THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par *L'INSTITUT NATIONAL POLYTECHNIQUE DE TOULOUSE*

Discipline ou spécialité : *Réseaux et Télécommunications*

Présentée et soutenue par *Sakuna CHAROENPANYASAK*

Le *23/06/2008*

Titre : *Optimisation inter-couches du protocole SCTP en réseaux ad hoc*

JURY

M.	Michel DIAZ	Président	Directeur de recherche du CNRS
M.	Khaldoun AL AGHA	Rapporteur	Professeur Université PARIS 11
Mme.	Pascale VICAT-BLANC PRIMET	Rapporteur	Directeur de recherche INRIA
M.	Fabrice ARNAL	Membre	Ingénieur de recherche Thales Alenia Space
M.	André-Luc BEYLOT	Membre	Professeur INPT/ENSEEIHT
Mme.	Béatrice PAILLASSA	Membre	Professeur INPT/ENSEEIHT

Ecole doctorale : *MATHEMATIQUES, INFORMATIQUE ET TELECOMMUNICATIONS DE TOULOUSE*
Unité de recherche : *UMR 5505*

Directeur(s) de Thèse : *Mme. Béatrice PAILLASSA*

Optimisation inter-couches du protocole SCTP en réseaux ad hoc

Sakuna CHAROENPANYASAK

Thèse dirigée par **Béatrice PAILLASSA**

"Il ne faut pas attendre d'être parfait pour commencer quelque chose de bien"
Abbé Pierre

"There is nothing either good or bad but thinking makes it so"
William Shakespeare

"Imagination is more important than knowledge"
Albert Einstein

Remerciements

Cette thèse s'est déroulée au sein du laboratoire IRIT (Institut de Recherche en Informatique de Toulouse) site de l'ENSEEIHT, Institut National Polytechnique de Toulouse (INPT) entre septembre 2004 et juin 2008 sous la direction du Professeur Béatrice PAILLASSA. Mes premiers remerciements iront à Béatrice PAILLASSA, pour m'avoir soutenue pendant ma thèse. J'aimerais lui adresser mes plus vifs remerciements pour tout son dynamisme et ses compétences scientifiques qui m'ont permis de mener à bien cette étude. Ce travail n'aurait jamais pu aboutir sans elle, qui a toujours su me consacrer des moments de son temps, me guider et me conseiller, et me témoigner son soutien et sa confiance. Je souhaite lui transmettre l'expression de ma reconnaissance et ma plus profonde et sincère gratitude.

Je remercie tout particulièrement les membres de mon jury de thèse, qui ont accepté de juger ce travail et de participer au jury. Je suis profondément reconnaissante à mes deux rapporteurs Madame Pascale VICAT-BLANC PRIMET, Directrice de recherche à INRIA, Lyon et Monsieur Khadoun AL AGHA, Professeur à Université Paris-Sud 11 d'avoir accepté de lire et évaluer ma thèse. Je voudrais aussi exprimer ma gratitude à Monsieur Michel DIAZ, Directeur de recherche au LAAS-CNRS de Toulouse pour avoir présidé le jury. Je tiens également à remercier Monsieur André-Luc BEYLOT, Professeur à l'ENSEEIHT/INPT et Monsieur Fabrice ARNAL, Ingénieur de recherche de Thales Alenia Space France-Toulouse, qui m'ont également fait l'honneur d'accepter de participer au jury.

Toute mon amitié à Farid Jaddi avec qui j'ai partagé le bureau tout au long de mes deux premières années de thèse et qui a su m'accompagner dans cette grande expérience scientifique mais surtout personnelle qu'est une thèse ; merci pour les moments d'amitié et de complicité que nous avons partagés. Je tiens aussi à remercier chaleureusement Sylvie Eichen, avec qui j'ai eu le plaisir de partager bien d'autres choses, ton humanisme, ton éternal humeur, et ton soutien aux moments les plus difficiles font de toi un être humain merveilleux. Je n'oublierai pas les aides permanentes reçues du personnel du laboratoire, Je remercie en particulier Stéphane Jublin.

Je remercie également l'ensemble des enseignants et doctorants qui m'ont offert un excellent cadre de travail ainsi qu'un séjour extrêmement agréable. Merci Michel Daydé, Christian Fraboul, Emmanuel Chaput, Jérôme Ermont, Riadh Dhaou, Claudia Betous, Marc Boyer, Jean-Luc Scharbarg, Julien Fasson, Anas Abou El Kalam, "Sylvie Armengaud et Boubounette", Hussein Charara, Mahamadou Issoufou Tiado, Alexandra Niculae, Rahim Kacimi, Clovis Tauber, Cholatip Yawut, Wasimon Panichpattanakul, Nassima Izerrouken «la belle», Zehor Oukzili, Mohamad Salhani et Henri Bauer. Je remercie Mr. Propre qui s'occupe de mon bureau et a toujours un mot gentil.

Je suis très reconnaissante envers l'ensemble du personnel de l'Université de la Thaïlande pour leur professionnalisme et l'esprit de service qu'ils ont toujours eu à mon égard. Merci aux Prof. Sinchai Kamolphiwong, Prof. Amnuay Sitticharearanchai, Prof. Wiwat Sutiwipakorn et sa femme, Prof. Chusak Limsakul, Prof. Krerkchai Thongnoo, Prof. Thanate Kaorapapong et bien d'autres qui ont contribué au bon déroulement de cette thèse.

Je voudrais adresser aussi un grand merci à tous mes amis thaïlandais. Merci, P'Jan, P'Nee, P'Anchalee, P'Ooy, Geuy, P'Lee, "P'Yao et sa famille", P'Mam, P'Tot, N'Nut, Mon, Pang, P'Bus, Bee, et beaucoup d'autres. Votre présence a rendu mon séjour à Grenoble inoubliable. Je remercie P'Mou, qui a su faire preuve d'une patience exceptionnelle. Merci de son soutien et de ses encouragements tout au long de ma thèse. Je lui dédie ce travail.

Enfin, j'adresse une pensée toute particulière à ma grande famille et notamment ma grand-mère et mon père. Cette page serait loin de suffire pour vous exprimer toute ma reconnaissance et mon affection.

A toutes ces personnes je dis un grand merci.

Table des matières

Table des figures

Liste des tableaux

Introduction

Motivation

Les réseaux ad hoc utilisent une transmission sans fil directe entre les équipements sans passer par des stations de base : ils n'ont pas d'architecture. La transmission entre les noeuds adjacents est gérée par un protocole d'accès. Lorsque le récepteur n'est pas à portée directe de transmission, des équipements intermédiaires situés sur le chemin entre la source et la destination vont servir de relais. Les réseaux MANETs (Mobile Ad hoc Networks) du nom du groupe éponyme de l'IETF (Internet Engineering Task Force) sont des réseaux ad hoc constitués d'éléments mobiles qui forment des topologies dynamiques. Leurs protocoles d'accès et de routage ont fait l'objet de nombreuses recherches qui ont abouti récemment à la standardisation de solutions spécifiques. Cependant l'utilisation de ces solutions ne résout pas les problèmes de performances au niveau des protocoles supérieurs.

Les réseaux ad hoc, outre la transmission sans fil et la capacité de routage, sont également caractérisés par la mobilité de leurs éléments. A tout instant un équipement peut se déplacer et provoquer des ruptures de routes et donc des pertes de données dans le réseau. Les protocoles de transport qui sont chargés de fiabiliser la communication sur des réseaux à pertes sont affectés par ce type de comportement. Alors que les protocoles de transport ont été conçus pour s'adapter aux problèmes de congestion dans les réseaux filaires, il s'agit à présent pour eux de s'adapter à la mobilité en utilisant au mieux les ressources limitées du système.

Une première génération de travaux sur les réseaux non filaires a mis en avant et proposé des solutions pour améliorer les performances du protocole TCP en environnement sans fil, en l'adaptant à la mauvaise qualité de transmission des liens sans fils. L'objectif principal de ces travaux était d'éviter que le transport ne diminue le débit de transmission, alors que des protocoles ou des mécanismes spécifiques pouvaient être définis pour corriger les erreurs de transmission. La deuxième génération de travaux se préoccupe de la transmission en environnements multisauts et traite de la perte de route due à la mobilité. Deux types de solutions sont proposées : des solutions dites avec retour, car prenant en compte l'état du système de transmission, et des solutions sans retour, qui optimisent au mieux les processus de TCP. Si le gain de performance obtenu avec le premier type de solution peut s'avérer plus important que par le second il n'en présente pas moins l'incon-

vénient de modifier le protocole TCP pour mettre en oeuvre le retour d'état. Une autre façon d'appréhender ce retour est alors de s'appuyer sur les informations communiquées par les couches inférieures. Le protocole de transport n'est plus limité aux interactions couches inférieures couches supérieures du modèle OSI (Open Systems Interconnection), chacun gérant son propre espace de données, et il peut s'appuyer sur une vision croisée (cross layer) de la pile de protocoles.

C'est cette approche d'optimisation, par communication inter-couches, que nous étudions dans cette thèse.

Le protocole SCTP est un protocole proche de TCP qui tient compte de l'expérience acquise avec TCP pour améliorer certains aspects de la communication fiable. Les deux innovations majeures de SCTP sont : la livraison séquencée de messages utilisateurs dans des flux multiples, et la livraison de messages par des interfaces multiples. SCTP, par son option de multihoming, gère l'association entre des équipements multidomiciliés, c'est à dire possédant plusieurs adresses internet. Cette caractéristique améliore fortement la robustesse du système, puisqu'en cas de panne sur une interface, le trafic est automatiquement basculé sur une autre, comme l'attestent les travaux effectués dans les réseaux traditionnels. Sachant que les réseaux ad hoc sont assujettis à des pannes, notre objectif est d'utiliser la caractéristique de multihoming de SCTP pour améliorer les performances du transport. Plus généralement les réseaux ad hoc n'étant pas constitués de façon systématique d'équipements multidomiciliés, nous avons cherché à optimiser le protocole SCTP par une communication inter-couches.

Contributions

Dans cette thèse, nous étudions les possibilités d'améliorer les performances en réseaux ad hoc du protocole de transport existant SCTP, normalisé par l'IETF, par une adaptation du protocole à la mobilité et à la topologie du réseau.

Nous montrons l'importance du routage sur les performances du transport et en déduisons l'intérêt d'étudier les possibilités de communications entre ces deux couches. Nous proposons deux types d'adaptation protocolaires, une adaptation locale à un noeud, et une adaptation répartie qui nécessite un partage d'informations sur plusieurs noeuds. Elles optimisent les performances du protocole SCTP, en terme de nombre de paquets transmis, sans avoir besoin de modifier celui-ci.

La première optimisation, adapte le protocole de transport selon l'état du réseau ad hoc perçu par le protocole de routage. Elle conditionne l'activation du processus de congestion à la détection des ruptures de routes. Cette proposition, est représentative des travaux menés dans la littérature. Notre principal apport est l'étude détaillée que nous en avons faite. Nous avons implanté ce fonctionnement sur un simulateur NS2 avec un routage AODV et examiné le domaine de validité de l'optimisation. Celui-ci se détermine par le degré d'encombrement du réseau, et la longueur des chemins empruntés.

Nous proposons également une optimisation inverse où il s'agit d'adapter le routage au fonctionnement du protocole transport, par une adaptation répartie. L'originalité de

notre démarche est d'utiliser des chemins multiples gérés au niveau transport, et non pas au niveau réseau par l'algorithme de routage, comme le proposent de nombreux travaux. L'intérêt est de réutiliser un protocole standard, tout en bénéficiant de fonctions de fiabilité du transfert, pour augmenter la performance du transport. Nous montrons que l'utilisation d'une gestion de chemins multiples par les stations terminales, nécessite, en réseaux ad hoc, une communication inter-couches répartie entre éléments des couches transport et réseau. Le protocole de routage s'adapte à la topologie des flux de transmissions. Nous étudions par simulation les politiques de retransmissions, ainsi que le domaine de validité de l'optimisation, en fonction de la métrique "degré de similitude" qui reflète le nombre d'éléments communs entre deux chemins transport.

Ce travail se conclut par la proposition d'une architecture d'adaptation pour utiliser selon leurs domaines de validité les optimisations proposées.

Organisation de la thèse

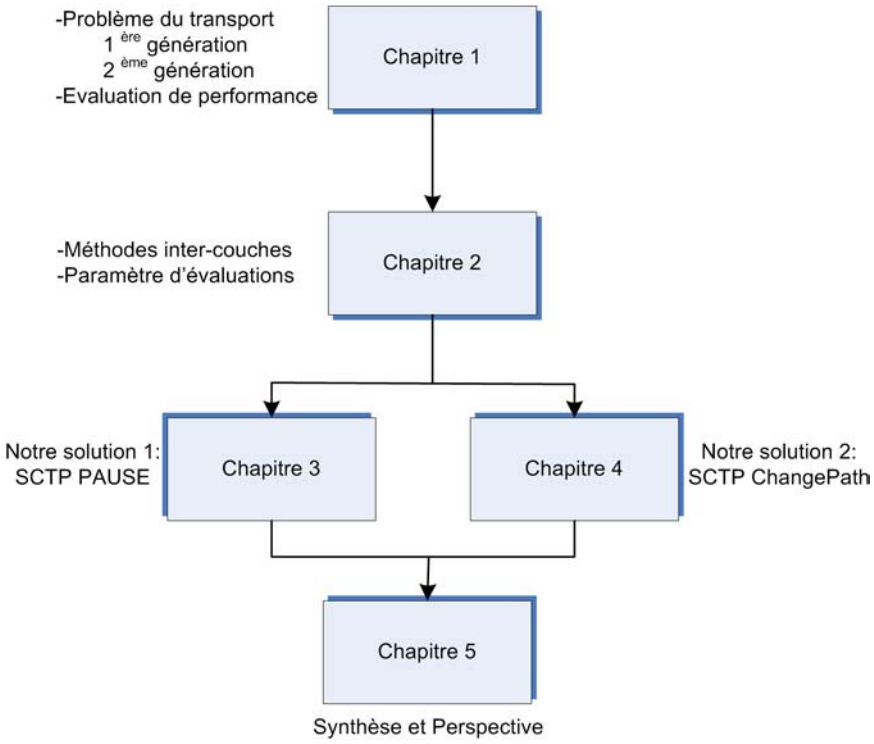


FIG. 1 – Organisation de la thèse

Le premier chapitre analyse la problématique de bonnes performances des protocoles de transport fiables dans les réseaux ad hoc. L'analyse menée à travers les travaux publiés dans le domaine, et nos expérimentations par simulation, conduisent à mettre en avant le protocole de routage comme un facteur prépondérant de performance du transfert de données de bout en bout.

Le deuxième chapitre s'intéresse aux moyens de prendre en compte l'influence du routage sur le transport pour améliorer les performances. Il présente les méthodes de conception de réseau prenant en compte les communications inter-couches : les approches cross layer. Une synthèse des différents travaux de la littérature ainsi qu'une classification des optimisations selon leur niveau de conception est effectuée. Nous décrivons également plus avant le protocole SCTP et définissons les paramètres nécessaires aux évaluations des optimisations présentées dans les chapitres suivants.

Le chapitre 3 propose et évalue l'optimisation que nous nommons SCTP PAUSE. Nous appliquons une méthode de communication inter-couches localisée à la source où le niveau transport utilise des informations du niveau réseau. Il s'agit d'adapter la transmission des informations à la mobilité du réseau, en bloquant l'émission et le processus de congestion en cas de mobilité détectée par le niveau réseau.

Le chapitre 4 propose et évalue une deuxième optimisation qui repose sur une augmentation de la robustesse du réseau, grâce à l'établissement de plusieurs routes pour un même flux d'information. Cette méthode, SCTP ChangePath (ou SCTP CP), est mise en oeuvre par le protocole SCTP sur un réseau AODV, et, une communication inter-couches répartie sur les noeuds du réseau ad hoc.

Pour terminer nous exposons la synthèse et les perspectives d'utilisation de nos résultats dans **le chapitre 5**. En s'appuyant sur les résultats des chapitres précédents nous proposons une architecture adaptative par communication inter-couches pour optimiser SCTP en réseaux ad hoc.

Chapitre 1

Problème du transport dans les réseaux ad hoc

Sommaire

Dans ce premier chapitre, nous nous intéressons à la problématique des bonnes performances des protocoles de transport fiables dans les réseaux ad hoc. Dans un premier temps, nous présentons les principales caractéristiques des protocoles de transport standards de l'IETF, puis celles de protocoles issus de la recherche et dont la conception a été adaptée à un environnement sans fil, puis plus récemment, à un environnement mobile.

Dans un second temps, nous illustrons notre sujet d'étude en simulant le fonctionnement du protocole TCP en réseaux ad hoc. Plusieurs algorithmes de contrôle de congestion sont évalués et deux protocoles de routage ad hoc sont pris en compte, à savoir DSR et AODV.

1.1 Les protocoles transport IETF

Avant de présenter les travaux de recherche menés sur le transport en environnement sans fil et plus récemment ceux menés sur les réseaux sans fil avec capacité de routage et mobilité (les réseaux MANETs : Mobile Ad hoc Networks), nous présentons l'état de la standardisation dans le domaine.

L'IETF, organisme chargé de standardiser les protocoles du monde internet, qui sont actuellement adoptés par la majorité des systèmes de communication, propose quatre protocoles de niveau transport : TCP (Transmission Control Protocol) et UDP (User Datagram Protocol), les deux protocoles pères de l'internet, ainsi que DCCP (Datagram Congestion Control Protocol) et SCTP (Stream Control Transmission Protocol), les deux protocoles fils de la convergence. Les deux premiers ont été spécifiés pour un monde de données alors que les deux derniers, plus récents, prennent en compte les besoins spécifiques au transfert d'information multimédia.

1.1.1 TCP (Transmission Control Protocol)

TCP a été défini pour fournir un service de transfert de données fiable entre deux ordinateurs "maîtres" raccordés par un réseau de type "paquets commutés" utilisant le protocole IP (Internet Protocol), et par tout système résultant de l'interconnexion de ce type de réseaux. La fiabilité du transfert est obtenue par différents mécanismes tels que l'établissement de connexion, la gestion du timers de retransmissions ou encore le contrôle de la fenêtre de retransmissions. La spécification initiale, qui a été définie dans le Request For Comments RFC 793 [Post81] de 1981, a fait l'objet depuis lors de très nombreux travaux. Le RFC 4614 [DBEB06], publié en 2006, liste de nombreux RFC qui améliorent ou corrigent la spécification initiale. Parmi ceux-ci, nous détaillerons plus précisément ceux relatifs aux options de TCP, nommées TCP Sack et TCP Westwood.

1.1.2 UDP (User Datagram Protocol)

UDP a été spécifié dans le RFC 768 [Post80] pour le même type d'environnement que TCP, un transfert de données sur réseau IP, mais avec un objectif de service de transfert minimal. Il n'y a aucune garantie sur la correction de la délivrance des informations. C'est pour cela qu'il est souvent décrit comme étant non-fiable, même si, pour un paquet UDP, l'exactitude du contenu des données est assurée grâce à une somme de contrôle (checksum). En cas de mauvais contrôle le récepteur détruit le paquet. Une version de UDP, UDP lite [LDPJ04] permet de ne pas détruire systématiquement le datagramme lorsque celui-ci a été endommagé. Dans cette version, le paquet est divisé en deux parties, une partie couverte par le checksum et l'autre non. UDP, de même que UDP lite, n'établit aucune connexion et il ne fait aucun contrôle sur le séquencement des messages reçus.

1.1.3 DCCP (Datagram Congestion Control Protocol)

Récemment standardisé par l'IETF (en 2006) [KoHF06], ce protocole fournit un service de transfert de données qui est adapté à des informations ne nécessitant pas de garanties de transfert aussi rigoureuses que celles de TCP, tels la voix sur IP ou encore les jeux en lignes [FlHK06]. L'objectif est de transférer les données en temps réel plutôt que dans le bon ordre. Contrairement à TCP, le protocole DCCP ne se préoccupe pas de retransmettre les informations perdues. Il est cependant capable de détecter la perte d'informations et ainsi de prévenir l'application.

Ses principaux mécanismes sont l'établissement de connexion, le renvoi d'acquittement par messages (TCP acquitte par segments), ainsi que des contrôles de congestions qui sont négociables à l'initialisation du transfert. La négociation se fait pour chaque sens du transfert, la connexion étant unidirectionnelle, contrairement à TCP.

Deux types de contrôle (CCID : Control Congestion Identifier) sont actuellement standardisés.

- Le premier, similaire à celui de TCP, est spécifié dans le RFC 4341 [FlKo06]. L'émetteur gère une fenêtre d'émission et transmet tant que sa fenêtre n'est pas pleine. Les messages sont explicitement acquittés par le récepteur, la liste des messages reçus est indiquée dans le message d'acquittement. La congestion du réseau est détectée par le mécanisme d'acquittement ou par le mécanisme ECN (Explicit Congestion Notification spécifié en [RaFB01]). ECN s'appuie sur un marquage des paquets IP par les noeuds de réseau engorgé, puis une notification de cet engorgement via le marquage d'un message TCP qui est émis par le destinataire du paquet reçu marqué vers l'origine de ce dernier. En cas de congestion de réseau détectée, l'émetteur diminue par deux sa fenêtre d'émission. Il ralentit ainsi rapidement son débit.
- Le second mécanisme de contrôle, défini par le RFC 4342 [FlKP06], prévoit une adaptation de débit moins brusque que précédemment. Le contrôle de congestion est effectué à partir d'une équation définie pour diminuer graduellement le débit en cas de congestion : c'est le mécanisme TCP-Friendly Rate Control (TFRC). L'émetteur émet selon un débit qui est mis à jour en fonction de l'estimation par le receveur des paquets perdus, ou marqués via ECN, par le réseau.

1.1.4 SCTP (Stream Control Transmission Protocol)

Initialement prévu pour transmettre des messages de signalisation téléphonique, SCTP a été standardisé en 2000 [SXMS00] et amélioré en 2007 dans le RFC 4960 [Stew07], conformément aux problèmes recensés dans le document RFC 4460 [SAPC06].

SCTP est globalement semblable à TCP ; les principaux points de divergence sont une orientation message, une phase d'établissement de connexion davantage sécurisée, un contrôle d'erreur qui peut être optionnel pour certaines données ; le positionnement d'un flag dans une donnée indique au récepteur que celle-ci n'est pas soumise au contrôle et qu'elle doit être délivrée à la couche supérieure (unordered service).

La principale originalité du protocole est la notion d'association. Une association est assimilable à une "méta connexion" permettant de gérer un transfert d'information composé de plusieurs flux (multistreaming). De plus, une association peut être en relation avec différentes adresses IP, ce qui lui permet d'être multi domiciliée dans plusieurs réseaux IP : c'est le **multihoming**.

L'intérêt majeur du multihoming auquel nous allons nous intéresser dans ce document est sa capacité à augmenter la robustesse du transfert de données face aux ruptures de réseaux. En cas de rupture d'un chemin primaire, référencé dans l'association par son adresse IP, le flux d'informations peut être basculé sur un chemin secondaire, également référencé dans l'association par son adresse IP. Nous verrons de façon détaillée ce fonctionnement dans le chapitre 4 et comment nous nous proposons d'utiliser le multihoming en réseaux ad hoc.

Les principales caractéristiques des protocoles transport de l'IETF sont indiquées dans le tableau 1.1.

Les caractéristiques	TCP	UDP	DCCP	SCTP
Request for comment (RFC)	RFC 793 [Post81]	RFC 768 [Post80]	RFC 4340 [KoHF06]	RFC 4960 [Stew07]
Ordonnancement	Numérotation octets pour assurer l'ordre	Non assuré	Non assuré	Numérotation message pour un transfert ordonné, ou non
Contrôle de congestion	Assuré, par mécanismes fixés (Additive Increase and Multiplicative Decrease : AIMD)	Non assuré	Assuré, par choix de mécanismes CCID(Control congestion Identifier)	Assuré, par mécanismes fixés (AIMD)
Etablissement de connexion	En 3 phases	Pas de connexion	En 3 phases	En 4 phases
Service de transfert fiable	Acquittement octets, retransmission	Pas d'assurance de livraison	Pas d'assurance de livraison mais détection de non livraison par acquittement de messages	Acquittement message, re-transmission sélective
Multihoming	Pas supporté	Pas supporté	Pas supporté	Supporté
Multistreaming	Pas supporté	Pas supporté	Pas supporté	Supporté

TAB. 1.1 – Caractéristiques des protocoles de transport IETF

1.2 Exemples de protocoles transport de 1ère génération pour les réseaux sans fil à un saut

Au milieu des années 90 de nombreux travaux ont été menés pour transmettre des données dans les réseaux téléphoniques sans fil puis dans les réseaux locaux WiFi en 802.11. Que ce soit dans les réseaux locaux informatiques ou dans les réseaux téléphoniques longue distance, les architectures sont constituées d'une infrastructure de points d'accès, appelées encore stations de base. Le transfert de données entre deux équipements s'effectue par l'intermédiaire de ces points. Les éléments qui accèdent directement à un point d'accès forment une cellule, les cellules sont reliées entre elles par une structure filaire connectant les points d'accès. Ces architectures avec infrastructure sont également appelées architectures à un saut, par opposition aux architectures sans infrastructure et multisauts que nous présentons dans la prochaine section.

Nous indiquons à la suite quels sont les facteurs spécifiques à l'environnement sans fil qui ont conduit à des améliorations protocolaires, puis, nous présentons une synthèse des principes proposés.

