.edu/.

# Conclusion

## Bilan des contributions

Le bilan concernant notre cartographie de l'Internet est très positif. Notre logiciel est actuellement le plus performant en terme d'exploration massive de l'Internet. Les cartes obtenues par *nec* possèdent une quantité relative de liens beaucoup plus importante que les cartes produites par d'autres logiciels. De plus notre logiciel peut aussi cartographier l'Internet IPv6. Pour les cartes IPv4, nous avons pu construire des recouvrements routeurs–systèmes autonomes d'une grande précision qui pourront être utilisés pour simuler les interactions intra et inter-domaines. Enfin, les traces obtenues nous ont permis d'étudier les délais des chemins et des liens de l'Internet IPv4 avec précision. Nous avons observé que la longueur en nombre de sauts des chemins est faiblement corrélée au délai de ces mêmes chemins. Nous avons mis en évidence que la distribution des liens forme une distribution à queue lourde et que plus de deux tiers des liens de l'Internet ont un délai mesuré inférieur à 10ms. Ces travaux sont le résultat du travail collectif de huit personnes (deux permanents, un doctorant et cinq stagiaires). Notre logiciel ainsi que des cartes sont disponibles gratuitement à [MH]. L'intérêt qualitatif et quantitatif des résultats produits par *nec* a permis la publication de deux articles internationaux de recherche ainsi que deux articles nationaux de recherche. Ces résultats sont actuellement utilisés par d'autres laboratoires et en particulier par Matthieu Latapy et Jean-Loup Guillaume du LIAFA dans leurs travaux concernant l'optimisation des explorations des grands réseaux d'interaction [LG05].

Notre contribution à l'étude de la topologie de l'Internet est plus modeste. En effet j'ai beaucoup étudié cette problématique durant mon doctorat et j'ai déjà publié, avec mon directeur de thèse Jean-Jacques Pansiot, huit articles de recherche concernant la topologie de l'Internet et les générateurs de topologies de type Internet. L'obtention de cartes réelles de l'Internet fut cependant propice à une étude réaliste et approfondie de la robustesse de l'Internet. Notre étude a montré que sous attaque adaptative, la suppression de seulement 4% des routeurs laisse des groupements de tout au plus 100 routeurs dans une carte de plus de 100k routeurs et que la suppression de 1.5% des routeurs réduit la taille de la plus grande composante connexe 50% de la taille initiale du réseau. Nous avons heureusement montré dans [Mag03b] que ces seuils dépendent de la taille du réseau ce qui signifie que détruire l'Internet lui-même nécessiterait des attaques simultanées sur des centaines de milliers de routeurs. Comme l'efficacité des attaques dépend fortement de la vue partielle qu'elles possèdent de la topologie, cela autorise une attaque focalisée sur une zone, mais rend une attaque globale presque impossible. Les résultats de notre étude sur la robustesse de l'Internet a donné lieu à la publication de deux articles internationaux de recherche. De plus, ils sont actuellement utilisés par Juan Wang, doctorant au département d'ingénierie électrique de l'Imperial College London, dans le cadre d'une étude sur la résilience des réseaux.

Malgré les très nombreuses études réalisées sur le multipoint ces dernières années, le bilan de notre contribution dans ce domaine me semble important sur deux points. Premièrement, la possibilité d'une saturation des routeurs par création d'états multipoints ne saurait être niée par les chercheurs (cf. section 3.3.2) et elle est si critique, que des attaques par déni de service

peuvent efficacement être menées en s'appuyant sur cette saturation comme nous le montrons au chapitre quatre. Deuxièmement, il est encore possible de définir et de proposer des mécanismes visant à améliorer l'efficacité et la sécurité des protocoles multipoints et de promouvoir leur déploiement dans l'Internet. Par conséquent :

- Nous avons proposé un mécanisme simple conçu pour profiter de multiples plus courts chemins dans le but d'augmenter les chances qu'un nouveau membre puisse joindre un arbre multipoint tout en distribuant les états multipoints parmi les routeurs non saturés. Nous avons montré que lorsque de tels réseaux sont très saturés (*i.e.*, plus de 50%), l'augmentation des messages *join* alternatifs fructueux va de 25% à 55% selon la densité de session. De plus notre mécanisme ne nécessite que des modifications mineures des protocoles multipoints de gestion d'arbres multipoints et il ne stocke aucune information dans les routeurs.
- Nous avons simulé l'impact d'attaques distribuées par déni de service basées sur le protocole PIM-SSM dans des topologies de type Internet. Les résultats montrent que ces attaques peuvent se révéler d'une efficacité redoutable car étonnamment elles atteignent la cible et son périmètre proche sans quasiment surcharger le réseau dans son ensemble. De plus il n'est pas nécessaire d'avoir un grand nombre d'attaquants pour réussir à saturer n'importe quel routeur de l'Internet. Cependant, il est relativement facile de créer des mécanismes de contrôle des branches inutiles. Nous sommes actuellement en train de finaliser la définition d'un tel mécanisme, à rajouter à la spécification du protocole PIM-SM.

Cinq personnes (quatre permanents et un doctorant) ont participé aux travaux concernant le multipoint. La pertinence des résultats de nos études et de nos propositions a permis la publication de deux articles internationaux de recherche ainsi que trois articles nationaux de recherche.

Le bilan concernant la proposition SSMSDP est très satisfaisant. Nous avons fourni une bibliothèque multipoint pour les applications multi-sources (ASM) en s'appuyant uniquement sur le protocole Source Specific Multicast au niveau de la couche réseau. Nous avons défini un nouveau protocole et nous l'avons implémenté dans cette bibliothèque afin de permettre à des applications ASM de découvrir des sources SSM. Nous avons donné des éléments d'évaluation de cette architecture et avons conclu que la fonctionnalité de découverte de source devrait être située dans les hôtes pour une majorité d'applications d'où l'intérêt de notre implémentation. Ces travaux sont le résultat de la collaboration de quatre personnes (deux permanents, un doctorant et un stagiaire) et cette bibliothèque fait partie des délivrables du projet européen *6net*. Elle est disponible gratuitement sur le site web de Mickaël Hoerdt.

Enfin le bilan de notre contribution aux applications distribuées basées sur des réseaux recouvrants reste difficle à évaluer en raison de sa jeunesse mais il semble prometteur. Nous avons proposé une architecture distribuée d'adressage, de routage et de nommage conçue pour les réseaux de recouvrants à topologies aléatoires et déployés sur l'Internet. Nos résultats de simulation présentés au chapitre six sont encourageants. Nous avons décrit comment faire face à la dynamicité du réseau et là encore nos résultats de simulation ont montré qu'un certain niveau de robustesse est réalisable. Nous implémentons actuellement nos mécanismes d'adressage, de routage et de nommage dans un intergiciel réseau pour hôtes. Cet intergiciel écrit en C et utilisant les *sockets* tourne sur LINUX. Il nous permettra d'évaluer l'efficacité de notre architecture en situation réelle et confirmera ou infirmera nos résultats de simulation. Le projet DHARMA a réuni les efforts de neuf personnes (deux permanents, un doctorant et six stagiaires). Notre intergiciel n'est pas encore disponible mais devrait le devenir bientôt. L'architecture de DHARMA a fait l'objet de sept publications internationales.

# Applications des contributions

Il est important, surtout en recherche appliquée, de pouvoir proposer des solutions concrètes à des problèmes donnés. En informatique, ces solutions se présentent le plus souvent sous la forme de logiciels ou de normes ratifiées par de grands organismes (*e.g.*, ISO, ITU, IETF). Quasiment tous les travaux entrepris dans ce mémoire ont eu pour but de proposer l'un ou l'autre de ces types de solutions. Le déploiement possible à grande échelle d'une solution est de plus très apprécié.

La contribution concrète de notre projet de cartographie Internet est le logiciel *nec*. C'est un logiciel open-source que quiconque peut télécharger et utiliser. De plus nous fournissons aussi des cartes générées grâce aux données de *nec* afin d'éviter aux chercheurs pressés, de longues campagnes de collecte. Les applications liées à *nec* sont très nombreuses :

- Etude de la topologie de l'Internet et recherche de ses caractéristiques intrinsèques (distribution des degrés, des arbres, des chemins multiples, distance moyenne, diamètre, rayon, etc).
- Etude des routes suivies par les paquets IP (routes parallèles multiples, variations des routes dans le temps, asymétrie aller-retour, etc).
- Etude des délais sur les routes et les liens (valeurs min-max, moyennes, asymétrie, etc).
- Etude des recouvrements routeurs–systèmes-autonomes (routeurs par AS, liens IP par connexion BGP, etc).
- Support pour la simulation réaliste de protocoles réseaux destinés à être déployés dans l'Internet aux niveaux IP et supérieurs.
- Support pour la simulation réaliste de protocoles fonctionnant au niveau intra-domaine et/ou inter-domaine.

En ce qui concerne le domaine du multipoint, nous avons proposé dans ce mémoire, plusieurs solutions pratiques à diverses situations problématiques. Les applications de ces contributions sont les suivantes :

- Notre proposition de distribution des états multipoints par utilisation des plus courts chemins multiples peut s'appliquer à tout protocole de création et de gestion d'arbre multipoint et en particulier au standard de facto PIM-SM/SSM. Elle permet d'augmenter les chances d'adhésion réussie des nouveaux membres et surtout d'améliorer la répartition des états dans le réseau ce qui favorise l'extensibilité de ces protocoles. Nous rédigeons actuellement un Internet Draft contenant les modifications à apporter à un protocole de ce type afin de supporter notre mécanisme.
- Notre proposition de contrôle et de destruction de branches inutiles peut là encore s'appliquer à tout protocole de création et de gestion d'arbre multipoint et en particulier à PIM-SSM. Elle permet d'empêcher des attaques par déni de service distribuées sur des routeurs multipoints. Nous envisageons de rédiger un Internet Draft contenant les ajustements nécessaires au contrôle de ces branches parasites.
- La contribution concrète du projet SSMSDP est l'intergiciel Libssmsdp. C'est un intergiciel open-source que quiconque peut télécharger et utiliser. De plus, deux Internet Draft ont été rédigé pour décrire en détail l'architecture de cet intergiciel [HBP03] et encourager des chercheurs externes à l'évaluer. Les applications de cette solution sont très variées puisque cet intergiciel permet d'interfacer toute ancienne application multipoint de type ASM avec un réseau SSM en effectuant de la découverte automatique de source et de la gestion de groupe.

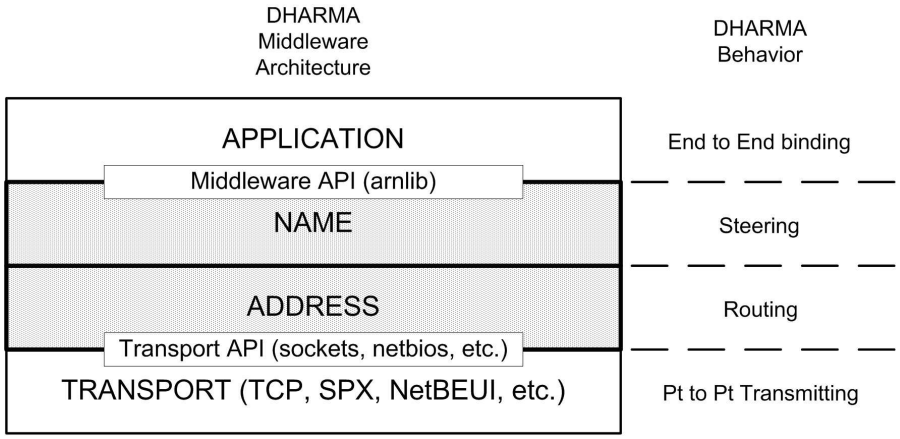


FIG. 6.13 – Structure de l'intergiciel DHARMA.

La contribution du projet DHARMA sera un intergiciel open-source permettant aux applications de lui déléguer la gestion de leurs recouvrements. La structure de l'architecture est présentée sur la figure 6.13. Les applications de notre architecture sont vastes :

- Support de la mobilité des membres : les connexions entre les membres d'un recouvrement sont effectuées en utilisant les noms et non les adresses. C'est pourquoi, lorsqu'un membre se déplace, la connexion n'est pas rompue. Ce membre obtient une nouvelle adresse par ses nouveaux voisins et avertit son ou ses correspondants ainsi que les intermédiaires (membres routant les paquets) soit directement soit en mettant à jour les serveurs de noms. Ainsi les paquets grâce au nom seront routés vers la nouvelle adresse. Les intermédiaires avertis peuvent rerouter les paquets durant le déplacement du membre. Les membres des applications utilisant notre intergiciel peuvent donc bénéficier de la mobilité sans qu'elle soit fournie par le réseau lui-même.
- Support de la sécurité de bout en bout : l'utilisation des noms pour créer des connexions sécurisées de bout en bout au lieu d'utiliser IPSec apporte plusieurs avantages. En effet dans IPsec, une Security Association (SA) est définie de manière unique par un triplet qui consiste en un Security Parameter Index (SPI), une adresse IP et un protocole de sécurité (AH or ESP) [KA98]. Si la destination est un mobile ou se trouve dans un réseau privé, la SA sera invalidée. Les membres des applications utilisant notre intergiciel peuvent donc bénéficier d'une sécurité de bout en bout même s'ils sont mobiles ou se trouvent dans des espaces d'adressage différents.
- Support du multipoint applicatif : plusieurs niveaux de nommage peuvent être fournis par notre architecture afin de permettre la correspondance entre une ressource (personne, service, groupe, etc) et un ou plusieurs membres (hôtes) d'un recouvrement. Les entrées sont stockées dans les tables de noms qui contiennent déjà les correspondances noms-adresses. Un intermédiaire résoud le nom de groupe en noms de membres du recouvrement puis en adresses et duplique le paquet si les adresses sont sur des chemins divergents ce qui permet de fournir du multipoint applicatif.
- Support de la qualité de service dans le recouvrement : les membres d'un recouvrement sont des machines hôtes. Elles possèdent par conséquent suffisamment de ressources pour fournir de la qualité de service nécessitant du stockage d'état (*i.e.*, à la RSVP). Les membres qui sont des intermédiaires de communication peuvent par conséquent fournir des débits, des priorités et des buffers différents selon les connexions.

# Encadrement doctoral

Une habilitation à diriger les recherches valide principalement la capacité à encadrer de jeunes chercheurs. Il est difficile de recruter de jeunes chercheurs mais les encadrer représente un réel challenge pour un jeune maître de conférence accaparé par ses cours, ses tâches administratives et sa quête de financements. J'ai cependant eu la chance de pouvoir encadrer plusieurs jeunes chercheurs très tôt dans mon propre parcours.

De février à juin 2002, j'ai eu l'opportunité de pouvoir co-encadrer à 33% avec Jean-Jacques Pansiot et Dominique Grad, le stage de DEA de Mickaël Hoerdt qui s'intitulait "Source Specific Multicast (SSM) : vers un routage multipoint inter-domaine efficace". J'avais moi-même fait une thèse sur un protocole multipoint et je pouvais donc aider Mickaël Hoerdt dans ce domaine. Le but de son stage a consisté, après étude de l'existant sur l'extensibilité des protocoles de routage multipoint, à étudier l'utilisation des plus courts chemins multiples dans la diffusion de données et la construction d'arbres de routage multipoint. Mickaël Hoerdt a obtenu son DEA avec succès. Il a ensuite débuté une thèse en octobre 2002, que j'ai co-encadré à 50% avec Jean-Jacques Pansiot et qui s'intitulait "Routage multicast inter-domaine et passage à l'échelle". Le financement de cette thèse s'est effectué dans le cadre du projet européen *6net*. Les travaux effectués par Mickaël Hoerdt et moi-même durant sa thèse et mes premières années de maître de conférence nous ont permis de publier dans deux journaux de recherche et dans les actes de trois conférences internationales et de quatre conférences nationales. Mickaël Hoerdt a obtenu son doctorat avec succès le 20 septembre 2005 et a effectué un post-doctorat en Norvège d'octobre 2005 à octobre 2006. Il a obtenu sa qualification par la section 27 du CNU au début 2006. Il est actuellement assistant de recherche à l'université de Lancaster au Royaume-Uni.

Parallèlement, de février à juin 2003, j'ai co-encadré à 50% avec Jean-Jacques Pansiot le stage de DEA de Dragos Manzateanu qui s'intitulait "Multipoint applicatif et réseaux Pair à Pair". Le but de son stage a consisté à étudier les critères importants pour pouvoir comparer les diverses propositions de multicast applicatif avec le routage multicast de niveau réseau, comme les délais de diffusion, la surcharge réseau, la rapidité d'adaptation aux changements du groupe ou du réseau, en fonction de la taille et du nombre des groupes de diffusion. Dragos Manzateanu a obtenu son DEA avec succès et travaille actuellement dans une entreprise privée en Roumanie, son pays d'origine. Notre travail a abouti à la rédaction d'un rapport de recherche non publié.

Ma titularisation en octobre 2003 m'a donné plus d'autonomie et m'a permis d'encadrer à 100% de février à juin 2004 le stage de DEA de Anthony James qui s'intitulait "Mécanismes d'adressage et de routage hiérarchiques sans états" et qui concernait directement mon propre thème de recherche. Le but de son stage a consisté à explorer de nouvelles méthodes d'adressage et de routage destinées à des réseaux de topologie libre tels que l'Internet, les recouvrements, les réseaux P2P et les réseaux Grid. De nouveaux mécanismes situés au niveau applicatif ont été proposés pour introduire une hiérarchie permettant le support de réseaux à très grande échelle tout en offrant un routage à faible coût. Anthony James a obtenu son DEA avec succès et travaille maintenant dans une entreprise privée aux Pays-Bas.