1.2.1 Les problèmes de TCP dans les réseaux sans fil

Les études menées sur TCP en environnement sans fil, dont on trouvera une synthèse en [Pent00], indiquent une dégradation de performances, par rapport à son fonctionnement en environnement filaire. La dégradation de performance s'explique par différents facteurs ([NJMS01]) :

- **Limitation de la bande passante**
  Contrairement à un réseau filaire, la largeur de bande disponible est réduite : des débits de 11 à 50 Mbps pour les réseaux locaux sans fil de type 802.11 à comparer avec des débits de 100 Mbps à 1 Gbps pour les réseaux locaux filaires type IEEE 802.3 (Ethernet), et ceux de l'ordre du 10 Gbps pour les fibres optiques des réseaux longue distance.
- **Taux d'Erreur Lien**
  Le taux d'erreurs bits (Bit Error Rate, BER) sur des liens sans fil est de l'ordre de 10^{-2} jusqu'à 10^{-3} alors que celui sur des liens filaires est de l'ordre de 10^{-6} jusqu'à 10^{-12}. Ces erreurs provoquent des destructions de paquets qui empêchent l'émetteur de recevoir un acquittement avant l'expiration du timer de retransmission ; celui-ci doit alors, conformément aux procédures de contrôle de congestion, diminuer sa fenêtre et retransmettre. TCP assimile l'altération temporaire à une congestion. Les altérations sur la liaison sans fil surviennent par bouffées, burst, qui peuvent provoquer plusieurs pertes de paquet sur une période de temps inférieure au temps pour émettre une donnée et recevoir l'acquittement (temps aller/retour). Il est alors intéressant d'utiliser un mécanisme de retransmission qui soit capable de retransmettre plusieurs paquets par fenêtre (comme le montre [BPSK97] avec l'option SACK). Notons que dans le cas de réseaux sans fil utilisés en accès d'un réseau longue distance filaire, ce qui est un cas d'utilisation très répandu, la perte de paquet sur la liaison sans fil au dernier saut provoque une retransmission de bout en bout, génératrice de trafic additionnel sur le lien sans fil, et donc d'encombrement et source d'erreurs additionnelles.
- **Mobilité de l'utilisateur**
  Quand un mobile se déplace d'une cellule dans une autre, un ensemble d'informations

doit être transféré de son ancienne station de base à sa nouvelle station. Cette procédure de réassociation, nommée Handover, qui dépend fortement de la technologie de transmission utilisée, peut provoquer de courtes interruptions de la connectivité. Le protocole transport assimile ces périodes de déconnexion à des phases de congestion et déclenche inutilement la procédure de contrôle de congestion qui diminue le débit du flux. L'interruption momentanée de la communication peut également être liée à : un mobile qui sort de sa zone de portée de transmission ou bien des signaux radio bloqués par des bâtiments ou autres obstacles.

1.2.2 Les solutions

Les solutions proposées pour améliorer les performances du protocole de transport, (dont une vue globale est présentée en [Elaa02]) peuvent être regroupées en trois catégories : découpage de la connexion, utilisation de mandataires de service transport, contrôle de niveau liaison.

1. Découpage de connexion (Split Connection) L'idée principale de cette approche est de séparer les problèmes de liaison sans fil des problèmes classiques des réseaux traditionnels, et ainsi d'associer des mécanismes spécifiques à chaque technologie de transfert. Des exemples de propositions conformes à cette approche sont :
   - **I-TCP** (Indirect-TCP) proposé par Bakre et Badrinath [BaBa95].
     I-TCP est adapté à un schéma de mobilité de type terminal fixe-terminal mobile. Quand un utilisateur mobile souhaite communiquer avec un utilisateur fixe, le protocole I-TCP découpe la connexion TCP traditionnelle en deux tronçons, en introduisant un point intermédiaire : la station de base. Le protocole I-TCP résidant sur le mobile émet une demande d'établissement de connexion TCP, pour l'utilisateur fixe, à la station de base. Concernant l'acquittement, un paquet transmis du poste fixe vers le mobile est d'abord acquitté sur le premier tronçon de connexion par la station de base, puis est transmis sur le second tronçon.
     Avec cette solution, les acquittements n'ont plus de signification de bout en bout, chaque connexion gérant ses propres acquittements.
   - **M-TCP** (Mobile-TCP) proposé par Brown et Singh [BrSi97].
     De même que I-TCP, M-TCP découpe la connexion en deux tronçons. Sa principale innovation est l'introduction d'un nouvel élément : le superviseur. Son rôle est de servir de passerelle vers le réseau filaire fixe à plusieurs stations de bases. Le serveur présente l'avantage d'alléger la charge des stations de base et de gérer les acquittements en conservant la sémantique de bout en bout de TCP.
     A la différence de I-TCP, les acquittements émis par le serveur ne sont envoyés que lorsque les messages correspondants ont été bien reçus par le mobile. Il faut d'abord que le mobile ait acquitté le paquet sur le dernier tronçon de la connexion pour que l'acquittement soit transmis sur le premier tronçon.
2. Mandataire de service transport (proxy)
   Le protocole **Snoop** proposé par Balakrishnan et al [BaSK95] se propose d'améliorer les performances de TCP de façon complètement transparente ; aucun logiciel additionnel n'est requis par les éléments mobiles. Snoop utilise une méthode d'espionnage qui examine tous les paquets de la connexion. Grâce à cet espionnage la station de base va se comporter en mandataire de la source. En cas d'erreur dans le sens fixe/mobile, c'est la station de base qui retransmet les paquets perdus sur le réseau sans fil. De même, lorsque l'émission des données par le mobile vers la

station fixe est perturbée sur la liaison sans fil, c'est la station de base qui se charge d'émettre des acquittements sélectifs.

3. Contrôle de niveau liaison

   Ces approches mettent en oeuvre explicitement l'idée sous-jacente de SNOOP présentée ci-dessus : il s'agit d'effectuer un contrôle d'erreur de niveau liaison de données sur le lien sans fil, de façon à éviter le contrôle de bout en bout.

   Le protocole **Airmail** (Asymmetric reliable mobile access in link-layer) proposé par Ayanoglu et al. [APLS95] utilise des codes de correction d'erreurs (Forward Error Correction : FEC) et l'émission automatique de messages de contrôle (Automatic Reply Control) de façon à fiabiliser la transmission du lien sans fil, et donc cacher à TCP les erreurs de transmission, pour éviter à celui-ci de diminuer son débit.

   Le protocole prend en compte l'asymétrie du lien entre le mobile et la station de base ; il propose des mécanismes adaptés à chaque sens de transmission. Ainsi, alors que la station de base émet périodiquement son statut, le mobile, lui, afin d'économiser sa batterie, ne transmet que sur évènement.

Les principales caractéristiques des protocoles transport de 1ère génération pour réseaux sans fil sont synthétisées dans le tableau 1.2.

Caractéristiques	I-TCP	M-TCP	Snoop	Airmail
BER élevé	✔		✔	✔
Burst d'erreurs			✔	✔
Handoff	✔	✔	✔	✔
Longue durée de déconnexion	✔	✔		
Déconnexion fréquente	✔	✔		
Sémantique de bout à bout (ACK-TCP)		✔	✔	✔

TAB. 1.2 – Caractéristiques d'environnement des protocoles de transport de 1ère génération

1.3 Exemples de protocoles transport de 2ème génération pour les réseaux sans fil multisauts

La deuxième génération de travaux sur les problèmes de performance du transport fiable s'est intéressée aux environnements de transmission filaires multisauts. Ce type de transmission est mis en oeuvre par des architectures sans infrastructure où la communication s'effectue directement entre les équipements sans passer par une station de base.

Cette communication est également appelée communication **ad hoc**. La transmission entre les noeuds adjacents est gérée par un protocole d'accès. Lorsque le récepteur n'est pas à portée directe de transmission, des équipements intermédiaires situés sur le chemin entre la source et la destination vont servir de relais. Le réseau de transmission est alors constitué non plus d'un lien sans fil, mais de plusieurs liens : il est multisauts. Les réseaux ad hoc sont des réseaux multisauts qui ont comme caractéristique supplémentaire d'être auto-organisés. Dans ces réseaux il n'y a pas d'administration centralisée, tout équipement est libre de quitter ou de rejoindre le réseau à tout moment. Parmi les réseaux ad hoc se trouvent les réseaux ad hoc mobiles appelés réseaux MANETs, du nom du groupe éponyme de l'IETF (Mobile Ad hoc Networks). Les réseaux MANETs sont constitués d'éléments mobiles qui forment des topologies dynamiques.

Dans la suite de ce document nous appelons indifféremment réseaux ad hoc, ou réseaux MANETs, les réseaux sans fil multisauts, auto-organisés et mobiles.

1.3.1 Problèmes de TCP dans les réseaux ad hoc

Les principales raisons qui entraînent une dégradation de performance dans les réseaux ad hoc proviennent, comme précédemment, de la qualité du lien sans fil, mais également, de la qualité du chemin. Plus précisément, une raison à la dégradation de performance provient du comportement erroné de TCP que nous avons mentionné dans la section précédente : le protocole déduit à tort un état de congestion du réseau et diminue inutilement son débit d'émission. Cette mauvaise interprétation de l'état du réseau est provoquée par des pertes de données qui ne sont pas dues à de la congestion.

Pertes de données dans les réseaux ad hoc

Les réseaux ad hoc présentent un taux d'erreurs plus important que celui des réseaux traditionnels en raison du taux d'erreurs (BER) des liaisons sans fil, mais également des pertes liées aux collisions de la couche d'accès au médium, ou encore de problèmes des noeuds cachés [MaMu04].

Comme dans les réseaux à un saut, la perte de données est synonyme d'une perte de connectivité. Celle-ci, outre les raisons de qualité mentionnées ci-dessus, est également due à la mobilité des utilisateurs. Dans un réseau ad hoc, comme les postes utilisateurs sont utilisés par leurs voisins pour relayer l'information, un déplacement de leur part peut occasionner une rupture du chemin de relayage.

Coupures de chemin dans les réseaux ad hoc mobiles

La mobilité des noeuds du réseau provoque des modifications de topologie qui sont géré par la couche réseau. Le protocole de routage est en charge du rétablissement du chemin en cas de rupture du chemin courant ; une fois la rupture de chemin détectée, il exécute un processus de rétablissement dont le délai est non négligeable.

Le temps de rétablissement du chemin est fonction du nombre de noeuds dans le réseau, des types de transmission des noeuds, de la topologie courante du réseau, de la bande passante disponible et de la nature du protocole de routage. Si le temps de rétablissement est plus grand que la valeur du timer de retransmission, l'émetteur suppose alors que le réseau est congestionné et retransmet les paquets perdus. Ces retransmissions peuvent, si le chemin n'a toujours pas été rétabli, mener à un gaspillage de bande passante et de batterie. De plus de par le contrôle de congestion, lorsqu'un nouveau chemin aura été mis en place, le débit sera inutilement faible.

Impact de la longueur du chemin

La possibilité d'une coupure de chemin augmente avec la longueur de chemin : étant donnée une probabilité de coupure de lien (pl), la probabilité d'une coupure de chemin (Pb) pour un chemin de la longueur k peut être simplement obtenue par $\text{Pb} = 1-(1-pl)^k$.

Lorsque la longueur de chemin augmente, le débit de TCP se dégrade comme nous le montrons sur la Figure 1.1. La Figure 1.2 montre la variation de la probabilité Pb avec la longueur de chemin pour $pl = 0.1$.

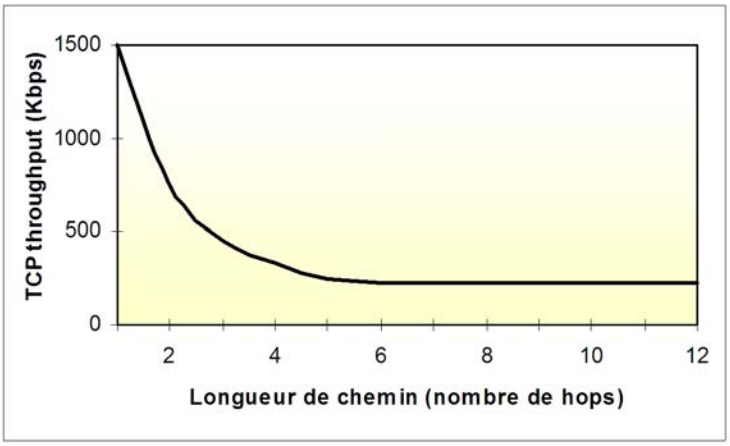


FIG. 1.1 – Variation de throughput de TCP avec la longueur de chemin

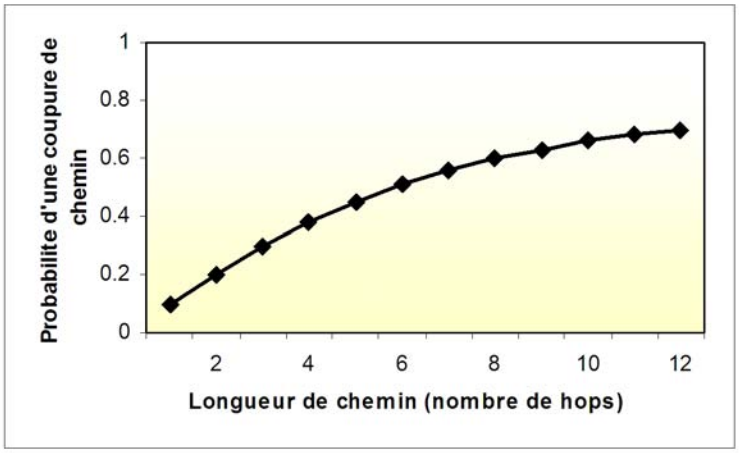


FIG. 1.2 – Variation de Pb avec la longueur de chemin (pl = 0.1)

Comportement asymétrique de liens et de chemins

Les propriétés du canal radio utilisé dans les réseaux ad hoc peuvent mener à l'existence de liens entre noeuds qui sont asymétriques. Dans ce cas, un paquet est correctement délivré à un noeud dans le sens aller, mais l'acquittement, lui, ne peut pas être reçu dans le sens inverse. Cette impossibilité de réception peut n'être que temporaire, le temps que le canal radio devienne à nouveau bidirectionnel.

Le TCP traditionnel se fonde sur un acquittement (ACK) de bout en bout pour assurer la fiabilité. Dans les réseaux ad hoc utilisant le protocole d'accès IEEE 802.11, chaque paquet ACK peut nécessiter un échange de trames RTS-CTS (Request To Send - Clear To Send) qui peut mener à une consommation significative de largeur de bande sur le chemin inverse, qui peut ou pas, être symétrique au chemin aller. Si le chemin inverse est le symétrique du chemin aller, ceci entraînera une réduction du débit sur le chemin aller. Ceci peut être évité par un routage qui émettrait les acquittements sur un chemin entièrement différent, ou partiellement différent, de celui du chemin aller inverse, même si les liens sont symétriques. Notons qu'une coupure de chemin sur un chemin d'acquittement affectera inutilement le protocole.

Pic de signalisation et recherche de route

La construction des tables de routage par la couche réseau nécessite l'envoi de nombreux paquets de signalisation. Dans le cas des protocoles réactifs, c'est à dire qui établissent leur route quand des données sont prêtes à être envoyées, le médium va leur être quasiment dédié durant une période assez longue. Le fort coût d'accès au médium en temps et en espace fait que cette période peut-être suffisamment longue pour produire des expirations du timer de retransmission sur certaines connexions. Il n'y a pas réellement de congestion dans le réseau mais seulement un pic de charge de signalisation. Le pic de signalisation peut se produire à l'établissement du chemin mais surtout en cas de rupture du chemin, lorsque le protocole de routage cherche à rétablir une route.

Contrôle de flux par fenêtres glissantes

TCP emploie une fenêtre coulissante pour contrôler le flux et éviter des engorgements du récepteur ; les paquets sont transmis tant que la fenêtre d'émission n'est pas pleine et les acquittements reçus permettent, en vidant la fenêtre, de transmettre de nouveaux paquets. Ceci évite l'utilisation de différents temporisateurs à grain fin pour la transmission de chaque flux TCP. Une telle conception a été choisie pour améliorer le passage à l'échelle du protocole dans les réseaux Internet où des millions de connexions de TCP peuvent être établies avec quelques serveurs fortement chargés. Dans les réseaux ad hoc, le mécanisme de fenêtre coulissante peut, lorsque le protocole d'accès n'est pas équitable, ne pas être adapté. Par exemple, les protocoles d'accès tels que le protocole de CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) montrent une inéquité d'accès à court terme, car un noeud qui a capturé le canal a une probabilité plus élevée de capturer à nouveau le canal. D'après [MaMu04], cette injustice peut mener à la réception d'un groupe de paquets d'acquittement, qui provoquera un effet de burst sur l'émission de traffic.

1.3.2 Solutions de TCP pour les réseaux ad hoc

Nous présentons les améliorations amenées par les chercheurs au protocole TCP en deux catégories : celles qui utilisent des informations sur l'état du réseau grâce à un "retour" et celles qui n'en utilisent pas.

1.3.2.1 TCP avec retour

TCP-Feedback

C'est une modification du protocole TCP proposée par Chandran et al [CRVP98] dont le fonctionnement est le suivant. L'émetteur a deux états : il est actif ou en arrêt momentané (snooze). Dans l'état actif, la couche de transport est contrôlée par le protocole TCP standard. Dans l'état snooze, l'émetteur reçoit un retour de l'état du réseau (Feedback) via l'échange de messages spécifiques. Ces messages annoncent, soit un échec de routage (RFN, Route Failure Notification), soit l'établissement d'un nouveau chemin (RRN, Route Re-establishment Notification). Lorsqu'il n'y a pas de route disponible, l'émetteur entre dans l'état snooze, il bloque alors son fonctionnement et arrête l'incrémentation de son timer. Quand l'émetteur reçoit un RRN, c'est la fin du blocage, il reprend son fonctionnement TCP.

TCP-ELFN

Hollande et Vaidya ont proposé également d'utiliser un retour de l'état de la transmission au niveau de TCP avec "Explicit Link Failure Notification (ELFN)" [HoVa99], [HoVa02]. Cette fois-ci, l'état n'est pas considéré au niveau réseau mais, au niveau liaison. En cas d'un échec de transmission, l'émetteur entre dans un mode appelé par les auteurs standby, qui est l'équivalent de l'état snooze de TCP-F. Le fonctionnement dans cet état est quasi similaire à celui de TCP-F, excepté le traitement de messages "Explicit Link Failure Notification" (au lieu de messages RFN), et l'envoi de sondes. Dans le mode standby, l'émetteur envoie périodiquement un petit paquet pour sonder le réseau et voir si une route a été établie. S'il y a une nouvelle route, l'émetteur quitte le mode standby, reconstitue son timer de retransmission et reprend un fonctionnement TCP standard.

TCP-Bus

TCP "BUffering capability and Sequence information" est proposé par Kim et al. [KiTC01]. L'idée est d'une part, d'avoir un retour du réseau grâce aux noeuds intermédiaires et d'autre part, de mémoriser les données dans ces noeuds en cas de rupture du chemin. Deux messages de contrôle sont utilisés pour informer la source d'une rupture de route ainsi que du rétablissement de route. Ces messages sont appelés "Explicit Route Disconnection Notification (ERDN)" et "Explicit Route Successful Notification (ERSN)". Lorsque l'émetteur reçoit un message ERDN, la source arrête son émission et les paquets sont mémorisés le long du chemin par les noeuds intermédiaires. La mémorisation évite les pertes de données, le temps que le chemin soit rétabli.

ATCP

ATCP (Ad hoc TCP), proposé par Liu et Singh [LiSi01], ajoute une couche intermédiaire entre la couche IP et la couche TCP. La fonction ATCP nommé "couche" par les auteurs est présente uniquement sur l'émetteur de TCP. Cette couche contrôle l'état de TCP et l'état du réseau qui est connu par les messages d'ECN (Explicit Congestion Notification) [RaFl99] [RaFB01] et les messages ICMP (Internet Control Message Protocol).

ATCP définit quatre états : “normal”, “congestionné”, “perte”, et “déconnexion”. En phase d'établissement de connexion TCP, l'émetteur est en état “normal”. Lorsque le réseau est congestionné, c'est à dire que ATCP a reçu un ACK ou un paquet de données avec le drapeau ECN, ATCP entre dans l'état “congestionné” et retransmet rapidement les données perdues de TCP. La réception de trois ACKs identiques provoque l'entrée dans l'état “perte”. Dans l'état “déconnexion”, les paquets n'atteignent pas leur destination ce qui provoque l'émission de messages ICMP vers la source. Sur réception d'un message ICMP, ATCP diminue la fenêtre d'émission à un segment.

1.3.2.2 TCP sans retour

Fixed RTO

Dans la solution TCP à timer fixe, “Fixed RTO” [DyBo01], aucun retour d'état n'est utilisé. Après une expiration de timer, si l'émetteur ne reçoit pas d'acquittement avant que le timer n'expire une deuxième fois, l'émetteur déduit que le réseau n'est pas congestionné mais qu'il y a un problème de routage ; il retransmet le paquet perdu mais n'augmente pas sa valeur de timer (RTO). Le RTO reste fixe jusqu'au rétablissement de la route.

TCP DOOR

TCP DOOR (Detection of Out-Of-Order and Response) est proposé par Wang et Zhang [WaZh02]. L'objectif est d'améliorer les performances sans utiliser de retour réseaux, mais en étant de capable de détecter et de réagir à des changements de routes. Le protocole assimile les changements de routes à des évènements de hors séquences (out-of-order : OOO). Ces hors séquences peuvent être fréquents dans les réseaux MANETs en raison de la mobilité des noeuds et des chemins. La détection du hors séquence est effectuée, à l'émetteur, sur les acquittements et, au récepteur, sur les données, grâce à une nouvelle numérotation (dans le TCP de base le numéro n'est pas incrémenté lors des retransmissions). Lorsque le récepteur détecte, grâce au nouveau numéro TPSN (TCP Packet Sequence Number), un hors séquence sur les données reçues, il renvoie l'information à l'émetteur en positionnant un flag dans l'acquittement correspondant. De même, l'émetteur vérifie le flux retour grâce à une numérotation spécifique dans l'acquittement. En cas de détection de hors séquence, l'émetteur dispose de deux mécanismes, l'un permet, comme dans TCP-Feedback, de bloquer le contrôle de congestion, car il y a une perte de route momentanée, l'autre d'accélérer momentanément le recouvrement d'erreur. Si le contrôle de congestion a été récemment effectué et qu'un évènement de hors séquence est détecté, l'émetteur déduit qu'il y avait une coupure momentanée de la connexion et non pas une congestion.

1.4 Performances en réseau ad hoc : étude par la simulation

Dans la littérature, les propositions de protocoles de transport pour réseaux ad hoc sont évaluées et comparées généralement au protocole TCP et à une optimisation du même ordre, afin de montrer qu'elles améliorent les performances dans un contexte donné. Notre objectif étant d'améliorer des performances, le problème est alors d'évaluer l'existant en ayant un cadre d'évaluation commun à différentes propositions, afin de déterminer quelle est l'optimisation adaptée à un contexte.

Pour aborder ce problème nous évaluons par simulations le protocole TCP, avec un outil, NS2, et des paramètres généralement utilisés par la communauté. Notre objectif de simulation est de répondre à deux questions.

- La première porte sur la notion même de l'optimisation de niveau transport dans les réseaux ad hoc. Nous souhaitons déterminer quel est l'intérêt d'optimiser le niveau transport par rapport au niveau réseau.
- La deuxième question aborde la notion de domaine d'optimisation. Existe-t-il des mécanismes qui soient performants quelles que soient les conditions réseau ?

La modélisation du système que nous avons évalué est composée d'une fonction de transport, d'une fonction de routage, et d'un environnement variable qui est représenté par un modèle de simulation. Nous présentons à la suite chacun de ces éléments.

1.4.1 Modélisation du transport

Rappelons brièvement le principe de base des algorithmes de congestion **slow start et congestion avoidance** dont les principes sont définis dans le RFC 2581 [StAP99].

Concernant le contrôle de congestion, il s'agit d'adapter le taux d'envoi des segments en fonction du taux d'arrivée des acquittements (ACKs). L'objectif est d'arriver à un niveau d'autosynchronisation où la cadence d'émission (réglée par la taille de la fenêtre) est égale à la cadence où les paquets quittent le réseau (réception de ACK). Les ACK reçus sont utilisés pour augmenter la taille de la fenêtre de congestion (congestion window : CWND), et donc le débit d'émission, de façon à exploiter au mieux la bande passante disponible. L'algorithme slow start augmente de façon exponentielle la taille de la fenêtre pour, à l'initialisation, remplir rapidement le réseau de paquets en transit, alors que l'accroissement devient ensuite linéaire, en phase de congestion avoidance, pour s'adapter à la cadence de sortie de réseau des paquets. Lorsqu'une congestion est détectée, la taille de la fenêtre est diminuée.

La valeur de la fenêtre, sa fonction de croissance avec les seuils pour passer d'un accroissement exponentiel à un accroissement linéaire, font l'objet de plusieurs variantes de TCP. Les différences entre les versions de TCP viennent également des algorithmes appliqués par TCP pour assurer la fiabilité du transfert. Deux modes de recouvrement sont spécifiés dans les standards :

- Le **fast retransmit** : lorsque l'émetteur progresse dans sa transmission de données, mais qu'il n'y a pas de progression dans les acquittements qu'il reçoit, il suppose qu'il y a eu un problème et déclenche la retransmission d'un segment, l'idée étant de ne pas attendre que le temporisateur de retransmission expire pour retransmettre. Le segment qui semble manquer est réémis dès réception de trois ACKs dupliqués

(DUPACKs).

– Le **fast recovery** consiste à essayer de continuer à transmettre des segments grâce à une diminution adéquate de la fenêtre (afin de ne pas interrompre l'horloge des ACKs qui la font glisser), puis à passer en congestion avoidance.