Suite à une collaboration mise en place avec l'équipe GRTC de l'Université de Haute Alsace, j'ai co-encadré avec Pascal Lorenz et Hervé Guyennet la thèse de doctorat de Fatiha Djemili Tolba d'octobre 2004 à octobre 2007. Cette thèse concerne la gestion de la puissance et des groupements dans les réseaux ad-hoc. De même je co-encadre, avec Pascal Lorenz, la thèse de Khaldoon Shami débutée en mars 2005 et qui concerne l'étude de l'architecture de communication de niveau applicatif pour réseaux recouvrants que j'ai définie début 2003 [Mag03a] ainsi que dans le dernier chapitre de ce mémoire. Fatiha Djemili Tolba a soutenu sa thèse le 5 septembre 2007 et Khaldoon devrait la soutenir au courant de l'année 2008. Les travaux effectués par Fatiha Djemili Tolba ont donné lieu à une publication dans un journal international et trois publications dans des actes de conférences internationales. Ceux effectués par Khaldoon

ont donné lieu à une publication dans un journal international et deux dans des actes de conférences internationales. Une grande partie des travaux décrits dans ce mémoire n'aurait pu voir le jour sans les efforts de ces jeunes chercheurs et j'espère que mon aide leur a été profitable. J'ai beaucoup appris à leurs côtés et j'espère pouvoir continuer à leur faire bénéficier de mon expérience dans la recherche, aussi limitée soit-elle.

# Perspectives de recherche

L'Internet forme désormais la structure inexorable de l'ère de l'information et de la communication de notre village planétaire. Cependant les mutations considérables de taille et de forme qu'il a subi depuis sa création, ont fait apparaître des failles importantes dans son architecture. L'Internet est là pour perdurer mais le fera-t-il avec ses protocoles actuels ? Malgré les pressions commerciales, cela n'est pas si sûr. A peine arrivé à l'heure du "tout IP", les acteurs du monde des réseaux s'aperçoivent des fortes limitations de celui-ci. Les opérateurs butent sur les protocoles de routage lourds, une faible gestion de la qualité de service, une absence de gestion de réseau hors-bande, etc. De plus ils ont beaucoup de mal à déployer des services réseaux avancés (*e.g.*, multipoint). Ils sont enclins à déployer des technologies plus adaptées aux coeurs de réseaux (*e.g.*, MPLS) ou à conserver des technologies existantes (*e.g.*, ATM) sous IP. Les fournisseurs d'accès sont freinés par les limitations de l'espace d'adressage IPv4 et les nombreux problèmes de sécurité réseau. Les équipements mobiles sont limités par la quasi absence de mobilité au niveau IP (car certains routeurs doivent supporter la mobilité). Les RFCs concernant la mobilité IPv4 (août 2002) et IPv6 (juin 2004) ne sont encore que des "proposed standard". Les réseaux privés sont forcés par manque d'adresses publiques de faire un usage massif de points NAT qui brisent le modèle TCP/IP de bout en bout. Les utilisateurs souffrent d'un manque d'identification (*e.g.*, usurpation) et de protection (*e.g.*, virus, vers, troyens, etc). Les applications sont rigidifiées par les lacunes sémantiques des protocoles (*e.g.*, l'URL d'une ressource contient de manière inhérente l'adresse IP de la machine qui stocke cette ressource). Il ne s'agit pas de jeter la pierre aux créateurs de la célèbre suite de protocoles TCP/IP. Ils n'imaginaient probablement pas le succès énorme qu'allait avoir leur invention. Le problème est plutôt comme le souligne Balakrishnan *et al.* dans [BLR+04] d'avoir gelé prématurément mais définitivement une architecture qui n'était encore qu'un "work-in-progress".

On pourrait être tenté de dire que le nouveau protocole IPv6 va résoudre tous les problèmes décrits ci-dessus. Au risque de décevoir ses promoteurs, j'aurais tendance à dire qu'IPv6 ne résoudra pas grand-chose si ce n'est la limitation de la taille de l'espace d'adressage. Avec 128 bits au lieu de 32, on pourra effectivement attribuer des adresses à toute entité imaginable mais je vous laisse évaluer la taille des tables de routage BGPv6 des routeurs de coeur. Certes l'espace d'adressage en IPv6 est hiérarchisé mais de manière figée. Cela peut aboutir à une utilisation inefficace de l'espace d'adressage. De plus un déplacement de machine ou de réseau entrainera toujours une renumérotation car la distribution automatique d'adresse n'est possible qu'au dernier niveau de la hiérarchie. Tous les problèmes liés à la fusion d'une identification et d'une localisation dans une même adresse IPv4 et qui sont autant de freins à l'implémentation efficace de la mobilité et de la sécurité, se retrouvent à l'identique dans IPv6 puisque celui-ci conserve la même sémantique. La conséquence de la définition actuelle d'IPv6 est que depuis 10 ans (les premières RFCs sur IPv6 datent de 1995), le déploiement d'IPv6 est resté embryonnaire avec seulement 4 RFCs ayant le status de "draft standard" et une utilisation majoritairement cantonnée à la communauté scientifique. Malgré ces difficultés, il est évident que le protocole IPv6 va s'imposer dans les années à venir ne serait-ce que pour ces facultés gigantesque d'adressage.

La frustration liée aux limitations des protocoles réseaux et à leur complexité de mise en oeuvre a probablement été à l'origine de l'engouement pour les techniques faisant appel à des

réseaux recouvrants (*i.e.*, recouvrements). La liberté et la rapidité que procure le développement d'une implémentation de niveau hôte la rend attractive malgré les pertes inévitables de performances. Le multipoint en est un exemple frappant : il a fallu attendre une bonne dizaine d'années pour que les chercheurs délaissent le multipoint de niveau réseau et se ruent vers la création de solutions applicatives [hCRSZ01, LN01, BBK02, BKK+03, KB04]. La mobilité au niveau des couches hautes suit un chemin similaire [SB00, VdaFR03a]. De plus l'émergence de nouvelles classes d'applications (*e.g.*, P2P, Grid) et de nouveaux réseaux (*e.g.*, ad hoc sans fil, réseaux de capteurs, réseaux maillés sans fil) a entraîné une effervescence de recherches tous azimuts dans le domaine des recouvrements. Non seulement les recherches de ces domaines variés vont s'enrichir mutuellement mais en plus les nouveaux résultats qu'elles vont produire (*e.g.*, adressage dynamique, routage local, nommage par tables de hachage distribuée, résilience, etc), vont probablement constituer les fondements d'une nouvelle architecture pour l'Internet.

Ainsi c'est depuis les couches les plus extérieures que les couches centrales réseau et transport vont évoluer et devenir enfin capable d'implémenter efficacement les nouveaux paradigmes de l'Internet que sont entre autres la mobilité, la sécurité et la diffusion. Les notions d'Internet ambiant et d'ubiquité ne pourront en effet se concrétiser sans implémenter ces nouveaux paradigmes. Le modèle en couche ne me semble absolument pas dépassé à cet égard malgré l'avènement des architectures "crosslayer". Ces architectures servent le plus souvent à implémenter efficacement des fonctionnalités qui sont orthogonales aux fonctions de communication (*e.g.*, économie d'énergie) et elles ne remettent pas en cause à mon avis l'efficacité des piles de protocoles. Au contraire, le modèle en couches est à l'origine de la force des protocoles réseaux et les failles de ces derniers sont souvent liées à une violation des principes de ce modèle. Une adresse IP de la couche réseau par exemple n'a pas sa place à l'intérieur d'un URL de la couche application. Le défi des architectures de recouvrement d'aujourd'hui (et des protocoles de l'Internet de demain) consiste à créer des niveaux d'indirection supplémentaires permettant de mieux isoler les protocoles les uns par rapport aux autres afin d'obtenir une infrastructure globale plus souple supportant un réseau plus dynamique.

Par rapport à la situation actuelle, au moins deux niveaux supplémentaires sont requis. Un niveau de nommage de point de connexion (correspondant aux noms dans l'architecture dharma vue au chapitre six) servant à l'établissement des connexions de bout en bout et un niveau de nommage d'entité (telle qu'une personne, un service ou un groupe) servant à définir la sémantique de la connexion (*i.e.*, à quoi sert-elle ?). Ainsi une application utilisera des noms d'entités, qui se réfèreront à des noms de connexion, qui eux seront finalement liés à des adresses réseaux. Par exemple, une connexion servant à effectuer une transaction financière entre un utilisateur et sa banque, sera définie par des identifiants d'entités indépendants des identifiants des hôtes terminaux utilisés (*i.e.*, points de connexion). De même, les identifiants des points de connexion seront eux-mêmes indépendants des adresses réseaux (*i.e.*, de la position géographique) de ces mêmes hôtes.

Des technologies dites disruptives émergent continuellement dans le domaine des réseaux. Pour être considérée comme telle d'après Christensen [Chr97], une technologie doit être dix fois plus performante que n'importe quelle autre alternative, dix fois moins coûteuse et proposer des fonctionnalités dix fois supérieures à l'existant. L'avènement des technologies de pair-à-pair et de grille par rapport au modèle client-serveur et celui des réseaux recouvrants par rapport à l'architecture réseau actuelle de l'Internet, illustre cette tendance disruptive au niveau de la couche application même si la définition de Christensen ne peut leur être appliquée tel quel actuellement. Dans le domaine des communications sans fil, les technologies de réseaux maillés sans fil (*wireless mesh networks* en anglais) souhaitent aussi accéder à ce statut au niveau de la couche liaison de données. Les réseaux maillés sans fil sont des systèmes pair-à-pair ad hoc multi-sauts dans lesquels tous les équipements s'entraident afin de transmettre les paquets dans le réseau surtout dans des conditions adverses. Ces réseaux ad hoc particuliers peuvent être déployés ra-

pidement et ils fournissent un système de communication flexible et fiable qui peut être étendu à des milliers d'équipements. Ces réseaux sont auto-configurables et auto-réparables. Ils se basent d'une part sur une technologie de réseau ad hoc sans fil telle que Bluetooth, 802.11 ou encore QDMA (Quad-Division Multiple Access). QDMA est une technologie radio développée et utilisée par l'armée américaine. Les caractéristiques de QDMA sont une connectivité IP de bout en bout, un accès large bande et le support de réseaux ad hoc pair-à-pair. Un réseau QDMA de test de 4000 noeuds est à l'étude par une filiale de Motorola. Les réseaux maillés sans fil se basent d'autre part sur une couche logicielle implémentant un protocole de communication qui gère le maillage des équipements et qui fournit les propriétés d'extensibilité et de fiabilité décrites précédemment.

Précisons que toutes ces technologies novatrices entrainent dans leur évolution un nombre conséquent de problèmes connexes qui sont tout autant de champs d'investigation pour la recherche réseau future. Citons, entre autres, les problèmes à résoudre dans les cas suivants :

- Le contrôle de la coopération équilibrée entre les membres d'un réseau pair-à-pair.
- Le renforcement de la notion de confiance et de réputation dans les systèmes pair-à-pair.
- L'implémentation distribuée d'annuaires génériques pour les niveaux multiples de nommage dans les réseaux recouvrants.
- Le chaînage efficace de connexions TCP dans les réseaux recouvrants.
- La mise en oeuvre de la confidentialité et de l'authentification dans les communications sans fil aussi bien pour le trafic de données que celui de contrôle.
- La gestion optimale de l'énergie dans les éléments des réseaux de capteurs sans fil.

Il y aurait encore bien d'autres problématiques à présenter mais cela dépasserait le cadre de notre propos qui consiste principalement à montrer la quantité et la diversité des défis liés à ces nouvelles technologies.

Nous pouvons conclure que l'Internet d'aujourd'hui n'est pas arrivé à maturité mais au contraire se trouve à l'aube d'une mutation majeure de ses protocoles internes. Comme le montre la situation actuelle de l'Internet, il reste beaucoup à faire pour effectuer cette transition. Nous sommes persuadés cependant que les évolutions discutées ci-dessus seront inévitables à plus ou moins long terme. Reste à savoir la forme exacte que prendront ces mutations. Quoi qu'il en soit, nous espérons, au travers de ce mémoire d'habilitation, avoir oeuvré utilement quoique modestement à la connaissance de l'Internet et à l'évolution de son architecture.

# Bibliographie


[ABKM01] David Andersen, Hari Balakrishnan, Frans Kaashoek, and Robert Morris. Resilient overlay networks. In *Proceedings of the 18th ACM SOSP*, October 2001.

[AHU83] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983.

[AJB00] Réka Albert, Hawoong Jeong, and Albert-László Barabási. Error and attack tolerance of complex networks. *Nature*, (406) :378–382, 2000.

[aMASK98] V. Hardman an M-A. Sasse and I. Kouvelas. Successful multiparty audio communication over the internet. *Communications of the ACM*, 41(5), May 1998.

[ANS02] Andrew Adams, Jonathan Nicholas, and William Siadaa. "draft-ietf-pim-dm-new-v2-01.txt" : Protocol independent multicast-dense mode (PIM-DM) : Protocol specification (revised). http ://www.ietf.org/internet-drafts/draft-ietf-pim-dm-new-v2-01.txt, February 2002.

[AWSBL99] William Adjie-Winoto, Elliot Schwartz, Hari Balakrishnan, and Jeremy Lilley. The design and implementation of an intentional naming system. In *Proceedings of 17th ACM SOSP*, 1999.

[BA97] Fred Baker and Randall Atkinson. Rip-2 md5 authentication. Request For Comments 2082, Internet Engineering Task Force, January 1997.

[Baa88] Sara Baase. *Computer Algorithms*. Addison-Wesley, 2nd edition, 1988.

[Bal97a] Tony Ballardie. Cbt version 2 multicast routing. Request For Comments 2189, Internet Engineering Task Force, September 1997.

[Bal97b] Tony Ballardie. Core based trees multicast routing architecture. Request For Comments 2201, Internet Engineering Task Force, September 1997.

[BBBC01] Paul Barford, Azer Bestavros, John Byers, and Mark Crovella. On the marginal utility of network topology measurements. In *Proceedings of ACM SIGCOMM Internet Measurement Workshop'01*, pages 5–17, November 2001.

[BBK02] Suman Banerjee, Bobby Bhattacharjee, and Christopher Kommareddy. Scalable application layer multicast. In *Proceedings of ACM SIGCOMM'02*, 2002.

[BC99] Hal Burch and Bill Cheswick. Mapping the internet. *IEEE Computer*, 32(4) :97–98, 1999.

[BC01] Andre Broido and Kc Claffy. Internet topology : Connectivity of ip graphs. In *Proceedings of SPIE ITCom'01*, Denver, CO, USA, August 2001.

[BCKR98] T. Bates, R. Chandra, D. Katz, and Y. Rekhter. RFC 2283 : Multiprotocol extensions for BGP-4., February 1998.

[BDPT02] Tian Bu, Nick Duffield, Francesco Lo Presti, and Don Towsley. Network tomography on general topologies. In *Proceedings of ACM SIGMETRICS'02*, 2002.

[BFC93] Tony Ballardie, Paul Francis, and Jon Crowcroft. Core based trees : an architecture for scalable inter-domain multicast routing. In *SIGCOMM*, pages 85–95, San Francisco, California, September 1993.

[BGT02] Tian Bu, Lixin Gao, and Don Towsley. On characterizing bgp routing table growth. In *Proceedings of IEEE GLOBECOM'02*, pages 2197–2201, 2002.

[BH] Frédéric Beck and Mickaël Hoerdt. *Libemu 1.0*. Université Louis Pasteur, http ://clarinet.u-strasbg.fr/ hoerdt/libemu/.

[Bha03] S. Bhattacharyya. An overview of source-specific multicast (ssm). Request For Comments 3569, Internet Engineering Task Force, July 2003.

[BHP03] Frédéric Beck, Mickaël Hoerdt, and Jean-Jacques Pansiot. Source discovery protocol in ssm network. Internet draft, Internet Engineering Task Force, June 2003.

[BKK+03] Suman Banerjee, Christopher Kommareddy, Koushik Kar, Bobby Bhattacharjee, and Samir Khuller. Construction of an efficient overlay multicast infrastructure for real-time applications. In *Proceedings of IEEE INFOCOM'03*, 2003.

[BLR+04] Hari Balakrishnan, Karthik Lakshminarayanan, Sylvia Ratnasamy, Scott Shenker, Ion Stoica, and Michael Walfish. A layered naming architecture for the internet. In *Proceedings of ACM SIGCOMM'04*, pages 343–352, August 2004.

[BT02] Tian Bu and Don Towsley. On distinguishing between internet power law topology generators. In *Proceedings of IEEE INFOCOM'02*, New York City, NY, USA, June 2002.

[BZB+97] Roy Braden, Lixia Zhang, Steve Berson, Shai Herzog, and Sugih Jamin. Resource reservation protocol (rsvp) – version 1 functional specification. Request For Comments 2201, Internet Engineering Task Force, September 1997.

[CA02] Robert Chalmers and Kevin Almeroth. On the topology of multicast trees. *IEEE/ACM Transactions on Networking*, 2002.

[CbAH02] Reuven Cohen, Daniel ben Avraham, and Shlomo Havlin. Percolation critical exponents in scale-free networks. *Phys. Rev.*, (E 66), 2002. 036113.

[CCG+02] Qian Chen, Hyunseok Chang, Ramesh Govindan, Sugih Jamin, Scott Shenker, and Walter Willinger. The origin of power laws in internet topologies revisited. In *Proceedings of IEEE INFOCOM'02*, New York City, NY, USA, June 2002.

[CDF+02] Cain, Deering, Fenner, Kouvelas, and Thyagarajan. "RFC 3376" internet group management protocol, version 3. http ://www.ietf.org/rfc/rfc3376.txt, October 2002.

[CDG+02] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. Security for structured peer-to-peer overlay networks. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI'02)*, December 2002.

[CDZ97] Kenneth Calvert, Matthew Doar, and Ellen Zegura. Modeling internet topology. *IEEE Transactions on Communications*, pages 160–163, December 1997.

[CEbAH01] Reuven Cohen, Keren Erez, Daniel ben Avraham, and Shlomo Havlin. Breakdown of the internet under intentional attack. *Phys. Rev. Lett.*, (86) :3682–3685, 2001.

[CGJ+02] Hyunseok Chang, Ramesh Govindan, Sugih Jamin, Scott Shenker, and Walter Willinger. Towards capturing representative as-level internet topologies. Technical report, University of Michigan, 2002.

[Chr97] Clayton Christensen. *The Innovator's Dilemma*. HarperCollins Publishers, 1997.