Les différentes versions de TCP proposent des algorithmes spécifiques pour augmenter la taille de la fenêtre et gérer les retransmissions. L'évolution de la fenêtre en cas de perte de donnée est illustrée sur la Figure 1.3. Nous allons nous intéresser aux versions du protocole TCP NewReno, TCP Sack et TCP Westwood qui d'après leurs caractéristiques sont adaptées à un environnement sans fil. Les premières permettent de gérer des pertes multiples, la dernière d'adapter le débit au plus près de celui de la liaison. Nous évaluerons par simulation l'intérêt d'utiliser ces options en les comparant avec la version de base TCP Reno.

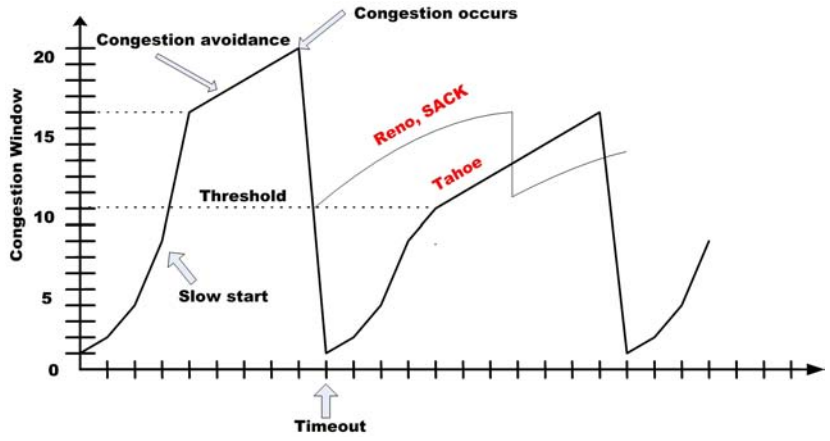


FIG. 1.3 – Évolution de la fenêtre de congestion avec retransmission rapide

1.4.1.1 TCP Reno/NewReno

La version de TCP dite Reno, proposée par Jacobson dans [Jaco88] et [Jaco90], a introduit le fast retransmit (dans la version historique, dite Tahoe [Jaco88], ce n'était qu'après l'expiration du timer que la perte était détectée). La perte de paquet, signe d'une congestion, déclenche la retransmission des paquets perdus. La retransmission s'effectue avec une fenêtre de congestion réduite à un segment, même si la perte s'est produite avec une taille de fenêtre importante. Ceci n'est pas efficace puisque la congestion s'est produite pour une valeur de fenêtre qui était supérieure à un, il n'est pas utile de la réduire autant. La proposition de Jacobson est de diminuer la valeur de la fenêtre non pas à un mais à la moitié de la valeur courante qu'avait celle-ci lors de la perte et ensuite d'augmenter sa taille de façon linéaire, pour transmettre de nouvelles données (algorithme de fast recovery).

Si plusieurs études, telle celle de floyd [FaFl96], ont montré que TCP Reno offre de meilleures performances par rapport à Tahoe, en cas de perte d'un seul paquet dans une fenêtre de congestion, en revanche, en cas de pertes multiples, les performances de TCP Reno se dégradent car il ne peut pas éviter les expirations à répétition de timers. La version TCP NewReno spécifiée dans le RFC 3782 [FlHG04] modifie légèrement le fast recovery pour prendre en compte les pertes multiples. Dans TCP Reno le premier acquittement

partiel stoppe la phase de recouvrement rapide alors que pour NewReno l'acquittement partiel est compris comme une indication de perte : l'émetteur retransmet le premier paquet non acquitté.

1.4.1.2 TCP Sack

Le mécanisme des acquittements sélectifs (Sack), RFC 2883 [FMMP00], permet également d'améliorer la retransmission des données en cas d'erreurs multiples. L'idée de Sack est de permettre au TCP récepteur de confirmer non seulement la dernière donnée reçue en séquence, mais également des données reçues hors séquence ; le TCP émetteur peut ainsi en déduire quelles sont les données manquantes, et donc optimiser (c'est-à-dire anticiper) la retransmission de celles-ci. Sack se base sur l'utilisation de deux options de TCP :

1. **Sack-permitted**
   Un TCP mettant en oeuvre Sack et voulant employer ce mécanisme devra inclure cette option dans son segment SYN (synchroniser), d'ouverture de connexion. De cette façon, il annonce à l'autre extrémité de la connexion qu'il supporte cette méthode d'acquittement. Sack ne sera utilisé pour la connexion que si les deux extrémités incluent cette option dans leur SYN.

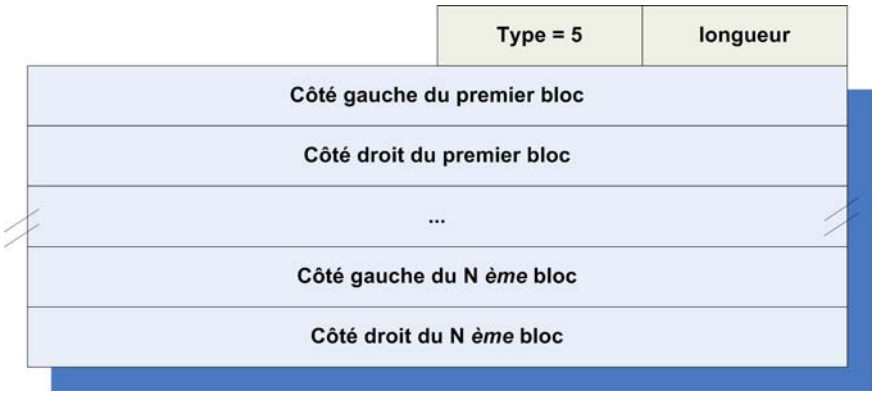


FIG. 1.4 – Format de l'option SACK

2. **Sack**
   Cette option (Figure 1.4), qui peut être vue comme un ensemble de champs supplémentaires au champ numéro d'acquittement, permet au récepteur d'indiquer à l'émetteur quels sont les blocs de données contigus reçus correctement mais hors séquence. Du fait des limitations de la taille des options dans l'en-tête TCP, le récepteur peut annoncer un maximum de quatre blocs de données hors séquence (trois blocs, dans le cas où l'option de l'estampillage temporel, ou timestamps, est utilisée).

1.4.1.3 TCP Westwood

TCP Westwood (TCP-W) n'est pas standardisé, et des travaux le concernant seront trouvés en [MCGL00], [ZPGS01], [GrMa04], [MGFC04]. Il a été spécifié pour des grands réseaux avec des débits élevés, des variations de charges importantes, des pertes d'informations dues à la congestion mais également à des erreurs de transmission. Il utilise l'algorithme de *fast recovery* en s'appuyant sur une estimation de la bande passante pour régler les mouvements de la fenêtre de congestion, et pour adapter les seuils des algorithmes de traitement de congestion (voir Figure 1.5).

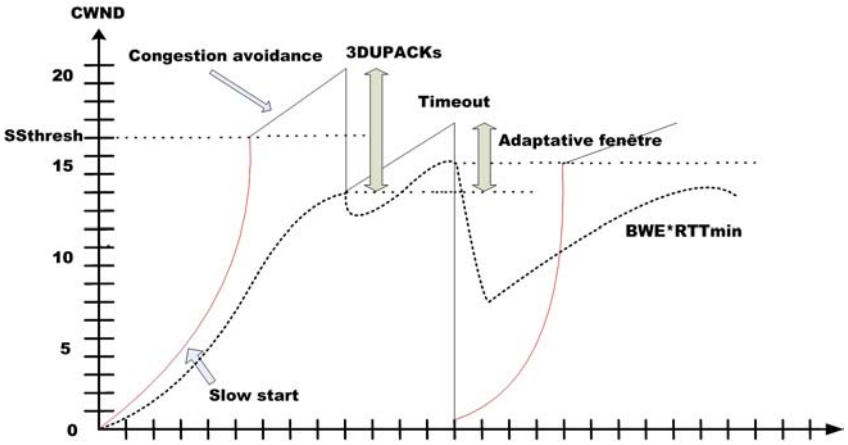


FIG. 1.5 – TCP Westwood

Pour évaluer la bande passante disponible, TCP-W mesure le taux de réception des accusés de réception. Pour cela, l'émetteur doit déduire le nombre de données envoyées vers le récepteur en fonction du temps. Lorsqu'un ACK parvient à la source, cela exprime qu'une certaine quantité d'informations est arrivée à destination. En effectuant une moyenne du nombre de données envoyées en fonction du temps, et s'il n'y a pas de pertes, on obtient une estimation de la bande passante utilisée par la source. Si les paquets transmis par la source sont de tailles variables, et qu'une perte survient et que celle-ci est annoncée par la présence de duplicate ACKs (DUPACKs), il n'est pas possible de savoir quel paquet s'est perdu. Il est alors utile d'effectuer des échantillonnages de mesures de taille de paquets pour pouvoir évaluer la quantité d'information perdue. Plus de précisions sur la mise à jour des variables ("Slow Start threshold (SSthresh)" et "Congestion Avoidance") et les procédures de mesures seront trouvées en [WVSG02].

1.4.2 Modélisation du réseau

Pour évaluer les versions de TCP nous considérons deux protocoles de routage standardisés dans les réseaux ad hoc : AODV (Ad hoc On-Demand Distance Vector) et DSR (Dynamic Source Routing), utilisés dans de nombreux travaux d'évaluations. Nous décrivons ci-après leurs caractéristiques principales.

1.4.2.1 AODV (Ad hoc On-Demand Distance Vector)

Le RFC 3561 [PeBD03] spécifie le fonctionnement du protocole AODV proposé par Perkins et Belding-Royer. Il définit deux types d'opération : la découverte de routes et la maintenance des routes. La première opération permet d'établir une route vers une nouvelle destination en utilisant deux types de message de contrôle : les RREQ (Route REQuest) et les RREP (Route REPly). La seconde opération permet de réparer une route lorsque celle-ci est brisée, elle utilise le message RERR (Route ERRor).

– **Découverte de routes**
Un noeud initie un processus de découverte de route lorsqu'il a besoin d'une route pour une nouvelle destination ou pour une destination pour laquelle la route est devenue invalide. Envisageons le cas, illustré à la Figure 1.6, où l'émetteur S veut communiquer avec la destination D. S'il n'existe pas encore de route valide entre eux, l'émetteur diffuse le message RREQ dans le réseau pour trouver le chemin entre les deux noeuds. RREQ est d'abord reçu par les plus proches voisins qui vont ensuite propager le message, jusqu'a atteindre le noeud destination (ou un noeud connaissant un chemin vers celui-ci). Chaque noeud intermédiaire enregistre dans sa table de routage l'adresse du noeud qui lui a transmis le RREQ, établissant ainsi le chemin de retour (Reverse Path). Si un noeud reçoit plusieurs copies d'un même RREQ, seule la première est conservée. AODV utilise un champ numéro de séquence dans sa requête pour éviter la retransmission infinie de la requête et être sûr d'utiliser les routes les plus récentes.

  Une fois que le message atteint le noeud destination (ou un noeud connaissant un chemin vers celui-ci), celui-ci transmet un message RREP (Route REPly) vers la source par le chemin de retour (voir Figure 1.7). Il parcourt le chemin suivi par la requête de route en sens inverse, et modifie les tables de routage des noeuds par lesquels il passe. En effet, l'information à conserver dans la table de routage est le noeud suivant, dans le chemin aller (Forward Path), et pas, celle qui a été précédemment enregistrée qui indique le noeud suivant, dans le chemin retour. Comme le chemin retour est l'inverse du chemin aller, ce fonctionnement suppose que les liens sont symétriques.

– **Maintenance de routes**
AODV prévoit deux optimisations pour réparer une route localement. La première est effectuée par le noeud qui détecte la panne, s'il a mémorisé dans sa table un chemin alternatif pour joindre la destination, il l'utilise et économise l'émission de messages de recherche et réponse de routes. La seconde économise l'émission de messages d'erreur de route. Lorsqu'un lien se brise le long d'une route active, le noeud précédant la cassure peut effectuer une découverte de route s'il se trouve au moins à mi-chemin de la source et de la destination sans prévenir la source. Ainsi, pour

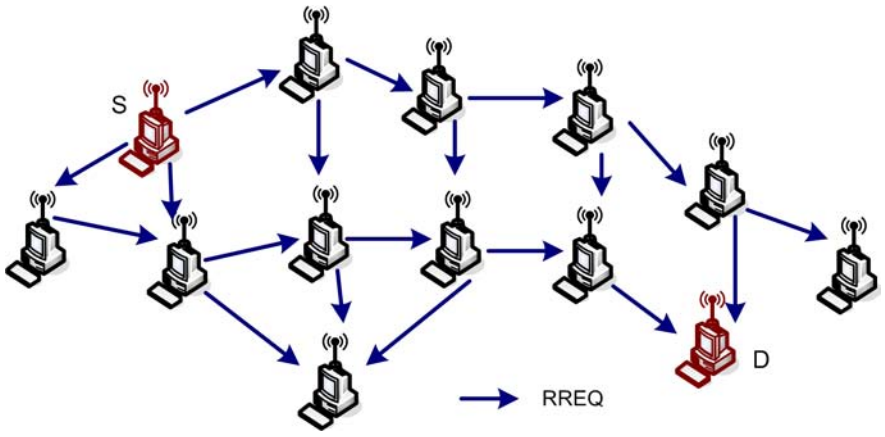


FIG. 1.6 – Procédure AODV de recherche de route

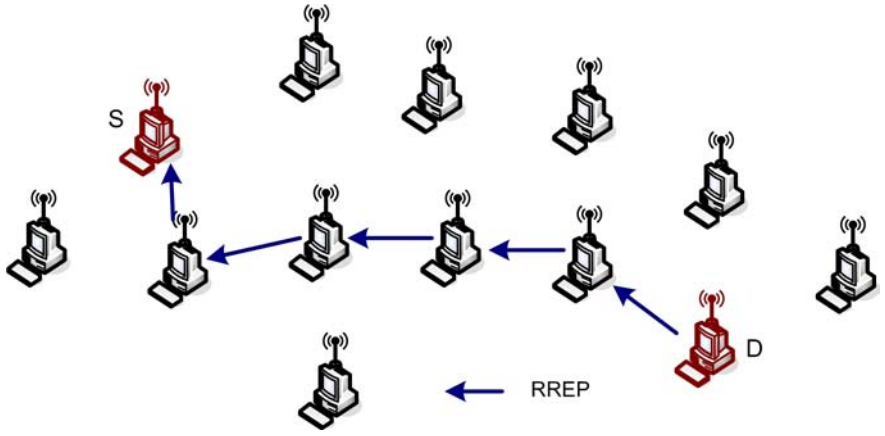


FIG. 1.7 – Réponse de la destination dans AODV

réparer la route, nous avons deux cas de figure :

- **Cas 1** : le noeud intermédiaire est plus proche du noeud destination (voir la Figure 1.8). Il incrémente le numéro de séquence de la destination et initie un processus de découverte de route. Si le noeud reçoit un RREP (ou un autre message de contrôle mettant à jour la route vers la destination), il met à jour ses informations de routage.

- **Cas 2** : le noeud intermédiaire est plus proche du noeud source. Il initie s'il a une route disponible une réparation et prévient la source. Elle pourra choisir de continuer sur cette route ou initier une nouvelle recherche (voir la Figure 1.9).

- **Détection de l'échec de lien**
  Le protocole AODV propose deux méthodes pour détecter la présence/non présence de liens : l'émission de message Hello et la prise en compte d'informations émanant directement de la sous-couche MAC.
  Dans la première méthode, le protocole définit un nouveau type de message de contrôle et une nouvelle table à chaque noeud. Chaque noeud envoie en diffusion à intervalles réguliers un message Hello et recueille les réponses des noeuds voisins. Ces

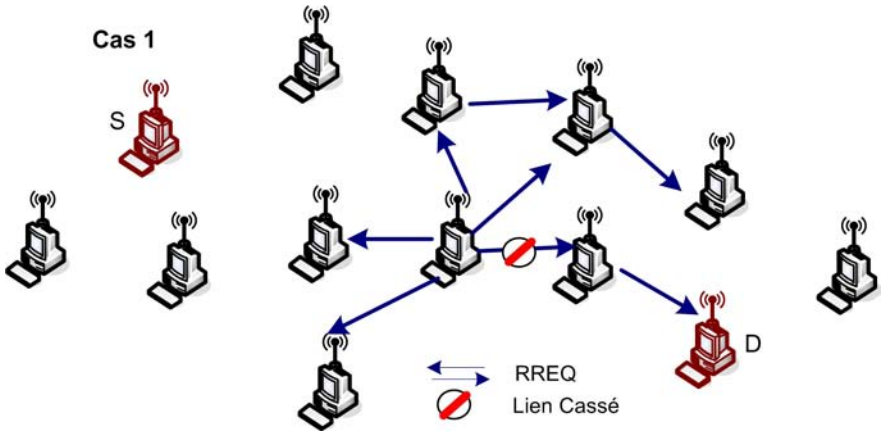


FIG. 1.8 – Cas 1 : Erreur dans AODV

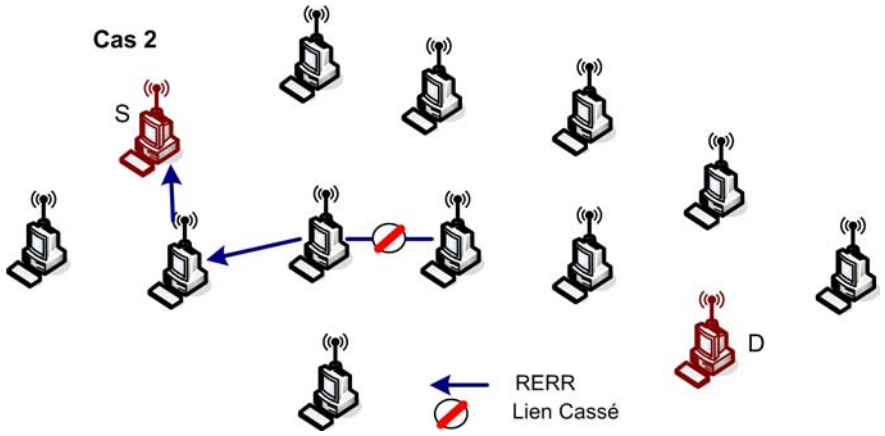


FIG. 1.9 – Cas 2 : Erreur dans AODV

informations sont stockées dans la Table des voisins (Neighbor Table) permettant ainsi à un noeud de savoir à chaque instant quels sont ses voisins. Si aucun message Hello, ou tout autre message, n'est reçu de la part d'un voisin durant un certain intervalle de temps, le voisin en question est supprimé de la table. L'intervalle de temps est fonction du nombre de messages Hello perdus (en raison de collisions) toléré et de l'intervalle de temps séparant deux émissions successives.
La seconde méthode se base sur les informations émanant directement de la sous-couche MAC 802.11. Un noeud considère qu'un lien est brisé lorsqu'il ne reçoit pas de trame MAC CTS en réponse à son émission de trame MAC RTS ou bien s'il ne reçoit pas d'ACK en réponse à une trame de données. La sous-couche MAC de 802.11 permet de définir le nombre de retransmission de ses paquets de contrôle avant l'abandon de la transmission. Dans les MANETs, où les changements de topologie sont fréquents et le temps que prend chaque noeud à détecter des signes de mobilité est primordial, cette méthode offre une meilleure convergence que la précédente et ne requiert pas de messages supplémentaires.

1.4.2.2 DSR (Dynamic Source Routing)

Le protocole DSR, proposé par Johnson et al. et décrit par le RFC 4728 [JoHM07] est un protocole qui, à la différence de AODV, n'utilise pas de table de routage mais un routage par la source. Dans cet algorithme, chaque paquet contient la liste complète des noeuds par lesquels il doit passer pour arriver à destination. L'avantage de cette méthode est que les noeuds intermédiaires n'ont pas besoin de maintenir d'informations sur la route puisque le paquet possède toutes ces informations. Un autre avantage du routage par la source est la possibilité de contrôler le trafic en lui indiquant de suivre le chemin le plus adapté à ses besoins de qualités de services ; une version de DSR proposée par Microsoft (LQSR : Link Quality Source Routing [Micr05]) inclut ainsi de la qualité de service. L'inconvénient de la méthode est l'augmentation de la taille des paquets qui grandit avec le nombre de noeuds à traverser. DSR convient donc mieux pour des réseaux de petite taille.

De même que AODV, DSR définit deux opérations : la découverte des routes et la maintenance de route.

– **Découverte de routes**
La procédure est très similaire à celle de AODV : il y a diffusion d'une requête de route et émission en unicast d'une réponse de route. La différence réside dans les traitements effectués par les noeuds intermédiaires. Pour AODV il y a mémorisation de pointeurs de chemins dans une table, alors que pour DSR, il y a écriture dans le champ "chemin du paquet". Le noeud intermédiaire enregistre son identifiant lors du relayage de la requête. Nous illustrons la procédure de recherche par un exemple (voir Figure 1.10) : le noeud 1 envoie un paquet Route Request à destination du noeud 10. Ce paquet se propage dans le réseau jusqu'à arriver au noeud destination ou à un noeud connaissant un chemin vers celui-ci. Le paquet contient l'adresse source, l'adresse de destination, un numéro d'identification, ainsi qu'un champ dans lequel se trouve la séquence des noeuds visités durant la propagation de la requête dans le réseau (route record).

  Quand un noeud reçoit un paquet Route Request, il vérifie s'il connaît un chemin vers la destination. Si ce n'est pas le cas, il ajoute son adresse dans le champ route record et transmet le paquet à ses voisins. Afin d'éviter les boucles et la multiplication des paquets Route Request, ce transfert n'a lieu que si l'adresse du noeud n'est pas déjà dans le champ route record.

  Quand le paquet atteint sa destination, celui-ci émet un paquet de réponse via le chemin indiqué dans le champ route record, si les liaisons sont symétriques, ou via un autre chemin (moyennant éventuellement une découverte de chemin). La Figure 1.11 montre le cas de connexions symétriques. Pour diminuer le coût de la recherche de route, chaque noeud peut garder en mémoire les routes qu'il a apprises, dans le cache de route.

– **Maintenance de routes**
Comme dans AODV un paquet Route Error est émis quand une route est inutilisable. Ce paquet, contenant l'adresse du noeud qui a détecté l'erreur et celle du noeud qui le suit dans le chemin, est envoyé à l'émetteur original du paquet, dont la "non transmission" a déclenché la détection de panne. Quand la source reçoit le paquet Route Error, le noeud concerné par l'erreur est supprimé du chemin sauve-

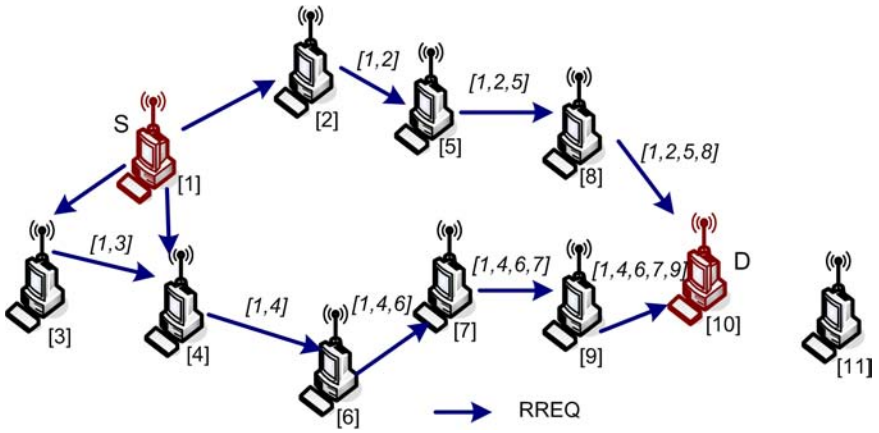


FIG. 1.10 – Découverte de route dans DSR

gardé et tous les chemins qui contiennent ce noeud sont tronqués à ce point là. Par la suite, une nouvelle opération de découverte de route vers la destination est initiée par l'émetteur.

DSR optimise ses algorithmes par un mécanisme de cache. Le cache est utilisé à la source pour mémoriser plusieurs routes vers une destination. En cas de panne la source n'initie pas de nouvelle recherche de route, elle utilise la route mémorisée dans le cache. Le mécanisme de cache est également utilisable sur les noeuds intermédiaires. Comme dans AODV, les noeuds intermédiaires, peuvent s'ils ont une route alternative dans leur cache réparer par eux-mêmes la route sans effectuer de recherche de route. Plus d'informations sur les avantages et inconvénients d'utiliser le cache sont disponibles en [HuJo00].

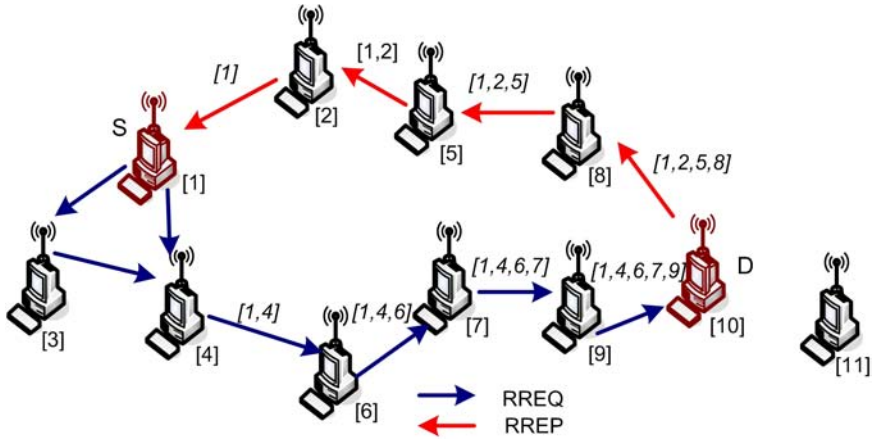


FIG. 1.11 – Renvoi du chemin dans DSR

1.4.3 Modèle de simulation

Nous avons utilisé l'environnement de simulation NS 2 (version 2.29) [NS] pour effectuer notre étude de performances. Nous utilisons les implantations des protocoles AODV et DSR intégrées à NS 2. Au niveau MAC, nous utilisons le modèle 802.11 implanté par le groupe Monarch, en conservant les valeurs par défaut des paramètres de ce modèle.