[CJJ+02] Eric Cronin, Sugih Jamin, Cheng Jin, Anthony Kurc, Danny Raz, and Yuval Shavitt. Constrained mirror placement on the internet. *IEEE Journal on Selected Areas in Communications*, (7) :1369–1382, September 2002.

[CJW01] Hyunseok Chang, Sugih Jamin, and Walter Willinger. Inferring as-level internet topology from router-level path traces. In *Proceedings of SPIE ITCom'01*, Denver, CO, USA, August 2001.

[CMK+02] Jun-Hong Cui, Dario Maggiorini, Jinkyu Kim, Khaled Boussetta, and Mario Gerla. A protocol to improve the state scalability of source specific multicast. In *Proceedings of the IEEE GLOBECOM'02*, pages 1910–1915, November 2002.

[Cow99] Lenore Cowen. Compact routing with minimum stretch. In *Proceedings of the 10th ACM-SIAM Symposium on Discrete Algorithms*, January 1999.

[CS03] Julian Chesterfield and Eve M. Schooler. An extensible rtcp control framework for large multimedia distributions. *IEEE Symposium on Network Computers and Applications (NCA-03)*, April 2003.

[Dee89] Stephen Deering. Host extensions for ip multicasting. Request For Comments 1112, Internet Engineering Task Force, August 1989.

[Dee91] Stephen Deering. *Multicast routing in a datagram internetwork*. PhD thesis, Standford University, December 1991.

[DH98] Steve Deering and R. Hinden. Internet protocol version 6 (ipv6) specification. Request For Comments 2460, Internet Engineering Task Force, December 1998.

[DM78] Yogan Dalal and Robert Metcalfe. Reverse path forwarding of broadcast packets. *Communications of the ACM*, 21(12) :1040–1048, December 1978.

[Doa96] Matthew Doar. A better model for generating test networks. In *Proceedings of IEEE GLOBECOM'96*, November 1996.

[EA98] Deborah Estrin and Al. RFC 2362 : Protocol independent multicast-sparse mode (PIM-SM) : Protocol specification., June 1998.

[EF94] K Egevang and P Francis. The ip network adress translator (nat). In *RFC1631*, May 1994.

[EFH+98] Deborah Estrin, Dino Farinacci, Ahmed Helmy, D Thaler, Stephen Deering, M Handley, Van Jacobson, Ching-Gung Liu, Puneet Sharma, and Liming Wei. Protocol independent multicast-sparse mode (pim-sm) : Protocol specification. Request For Comments 2362, Internet Engineering Task Force, June 1998.

[EGP98] Tamar Eilam, Cyril Gavoille, and David Peleg. Compact routing schemes with low stretch factor. In *Proceedings of the 17th ACM Symposium on Principles of Distributed Computing*, pages 11–20, August 1998.

[EHaPH99] Deborah Estrin, Mark Handley, and Ahmed Helmy an Polly Huang. A dynamic bootstrap mechanism for rendezvous-based multicast routing. In *INFOCOM*, New York,USA, March 1999.

[FCGF01a] Aiguo Fei, Junhong Cui, Mario Gerla, and Michalis Faloutsos. Aggregated multicast : an approach to reduce multicast state. In *Proceedings of IEEE GLOBECOM'01*, San Antonio, Texas, November 2001.

[FCGF01b] Aiguo Fei, Junhong Cui, Mario Gerla, and Michalis Faloutsos. Aggregated multicast with inter-group sharing. In *Proceedings of Networked Group Communications'01*, London, UK, November 2001.

[Fen97] William Fenner. Internet group management protocol version 2. Request For Comments 2236, Internet Engineering Task Force, November 1997.

[FFF99] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *Proceedings of ACM SIGCOMM'99*, pages 251–262, Cambridge, Massachusetts, USA, August 1999.

[FG01] Paul Francis and Ramakrishna Gummadi. Ipnl : A nat-extended internet architecture. In *Proceedings of ACM SIGCOMM'01*, 2001.

[FJJ+01] Paul Francis, Sugih Jamin, Cheng Jin, Yixin Jin, Danny Raz, Yuval Shavitt, and Lixia Zhang. Idmaps : A global internet host distance estimation service. *IEEE/ACM Transactions on Networking*, 9(5) :525–540, 2001.

[FJP+99] Paul Francis, Sugih Jamin, Vern Paxson, Lixia Zhang, Daniel Gryniewicz, and Yixin Jin. An architecture for a global internet host distance estimation service. In *Proceedings of IEEE INFOCOM'99*, March 1999.

[FK97] I. Foster and C. Kesselman. Globus : A metacomputing infrastructure toolkit. *International Journal of Supercomputing Applications*, 11(2) :115–128, 1997.

[FK99] I. Foster and C. Kesselman, editors. *The Grid : Blueprint for a Future Computing Infrastructure*. Morgan-Kaufmann, 1999.

[FM03] Bill Fenner and David Meyer. Distance vector multicast routing protocol. Request For Comments 3618, Internet Engineering Task Force, Nctober 2003.

[FP01] Sally Floyd and Vern Paxson. Difficulties in simulating the internet. *IEEE/ACM Transactions on Networking*, 9(4) :392–403, August 2001.

[Fra00] Paul Francis. Yoid : Extending the internet multicast architecture. Technical report, ATT Center for Internet Research at ICSI, 2000.

[FYT91] F.Teraoka, Y. Yokote, and M. Tokoro. A network architecture providing host migration transparency. In *Proceedings of ACM SIGCOMM'91*, 1991.

[GC01] M. Gritter and D. R. Cheriton. An architecture for content routing support in the internet. In *Proceedings of Usenix Symposium on Internet Technologies and Systems*, 2001.

[GMZ03] Christos Gkantsidis, Milena Mihail, and Ellen W. Zegura. Spectral analysis of internet topologies. In *Proceedings of 22nd IEEE INFOCOM*, 2003.

[GSGM03] Prasanna Ganesan, Qixiang Sun, and Hector Garcia-Molina. Yappers : A peer-to-peer lookup service over arbitrary topology. In *Proceedings of 22nd IEEE INFOCOM*, 2003.

[GT00] Ramesh Govindan and Hongsuda Tangmunarunkit. Heuristics for internet map discovery. In *Proceedings of IEEE INFOCOM'00*, Tel Aviv, Israël, March 2000.

[Har69] Frank Harary. *Graph Theory*. Addison-Wesley, 1969.

[HB96] John Hawkinson and Tony Bates. Guidelines for creation, selection and registration of an autonomous system (as). Request For Comments 1930, Internet Engineering Task Force, March 1996.

[HBMP04] Mickaël Hoerdt, Frédéric Beck, Damien Magoni, and Jean-Jacques Pansiot. A source discovery protocol for asm applications in ssm networks. In *Proceedings of the 3rd International Conference on Networking*, pages 324–330, Gosier, Guadeloupe, French Caribbean, March 2004.

[HBP03] M. Hoerdt, F. Beck, and J-J Pansiot. Multi-source communications over ssm networks : draft-hoerdt-mboned-multisource-ssm-02.txt, June 2003.

[HC99] Hugh Holbrook and David Cheriton. Ip multicast channels : Express support for large-scale single-source applications. In *Proceedings of ACM SIGCOMM'99*, pages 65–78, 1999.

[hCRSZ01] Yang hua Chu, Sanjay Rao, Srinivasan Seshan, and Hui Zhang. Enabling conferencing applications on the internet using an overlay multicast architecture. In *Proceedings of ACM SIGCOMM'01*, August 2001.

[hCRZ00] Yang hua Chu, Sanjay Rao, and Hui Zhang. A case for end system multicast. In *Proceedings of ACM SIGMETRICS'00*, 2000.

[Hef98] Andy Heffernan. Protection of bgp sessions via the tcp md5 signature option. Request For Comments 2385, Internet Engineering Task Force, August 1998.

[HJ95] Mark Handley and Van Jacobson. Sdp : Session description protocol. Internet draft, Internet Engineering Task Force, 1995.

[HKC+96] Kim Hubbard, Mark Kosters, David Conrad, Daniel Karrenberg, and Jon Postel. Internet registry ip allocation guidelines. Request For Comments 2050, Internet Engineering Task Force, November 1996.

[HKYH02] Petter Holme, Beom Jun Kim, Chang No Yoon, and Seung Kee Han. Attack vulnerability of complex networks. *Phys. Rev.*, (E 65), 2002. 056109.

[HM03] Mickaël Hoerdt and Damien Magoni. Completeness of the internet core topology collected by a fast mapping software. In *Proceedings of the 11th International Conference on Software, Telecommunications and Computer Networks*, pages 257–261, Split, Croatia, October 2003.

[HM04] Mickaël Hoerdt and Damien Magoni. Distribution of multicast tree states over the ipv6 network topology. In *Proceedings of the IEEE International Conference on Communications*, pages 1991–1995, Paris, France, June 2004.

[Hol01] Hugh Holbrook. *A channel model for multicast*. PhD thesis, Standford University, August 2001.

[HPMG03] Mickaël Hoerdt, Jean-Jacques Pansiot, Damien Magoni, and Dominique Grad. Répartition des états de routage multipoint par les plus courts chemins multiples. In *Actes des 5èmes rencontres francophones sur les aspects Algorithmiques des Télécommunications*, pages 139–145, Banyuls-sur-mer, France, Mai 2003.

[HPW00] Mark Handley, Colin Perkins, and Edmund Whelan. Session announcement protocol. Request For Comments 2974, Internet Engineering Task Force, October 2000.

[HSSR99] M Handley, H Schulzrinne, E Schooler, and J Rosenberg. Sip : Session initiation protocol. In *RFC2543*, March 1999.

[IAN] IANA, http ://www.ripe.net/ipv6/ipv6allocs.html. *Regional Internet registries databases (RIR)*.

[JJK+01] Sugih Jamin, Cheng Jin, Anthony Kurc, Danny Raz, and Yuval Shavitt. Constrained mirror placement on the internet. In *Proceedings of IEEE INFOCOM'01*, Anchorage, Alaska, USA, April 2001.

[KA98] S. Kent and R. Atkinson. Security architecture for the internet protocol. Request For Comments 2401, Internet Engineering Task Force, November 1998.

[KB04] Gu-In Kwon and John Byers. Roma : Reliable overlay multicast with loosely coupled tcp connections. In *Proceedings of IEEE INFOCOM'04*, March 2004.

[KFY04] Dmitri Krioukov, Kevin Fall, and Xiaowei Yang. Compact routing on internet-like graphs. In *Proceedings of IEEE INFOCOM'04*, March 2004.

[KMC+00] E Kohler, R Morris, B Chen, J Janotti, and F Kaashoek. The click modular router. In *ACM Transactions On Computer Systems*, August 2000.

[LAWD04] Lun Li, David Alderson, Walter Willinger, and John Doyle. A first-principles approach to understanding the internet's router-level topology. In *Proceedings of ACM SIGCOMM'04*, pages 3–14, August 2004.

[LBCX03] Anukool Lakhina, John W. Byers, Mark Crovella, and Peng Xie. Sampling biases in ip topology measurements. In *Proceedings of 22nd IEEE INFOCOM*, 2003.

[LG05] Matthieu Latapy and Jean-Loup Guillaume. Relevance of massively distributed explorations of the internet topology : Simulation results. In *Proceedings of 24th IEEE INFOCOM*, 2005.

[LK00] A.M. Law and W.D. Kelton. *Simulation Modelling and Analysis*. McGraw-Hill, 3rd edition, 2000.

[LM04] Zhi Li and Prasant Mohapatra. Impact of topology on overlay routing service. In *Proceedings of IEEE INFOCOM'04*, March 2004.

[LN01] Jorg Liebeherr and M. Nahas. Application-layer multicast with delaunay triangulations. In *Proceedings of IEEE Globecom'01*, November 2001.

[Mag] Damien Magoni. *network manipulator (nem)*. Université Louis Pasteur, https ://dpt-info.u-strasbg.fr/~magoni/nem/.

[Mag02] Damien Magoni. Nem : a software for network topology analysis and modeling. In *Proceedings of the 10th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 364–371, Fort Worth, Texas, USA, October 2002.

[Mag03a] Damien Magoni. A scalable and unifying architecture for deploying advanced protocols in the internet. In *Proceedings of the 10th International Conference on Telecommunications*, pages 1001–1007, Papeete, Tahiti, French Polynesia, February 2003.

[Mag03b] Damien Magoni. Tearing down the internet. *IEEE Journal on Selected Areas in Communications*, 21(6) :949–960, August 2003.

[Mag05] Damien Magoni. Network topology analysis and internet modelling with nem. *International Journal of Computers and Applications*, 27(4) :to appear, 2005.

[Mal98] Gary Malkin. Rip (routing information protocol) version 2. Request For Comments 2453, Internet Engineering Task Force, November 1998.

[MBW97] Sandra Murphy, Madelyn Badger, and Brian Wellington. Ospf with digital signatures. Request For Comments 2154, Internet Engineering Task Force, June 1997.

[MH] Damien Magoni and Mickaël Hoerdt. *network cartographer (nec)*. Université Louis Pasteur, https ://dpt-info.u-strasbg.fr/~magoni/nec/.

[MJ95] Steven McCanne and Van Jacobson. vic : A flexible framework for packet video. In *ACM Multimedia*, pages 511–522, 1995.

[MMB00] Alberto Medina, Ibrahim Matta, and John Byers. On the origin of power laws in internet topologies. *ACM Computer Communication Review*, 30(2), April 2000.

[MN98] Makoto Matsumoto and Takuji Nishimura. Mersenne twister : a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation*, 8(1) :3–30, 1998.

[Moc87] P. Mockapetris. Domain names - implementation and specification. Request for comments, Internet Engineering Task Force, November 1987.

[Mos01] R. Moskowitz. Host identity payload and protocol. Internet draft, Internet Engineering Task Force, October 2001.

[Moy94] John Moy. Multicast extensions to ospf. Request For Comments 1584, Internet Engineering Task Force, March 1994.

[Moy98] John Moy. Ospf (open shortest path first) version 2. Request For Comments 2328, Internet Engineering Task Force, April 1998.

[MP01a] Damien Magoni and Jean-Jacques Pansiot. Algorithm for an oriented multicast routing protocol. In *Proceedings of the IEEE International Conference on Communications*, pages 2593–2597, Helsinki, Finland, June 2001.

[MP01b] Damien Magoni and Jean-Jacques Pansiot. Analysis of the autonomous system network topology. *ACM Computer Communication Review*, 31(3) :26–37, July 2001.

[MP01c] Damien Magoni and Jean-Jacques Pansiot. Influence of network topology on protocol simulation. In *Proceedings of the 1st International Conference on Networking*, pages 762–770, Colmar, France, July 2001.

[MP01d] Damien Magoni and Jean-Jacques Pansiot. Internet topology analysis and modeling. In *Proceedings of the 16th IEEE Computer Communications Workshop*, Charlottesville, Virginia, USA, October 2001.

[MP01e] Damien Magoni and Jean-Jacques Pansiot. Oriented multicast routing algorithm applied to network level agent search. *Discrete Mathematics and Theoretical Computer Science*, 4(2) :255–272, August 2001.

[MP02a] Damien Magoni and Jean-Jacques Pansiot. Evaluation of internet topology generators by power law and distance indicators. In *Proceedings of the 10th IEEE International Conference On Networks*, pages 401–406, Singapore, August 2002.

[MP02b] Damien Magoni and Jean-Jacques Pansiot. Internet topology modeler based on map sampling. In *Proceedings of the 7th IEEE Symposium on Computers and Communications*, pages 1021–1027, Giardini Naxos, Sicily, Italy, July 2002.

[MP02c] Damien Magoni and Jean-Jacques Pansiot. Network layer search service using oriented multicasting. In *Proceedings of the 21st IEEE Joint Conference on Computer Communications and Networking*, pages 1346–1355, New York City, New York, USA, June 2002.

[nMEJNSW00] D. S. Callaway nad M. E. J. Newman, S. H. Strogatz, and D. J. Watts. Network robustness and fragility : Percolation on random graphs. *Phys. Rev. Lett.*, (85) :5468–5471, 2000.

[NPB03] Akihiro Nakao, Larry L. Peterson, and Andy C. Bavier. A routing underlay for overlay networks. In *Proceedings of ACM SIGCOMM'03*, pages 11–18, 2003.

[NSW01] M. E. J. Newman, S. H. Strogatz, and D. J. Watts. Random graphs with arbitrary degree distributions and their applications. *Phys. Rev.*, (E 64), 2001. 026118.

[OPN] OPNET Technologies Inc., http ://www.opnet.com. *network simulator (OPNET)*.

[Par] Parallel Computing Laboratory, http ://pcl.cs.ucla.edu/projects/glomosim/. *GloMoSim*.

[PG98] Jean-Jacques Pansiot and Dominique Grad. On routes and multicast trees in the internet. *ACM Computer Communication Review*, 28(1) :41–50, January 1998.

[PMAM98] Vern Paxson, Jamshid Mahdavi, Andrew Adams, and Matt Mathis. An architecture for large scale internet measurement. *IEEE Transactions on Communications*, 1998.

[Pos80] Jon Postel. User datagram protocol. Request For Comments 768, Internet Engineering Task Force, August 1980.

[Pos81a] Jon Postel. Internet control message protocol. Request For Comments 792, Internet Engineering Task Force, September 1981.

[Pos81b] Jon Postel. Internet protocol. Request For Comments 791, Internet Engineering Task Force, September 1981.

[Pos81c] Jon Postel. Transmission control protocol. Request For Comments 793, Internet Engineering Task Force, September 1981.

[pro] VINT project. *network simulator (ns-2)*. UCB/LBNL, USC/ISI, Xerox PARC, http ://www.isi.edu/nsnam/vint/.

[PSF+01] Christopher Palmer, Georgos Siganos, Michalis Faloutsos, Christos Faloutsos, and Phillip Gibbons. The connectivity and fault-tolerance of the internet topology. In *Proceedings of ACM SIGMOD/PODS Workshop on Network-Related Data Management (NRDM'01)*, Santa Barbara, May 2001.