Les performances des protocoles de routage et plus particulièrement celles de AODV et DSR, comme l'indique le document [CoMa99] étant fortement dépendantes de la mobilité des noeuds de réseau et de leur nombre, nous avons fait varier ces paramètres.

Ainsi :
- Nous avons effectué des simulations pour 10 et 50 noeuds ;
- l'aire de simulation est de 500m*500m pour 10 noeuds et de 1,000m*1,000m pour 50 noeuds ;
- le modèle de mobilité est Random Waypoint [CaBD02]. La vitesse de déplacement des noeuds varie de 1 m/s à 20 m/s et nous avons utilisé des temps de pause de 30 secondes ;
- la durée de simulation est de 500s pour 10 noeuds et de 1,000s pour 50 noeuds ;
- la portée de transmission de chaque noeud est fixée à 250 mètres.

Concernant le modèle de trafic, notre objectif premier étant de comprendre l'impact de la mobilité, et non d'étudier le mécanisme de congestion, nous avons choisi de simuler un réseau qui ne soit pas en surcharge.

- La taille de paquet TCP est 1,460 octets.
- Le traffic données est de type CBR (Constant Bit Rate) et le débit d'émission est de 447 Kbps ou 266 paquets/s. Nous avons effectué des simulations pour 4 et 20 connexions.

Les valeurs, telles que la durée de simulation, la vitesse des noeuds, le nombre de connexions, ont été fixées de façon à obtenir des résultats interprétables par rapport à ceux publiés dans la littérature sur les performances des optimisations de transport.

Les performances sont évaluées à partir des paramètres :

- **Goodput (unité : Kbps)** : le nombre de bits de données reçus, à l'exclusion du nombre de bits de données retransmis par le temps de simulation.
- **Délai de bout en bout (unité : seconde)** : le temps de réception des données - le temps d'émission des données / nombre de paquets de donnée reçus.
- **Surcharge de routage normalisée (normalized routing load)** : le nombre de bits de contrôle transmis/le nombre de bits de contrôle et de données transmis.
- **Pourcentage de données retransmises (%)** : (le nombre de bits de données retransmis / nombre de bits de données transmis)*100.

1.4.4 Résultats et interprétations

Les résultats de simulations sont interprétés et analysés selon deux questions :

- Quel est l'intérêt d'optimiser le niveau transport dans les réseaux ad hoc par rapport au niveau réseau ?
- Quel est l'impact des conditions environnementales sur les versions de TCP ?

Optimisation de niveau transport versus optimisation réseau

L'importance du protocole de routage sur les performances de TCP dans les réseaux ad hoc, a été constatée sur les versions de TCP Reno [HoVa02] et TCP Tahoe [AASS00]. Nous étendons ce résultat à deux versions plus récentes de TCP : TCP Sack et TCP Westwood.

Les versions TCP Reno et TCP Tahoe sont adaptées à un environnement filaire où les erreurs sont rares et isolées ; un seul recouvrement de paquet perdu par temps aller/retour est prévu. TCP Sack et TCP Westwood au contraire, permettent de retransmettre plusieurs paquets et devraient donc obtenir de meilleures performances.

Plus précisément, nous observons dans un premier temps le goodput en fonction de la vitesse de noeuds. Les critères de performance sont calculés sur différentes configurations des noeuds (10 et 50 noeuds). Chaque point calculé représente une vitesse (variant de 1 m/s à 20 m/s) et 10 scénarios sont simulés pour chaque vitesse.

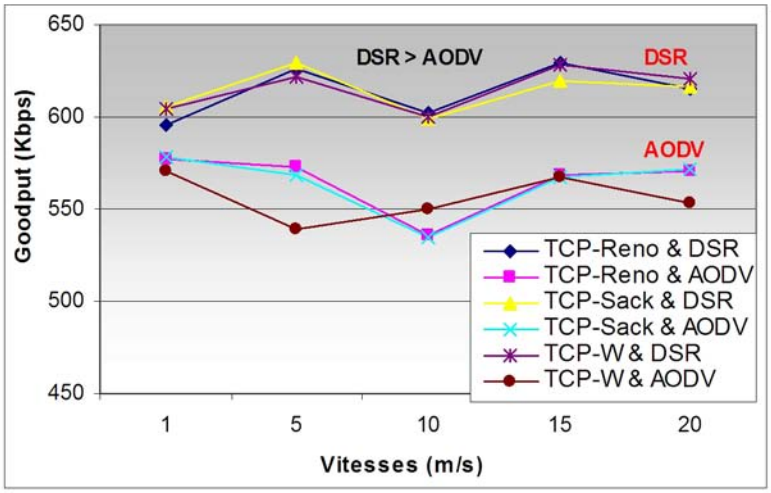


FIG. 1.12 – Comparaison goodput entre DSR et AODV pour 10 noeuds

1. Pour 10 noeuds, nous observons tout d'abord sur la Figure 1.12 que DSR obtient de meilleures performances, et ce quelle que soit l'option de transport. Ceci confirme les résultats présentés dans la littérature sur les performances respectives des deux protocoles de routage. En effet le mécanisme de cache de DSR lui permet de répondre plus rapidement que AODV ; d'autre part l'émission des messages Hello qui provoquent des engorgements de signalisation est également une raison avancée par

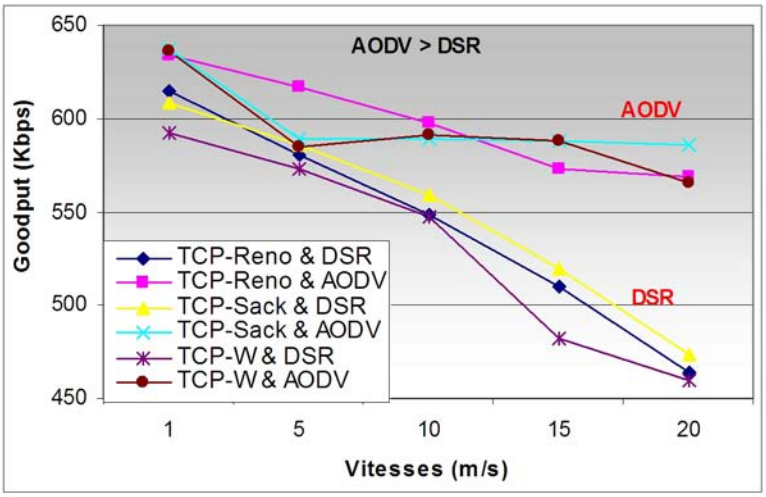


FIG. 1.13 – Comparaison goodput entre DSR et AODV pour 50 noeuds

[PeRD01] pour expliquer les moins bonnes performances de AODV.

2. Pour 50 noeuds, la tendance s'inverse (voir Figure 1.13) : AODV présente de meilleures performances quelle que soit l'option de transport simulée. Avec un plus grand nombre de noeuds, le problème des pannes de chemins, le nombre de noeuds par chemins augmentant, la politique de maintenance de routes d'AODV s'avère plus efficace et compense le traffic périodique des paquets Hello.

L'ordre de l'amélioration obtenue avec le protocole de routage idoine est autour de 25 % pour 50 noeuds tandis qu'il est autour de 5 % pour 10 noeuds. Nos résultats concordent avec ceux présentés en [BMJH98] et [SYQH04] qui ne prennent pas en compte le type du protocole transport utilisé. Pour des versions de base de TCP, DSR est davantage adapté à de petits réseaux, alors qu'AODV convient mieux à de grands réseaux. Nous constatons que les performances du niveau transport sont masquées par celles du protocole de routage.

Versions TCP et paramètres d'environnement

Une fois constatée l'importance du routage, nous détaillons le comportement des versions de TCP selon la vitesse de déplacement des éléments et le nombre de noeuds. Cette étude est loin d'être exhaustive, elle ne présente les résultats que sur un seul modèle de mobilité : le Random Waypoint. Cependant nous justifions notre choix par le fait que le réseau n'est pas conçu pour une mobilité particulière et que ce modèle est largement utilisé dans la littérature.

Les résultats de goodput pour les différentes versions de TCP sont détaillés sur les Figures 1.14 et 1.15.

1. Importance de la vitesse. Dans un petit réseau le lien entre le goodput et la vitesse, et ce quel que soit le protocole de routage, n'apparaît pas dans les résultats obtenus. Notons que ce résultat s'observe également sur d'autres versions telle TCP-ELFN [HoVa99]). Par contre, dans un réseau de 50 noeuds les performances sont inversement proportionnelles à la vitesse. Nous expliquons ce comportement par la prédominance des pannes de rupture de chemins sur les performances de transport

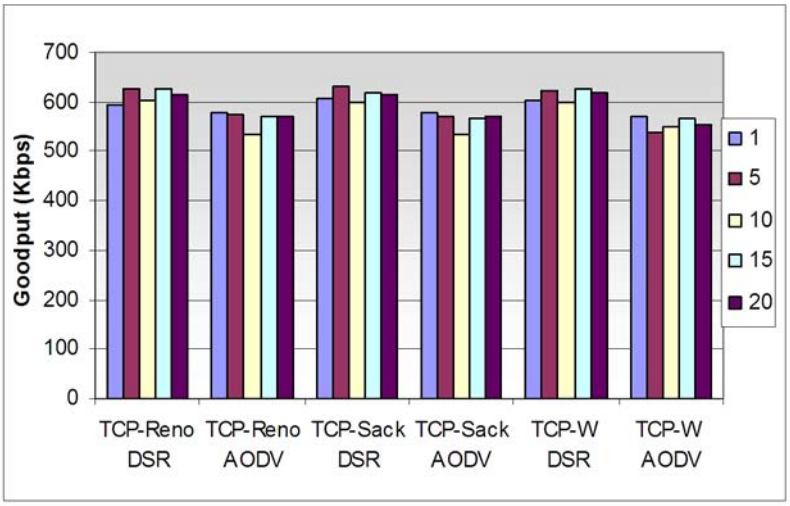


FIG. 1.14 – Goodput des versions protocoles de TCP pour 10 noeuds

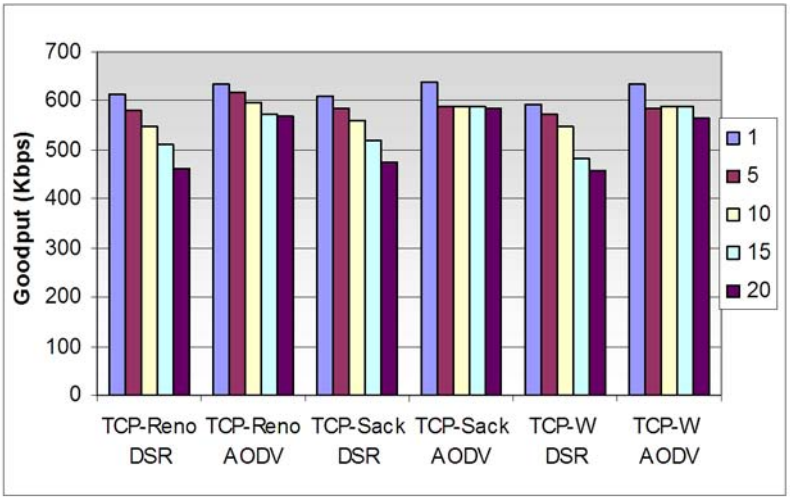


FIG. 1.15 – Goodput des versions protocoles de TCP pour 50 noeuds

par rapport aux autres types d'erreurs. Dans les petits réseaux, le nombre de rupture de chemins n'est pas lié directement à la vitesse de déplacement du modèle de mobilité utilisé : le nombre de noeuds composant le chemin est faible, il y a peu de rupture de chemin. Les performances de TCP sont directement influencées par les délais du réseau. Dans un réseau de plus grande envergure, le nombre de rupture de chemins est plus important selon la vitesse de déplacement, occasionnant ainsi plus de pertes (voir les Figures 1.16, 1.17). Les performances de TCP sont alors influencées par le temps de recouvrement de la route.

2. TCP Westwood versus TCP Sack : de nos expérimentations, il apparait que soit TCP-W, soit TCP Sack, obtiennent de meilleures performances selon la vitesse ou le nombre de noeuds. Néanmoins la différence de performance entre les deux versions ne semble pas très significative (autour de 1 %). Notons que d'après [DyBo01], l'utilisation de TCP avec renvoi retardé de l'acquittement (Delayed ACK), n'obtient pas de meilleurs résultats.

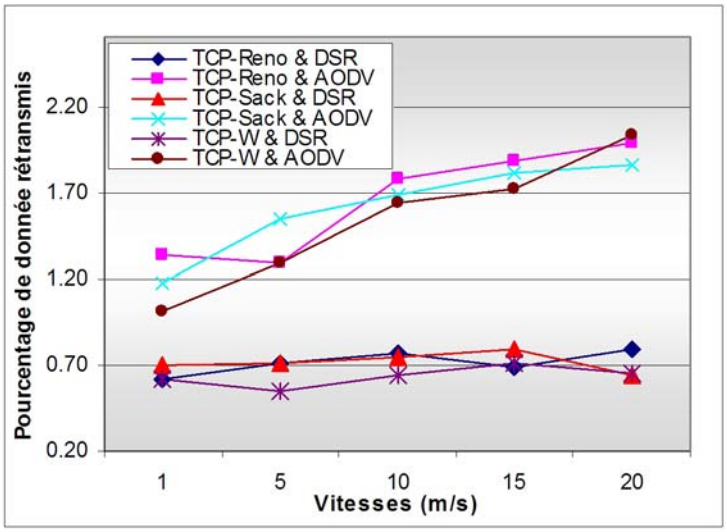


FIG. 1.16 – Pourcentage de données rétransmis dans les versions TCP pour 10 noeuds

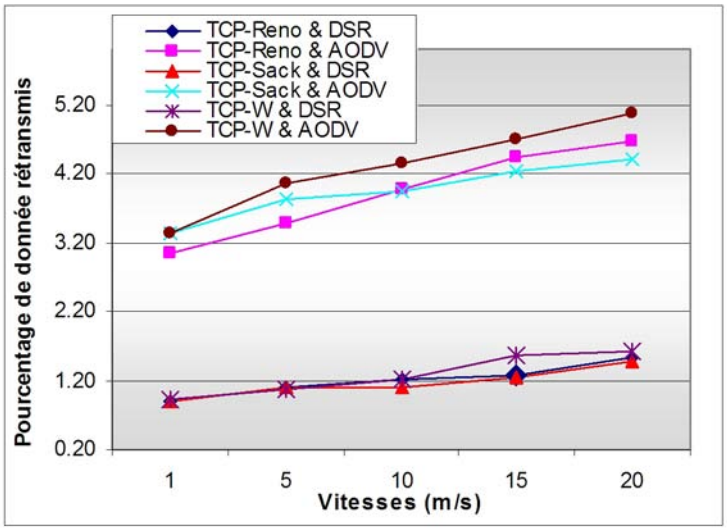


FIG. 1.17 – Pourcentage de données rétransmis dans les versions TCP pour 50 noeuds

1.4.5 Analyse des performances du protocole de routage

Nous expliquons les résultats de simulations que nous venons d'indiquer par la formulation des performances en fonction des paramètres d'environnement et de simulation. Nous simplifions le fonctionnement de AODV et de DSR en ne prenant en compte le mécanisme de cache que sur la source (les deux protocoles utilisent sur les noeuds intermédiaires un mécanisme de cache similaire). Le mécanisme de cache est alors présent uniquement avec DSR.

Analysons les facteurs de performance par la méthode suivante :

1. formulons les délais, de niveau réseau, d'un paquet pour les deux protocoles de routage AODV et DSR, puis,

2. évaluons celui obtenu pour chaque protocole dans le cas d'un petit réseau et dans le cas d'un grand réseau, en utilisant les résultats de simulation. La relation d'ordre entre le délai des deux protocoles de routage et le goodput est équivalente. Par souci de simplification nous utilisons la variable goodput réseau pour faire le lien avec le goodput de niveau transport que nous avons considéré précédemment ;
3. formulons le goodput transport en fonction du taux de perte et des performances du routage ;
4. en utilisant l'évaluation, montrons que les résultats de simulation sont cohérents ; Quel que soit le mécanisme transport, le goodput transport est meilleur pour AODV dans les grands réseaux et meilleur pour DSR dans les petits réseaux.

Le détail de notre démarche est présenté dans la table 1.3, la signification des symboles utilisés est indiquée dans la table 1.4.

Formulation

(F 1.1) : Formulation du délai pour AODV : il y a un délai d'obtention de route (Droute) et un délai de transmission de paquet (Dtrans).

(F 1.2) : Droute se compose d'un délai d'établissement de route initial (Dest) plus d'un certain nombre de réétablissement qui est fonction de la probabilité de rupture de route (Pb).

(F 2.1) : Formulation du délai pour DSR : le délai d'obtention de route, Droute, est équivalent à celui de AODV.

(F 2.2) : DSR utilise un cache de routes : en cas de rupture de route, il obtient directement la route mémorisée sans avoir à faire d'établissement. Quand la route est non-valide (probabilité ps), DSR obtient une nouvelle route. Nous approchons le temps perdu sur la route non valide par un délai d'établissement de route (Dest).

(F3) : Les délais d'établissement des routes en DSR et en AODV sont équivalents.

Evaluation-Cas 1 : Pour un petit réseau (N < N1),

En cas d'erreur, AODV qui n'a pas de mécanisme de cache, doit chercher une route, alors que DSR obtient directement une nouvelle route ; la probabilité **ps** que la route soit non valide étant très petite, elle est considérée comme nulle. Par la simulation nous avons un délai de transmission de paquet de DSR plus important que celui de AODV (**r 1.1** et voir la Figure 1.19). Ceci s'explique par la surcharge de routage générée par DSR qui rajoute un champ chemin dans les paquets de routage (voir le Figure 1.18). Comme par ailleurs, le délai d'établissement de la route est similaire entre les deux protocoles (la procédure est équivalente, **F3**), les meilleures performances de DSR obtenues par simulation (Figure 1.12) sont dues à son mécanisme de cache qui masque ses moins bonnes performances de transmission (**r 1.2**). Son goodput réseau est donc meilleur que celui de AODV (**r 1.3**).

Evaluation-Cas 2 : Pour un réseau de taille suffisante (N > N1),

Pour **AODV**	
(F 1.1)	D.AODV = Dtrans.AODV + Droute.AODV
(F 1.2)	Droute.AODV = (1 + pb) Dest.AODV
Pour **DSR**	
(F 2.1)	D.DSR = Dtrans.DSR + Droute.DSR
(F 2.2)	Droute.DSR = (1 + 2*pb*ps) Dest.DSR
(F3)	(Dest.DSR) eq (Dest.AODV) eq Dest
Cas 1	N < N1, pb = i1 ; ps = j1 et j1 eq to 0
(r 1.1)	Dtrans.DSR > Dtrans.AODV ;
(r 1.2)	Pb*Dest > Dtrans.DSR - Dtrans.AODV
(r 1.3)	Ng.DSR > Ng.AODV
Cas 2	N > N1, pb : = i2 > i1 ; ps = j2 > j1
(r 2.1)	Dtrans.DSR > Dtrans.AODV ;
(r 2.2)	Droute.DSR > Droute.AODV
(r 2.3)	Ng.AODV > Ng.DSR
(F4)	Tg.Ti (rp) = f (Pl.Ti, D.rp)
(r3)	Pl << 1 => Tg.eq.f(D.rp)
(r4)	Ng.AODV < Ng.DSR => $\forall i$, Tg.Ti (AODV) < Tg.Ti (DSR) et
	Ng.AODV > Ng.DSR => $\forall i$, Tg.Ti (AODV) < Tg.Ti (DSR)

TAB. 1.3 – Analyse de l'impact du routage sur le transport

Les symboles	Sens des signes
AODV	Ad hoc On-Demand Distance Vector
D	Délai
Dest	Délai d'établissement de route
Droute	Délai de route
Dtrans	Délai de la transmission de paquet
DSR	Dynamic Source Routing
eq	Opérateur équivalent
N	Nombre de noeuds
Ng	Goodput de réseau
N1	Valeur de nombre entier
pb	Probabilité de rupture de route
ps	Probabilité à la vieille route est utilisée
Pl	Nombre de perte de paquet
rp	Protocole de routage
Tg	Goodput de protocole transport
Ti	Goodput de protocole transport pour les versions i tels que Reno et Sack
Exemple :	
Ng.rp	Goodput de réseau pour le protocole de routage
D.rp	Délai pour le protocole de routage

TAB. 1.4 – Symboles de formulation

Le délai de transmission d'un paquet est nettement plus important avec DSR que AODV, (Figure 1.21) en raison de la surcharge générée par le champ chemin (Figure 1.20 et **r 2.1**). De plus, sur de grands réseaux, la probabilité **ps** d'avoir une route mémorisée

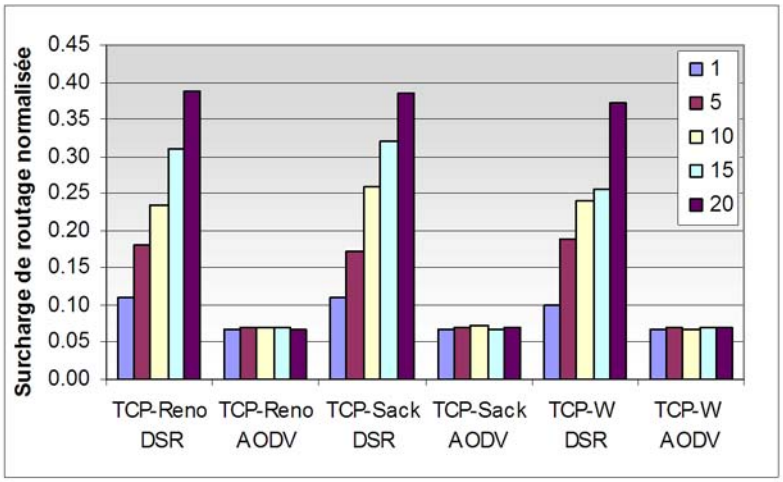


FIG. 1.18 – Surcharge de routage normalisée pour 10 noeuds

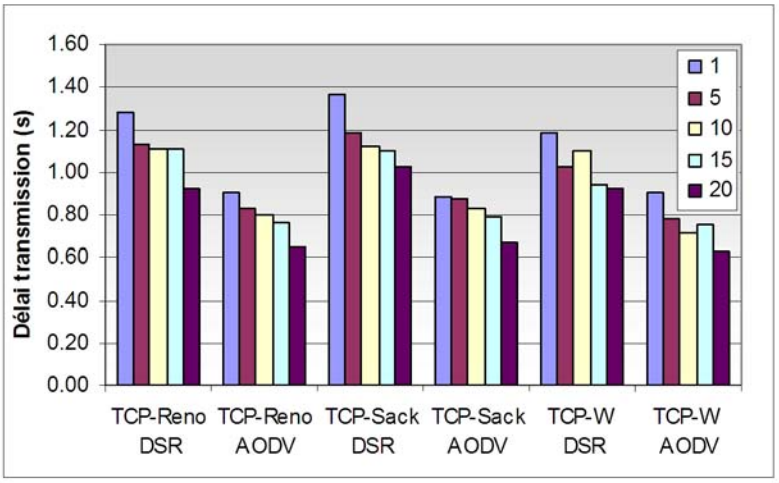


FIG. 1.19 – Délai transmission pour 10 noeuds

en cache qui ne soit pas valide devient non négligeable, ce qui explique que le délai de DSR soit supérieur à celui de AODV (**r 2.2**). Ainsi, le goodput réseau d'AODV est plus élevé que celui de DSR (**r 2.3**).

(F4) Le goodput de transport (Tg) de la version i, Ti, au-dessus d'un protocole de routage (rp) est influencé par le nombre de la perte de paquet (Pl) mais également par le délai du protocole de routage (D.rp).

(r3) D'après nos simulations, (Figures 1.16 et 1.17) : nous observons que Pl est faible de l'ordre 10^{-6}, ou $Pl << 1$ (**r3**). Le facteur de performances le plus important du protocole transport n'est pas le temps de retransmission provoqué par les pertes en cas de rupture de route, et qui diffère selon les variantes de TCP, mais le délai réseau.