[PSM04] and Rami Lehtonen Pekka Savola and David Meyer. "draft-ietf-mboned-mroutesec-02.txt" : Pim-sm multicast routing security issues and enhancements. Internet draft, Internet Engineering Task Force, June 2004.

[QYZS03] Lili Qiu, Yang Richard Yang, Yin Zhang, and Scott Shenker. On selfish routing in internet-like environments. In *Proceedings of ACM SIGCOMM'03*, pages 151–162, 2003.

[RD01] A Rowstron and P Druschel. Pastry : Scalable, distributed object location and routing for large scale p2p systems. In *Proceedings of IFIP/ACM Middleware*, November 2001.

[RFHaSS01] S Ratnasamy, P Francis, M Handley, and R Karp ans S Schenker. A scalable content addressable network. In *Proceedings of SIGCOMM*, August 2001.

[RG95] Yakov Rekhter and Phill Gross. Application of the border gateway protocol in the internet. Request For Comments 1772, Internet Engineering Task Force, March 1995.

[RKDC01] A Rowstron, A Kermarrec, P Druschel, and M Castro. Scribe : the design of a large scale event notification infrastructure. In *Proceedings of NGC*, November 2001.

[RL95] Yakov Rekhter and Tony Li. A border gateway protocol 4 (bgp-4). Request For Comments 1771, Internet Engineering Task Force, March 1995.

[RMK+96] Yakov Rekhter, Robert Moskowitz, Daniel Karrenberg, Greet Jan de Groot, and Eliot Lear. Address allocation for private internets. Request For Comments 1918, Internet Engineering Task Force, February 1996.

[RVC01] Eric Rosen, Arun Viswanathan, and Ross Callon. Multiprotocol label switching architecture. Request For Comments 3031, Internet Engineering Task Force, January 2001.

[SAZ+02] Ion Stoica, Daniel Adkins, Shelley Zhuang, Scott Shenker, and Sonesh Surana. Internet indirection infrastructure. In *Proceedings of ACM SIGCOMM'02*, 2002.

[SB00] Alex Snoeren and Hari Balakrishnan. An end-to-end approach to host mobility. In *Proceedings of 6th ACM MobiCom*, 2000.

[SCFJ96] Henning Schulzrinne, Stephen Casner, Ron Frederick, and Van Jacobson. Rtp : A transport protocol for real-time applications. Request For Comments 1889, Internet Engineering Task Force, January 1996.

[SE01] Pyda Srisuresh and Kjeld Borch Egevang. Traditional ip network address translator. Request For Comments 3022, Internet Engineering Task Force, January 2001.

[SH03] Pekka Savola and Brian Haberman. "draft-ietf-mboned-embeddedrp-06.txt"embedding the address of the rp in ipv6 multicast address. Internet draft, Internet Engineering Task Force, Oct 2003.

[SMA03] Neil Spring, Ratul Mahajan, and Tom Anderson. Quantifying the causes of path inflation. In *Proceedings of the ACM SIGCOMM'03*, August 2003.

[sMaatM03] Dragoş Mânzˇ aţeanu and Damien Magoni. Performance evaluation of an application–layer multicast protocol. Technical Report ULP-LSIIT-RR-2003-03, Université Louis Pasteur, August 2003.

[SMK+01] Ion Stoica, Robert Morris, David Karger, Frans Kaashoek, and Hari Balakrishnan. Chord : A scalable peer-to-peer lookup service for internet applications. In *Proceedings of ACM SIGCOMM 2001*, pages 149–160, August 2001.

[SMKK02] Ion Stoica, R Morris, David Karger, and D Kaashoek. Chord : a scalable peer to peer lookup service for internet applications. In *SIGCOMM*, pages 149–160, 2002.

[SMW02] Neil Spring, Ratul Mahajan, and David Wetherall. Measuring isp topologies with rocketfuel. In *Proceedings of ACM SIGCOMM'02*, Pittsburgh, PA, USA, August 2002.

[SNA03] K. Sarac, P. Namburi, and Kevin C. Almeroth. Ssm extensions : Network layer support for multiple senders in ssm. In *ICCCN*, pages ?– ?, Dallas, October 2003.

[SSWY01] Yuval Shavitt, Xiaodong Sun, Avishai Wool, and Bulent Yener. Computing the unmeasured : An algebraic approach to internet mapping. In *Proceedings of IEEE INFOCOM'01*, Anchorage, Alaska, USA, April 2001.

[SWA03] Neil Spring, David Wetherall, and Tom Anderson. Scriptroute : A public internet measurement facility. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems (USITS)*, March 2003.

[TDG+01] Hongsuda Tangmunarunkit, John Doyle, Ramesh Govindan, Sugih Jamin, Scott Shenker, and Walter Willinger. Does as size determine degree in as topology ? *ACM Computer Communication Review*, 31, 2001.

[TGJ+02] Hongsuda Tangmunarunkit, Ramesh Govindan, Sugih Jamin, Scott Shenker, and Walter Willinger. Network topology generators : Degree-based vs. structural. In *Proceedings of ACM SIGCOMM'02*, pages 147–159, Pittsburgh, Pennsylvania, USA, August 2002.

[TGSE01] Hongsuda Tangmunarunkit, Ramesh Govindan, Scott Shenker, and Deborah Estrin. The impact of routing policy on internet paths. In *Proceedings of IEEE INFOCOM'01*, Anchorage, Alaska, USA, 2001.

[TH00] David Thaler and Mark Handley. On the aggregatability of multicast forwarding state. In *Proceedings of IEEE INFOCOM'00*, pages 1654–1663, 2000.

[TN98] Jining Tian and Gerald Neufeld. Forwarding state reduction for sparse mode multicast communication. In *Proceedings of IEEE INFOCOM'98*, pages 711–719, 1998.

[TPSF01] Sudhir Tauro, Christopher Palmer, Georgos Siganos, and Michalis Faloutsos. A simple conceptual model for the internet topology. In *Proceedings of IEEE GLOBECOM'01*, San Antonio, Texas, November 2001.

[TTNS05] Yoshio Tanaka, Hiroshi Takemiya, Hidemoto Nakada, and Satoshi Sekiguchi. Design, implementation and performance evaluation of gridrpc programming middleware for a large-scale computational grid. In *Proceeding of the 5th IEEE/ACM International Workshop on Grid Computing*, pages 298–305, 2005.

[Uni] University of Oregon, Advanced Network Technology Center, http ://www.routeviews.org/. *BGP data from route-views*.

[VCF+02] Vida, Costa, Fdida, Deering, Fenner, Kouvelas, and Haberman. "draft-vida-mld-v2-06.txt" multicast listener discovery version 2 (mldv2) for ipv6. http ://www.ietf.org/internet-drafts/draft-vida-mld-v2-06.txt, November 2002.

[VdaFR03a] Aline Viana, Marcelo Dias de amorim, Serge Fdida, and José F. Rezende. Indirect routing using distributed location information. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 224–234, March 2003.

[VdaFR03b] Aline Viana, Marcelo Dias de amorim, Serge Fdida, and José F. Rezende. Routage basé sur ancre dans les réseaux à large échelle auto-organisables. In *Actes du Colloque Francophone sur l'Ingenierie des Protocoles*, October 2003.

[Wax88] Bernard Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6(9) :1617–1622, December 1988.

[WK00] Tina Wong and Randy Katz. An analysis of multicast forwarding state scalability. *IEEE/ACM Transactions on Networking*, 8(3) :419–427, 2000.

[WPD88a] Waitzman, Partridge, and Deering. Distance vector multicast routing protocol. Request For Comments 1075, Internet Engineering Task Force, November 1988.

[WPD88b] Dan Waitzman, Craig Partridge, and Stephen Deering. Distance vector multicast routing protocol. Request For Comments 1075, Internet Engineering Task Force, November 1988.

[WR02] Marcel Waldvogel and Roberto Rinaldi. Efficient topology-aware overlay network. In *Proceedings of ACM SIGCOMM HotNets'02*, 2002.

[XMK03] Zhichen Xu, Mallik Mahalingam, and Magnus Karlsson. Turning heterogeneity into an advantage in overlay routing. In *Proceedings of 22nd IEEE INFOCOM*, 2003.

[YDK03] Ben Y.Zhao, Anthony D.Joseph, and John Kubiatowicz. Locality aware mechanisms for large-scale networks. In *Paper*, 2003.

[Zap00] Daniel Zappala. Alternate path routing for multicast. In *Proceedings of IEEE INFOCOM'00*, pages 1576–1585, 2000.

[ZCD97] Ellen Zegura, Kenneth Calvert, and Michael Donahoo. A quantitative comparison of graph-based models for internetworks. *IEEE / ACM Transactions on Networking*, 5(6) :770–783, December 1997.

[ZF01] Daniel Zappala and Aaron Fabbri. Using ssm proxies to provide efficient multiple-source multicast delivery. In *IEEE Globecom, Sixth Global Internet Symposium*, November 2001.

[ZKJ01] B Zhao, J Kubiatowicz, and A Joseph. Tapestry : An infrastructure for fault-tolerant wide-area location and routing. In *Tech. Rep. UCB/CSD-01-1141*, April 2001.