Nos résultats sont en accord avec ceux publiés dans la littérature sur d'autres versions de transport fiable ([AASS00], [DyBo01], [HoVa02]). Nous en déduisons que le délai d'éta-

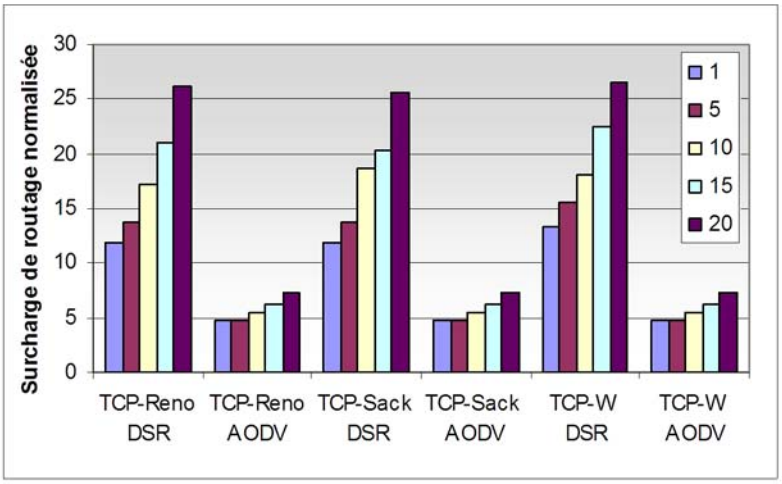


FIG. 1.20 – Surcharge de routage normalisée pour 50 noeuds

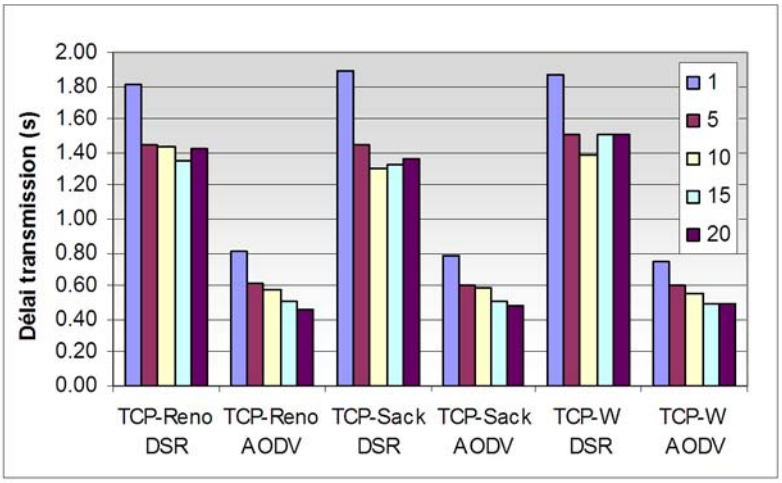


FIG. 1.21 – Comparaison délai de bout en bout pour 50 noeuds

blissement de route est plus important pour la performance de transport que le processus de rétablissement d'erreur. Le choix d'une option de contrôle de congestion ou d'une autre n'est pas un facteur primordial pour améliorer les performances (**r4**). L'optimisation de routage est plus importante que celle du transport.

1.5 Conclusion

Les protocoles de transport sont historiquement conçus pour s'adapter à la qualité de service du système de transmission, que ce soit dans le monde OSI ou dans le monde IETF, en proposant de faire ou de ne pas faire de contrôle. Plus précisément, un protocole de transport fiable tel que TCP inclut des mécanismes d'auto-adaptation à l'encombrement du système de communication via son contrôle de congestion. Dans ce chapitre, nous avons exprimé un nouveau besoin d'adaptation : l'adaptation à la mobilité. La mobilité que nous considérons concerne l'utilisateur mais également le système de communication, car dans le réseau ad hoc, tous les équipements sont mobiles et ils forment une topologie de connexion dynamique.

L'analyse que nous avons menée dans ce chapitre à travers les travaux publiés dans le domaine, et nos expérimentations par simulation, conduisent à mettre en avant le protocole de routage comme un facteur prépondérant de performance du transfert de données de bout en bout. La suite de ce travail aura pour objectif d'améliorer la stratégie de transport pour prendre en compte la mobilité, tout en s'appuyant sur le routage.

Chapitre 2

Optimisation de transport par communication inter-couches

Sommaire

Dans le premier chapitre nous avons présenté et analysé les problèmes de performance du transport fiable, de type TCP, dans les réseaux ad hoc et montré la dépendance entre les performances du protocole de transport et le fonctionnement du protocole réseau.

Dans ce chapitre, nous nous intéressons à l'approche de communication inter-couches (cross layer) mise en oeuvre pour optimiser les performances d'un protocole d'un niveau donné, en fonction de ses dépendances à d'autres niveaux. Nous présenterons également plus avant le protocole SCTP qui constitue la base des expérimentations que nous proposons dans les chapitres suivants.

2.1 Approches cross layer

Nous conservons le terme anglais cross layer pour désigner la nouvelle approche de conception de réseau à l'étude ces dernières années. Nous employons la désignation française "inter-couches" lorsque nous traitons de la communication entre les couches (dans la conception cross layer).

2.1.1 Motivation générale pour la conception cross layer

L'architecture actuelle des protocoles réseaux, que ce soit le modèle OSI ou le modèle TCP/IP, repose sur un ensemble de modules devant chacun fournir un service précis et ce de façon autonome. Ces modules ont été organisés hiérarchiquement dans une pile, chaque module se trouvant au dessus du service dont il a besoin pour fournir le sien. De façon à assurer la maintenabilité des systèmes de communication face aux évolutions technologiques, les interfaces entre modules sont limitées : seules les interactions de services entre modules directement adjacents sont autorisées, chacun gérant son propre espace de données. Cependant, dès l'apparition d'influences entre deux couches, l'envie est forte que ces couches puissent se communiquer un ensemble de variables statistiques, qui aideront à la résolution d'un problème. C'est ce que proposent les architectures cross layer notamment à travers des retours d'informations [RaIy06].

Un exemple de retour particulièrement intéressant dans les réseaux ad hoc, qui permet d'optimiser des protocoles de différents niveaux, concerne la qualité de la liaison. Cette information, déterminée par un ensemble de paramètres de la couche physique, est utilisable par la couche MAC/LLC (Logical Link Control) pour décider d'utiliser ou non des mécanismes de contrôle d'erreur ou encore pour adapter la portée de la transmission. Cette information peut également être utilisée par le routage pour sélectionner les noeuds qui constituent un chemin.

Un grand nombre de protocoles cherchant à améliorer TCP utilisent des mécanismes cross layer. Ceux-ci peuvent demander une coopération importante de la part des noeuds de routage et nécessitent souvent l'utilisation de messages, ou de champs de messages, spécifiques. Nous présentons à la suite deux mécanismes typiques de cross layer.

Contrôle des pertes de routes

Le principe de ce mécanisme est de faire envoyer par un noeud qui détecte la perte d'une route, un paquet de retour à la source. A la réception de celui-ci, l'émetteur TCP, que ce soit dans le protocole TCP-F [CRVP98] ou ELFN [HoVa99], va bloquer son fonctionnement : gel des horloges, des émissions et des fenêtres de régulation de débit. Ainsi, à la fin du blocage (TCP-F attend un paquet de notification de reconstruction de la route alors que ELFN teste par l'envoi de paquets spéciaux si la route est à nouveau disponible), TCP va pouvoir reprendre immédiatement ses envois, à un débit concordant avec la réalité du réseau, et donc, ne pas souffrir d'un départ lent de sa fenêtre de congestion. Dans TCP-Bus, [KiTC01], lorsque les noeuds reçoivent un retour de perte de route, ils mémorisent les messages en transit. De cette façon, les messages ne seront pas perdus, et en augmentant suffisamment le RTO de ces paquets, ils ne provoqueront pas d'expiration de timers intempestives, ce qui a pour effet de rendre TCP transparent à l'échec de routage.

Contrôle de la congestion

Le contrôle de congestion est une fonction des protocoles de transport qui a fortement bénéficié des mécanismes de cross layer, comme en attestent le mécanisme ECN [RaFl99]

[RaFB01], et le protocole ATCP [LiSi01]. Venu du monde filaire, le mécanisme ECN permet à un noeud routeur, selon l'occupation moyenne de son buffer, de positionner un bit dans l'entête IP du paquet qu'il est en train de traiter, pour signifier qu'une congestion est probable. Le paquet, en arrivant au récepteur de la connexion TCP, va positionner le bit ECN-echo dans quelques-uns des acquittements suivants. L'ECN-echo va avoir pour effet d'inhiber l'accroissement de la taille de la fenêtre, CWND, à l'arrivée de l'acquittement au niveau de l'émetteur TCP. ATCP est un protocole cross layer ; il inclut une nouvelle couche entre la couche réseau et la couche TCP qui a pour but de filtrer les signes de congestion émanant du réseau. Celle-ci garde trace des acquittements reçus et des horloges de retransmissions. Ainsi, lorsque trois acquittements dupliqués se succèdent, ou lorsque une expiration de timer est proche, ATCP bloque TCP et effectue lui-même la retransmission des paquets nécessaires, jusqu'au moment où un acquittement non dupliqué arrive. TCP est alors débloqué et l'acquittement lui est délivré.

2.1.2 Architectures de communication inter-couches

Dans l'approche cross layer, les échanges d'informations se font entre des couches qui ne sont pas forcément adjacentes : la performance globale est optimisée en adaptant chaque niveau en fonction de l'information disponible. Les architectures protocolaires, pour mettre en oeuvre les mécanismes d'adaptation, s'inspirent de trois modèles : communication directe entre couches, interactions vers une entité intermédiaire, nouvelles abstractions. Ces trois modèles, exposés par Srivastava et Motani [SrMo05], sont représentés sur la Figure 2.1.

2.1.2.1 Communication directe entre couches

Une façon pour un protocole de s'adapter est de permettre des communications directes entre les couches qu'elles soient ou non adjacentes, comme indiqué sur la Figure 2.1-A. De nouvelles interfaces sont utilisées pour partager les informations entre les couches. Par cette méthode, des flux d'informations peuvent être redirigés d'une (de) couche(s) inférieure(s) vers une (des) couche(s) supérieure(s), ou du(e) niveau(x) supérieur(s) vers un (des) niveau(x) inférieurs. Un problème inhérent à cette méthode est la complexité de gestion des espaces de mémoire partagée entre les niveaux, qui peut poser un problème d'implantation dès lors que les ressources des équipements sont limitées. De plus, ce type d'architecture pose des problèmes de maintenance dans la mesure où l'ajout, le retrait ou l'évolution d'éléments, influe sur de nombreuses interactions [KaKu05].

2.1.2.2 Interactions vers une entité intermédiaire

Ce modèle introduit un point de synchronisation qui facilite la communication entre les couches (voir la Figure 2.1-B). Il s'agit de spécifier les interactions des différentes couches vers une entité intermédiaire. Le partage de données s'effectue par un service de stockage/récupération d'informations sur toutes les couches ainsi que par la possibilité de positionner dans une couche spécifique des variables optimisées par une autre couche. Un exemple de cette approche est l'interface Xian qui permet à un réseau local de type WiFi 802.11 de propager ses informations vers tous les niveaux protocolaires ([ACLL06], [ACLL08]).

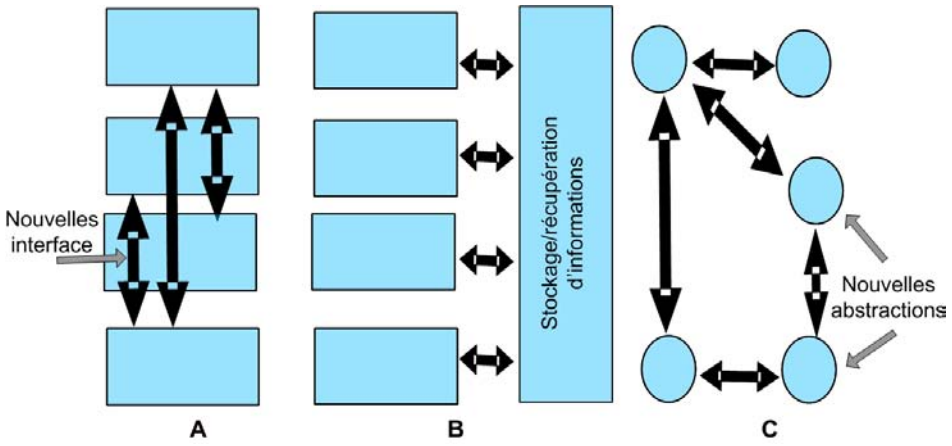


FIG. 2.1 – Modèles d'architectures cross layer

2.1.2.3 Nouvelles abstractions

La Figure 2.1-C présente une autre façon d'organiser les protocoles. Ceux-ci ne sont plus considérés comme des modules organisés hiérarchiquement avec des interactions récursives, mais comme des ensembles de composants fournissant un service spécifié par un graphe. Cette méthode, qui remet en cause le principe de structuration en couches, par une approche fonctionnelle, au bénéfice d'une structuration service, par une approche composants, a comme avantage d'offrir une grande flexibilité d'évaluation tout en minimisant les problèmes d'interactions et donc de maintenance.

2.1.3 Considérations architecturales pour les réseaux ad hoc

Nous avons vu dans le premier chapitre que les couches traditionnelles de la pile protocolaire TCP/IP ne sont pas conçues pour faire face à la dynamique des communications sans fil de deuxième génération ou réseaux ad hoc. Les architectures cross layer peuvent fournir une solution en cassant la structure traditionnelle pour permettre des interactions entre des couches directement ou non adjacentes. L'objectif est d'obtenir un système qui soit plus sensible à son environnement, sa charge et son utilisation, en remplaçant l'interface restreinte entre couches adjacentes par une vue plus riche et plus complète des objectifs, problèmes et contraintes du réseau.

La conception de l'architecture protocolaire doit permettre de prendre en compte les modifications nécessaires à sa longévité. Un certain nombre de propositions pour des conceptions cross layer avec leurs architectures correspondantes ont été étudiées dans la littérature. Au delà des modèles de conception présentés dans la section précédente, nous pouvons les classer selon la façon dont ils obtiennent l'information, source des optimisations [RaDN07] :

- architectures basées sur l'information locale (du noeud et de ses différentes couches).
- architectures basées sur l'information locale et globale (du noeud, de ses différentes couches et des voisins).

2.1.3.1 Architectures avec vue locale du réseau

La majorité des architectures existantes dépendent d'une vue locale de l'information de l'état du réseau, dans la prise de décision de leur optimisation. Les interactions entre protocoles sont augmentées alors que les communications entre entités distantes sont diminuées, permettant par là même d'économiser la bande passante.

WIDENS (Wireless Deployable Network System) [KNBA04] [BoCo04] est une architecture développée pour les réseaux ad hoc (voir la Figure 2.2) pour être déployée en cas de services d'urgence (e.g. application télémédecine, catastrophes naturelles). WIDENS s'intéresse principalement aux contraintes de QoS, mobilité et sécurité. Il propose des extensions cross layer qui fournissent des informations d'état et une correspondance des paramètres, dans le but d'augmenter le pouvoir de re-configuration et l'adaptabilité des protocoles. Les interactions "inter-couches" s'effectuent via des interfaces bien définies entre les couches adjacentes. Les informations sont exploitées lorsque les optimisations isolées au niveau d'une couche n'empêchent pas la dégradation de performances. Les adaptations potentiellement problématiques sont évitées en tolérant seulement les interactions entre les couches adjacentes. Les informations des couches non adjacentes sont exploitées via des fonctions de correspondance de la couche inférieure ou de la couche supérieure. Ceci est accompli uniquement dans le cas où les adaptations cross layer au niveau de la couche adjacente sont inefficaces et qu'il est alors nécessaire que les autres couches s'adaptent également pour satisfaire les performances requises.

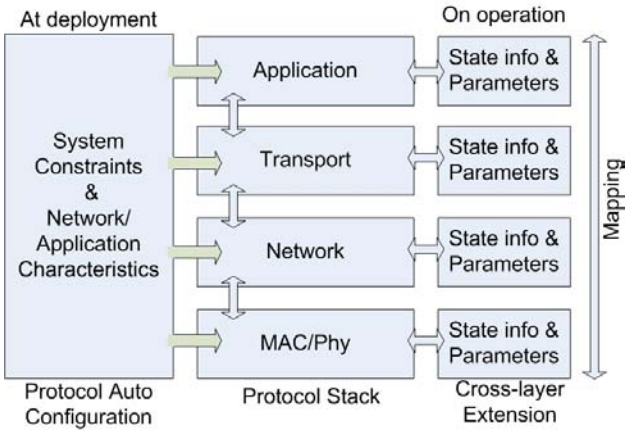


FIG. 2.2 – L'architecture de WIDENS

MobileMAN (Mobile Metropolitan Ad hoc Network), dont l'architecture est illustrée sur la Figure 2.3, s'appuie sur un composant fondamental, le "Network Status". Il contrôle les informations cross layer en respectant le principe de division des fonctionnalités et des responsabilités dans les couches. Chaque protocole peut accéder au "Network Status" pour partager ses données avec les autres protocoles, ceci en évitant la duplication de l'information de l'état interne, améliorant ainsi l'efficacité de la conception. L'intérêt et les gains de performances obtenus par l'architecture ont été étudiés en [CMTG04], [BoCD06_1], et des expérimentations à partir des protocoles AODV et OLSR (Optimized Link State Routing Protocol) ont été menées en [BoCD06_2].

ECLAIR (Efficient Cross Layer Architecture) [RaIy04] est une vue locale d'architecture (voir la Figure 2.4) qui propose un plan d'optimisation. Il se compose de deux composants

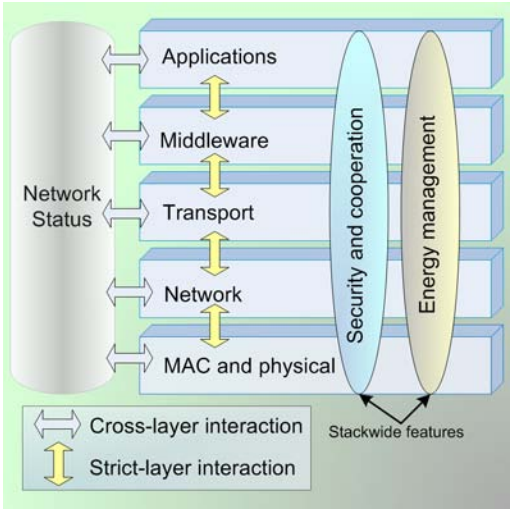


FIG. 2.3 – L'architecture de MobileMAN

principaux : un sous-système d'optimisation [OSS] qui constitue le moteur cross layer et contient de nombreux optimisateurs de protocole (PO), qui sont des composants "intelligents", et les "Tuning layers" (TL) qui fournissent les APIs (Application Programming Interfaces) des POs pour interagir avec les couches et manipuler les structures de données correspondantes. Par exemple, dans le cas de TCP, TCP tuning layer (TCPTL), peut positionner le contrôle de congestion, selon les résultats du PO (e.g : congestion avoidance ou slow start). Il n'y a aucun temps de processus additionnel dans la mesure où le sous-système d'optimisation s'exécute en parallèle avec la pile de protocole. L'avantage de ECLAIR, est sa grande capacité à supporter plusieurs types d'applications.

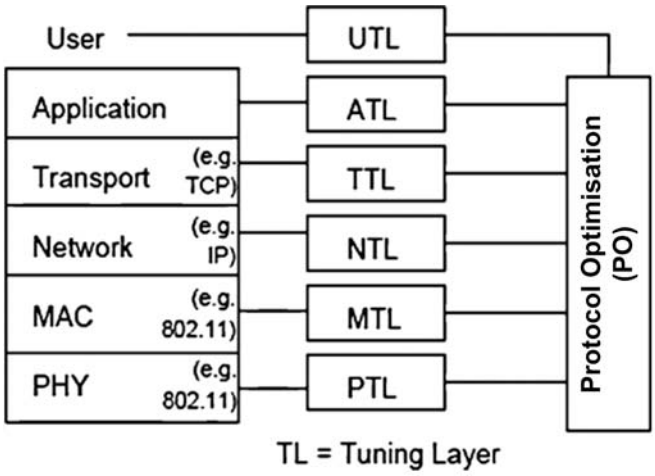


FIG. 2.4 – L'architecture de ECLAIR

POEM (Performance-Oriented Model) [GFTW05], [GuWo07] a été développé pour une auto-adaptation des protocoles dans le cadre de communications autonomes (voir la Figure 2.5). L'architecture utilise 3 plans : un plan de données, un plan de contrôle et un plan de gestion. L'optimisation est perçue comme une fonction du plan de contrôle qui ne doit pas perturber le fonctionnement normal des protocoles alors que la collecte

d'informations est du ressort du plan de gestion. Ainsi, les noeuds de réseau utilisent les données disponibles pour aider le protocole de routage à maintenir la sûreté du réseau en cas de problème.

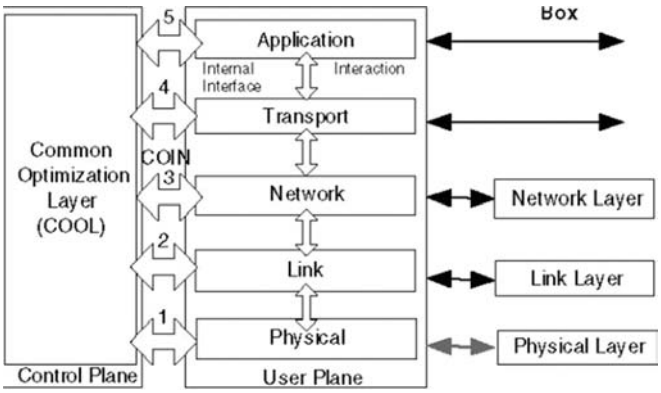


FIG. 2.5 – L'architecture de POEM

CATS (Cross-layer Approach to Self-healing) [SaKC05] est caractérisé par la présence d'un composant nommé Management Plane visible par toutes les couches. Le Management Plane est un composant actif qui influence le comportement des protocoles et même des paquets traversant la pile protocolaire (par exemple, le Management Plane peut changer l'adresse de destination d'un paquet ou encore influencer le protocole de routage pour stopper la réponse aux requêtes de routes) voir la Figure 2.6.

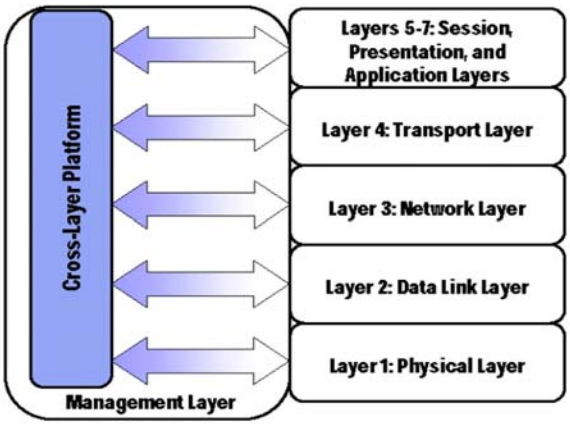


FIG. 2.6 – L'architecture de CATS

2.1.3.2 Architectures avec vue locale et globale du réseau

Les architectures de cross layer mentionnées ci-dessus se fondent sur une vue locale du réseau qui peut rendre difficile la détection, le diagnostic et la résolution de problème. Un point de vue global sera par exemple très utile pour mettre en oeuvre des politiques de contrôle de charge, de routage ou d'économie d'énergie. L'inconvénient de ce type d'approche est cependant le coût, en terme de bande passante et de complexité de maintenance, de l'état du réseau, de la collecte de statistiques, de la coordination d'actions globales.

Davantage concerné par la problématique de la qualité de services multimédia que par celle de la mobilité, GRACE (Global Resource Adaptation through CoopEration) [SYHH03], [YNAJ03] est un cadre d'adaptation de cross layer qui s'adresse à tous les niveaux : matériel, réseau, système opératoire et application, (voir Figure 2.7). Considérés comme des entités adaptives, les éléments du système global coopèrent les uns avec les autres pour réaliser une configuration optimale dont l'objectif est de maximiser l'utilisation du système selon l'état des ressources disponibles et les besoins des applications en termes de qualités de services multimédia. Pour obtenir une adaptation à grain fin, c'est à dire capable de prendre en compte les variations à court terme du système et de son environnement, il est nécessaire de recalculer fréquemment la configuration optimale, (un calcul par trame MPEG). Ce processus est coûteux en temps et complexe (résolution d'un problème d'allocation optimale). Pour résoudre ce problème des approches de hiérarchisation ont été proposées ; l'allocation de ressources est faite à un niveau global mais la décision exacte concernant l'utilisation des ressources est reportée sur les niveaux inférieurs qui sont davantage au courant de l'état instantané. Le résultat global de l'adaptation est mesuré de sorte qu'en cas de mauvaise utilisation des ressources (définie en rapport à des seuils), une optimisation globale puisse être déclenchée [Sach06].

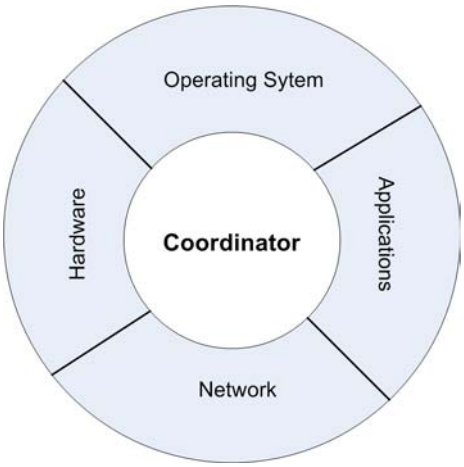


FIG. 2.7 – L'architecture de GRACE

CrossTalk [WSNB05] propose une vue globale du réseau à partir de différentes métriques (voir la Figure 2.8). Disposant d'une vue globale, un noeud peut utiliser l'information globale pour des processus de décision locaux, en même temps qu'une vue locale qui contient des informations spécifiques au noeud fournies par toutes ses couches protocolaires, ou ses composants systèmes. De façon à minimiser les échanges d'informations, aucun message spécifique n'est utilisé. Les informations issues de la vue locale sont jointes à des messages classiques en piggybacking. Le modèle de trafic influence alors beaucoup la véracité de la vue globale. Cependant même avec un modèle de données incertain pour construire la vue globale du réseau, CrossTalk, d'après les auteurs, permet d'améliorer les performances d'algorithmes de partage de charge.

2.1.4 Exemple d'interactions entre les protocoles

Les interactions entre protocoles ont particulièrement été étudiées pour adapter la transmission d'informations multimédia au système de transmission, filaire ou non, pour

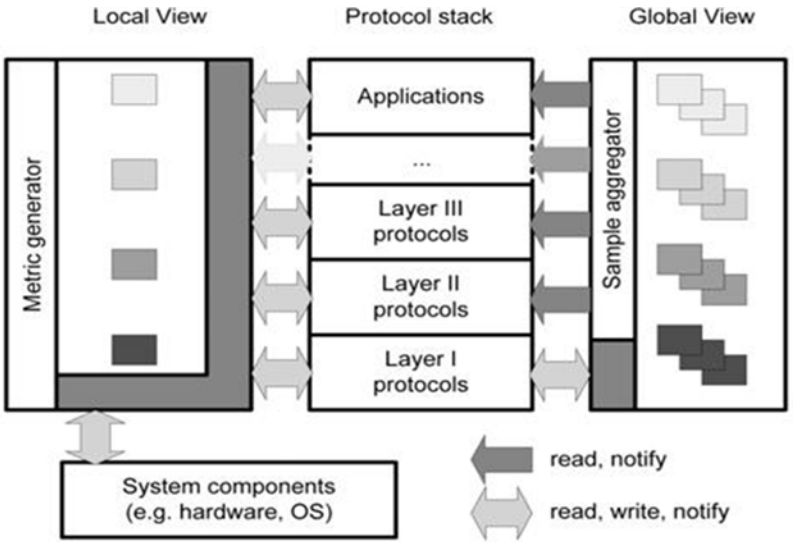


FIG. 2.8 – L'architecture de CrossTalk

mettre en oeuvre de la qualité de service. L'adaptation est double : le système de communication doit être adapté aux besoins de l'application (voir l'architecture GRACE présentée précédemment), mais l'application s'adapte également aux caractéristiques du système, (choix d'un codage adapté). On trouvera dans [SYZG05] une architecture de cross layer pour le transfert temps réel d'images dans les réseaux ad hoc. Plus récemment, le besoin d'économiser la puissance des équipements mobiles, (power control), a vu l'émergence de protocoles utilisant des informations autres que les leurs.

Nous illustrons par l'exemple l'approche cross layer en précisant les principales informations qui peuvent être échangées entre les couches selon la Figure 2.9.

2.1.4.1 Interactions couche physique et couches supérieures

L'information disponible à la couche physique est la puissance du signal transmis, le taux d'erreur et le codage/modulation des données.

Ce dernier paramètre peut être utilisé par la couche liaison de données pour adapter ses mécanismes de contrôle d'erreurs. La couche physique indique également la puissance du signal transmis afin qu'elle soit accordée par la couche MAC pour augmenter la portée de transmission. Le taux d'erreur peut être utilisé par la couche réseau comme une indication pour choisir, en cas d'interfaces multiples, l'interface de transmission la plus appropriée. D'une façon générale, les informations de la couche physique sont utilisées pour du contrôle de puissance, du routage à qualité de service. Notons que le routage à qualité de service peut utiliser plusieurs métriques comme proposé dans [SuHu05].

Dans le réseau 802.11 les informations de la couche physiques sont accessibles via la couche MAC. Un champ de la trame MAC fournit des indications de débit, qui sont directement liées à la qualité du signal reçu sur la liaison (le SNR). L'interface Xian est un exemple d'implantation permettant de récupérer ces données. Des exemples d'utilisation des informations de la couche MAC seront trouvés dans le projet MobileMan, pour définir des métriques de routage à qualité de service, ou dans [LoDP07], pour permettre à TCP de différencier les erreurs de transmission et éviter ainsi les diminutions inutiles de débit.

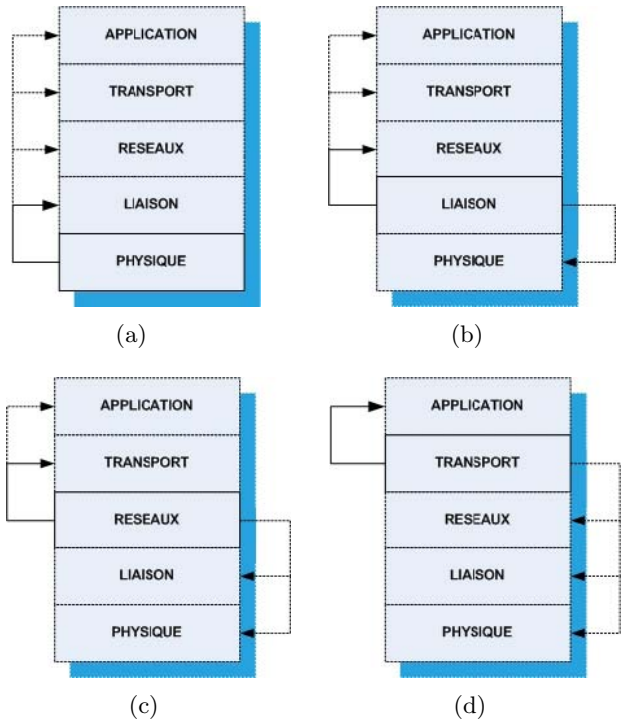


FIG. 2.9 – Exemple d'interactions entre les protocoles

2.1.4.2 Interactions couche liaison de données et couches supérieures ou inférieures

Dans la Figure 2.9 (b), l'information disponible à la couche liaison de données concerne le mécanisme de FEC (Forward Error Correction), la longueur de la trame, le nombre de trames retransmises.

La couche liaison de données peut adapter son schéma de correction d'erreur à base de FEC selon le délai et les pertes de paquet de la couche transport, ainsi que le taux d'erreur de la couche physique.

Notons que de nombreuses interactions ont été définies dans les réseaux sans fil avec infrastructure pour prendre en compte le déplacement inter cellule des utilisateurs. L'approche cross layer a d'ailleurs été intégrée dans les travaux de l'IETF pour améliorer les performances du service Mobile IP ([Perk02], [PeCB07]). La version FMIP (Fast Handovers for Mobile IPv6) ([Kood05]) utilise des informations provenant de la couche liaison de donnés pour détecter plus rapidement le déplacement de l'utilisateur. La normalisation de ces informations est en cours d'étude par l'IEEE chargé de la normalisation des réseaux locaux, et elle devrait pouvoir être réutilisée dans le contexte des réseaux ad hoc.

2.1.4.3 Interactions couche réseau et couches supérieures ou inférieures

Les informations de routage, tel l'échec de route que nous prenons en compte pour adapter le transport dans cette thèse sont utilisés comme métrique de mobilité. Les données de routage sont également utilisées par l'application comme dans [MeGo05] pour des cas de transfert multimédia ou de services de répertoires.

Des informations de routage peuvent également être utilisées par les couches inférieures pour les nouveaux systèmes de transmission dits coopératifs. Ceux-ci utilisent un relais afin d'augmenter la qualité de la transmission par une diversité spatiale. Plutôt que d'utiliser plusieurs antennes sur un même équipement, les antennes sont virtuellement réparties entre la source et des relais, ce qui permet de conserver des équipements de taille réduite. Les relais sont des noeuds voisins qui peuvent être connus par cross layer grâce au protocole de routage.

2.1.4.4 Interactions couche transport et couches supérieures ou couche inférieures

Dans la Figure 2.9 (d), des exemples d'informations disponibles à partir de la couche transport utilisables par des applications sont le temps d'aller/retour, le timer de retransmission, la taille des messages, la fenêtre du récepteur, la fenêtre de congestion et le nombre de paquets perdus. Ce type d'interaction a plus particulièrement été pris en compte pour optimiser les applications multimédia.

2.2 L'approche cross layer et les autres approches d'optimisation

Nous situons l'approche d'optimisation cross layer par rapport aux optimisations proposées dans la littérature. Notre proposition de classification est illustrée sur la Figure 2.10.

Nous considérons deux grandes classes parmi les propositions améliorant le transfert fiable dans les réseaux ad hoc. Premièrement, une classe d'optimisations effectuées au niveau des *couche inférieures* : elles proposent des améliorations protocolaires adaptées aux réseaux ad hoc, comme un protocole MAC diminuant le nombre de collisions proposé dans [GBZT99] ou encore un protocole de routage améliorant la stabilité de route dans [AASS00]. Deuxièmement, une classe d'*optimisation interne* à la couche qui agit au niveau transport.

Parmi les optimisations internes nous retenons trois méthodes d'optimisation selon l'objet de leur action :

1. Protocole
2. Algorithme
3. Evénement

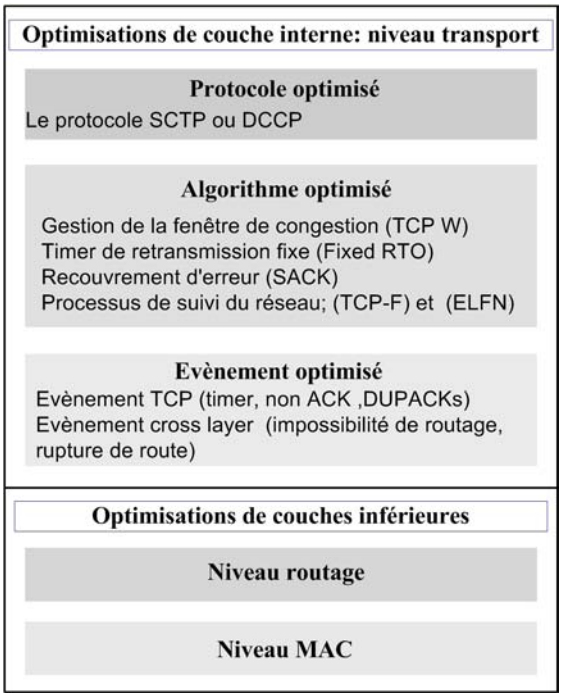


FIG. 2.10 – Classification des optimisations de transport

2.2.1 Méthodes d'optimisation

2.2.1.1 Protocole optimisé

Dans la mesure où le protocole TCP n'est pas adapté, une solution est alors de considérer un autre protocole de transport, qui lui pourra par exemple ne pas traiter de la fiabilité, tels SCTP ou DCCP, ou encore masquer le problème du réseau par un transfert multihoming disponible dans SCTP.

2.2.1.2 Algorithme optimisé

L'objectif est de conserver un protocole standard et d'améliorer ses performances par des algorithmes adaptés.

Les différents algorithmes qui peuvent être optimisés sont relatifs aux trois processus de base de TCP à savoir : la gestion de la fenêtre de congestion, le recouvrement d'erreur et la gestion du temporisateur. De nouveaux algorithmes peuvent également être introduits que ce soit dans les processus de base ou dans de nouveaux processus. Ainsi, parmi les différents transports que nous avons présentés dans le premier chapitre, nous pouvons considérer que TCP Westwood et TCP Sack introduisent de nouveaux algorithmes, pour les processus, respectivement, de gestion de la fenêtre de congestion et de recouvrement d'erreur, alors que, TCP-F et TCP-ELFN [HoVa02], définissent un nouveau processus de suivi du réseau (snooze ou standby).

Des exemples d'algorithmes optimisés pour les processus de base de TCP sont :

– **Recouvrement d'erreur** : utilisation d'une option duplicate ACK [VMPM99] ou d'une option sélective ACK [FMMP00] afin de réduire le temps de retransmission en

cas d'erreurs multiples. En réseau ad hoc, ce type d'erreur doit être pris en compte en raison des caractéristiques de la transmission sans fil,
– **Gestion de timer** : un timer de retransmission fixe [DyBo01] qui laisse à l'émetteur l'occasion de retransmettre le paquet perdu à intervalle périodique plutôt que d'attendre avant de retransmettre une période de temps qui croît avec le nombre d'échecs. S'il y a une rupture de route, il faut tester de façon périodique son rétablissement plutôt que d'attendre de plus en plus de temps avant de tester. Une autre proposition serait que, plutôt que d'attendre de plus en plus longtemps, attendre de moins en moins de temps : s'il y a longtemps que la rupture de chemin s'est produite, il y a de forte chance qu'elle soit bientôt réparée.

Des exemples de nouveaux processus pour TCP sont :

– **Processus de sondage** : le processus est chargé de surveiller l'état de réseau afin de détecter une période d'erreur de non-congestion, tel l'envoi de sondes dans ELFN.
– **Processus de pause** : un nouveau processus est chargé d'éviter les réactions classiques de TCP au problème de perte et lui éviter de diminuer le débit de transmission inutilement comme dans [BrSi97], [CRVP98] et [GMPG00].

2.2.1.3 Déclenchement d'algorithmes sur évènement optimisé

Cette méthode d'optimisation est généralement combinée avec des méthodes d'algorithmes optimisés. Par exemple, le processus de recouvrement d'erreur est déclenché sur un évènement timer, ou pour de meilleures performances (en filaire) sur la réception de 3 ACK. En réseau ad hoc, il est intéressant de définir au mieux les événements qui permettent de différencier la perte de transmission de celle de congestion, un événement sera spécifié pour détecter la perte de paquet due à une impossibilité de routage, à une rupture de chemin.

Nous séparons les événements internes des événements de cross layer, nommés ainsi car les processus TCP sont activés sur des événements produits par une autre couche. Des exemples d'événements internes sont, un retour vers la source de transmission d'informations issues de noeuds du réseau, tel un message TCP spécifique pour indiquer un échec de lien [HoVa02], ou bien un ACK absent non reçu avant la deuxième expiration de timer [DyBo01]. Concernant des exemples d'événements de cross layer, citons, le processus pause qui est activé par l'échec de lien ([CRVP98], [HoVa02]). L'émetteur est dans un mode pause, il gèle le timer et la fenêtre de congestion.

2.2.2 Gestion de la communication inter-couches

Nous considérons ci après les façons de mettre en oeuvre le cross layer.

2.2.2.1 Communication à l'initiative du noeud source

Les noeuds impliqués par le cross layer sont les noeuds sources. Cette technique peut être appliquée pour distinguer des pertes de paquets entre le niveau réseau et le niveau transport. Par exemple, comme nous le verrons plus précisément dans le chapitre 3, lorsque le noeud source reçoit un message d'erreur de route, il active un processus de pause.

2.2.2.2 Communication répartie

Cette technique implique plusieurs noeuds dans le processus de cross layer. Pour optimiser le protocole transport en diminuant l'impact du protocole de routage, c'est à dire l'impact du traitement des ruptures de route, une solution est de gérer la rupture de route au niveau du protocole de transport (par du multihoming). Dans ce cas, une interaction réseau transport est nécessaire sur tous les noeuds, car ceux-ci sont impliqués dans le processus de rétablissement de route et, s'il n'y a pas de communication interniveaux, le rétablissement qui est fait à deux niveaux à la fois peut amener, comme nous le verrons le chapitre 4, à des incohérences.

2.3 Intérêt de SCTP pour des optimisations inter-couches

2.3.1 Pourquoi SCTP

SCTP tient compte de l'expérience acquise avec TCP pour améliorer certains aspects de la communication fiable. Par exemple, l'établissement d'une connexion (que SCTP nomme association) se fait avec un échange de quatre paquets (et non pas trois comme avec TCP), pour offrir une meilleure protection contre les dénis de service. Les SYN cookies, un ajout de sécurité en TCP, sont ici partie intégrante du protocole.

Les deux innovations majeures de SCTP sont :

- La livraison séquencée de messages utilisateurs dans des flux multiples, avec une option individualisée par flux concernant le respect de l'ordre de livraison d'arrivée des messages utilisateurs,
- La livraison de messages par des interfaces multiples. SCTP, par son option de multihoming, gère l'association entre des équipements multidomiciliés, c'est à dire possédant plusieurs adresses internet, ou, entre un équipement émetteur multidomicilié d'un coté, et un équipement sans multihoming de l'autre côté. Chaque terminal fournit, pendant l'établissement de l'association, une liste d'adresses transport (c'est-à-dire, plusieurs adresses IP combinées avec des numéros de ports SCTP). L'association se charge des transferts par toutes les combinaisons source/destination, qui pourraient être produites à partir des listes fournies par les terminaux. Cette caractéristique améliore fortement la robustesse du système, puisqu'en cas de panne sur une interface, le trafic est automatiquement basculé sur une autre.

Des travaux effectués dans les réseaux traditionnels, confirment que le multihoming améliore les performances du transfert de données de bout en bout dans le cas des pannes de réseau [CaAS06]. Sachant que les réseaux ad hoc sont assujettis à des pannes [SrMo05], il parait naturel de vouloir utiliser du multihoming et donc SCTP, qui possède cette caractéristique et a l'avantage d'être un standard IETF. Nous verrons dans les chapitres 3 et 4, les adaptations de cross layer que nous proposons pour SCTP.

2.3.2 A propos des différences de contrôle entre TCP et SCTP

De nombreuses évaluations du protocole SCTP comparant ses performances avec celles de TCP sont disponibles dans la littérature [AlAI02] et [CSIA03]. Si globalement les deux protocoles sont équivalents, certaines variations dans l'affectation des paramètres peuvent

expliquer les différences de performance constatées.

Nous précisons dans cette section les paramètres de SCTP en relevant les divergences des spécifications initiales et des spécifications plus récentes. Ce sont ces dernières que nous avons appliquées dans cette thèse.

2.3.2.1 Modifications sur le contrôle de congestion de SCTP

Le document RFC 4460 [SAPC06], contient une compilation de tous les défauts trouvés dans le document de base de la spécification de SCTP, le RFC 2960. Ces défauts peuvent être de nature éditoriale ou technique. Considéré comme un document d'accompagnement pour l'implantation de SCTP, il clarifie les imprécisions et corrige des erreurs du document original.

Le contrôle de congestion de SCTP est globalement équivalent à celui de TCP. Les premiers travaux publiés dans la littérature pour comparer le protocole SCTP avec le protocole TCP ont été menés à partir du document RFC 2960. Nous précisons ci-après les principales divergences avec le document RFC 4460, que nous avons prises en compte dans nos évaluations en mettant à jour le module correspondant dans le simulateur. Les nouvelles versions du simulateur NS prennent en compte partiellement ces modifications, nous recommandons vivement de vérifier les points ci après :

Taille de la fenêtre de congestion

Limite le rythme auquel l'expéditeur est autorisé à émettre ses données sur le réseau.

1. La valeur initiale de CWND avant la transmission de données ou après une période à vide suffisamment longue doit être :

- **Ancien texte**, **2*MTU**.
- **Nouveau texte**, **min(4*MTU, max (2*MTU, 4380 bytes))**.

2. Quand le point terminal ne transmet pas d'informations sur une adresse de transport donnée, la valeur de CWND associée à l'adresse de transport concernée doit être ajustée :

- **Ancien texte**, **max(CWND/2, 2*MTU) per RTO**.
- **Nouveau texte**, **max(CWND/2, 4*MTU) per RTO**.

3. Lors de la détection des pertes de paquet SACK, un point terminal fait ce qui suit :

- **Ancien texte**, **ssthresh = max(CWND/2, 2*MTU)** et **CWND = ssthresh**.
- **Nouveau texte**, **ssthresh = max(CWND/2, 4*MTU)** et **CWND = ssthresh**.

4. Une perte de paquet provoque la diminution de moitié de la valeur de CWND. Quand le temporisateur noté T3-rtx expire sur une adresse, SCTP effectue un démarrage lent :

- **Ancien texte**, **ssthresh = max(CWND/2, 2*MTU)** et **CWND = 1*MTU**.

– **Nouveau texte, ssthresh = max(CWND/2, 4*MTU)** et **CWND = 1*MTU**.

L'algorithme de retransmission rapide

Description du problème : La spécification originale de SCTP nécessite la réception de 4 ACKs avec le même numéro pour retransmettre rapidement un segment. Quand au protocole TCP, il s'appuie sur seulement 3 ACKs (Figure 2.11). De façon à pouvoir comparer les deux protocoles, en leur donnant un fonctionnement équivalent, nous avons modifié la valeur de détection de perte de SCTP et l'avons positionnée à la valeur de TCP : 3 ACKs.

Quand un point terminal détecte un trou de numérotation dans l'ordre d'arrivée des numéros de séquence TSN, il transmet un SACK à chaque réception de paquet de données jusqu'à ce que le trou soit comblé. Toutes les fois qu'un point terminal reçoit un SACK qui indique l'absence de paquets correspondants aux TSN manquants, il doit attendre encore 2 indications de manque (par réception de SACKs) sur le même TSN avant d'agir avec une retransmission rapide.

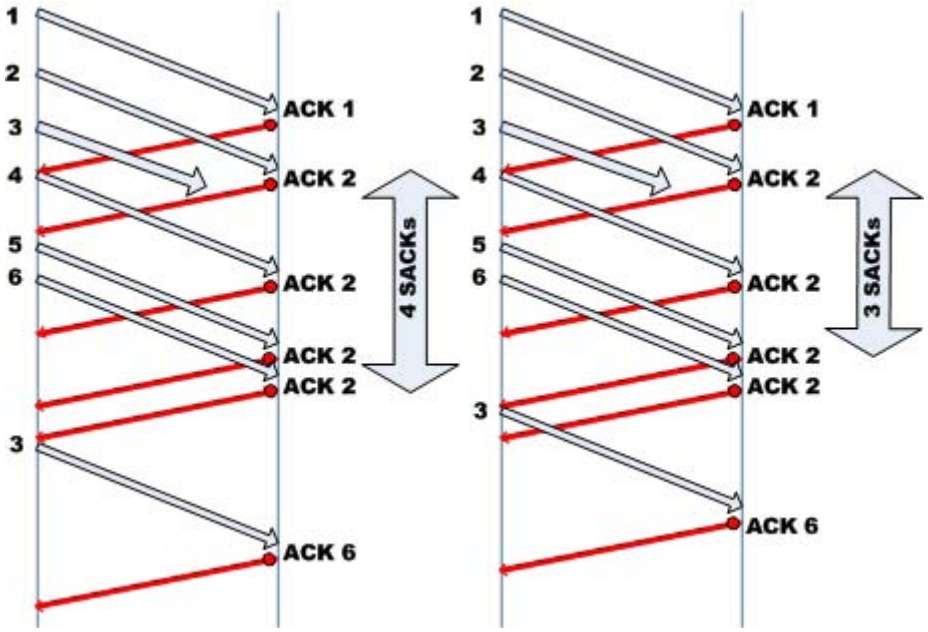


FIG. 2.11 – L'algorithme de la retransmission rapide

SCTP NewReno : Fast retransmit amélioré

Afin de résoudre les problèmes de Reno lorsque des pertes multiples ont lieu, une modification du *fast retransmit* a été introduite dans la version NewReno qui est utilisée dans SCTP. Ce changement consiste à mémoriser (variable recover) le numéro de séquence maximal envoyé, puis à utiliser cette information pour décider de la retansmission rapide d'autre segments. La procédure s'arrête lorsque tous les octets transmis ont été acquittés. Nous comparons SCTP et TCP en NewReno.

2.3.2.2 Gestion de timer de retransmission de SCTP

SCTP utilise le timer de retransmission (T3-rtx) pour assurer la livraison de données en l'absence de réception de messages. La durée de ce timer est notée RTO. Dans le cas d'un réseau multi domicilié, le réseau calculera un RTO séparé pour chaque adresse de destination différente.

Calcul de RTO

Les règles du calcul de SRTT (Smooth Round Trip Time), RTTVAR (Round Trip Time VARiance), et RTO sont légèrement modifiées par rapport à celles de TCP (variable beta de l'algorithme TCP prise à 4 dans l'algorithme SCTP).

1. Jusqu'à ce qu'une mesure de RTT soit disponible pour un paquet envoyé à l'adresse de destination, RTO a pour valeur celle du paramètre "RTO.Initial".
2. Quand la première mesure de RTT, R, est disponible :

   SRTT <- R,

   RTTVAR <- R/2, et

   RTO <- SRTT + 4 * RTTVAR.
3. RTO.Alpha = 1/8 et RTO.Beta = 1/4, quand une nouvelle mesure de RTT, R', est disponible,

   RTTVAR <- (1 - RTO.Beta) * RTTVAR + RTO.Beta * |SRTT - R'| et

   SRTT <- (1 - RTO.Alpha) * SRTT + RTO.Alpha * R'.

   Après le calcul, la mise à jour est : RTO <- SRTT + 4 * RTTVAR.

Règles de T3-rtx

1. Chaque fois qu'une donnée est envoyée à une adresse quelconque (y compris une retransmission), si le timer T3-rtx de cette adresse n'est pas armé, il doit être armé.
2. Quand toutes données envoyées à une adresse ont été acquittées, le T3-rtx de cette adresse est arrêté.
3. Quand un acquittement sélectif est reçu par la destination, le T3-rtx pour cette adresse doit être réarmé avec la valeur de RTO.

Expiration T3-rtx

1. Pour l'adresse de destination pour laquelle le timer expire, ajuster la variable de seuil ssthresh avec la régle CWND <- MTU.
2. Pour l'adresse de destination pour laquelle le timer expire, augmenter la valeur RTO : RTO <- RTO * 2.

2.3.3 Comparaison de performances entre TCP et SCTP

Dans l'article [KuJa04], les performances de TCP apparaissent meilleures que celles de SCTP dans les MANETs. Ceci provient des versions comparées, en particulier, de l'utilisation d'une option SACK pour SCTP, coûteuse en overhead, et d'une option ACK pour TCP. Dans cette section, nous montrons l'intérêt d'avoir des paramètres similaires pour SCTP et TCP et déterminons les valeurs de ceux-ci.

Nous utilisons un modèle très simple, comprenant une source et un destinataire (voir la Figure 2.12) et définissons trois scénarios de simulation sans perte pour tester l'implantation NS.

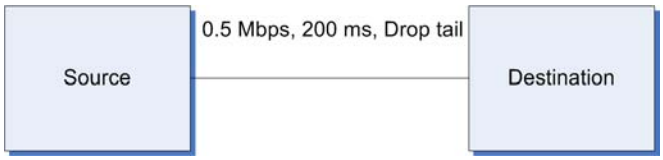


FIG. 2.12 – Scénario entre une source et une destination

Scénario 1 : TCP et SCTP utilisent les paramètres par défaut du simulateur NS2.

SCTP

- MTU = 1500 octets.
- Taille des données = 1468 octets.
- Taille de l'en-tête = 32 octets.
- Taille de la file d'attente de réception = 65536 octets.

TCP

- MTU = 1040 octets.
- Taille des données = 1000 octets.
- Taille de l'en-tête = 40 octets.
- Taille de la fenêtre = 30 paquets.