# Liste des tableaux

# Table des figures

# Annexe A

# network cartographer

L'outil *network cartographer* (*nec*) est un logiciel que nous avons développé dans le cadre de notre étude sur la cartographie de l'Internet. Il permet de collecter des traces via des serveurs Web *traceroute* ou des loupes (*i.e.*, looking glasses) qui servent à la création de la cartographie de l'Internet. Il est constitué de plus de 2000 lignes de code ISO C++ et peut être compilé sous UNIX ou Windows. Nous l'avons mis à la disposition de la communauté scientifique et il peut être gratuitement téléchargé et utilisé. L'archive disponible à l'adresse suivante contient le code source du logiciel ainsi que des fichiers de configuration pour une cartographie IPv4 :
`https://dpt-info.u-strasbg.fr/~magoni/nec/nec-0.3.1.tar.gz`

## A.1 Configuration de *nec*

Il fonctionne en ligne de commande avec des fichiers de configuration adéquats. En particulier, un fichier `.specif` contient une liste de paramètres permettant de configurer une recherche. Voici un exemple ci-dessous :

```
max_retrieve_sockets 20
targets_filename cibles.txt
tracers_filename traceurs.traceroute
max_stars 9
max_failures 20
```

Le paramètre `max_retrieve_sockets` permet de définir le nombre de connexions locales simultanées afin d'accélérer la recherche. Le paramètre `targets_filename` contient le nom du fichier contenant les adresses IP cibles des requêtes *traceroute*. Ces adresses peuvent être réelles ou fictives (*i.e.*, construites à partir de préfixes réseaux). Le paramètre `tracers_filename` contient le nom du fichier contenant les URL des serveurs *traceroute*. Le paramètre `max_stars` définit à partir de combien d'étoiles successives renvoyées par *traceroute* peut-on considérer la trace comme étant bloquée et donc avorter la recherche. Enfin le paramètre `max_failures` définit le seuil du nombre de requêtes ayant échouées au delà duquel le serveur *traceroute* n'est plus sollicité par *nec*.

## A.2 Exemple de carte créée par *nec*

Les cartes créées par *nec* sont des recouvrements : elles contiennent des informations de niveau routeur et de niveau système autonome. La correspondance entre les deux niveaux peut s'effectuer grâce au champ `as_number` contenu dans les informations sur les routeurs. Ce champ indique pour chaque routeur à quel AS il appartient. Pour les routeurs, chaque ligne

contient l'indice du routeur (de 0 à n-1), suivi de son adresse IP canonique (*i.e.*, la plus petite), de son nom de domaine lorsqu'il existe, du numéro de l'AS auquel il appartient et enfin d'un booléen précisant si ce routeur est un routeur BGP (cf. chapitre 1). Ci-dessous un exemple d'une carte de l'Internet IPv4 :

```
ROUTERS (subscript ip_address fqdn as_number is_bgp):
0 200.45.42.241 host042241.arnet.net.ar 7303 0
1 200.45.104.253 host104253.arnet.net.ar 7303 0
2 200.3.34.201 cli03ra-se-9-0-0.tasf.telecom.net.ar 43290 0
3 200.3.32.69 69.32.3.200.telecom.net.ar 43290 0
4 200.3.32.114 mun01rt-pos-1-1-0.tasf.telecom.net.ar 43290 0
...
8 66.28.4.65 p5-0.core01.jax01.atlas.cogentco.com 16631 0
9 66.28.4.138 p5-0.core01.atl01.atlas.cogentco.com 16631 1
...
14 65.208.85.113 500.POS3-0.IG2.MIA4.ALTER.NET 701 1
15 152.63.7.213 0.so-1-0-3.XL1.MIA4.ALTER.NET 701 1
...
```

Pour les liens, chaque ligne contient l'adresse IP canonique du routeur de départ, celle du routeur d'arrivée et une mesure moyenne du delai de ce lien en ms. Le délai vaut 0 lorsque sa vraie valeur n'est pas connue (*i.e.*, lien traversé uniquement dans l'autre sens).

```
LINKS (initial_router_addr final_router_addr link_delay_ms):
200.45.42.241 200.45.104.253 17.3685
200.45.42.241 200.3.45.81 20.9045
200.45.104.253 200.45.42.241 0
200.45.104.253 200.3.34.201 3.26549
200.45.104.253 200.3.34.205 2.89003
...
```

Pour les systèmes autonomes, chaque ligne contient l'indice de l'AS suivi de son numéro d'AS. Pour les connexions, chaque ligne contient le numéro de l'AS de départ et celui d'arrivée.

# Annexe B

# Sélection de publications

Cette annexe contient une sélection de publications représentatives de mes travaux de recherche. Ces publications complètent les résultats présentés dans les chapitres de ce mémoire.

1. Damien Magoni and Mickaël Hoerdt.
   Internet core topology mapping and analysis.
   *Computer Communications*,
   28(5) : 494–506, March 2005. (cf. chapitre 1)
2. Damien Magoni.
   Tearing down the Internet.
   *IEEE Journal on Selected Areas in Communications*,
   21(6) : 949–960, August 2003. (cf. chapitre 2)
3. Mickaël Hoerdt and Damien Magoni.
   Distribution of multicast tree states over the IPv6 network topology.
   In *Proceedings of the IEEE International Conference on Communications*,
   pages 1991–1995, June 2004. (cf. chapitre 3)
4. Mickaël Hoerdt, Frédéric Beck, Damien Magoni and Jean-Jacques Pansiot.
   A source discovery protocol for ASM applications in SSM networks.
   In *Proceedings of the International Conference on Networking*,
   pages 324–330, March 2004. (cf. chapitre 5)
5. Khaldoon Shami, Damien Magoni and Pascal Lorenz.
   Autonomous, scalable, and resilient overlay infrastructure.
   *KICS/IEEE Journal of Communications and Networks*,
   8(4) : 378–390, December 2006. (cf. chapitre 6)

## Résumé
L'Internet est devenu sans conteste en trois décades le support matériel majeur de l'ère de l'information. Durant cette courte période, il a subi deux mutations importantes. Tout d'abord, il a subi une mutation de taille : de quelques dizaines de systèmes interconnectés suivant une topologie fixée et bien connue, l'Internet est passé à une taille phénoménale de plus de 285 millions d'hôtes suivant une topologie désormais à dimension libre dûe au fait que son développement n'est plus centralisé. De plus, il a subi une mutation de forme : nous sommes passés des gros systèmes interconnectés par de simples lignes téléphoniques à des ordinateurs personnels mobiles interconnectés par des technologies radio en bordure de réseau et par des routeurs gigabits reliés par fibre optique en coeur de réseau. Ces deux mutations entraînent deux défis importants pour les chercheurs qui oeuvrent à l'amélioration de l'Internet. Premièrement, il est nécessaire de connaître avec le plus de précision possible, la topologie de l'Internet. Seules des méthodes macroscopiques et statistiques permettent désormais de connaître cette topologie car plus personne n'a autorité sur l'organisation et l'expansion du réseau Internet. De plus, étant donné sa dynamicité, il est aussi souhaitable que cette topologie soit mise à jour fréquemment. Deuxièmement, l'évolution rapide des technologies a fait que les paradigmes qui sous-tendaient à la création des protocoles initiaux de l'Internet ne sont plus valables à l'heure actuelle. Les notions de mobilité, de sécurité et de diffusion quasi-inexistantes à l'origine sont désormais ardemment souhaitées par les utilisateurs mais difficiles à mettre en oeuvre dans les protocoles actuels. Il faut donc trouver les moyens d'offrir ces nouvelles fonctionnalités en conservant les propriétés initiales des protocoles qui ont fait le succès d'Internet. Dans ce mémoire d'habilitation, nous présentons des contributions qui tentent de répondre à certains aspects de ces deux défis majeurs. Nous nous concentrons tout particulièrement sur la cartographie de l'Internet, le stockage d'états dans les routeurs exécutant des protocoles multipoints, l'implémentation de fonctionnalités réseaux dans les hôtes et la conception de réseaux recouvrants.

**Title** Internet topologies: from routers to overlays

## Abstract
The Internet has become in three decades, the main foundation of the information era. During this short time, it has gone through two heavy mutations. First, it has gone through a size mutation : from a dozen of connected systems, it has grown to a huge size of more than 285 million hosts. Second it has gone through a form mutation : from mainframes connected by phone lines, it has mutated to wireless-enabled laptops and gigabit routers connected by fiber optics. These mutations are bringing two great challenges to the Internet researchers. First we have to map the Internet topology with accuracy and timeliness in order to improve the efficiency of the forthcoming protocols. Second we have to develop new protocols for providing advanced communication functionalities such as mobility, security and multicast that are adapted to the new devices and usages of the Internet. To this end, we present here contributions that resolve some parts of the challenges described earlier. These are focused on Internet mapping, multicast state storing, host level networking and overlay design.

**Discipline doctorale** Informatique

**Mots-clés** Adressage, attaque, cartographie, Internet, multichemin, multipoint, nommage, protocole, recouvrant, recouvrement, réseau, routage, topologie.

## Intitulé et adresse du laboratoire
Laboratoire des Sciences de l'Image, de l'Informatique et de la Télédétection (LSIIT)
UMR n° 7005 du CNRS, Pôle API, Boulevard Sébastien Brant, 67412 Illkirch, France.