La taille de la file d'attente de réception de TCP est obtenue par l'émetteur dans la limite de la taille de fenêtre. Pour SCTP, la taille de fenêtre est la taille de la file d'attente de réception / la taille de MTU.

Dans la première simulation nous cherchons à découvrir l'impact de la taille de la MTU sur les performances des protocoles TCP et SCTP. A cet effet, nous gardons les caractéristiques normales de transmission de données qui incluent une taille de MTU de 1500 octets dans SCTP et une taille de MTU de 1040 octets avec TCP. Les résultats observés sont indiqués sur la Figure 2.13 qui représente l'évolution de la taille de la fenêtre avec le temps de simulation. Nous avons utilisé un temps de simulation de 100 secondes.

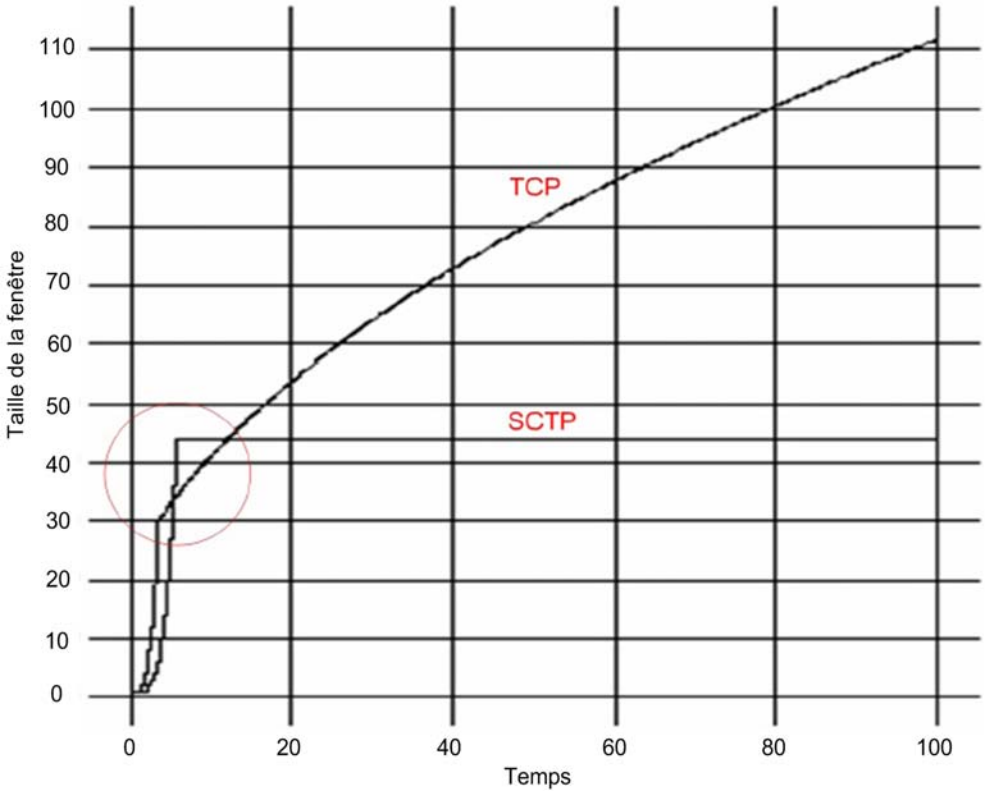


FIG. 2.13 – Scénario 1 : utilisation des valeurs par défaut

Résultat :

Nous pouvons voir que la taille de fenêtre de SCTP atteint un seuil plus important que celle de TCP avant que celle-ci ne passe en progression linéaire. De plus, alors que TCP continue à augmenter sa fenêtre, SCTP reste avec une fenêtre constante car il est limité par la fenêtre de réception (65536 octets).

Dans le deuxième scénario nous modifions les paramètres par défaut : la taille de la MTU est équivalente pour les deux protocoles, la taille de la fenêtre de TCP est choisie pour obtenir un seuil de basculement (progression exponentielle / progression linéaire) entre SCTP et TCP équivalent. D'après le scénario 1 cette valeur est de 44 paquets.

Scénario 2 : Nous changeons le MTU et la taille de la fenêtre de TCP.

SCTP

- MTU = 1500 octets.
- Taille des données = 1468 octets.
- Taille de l'en-tête = 32 octets.
- Taille de la file d'attente de réception = 65536 octets.

TCP

- MTU = **1500** octets.
- Taille des données = **1460** octets.
- Taille de l'en-tête = 40 octets.
- Taille de la fenêtre = **44** paquets.

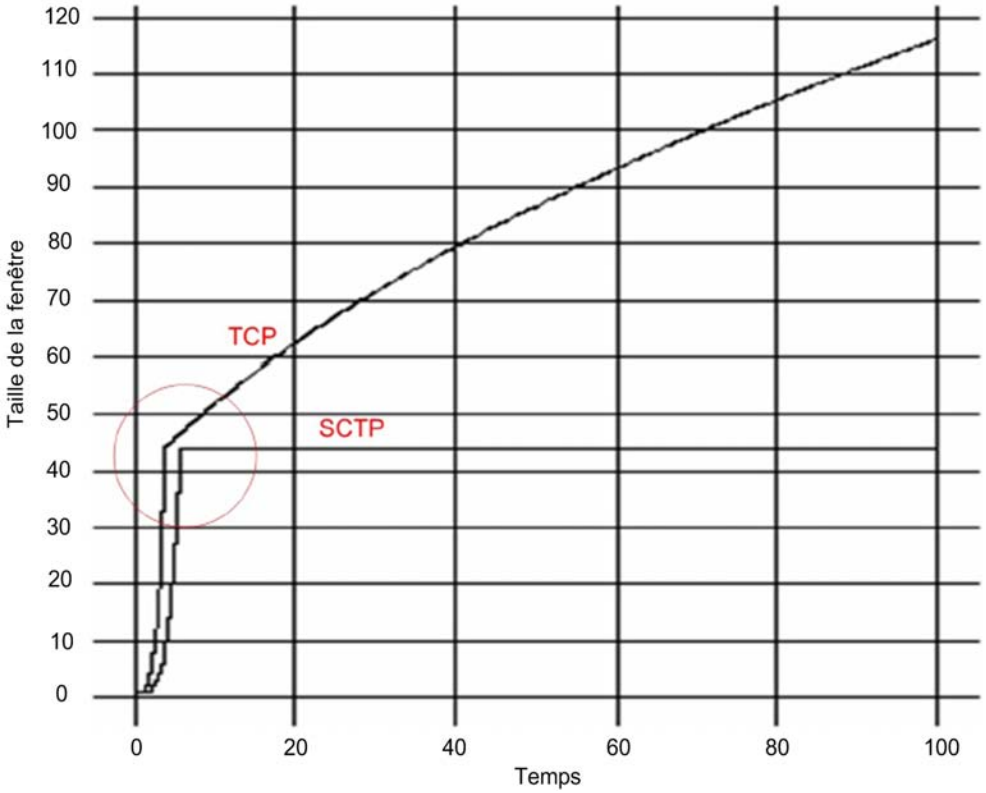


FIG. 2.14 – Scénario 2 : Changement des paramètres MTU et taille de fenêtre TCP

Résultat :

Les résulats illustrés dans la Figure 2.14, montrent un comportement d'émission en phase d'initialisation similaire. L'évolution des fenêtres des deux protocoles, quoique pas tout à fait identiques sont plus proches que précédemment. L'étape suivante est de modifier les paramètres de SCTP pour obtenir un accroissement linéaire équivalent à celui de TCP.

Scénario 3 : Nous changeons la taille de la file d'attente de réception de SCTP.

SCTP

- MTU = 1500 octets.
- Taille des données = 1468 octets.

– Taille de l'en-tête = 32 octets.
– Taille de la file d'attente de réception = 65536 x 2 = **131072** octets.

TCP

– MTU = 1500 octets.
– Taille des données = 1460 octets.
– Taille de l'en-tête = 40 octets.
– Taille de la fenêtre = 44 paquets.

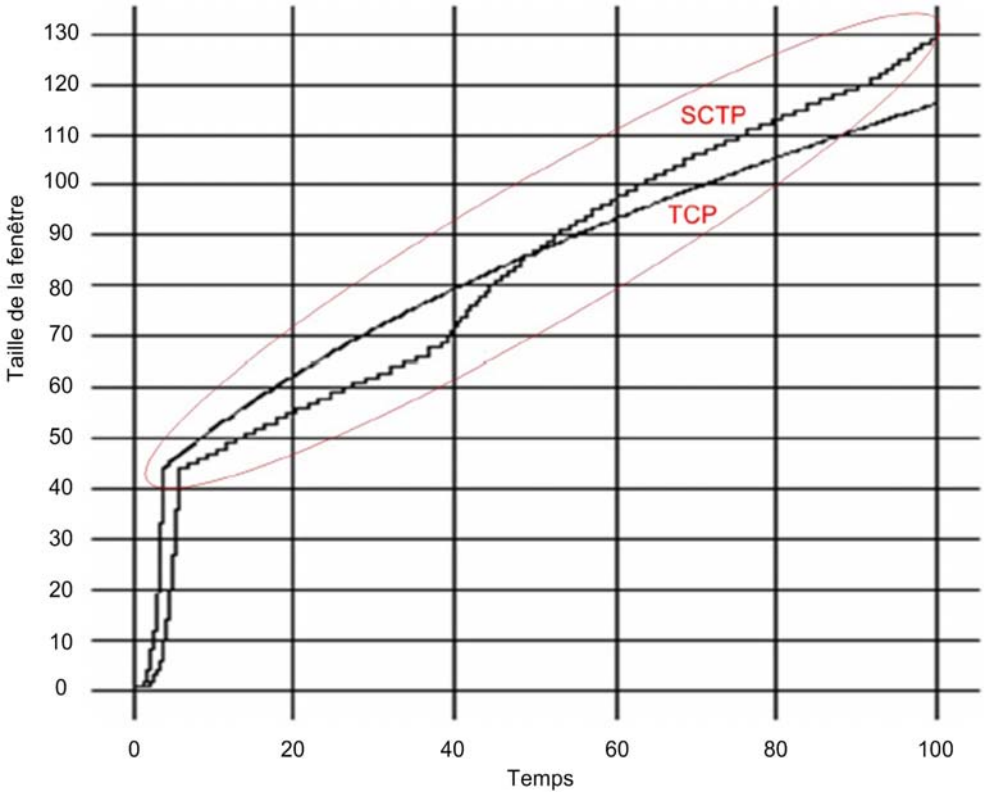


FIG. 2.15 – Scénario 3 : Changement du paramètre taille de la file d'attente de réception SCTP

Résultat :

Dans ce dernier scénario **nous obtenons un comportement similaire de SCTP, et de TCP** (voir la Figure 2.15). Notons que si la taille de la queue de réception du récepteur est suffisamment grande pour garder les paquets pendant le temps qu'un paquet perdu soit retransmis, la performance est alors améliorée.

2.4 Conclusion

Dans ce chapitre nous avons synthétisé les éléments de base des optimisations pour améliorer les performances du transport en réseaux ad hoc. Nous nous sommes intéressés à la méthode, au protocole et, aux paramètres permettant d'obtenir des évaluations cohérentes.

La méthode cross layer, s'appuie sur des échanges d'informations entre des couches qui ne sont pas forcément adjacentes, la performance globale est optimisée en adaptant chaque niveau en fonction de l'information disponible. Les architectures protocolaires, pour mettre en oeuvre les mécanismes d'adaptation, s'inspirent de trois modèles : communication directe entre couches, interactions vers une entité intermédiaire et nouvelles abstractions. Dans les chapitres suivants, nous utilisons le modèle d'interactions vers une entité intermédiaire. Celui-ci sera géré à la source dans le chapitre 3 et de façon répartie dans le chapitre 4. Le modèle choisi simplifie les évolutions d'architectures, par rapport à un modèle de communication directe entre couches, tout en préservant le découpage fonctionnel de l'architecture standard qui est remis en cause dans le modèle par composants. En nous référant à la classification des optimisations que nous avons effectuée, nos propositions se situent au niveau transport, sur des événements optimisés inter-couches, et des algorithmes optimisés.

Le protocole de transport sur lequel nous travaillons est SCTP en raison de sa capacité à gérer le multihoming. Quoique très proche de SCTP nous avons relevé des différences, en particulier sur les paramètres de congestion. Nous avons déterminé leurs valeurs pour obtenir des évaluations de performances pertinentes.

Chapitre 3

Optimisation de SCTP par communication inter-couches à l'initiative du noeud source

Sommaire

Nous avons précédemment dégagé l'intérêt d'améliorer les performances des protocoles par une méthode de cross layer. Celle-ci peut être mise en oeuvre de deux façons. La première est effectuée localement, à l'initiative du noeud source, alors que la seconde est répartie sur différents noeuds. Dans ce chapitre nous proposons et évaluons l'optimisation que nous nommons SCTP PAUSE qui relève du premier type d'optimisation. L'optimisation répartie sera présentée dans le chapitre suivant.

Au vu des résultats que nous avons obtenus dans le chapitre 1, concernant l'importance sur les performances de bout en bout du protocole de routage, nous avons choisi de travailler sur une optimisation inter-couches de niveau 3/4 ; le niveau transport utilise des informations du niveau réseau. Il s'agit d'adapter la transmission des informations à la mobilité du réseau, en bloquant l'émission et le processus de congestion en cas de mobilité, détectée par le niveau réseau.

3.1 Fonctionnement de SCTP

Le fonctionnement de base de SCTP est représenté sur la Figure 3.1.

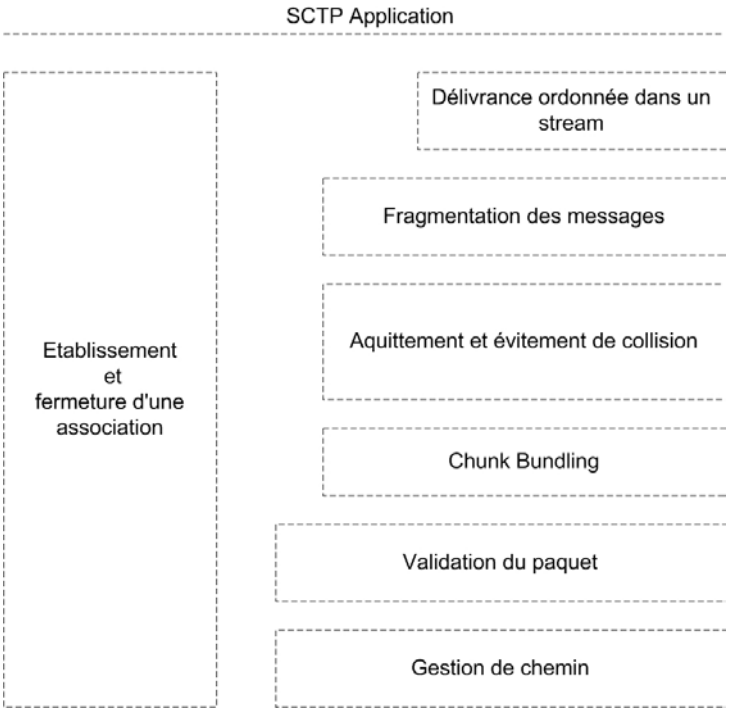


FIG. 3.1 – Fonctionnement de base de SCTP

Les sept blocs fonctionnels du protocole sont :

1. Etablissement et fermeture d'une association : SCTP établit une association avant d'émettre des messages. Lorsque SCTP stoppe l'acceptation des données, il initie, à la demande de l'application, la fermeture de l'association.
2. Délivrance ordonnée dans un stream : un stream dans le contexte de SCTP se réfère à une séquence de messages utilisateurs. Une association de SCTP peut soutenir de multiples streams.
3. Fragmentation des messages : SCTP fragmente les messages utilisateur pour s'assurer que la taille des paquets passés à la couche inférieure ne dépasse pas le MTU (Maximum Transmission Unit). Les fragments reçus sont réassemblés avant d'être transmis à la couche supérieure.
4. Acquittement et évitement de congestion : la fonction gère la fenêtre d'émission et retransmission, conformément aux algorithmes TCP de congestion avoidance présentés précédemment (cf chapitre 1).
5. Chunk Bundling : le paquet SCTP est formé de plusieurs entités (chunk), cette fonction gère l'assemblage du paquet SCTP et son réassemblage en réception.
6. Validation du paquet : protection des attaques de sécurité et des paquets de SCTP "périmés" provenant d'une association antérieure.
7. Gestion de chemin dans le multihoming : surveillance de l'état des chemins multiples que peut emprunter l'association.

Remarque : nous utilisons la terminologie de l'IETF pour les termes message (ce qui est manipulé par le protocole SCTP) paquet (ce qui est transporté dans un paquet IP).

Nous détaillons à la suite les fonctions (5) (Chunk Bundling) (6) (Validation de paquets) et (1)(Etablissement de connexion). Les fonctions (2) et (4) sont des fonctions classiques que nous avons déjà présentées ; la fonction (3) est analogue aux fonctions de fragmentation standard, sa mise en oeuvre est illustrée au travers du format des unités transmises ; la fonction (7) sera présentée dans le chapitre suivant.

3.1.1 Chunk Bundling : Format de Messages SCTP

Le paquet SCTP est composé d'une **en-tête commune** et de modules spécifiques appelés **chunks** (voir Figure 3.2).

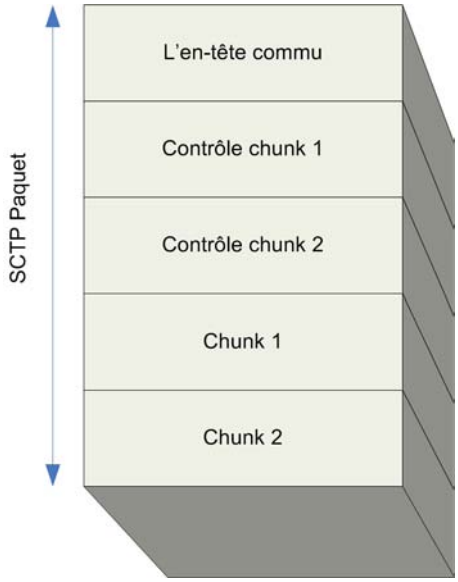


FIG. 3.2 – Le paquet SCTP

L'en-tête commune

Chaque paquet SCTP contient une en-tête commune qui a 4 champs différents (voir Figure 3.3). L'en-tête comporte 12 octets. Pour l'identification d'une association, SCTP utilise le même concept de port que TCP et UDP. Pour la détection d'erreurs de transmission, chaque paquet de SCTP est protégé par une somme de contrôle (checksum) à 32 bits (Adler-32 algorithme ou 32-bit CRC checksum), qui est plus robuste que le checksum à 16 bits de TCP et UDP. L'en-tête contient également une valeur à 32 bits appelée balise de vérification (verification tag). La balise de vérification est utilisée pour se prémunir contre l'insertion de vieux messages erronés ; elle est échangée à l'établissement de l'association.

Chunks

Les chunks sont divisés en deux classes : chunk de contrôle et chunk de données (voir Figure 3.4). Un chunk de contrôle est par définition un chunk qui contient des informations propres au contrôle et au maintien de l'association SCTP. Les chunks de données sont utilisés pour transporter des messages utilisateurs à travers l'association.

La Figure 3.5 détaille le format d'un chunk de données ;

FIG. 3.3 – En-tête commune

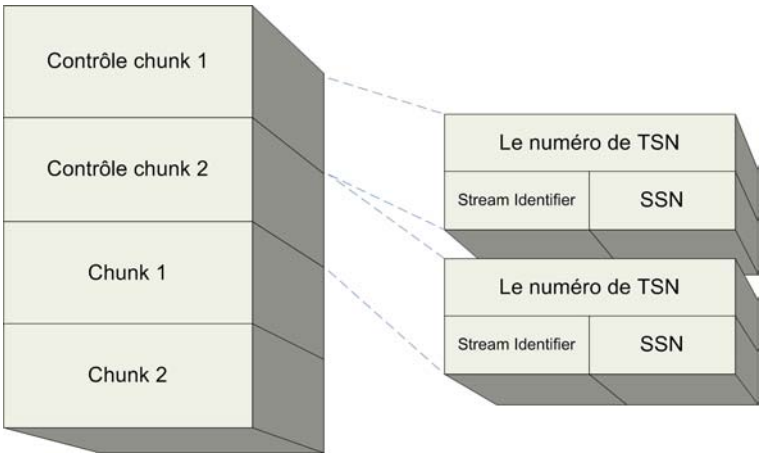


FIG. 3.4 – Les chunks : chunks de contrôles et de données

– Transmission Sequence Number (TSN) : est renvoyé dans un chunk de contrôle pour chaque chunk de données reçu ; le récepteur utilise TSN pour la détection de perte et la duplication de données.
– Identifiant de stream (SI : Stream Identifier) : identifie les flux des chunks de données Ceci est utilisé que lorsque SCTP transporte plusieurs flux dans une association (multistreaming).
– Numéro de séquence de Stream (SSN : Stream Sequence Number) : numéro de séquence pour chaque stream. Ces nombres sont utilisés au récepteur pour déterminer l'ordre de livraison.
– U, vient de "(U)nordered" : si cette valeur égale 1, alors ceci est un chunk de données non ordonné. Il n'y a pas de numéro de séquence de Stream pour cette donnée.
– B, vient de "(B)eginning fragment" : indique le premier fragment d'un message utilisateur.
– E, vient de "(E)nding fragment" : indique le dernier fragment d'un message utilisateur. Les valeurs combinées de B et E sont indiquées dans la table 3.1.
– Payload Protocol Identifier : précise un identifiant de l'application. Cet identificateur n'est pas utilisé par SCTP mais peut être utilisé par certains réseaux. Si cette valeur égale 0, alors aucun identifiant d'application n'est spécifié par la couche supérieure pour ces données.

Les types de Chunks

Le standard de l'IETF RFC 4960, spécifie différents types de chunks identifiés de

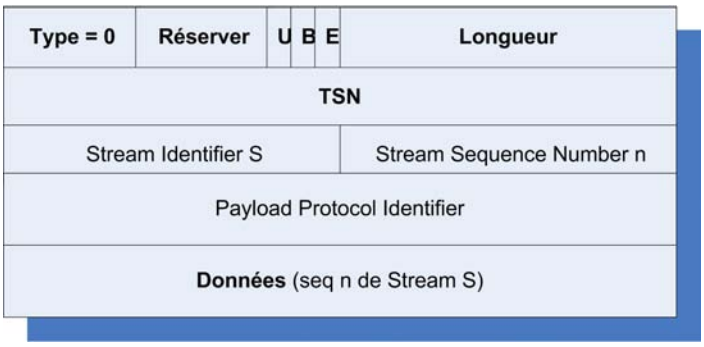


FIG. 3.5 – Chunk de données

B	E	Description
1	0	Le premier chunk d'un message d'utilisateur fragmenté
0	0	Le chunk milieu d'un message d'utilisateur fragmenté
0	1	Le dernier chunk d'un message d'utilisateur fragmenté
1	1	Pas faire de fragment de message

TAB. 3.1 – Détails de B et E dans le chunk de données

0 à 254. La valeur 255 est réservée pour une utilisation ultérieure, comme un champ d'extension (voir le tableau 3.2).

3.1.2 Validation de paquet et sécurité

3.1.2.1 Le mécanisme de cookie

Un mécanisme de cookie est utilisé pendant l'établissement de l'association, pour assurer la protection contre des attaques de sécurité. Le cookie est créé par le destinataire et les clés secrètes restent chez le destinataire.

Le mécanisme de cookie protège des attaques à l'aveugle (en anglais blind attack) qui émettent de nombreux chunk INIT d'établissement d'association, pour monopoliser les ressources du destinataire et provoquer ainsi un déni de service. Au lieu d'allouer directement la mémoire pour un Transmission Control Block (TCB), le destinataire SCTP crée un cookie avec l'information de TCB et l'envoie avec un message INIT-ACK. L'émetteur, malveillant de l'attaque à l'aveugle qui généralement se déplace de machine en machines, ne peut pas obtenir de cookie, parce que INIT-ACK est renvoyé à l'adresse source du message INIT. L'émetteur bienveillant obtient le cookie et le retourne dans le COOKIE-ECHO, où le destinataire peut valider le cookie et l'utiliser pour reconstruire le TCB.

3.1.2.2 La balise de vérification

Une balise de vérification est utilisée pour la protection contre les attaques de mascarade. Les règles de balise de vérification s'appliquent à l'émission et à la réception des paquets de SCTP qui ne contiennent pas de messages utilisés pour l'établissement et la libération d'une association (INIT, SHUTDOWN-COMPLETE, COOKIE-ECHO, ABORT, et SHUTDOWN-ACK). La valeur de balise de vérification est choisie par chaque extrémité de l'association pendant l'établissement d'une association.

Identifiant	**Nom de chunk**
0	Données utiles (DATA)
1	Initialisation (INIT)
2	Accusé-réception d'initialisation (INIT-ACK)
3	Accusé-réception sélectif (SACK)
4	Requête de chemin en vie (HEARTBEAT)
5	Accusé-réception de chemin en vie (HEARTBEAT-ACK))
6	Abondon (ABORT)
7	Fermeture (SHUTDOWN)
8	Acccusé-réception de fermeture (SHUTDOWN-ACK)
9	Erreur d'opération (ERROR)
10	Témoin d'état (COOKIE-ECHO)
11	Accusé-réception de témoin (COOKIE-ACK)
12	Réservé pour écho de notification de congestion explicite (ECNE)
13	Réservé pour fenêtre de congestion réduite (CWN)
14	Arrêt complet (SHUTDOWN-COMPLETE)
15-62	Réservé pour l'IETF
63	IETF-Blocs d'extension définis
64-126	Réservé pour l'IETF
127	IETF-Blocs d'extension définis
128-190	Réservé pour l'IETF
191	IETF-Blocs d'extension définis
192-254	Réservé pour l'IETF
255	IETF-Blocs d'extension définis

TAB. 3.2 – Les types de chunks

Après réception d'un paquet SCTP, chaque extrémité doit s'assurer que la valeur dans le domaine de balise de vérification du paquet reçu correspond à sa propre balise. Si la valeur reçue ne correspond pas, le récepteur détruit le paquet.

3.1.3 Etablissement et fermeture d'une association

3.1.3.1 Etablissement d'une association

L'établissement d'une association est effectué après échange de quatre messages (Figure 3.6).

1. L'émetteur envoie un message INIT au récepteur pour initialiser une association.
2. A la réception du message INIT, le récepteur envoie une réponse INIT-ACK à l'émetteur. Ce message INIT-ACK contient les informations de configuration qui constituent un COOKIE.
3. A la réception de ce message INIT-ACK, l'émetteur envoie une réponse COOKIE-ECHO.
4. Le récepteur analyse les informations contenues dans le paquet COOKIE-ECHO reçu. Il vérifie ensuite la validité de la clé afin de s'assurer qu'il est bien l'émetteur d'origine de ce message de COOKIE. Si la clé est valide, le récepteur renvoie un message de COOKIE-ACK à l'émetteur et considère l'association comme établie.

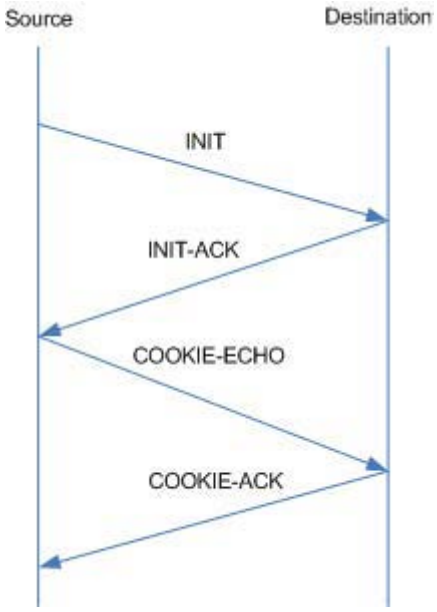


FIG. 3.6 – L'établissement d'une association SCTP

3.1.3.2 Fermeture d'une association

La fermeture de SCTP utilise une procédure en trois messages (Figure 3.7) :

1. L'émetteur envoie un message SHUTDOWN au récepteur, qui indique que le client est prêt à fermer la connexion.
2. Le récepteur répond en envoyant un message de SHUTDOWN-ACK.
3. L'émetteur envoie alors un message de SHUTDOWN-COMPLETE en retour au récepteur.

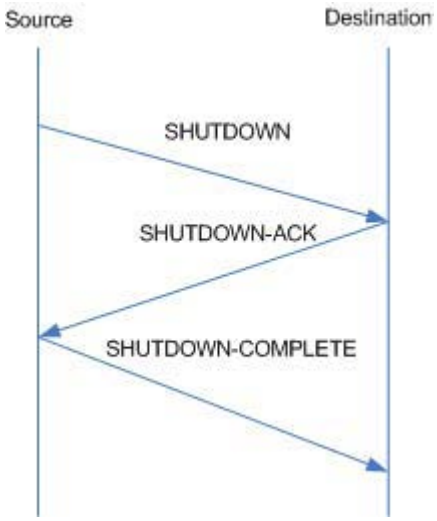


FIG. 3.7 – La fermeture d'une association SCTP

3.2 SCTP PAUSE

3.2.1 Idée de base

Nous voulons distinguer les pertes de paquets qui sont dues à la congestion de celles qui sont dues à un échec de routage. Pour cela nous nous proposons d'utiliser au niveau 4 une information de niveau 3 propre au protocole de routage. Cette information ne sera utilisée que sur la source.

Les protocoles de routage DSR et AODV ont tous deux une procédure de maintenance de route qui génère une information **erreur de route**. C'est celle-ci que nous utilisons. Nous avons choisi d'utiliser AODV car, au vu des simulations effectuées dans le premier chapitre, il apparaît que les performances de routage de AODV sont meilleures que celles de DSR pour 50 noeuds et nous souhaitons avoir suffisamment de noeuds dans le réseau pour obtenir des erreurs de route. Notons que les travaux de la littérature utilisent majoritairement des simulations de 50 noeuds, ce qui nous permet également de vérifier la cohérence de nos résultats (les informations fournies sur les paramètres de simulations ne sont pas suffisamment précises pour réellement comparer les résultats).

Nous nommons l'optimisation SCTP PAUSE. Son comportement en regard de la version standard SCTP est illustré sur la Figure 3.8. Nous y indiquons, dans un réseau ad hoc à deux sauts, comportant deux routes vers la destination, l'évolution de la fenêtre d'émission en cas de rupture de route. Nous présentons cette évolution en 3 étapes :

1. Premièrement, l'émetteur envoie ses paquets au récepteur en utilisant un noeud relais. La taille de la fenêtre d'émission de SCTP et de celle de SCTP PAUSE évoluent de façon similaire.

2. Dans une deuxième temps, le noeud intermédiaire se déplace provoquant une rupture de route avec pertes de paquets. Le réseau lance une procédure de découverte de route pour chercher une nouvelle route tandis que le niveau transport réduit la

fenêtre de congestion. La version SCTP PAUSE utilise la notification d'erreur pour bloquer sa fenêtre, et faire une pause dans l'augmentation de sa fenêtre, jusqu'à ce que la nouvelle route soit établie (information de route reply obtenue par communication entre AODV et SCTP). Ainsi la taille de la fenêtre n'est pas diminuée sur la nouvelle route.

3. Enfin, la source peut envoyer les paquets sur le nouveau chemin. La taille de la fenêtre de SCTP est moins importante que celle de SCTP PAUSE.

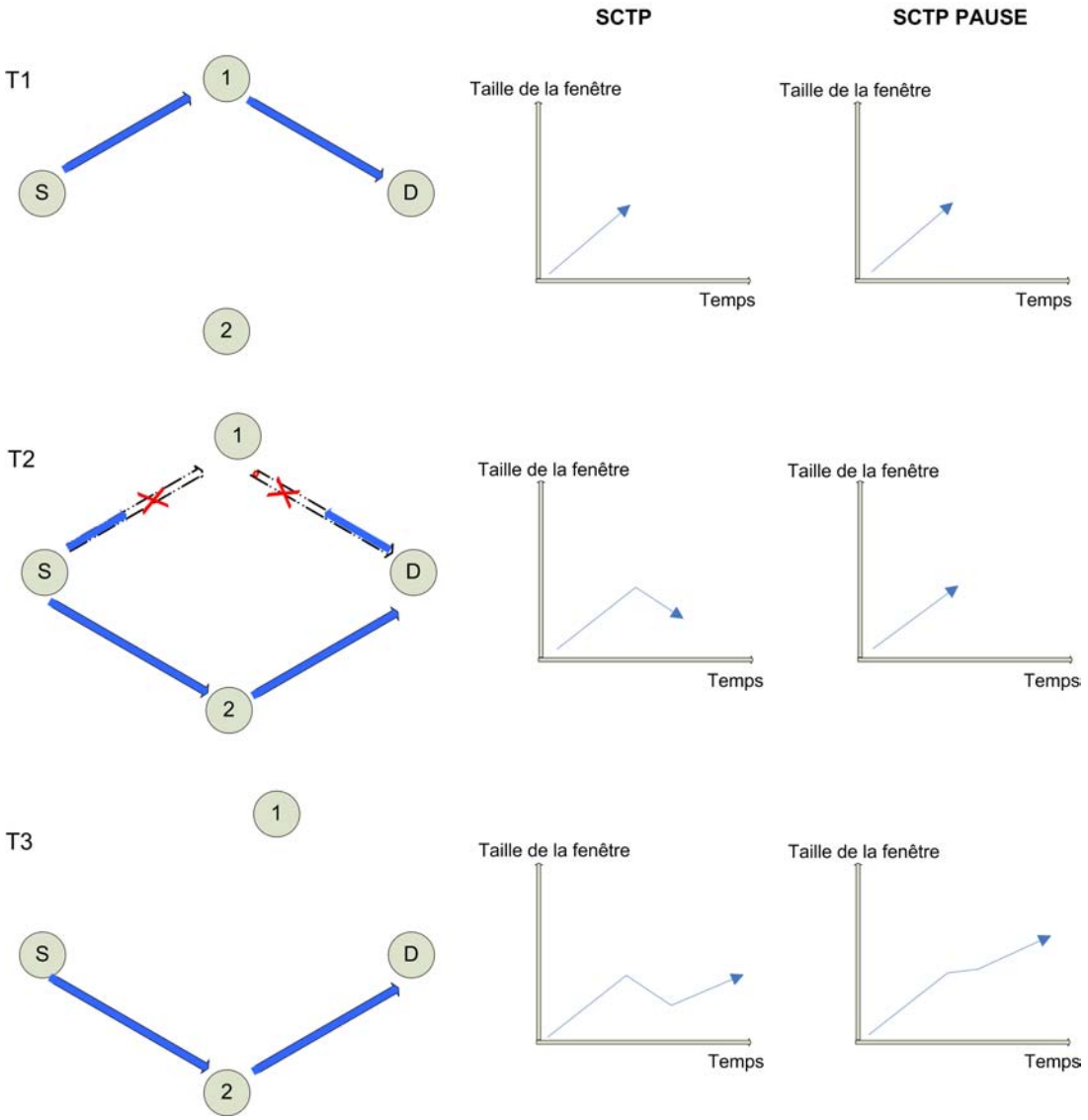


FIG. 3.8 – Idée de base pour SCTP PAUSE

3.2.2 Mise en oeuvre avec le protocole AODV

La méthode d'optimisation SCTP PAUSE utilise le processus d'erreurs de route de AODV. Nous détaillons son fonctionnement : à l'établissement de l'association puis en cas de rupture de route.

3.2.3 Etablissement d'une association entre les protocoles SCTP et AODV

SCTP établit l'association par les 4 messages (INIT, INIT-ACK, COOKIE-ECHO et COOKIE-ACK), AODV utilise 2 messages (Route Request et Route Reply).

La Figure 3.9 illustre le mécanisme de création de connexion par SCTP avec création de routes AODV.

- Un INIT de SCTP est envoyé ;
- AODV mémorise le paquet SCTP qu'il ne peut envoyer faute de route et fait une découvert de route. Il construit les routes par l'emploi d'un cycle de requêtes "route request/route reply". Lorsqu'un noeud source désire établir une route vers une destination pour laquelle il ne possède pas encore de route, il diffuse un paquet route request (RREQ) à travers le réseau. Les noeuds recevant le paquet mettent à jour leur information relative à la source et établissent des pointeurs de retour vers la source dans les tables de routage. Un noeud recevant un RREQ émettra un paquet route reply (RREP) s'il est la destination ;
- Quand AODV peut utiliser la route vers le noeud destination, SCTP poursuit l'établissement d'association en 4 étapes. INIT est transmis par le niveau routage, le noeud destination répond, il envoie un INIT-ACK. La source reçoit en réponse un COOKIE-ECHO, lequel reçoit enfin un COOKIE-ACK ;
- L'émission des données peut commencer.

3.2.3.1 Architecture d'interactions vers une entité intermédiaire

Le modèle d'architecture de communications que nous utilisons est un modèle de type interactions vers une entité intermédiaire (cf chapitre 2).

Cette architecture effectue le partage de données par un service de stockage/récupération /positionnement d'informations entre deux couches (couches transport et routage). Elle est illustrée sur la Figure 3.10. Les informations échangées entre AODV et SCTP sont gérées par le bloc Source Cross Layer Mechanism (SCLM).

Le bloc SCLM fonctionne de la façon suivante :

- Si le noeud source reçoit un message RERR sur le niveau routage => SCTP fait une pause des temps de timeouts. AODV fait une découverte de route et garde en mémoire les paquets SCTP pas encore transmis.

- Si le noeud source reçoit un message RREP sur le niveaux routage => SCTP arrête la pause. AODV envoie les paquets qui étaient dans la file d'attente.

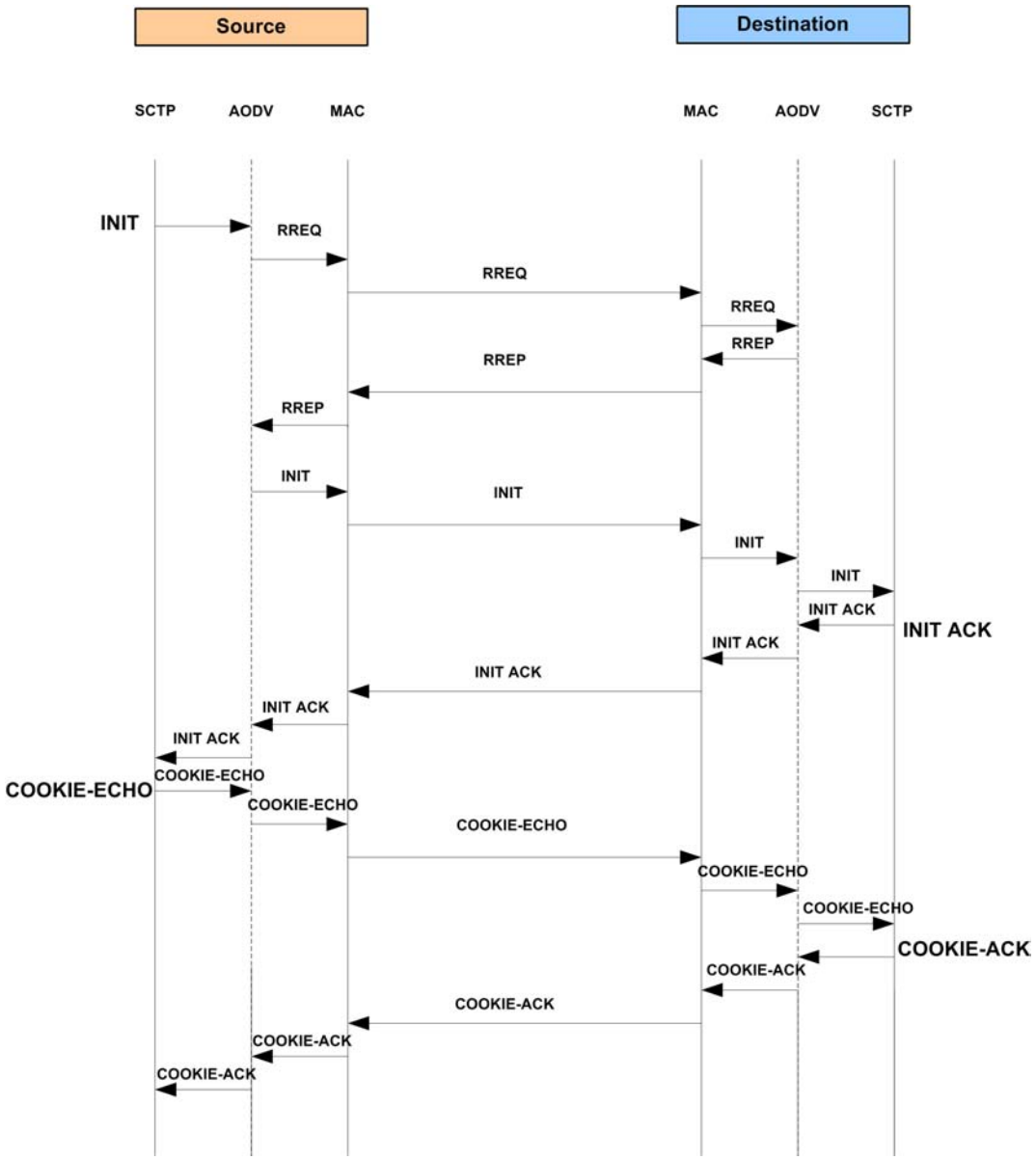


FIG. 3.9 – SCTP PAUSE dans le protocole AODV

La Figure 3.11 illustre le fonctionnement de SCTP PAUSE. Lorsqu'il se produit un échec de route détecté par les noeuds intermédiaires, la source reçoit le message RERR par le niveau routage AODV et utilise l'algorithme SCLM.

3.2.3.2 Conditions d'activation de SCTP PAUSE

SCTP PAUSE repose sur la réception des paquets d'erreurs de route générés par la procédure de maintenance de AODV. Précisons les conditions dans les quelles sont émises ces informations.

En cas de rupture de lien, le noeud qui détecte la panne :

1. effectue une réparation locale s'il a mémorisé dans sa table un autre voisin pour

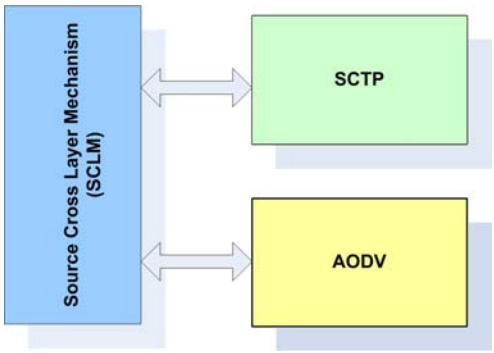


FIG. 3.10 – Mécanisme cross layer source

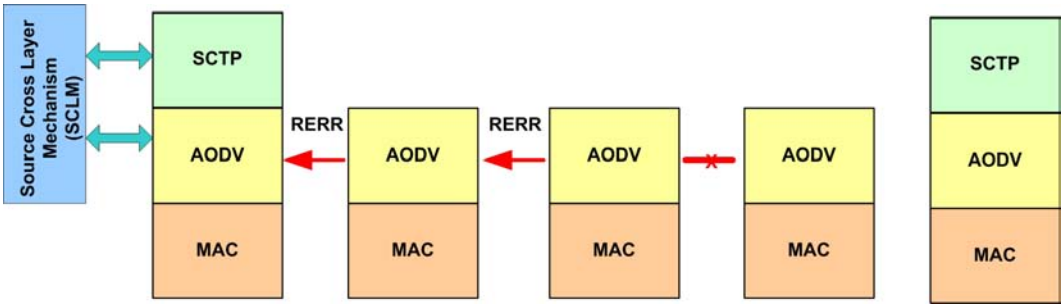


FIG. 3.11 – Le processus de cross layer par la source

arriver à destination.

2. effectue selon les conditions, une découverte de route sans transmettre l'erreur à la source.

3. si la découverte de route ne peut être exécutée ou bien que celle-ci est infructueuse, la notification d'erreur est envoyée à la source.

AODV définit deux fonctionnements : un processus de découverte locale (Local Repair) et un processus de découverte par la source sans condition. Lorsque la rupture de lien est détectée par un noeud qui est plus près de la source que de la destination, celui-ci utilise la valeur de saut (hop) contenue dans la table de routage :

- Si le nombre de sauts à destination < le nombre de sauts vers la source ; alors lancer une réparation locale avec découverte de route.
- Sinon transmettre l'erreur de route à la source.

Les modes de maintenance sont illustrés sur la Figure 3.12 et 3.13.

Selon les configurations, l'option SCTP PAUSE sera ou pas activée à la source. Ainsi sur la Figure 3.13 :

- Cas 1 : le noeud intermédiaire peut faire une découverte de route. Il n'émet pas d'erreur de routes, le SCTP PAUSE ne sera pas utilisé.
- Cas 2 : le noeud intermédiaire transmet le message RERR vers la source, il participera à la découverte de route initiée par la source. SCTP PAUSE sera utilisé.

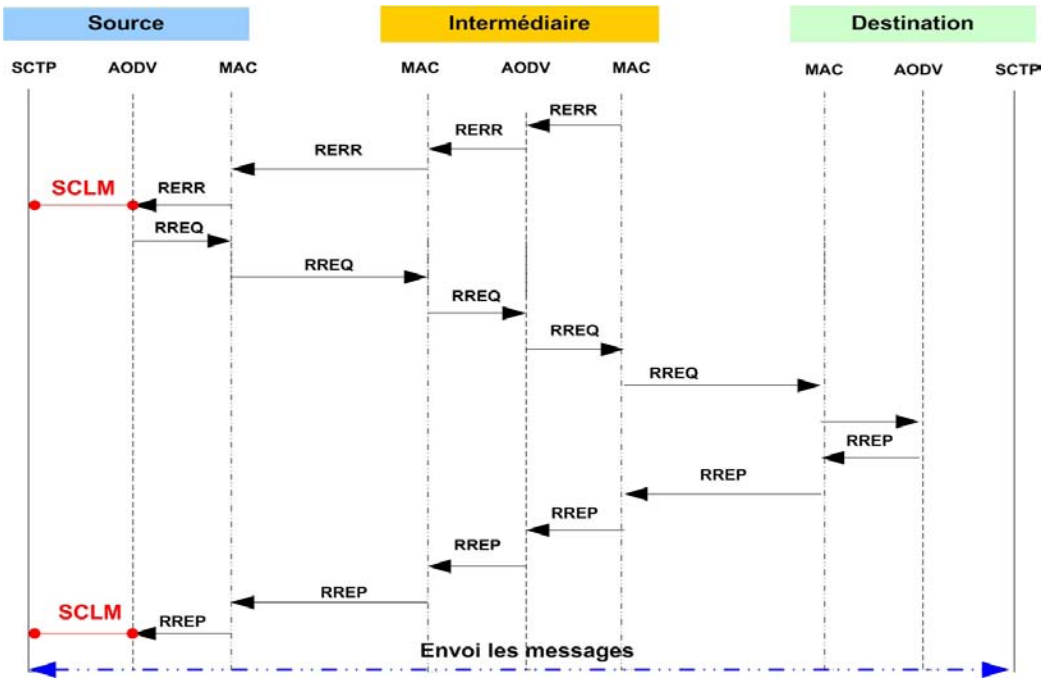


FIG. 3.12 – Nombre de hop à destination >= Nombre de hop de source

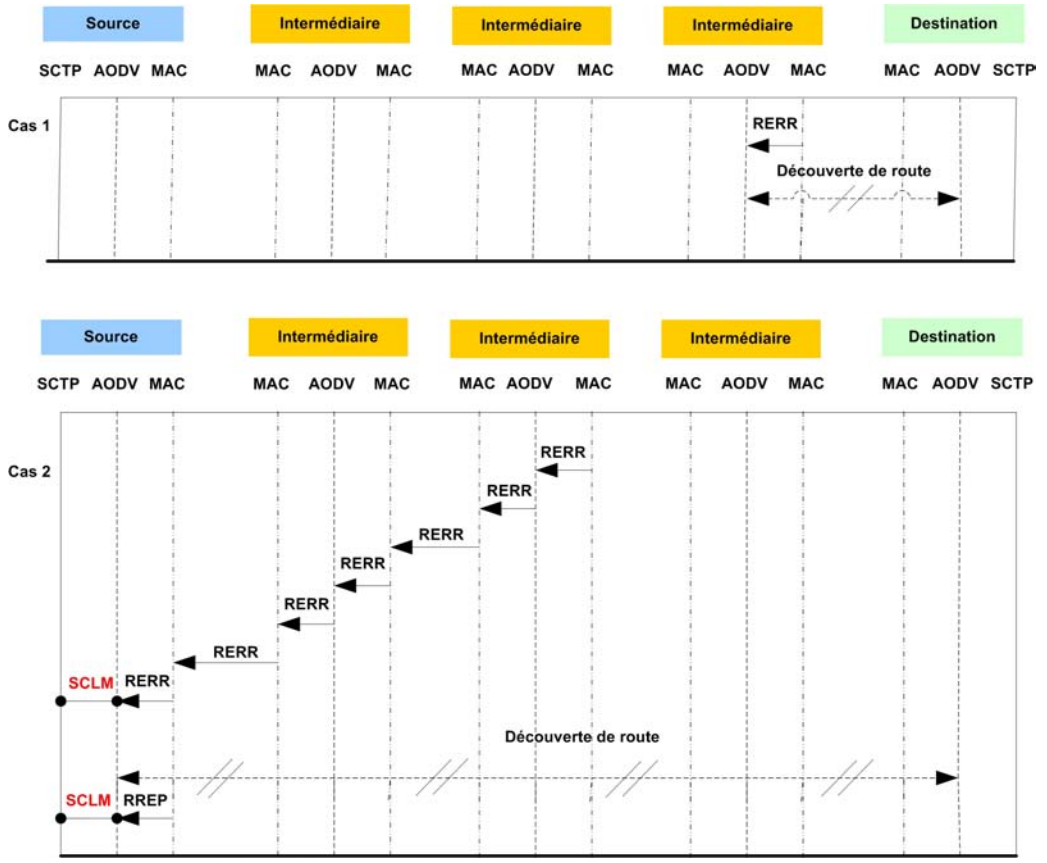


FIG. 3.13 – Modes de maintenance de AODV

3.2.4 Résultats et interprétations de l'évaluation de performances

Tous les résultats ont été obtenus avec les conditions ci-après :
- utilisation d'extension SCTP normalisée IETF et paramétrage approprié d'un SCTP conforme aux recommandations du RFC 4460.
- utilisation de l'option Newreno et d'une fenêtre initiale à Min (4*MTU, max (2*MTU, 4380 bytes)).
- lors de la détection des pertes de paquet ssthresh := max (cwnd/2, 4*MTU) et cwnd = ssthresh.
- SCTP applique l'algorithme de la retansmission rapide sur 3 SACKs.

Pour plus de précisions sur ces paramètres, se reporter au chapitre 2.

Concernant les paramètres d'environnement, nous utilisons les implantations des protocoles intégrées à NS 2 (version 2.29). Au niveau MAC, nous utilisons le modèle 802.11 implanté par le groupe Monarch. Nous gardons les valeurs par défaut des paramètres de ce modèle.
- Nous avons effectué des simulations pour 50 noeuds.
- L'aire de simulation est de 1,000m*1,000m.
- Nous fixons la durée de simulation à 1,000s.
- La portée de transmission de chaque noeud est fixée à 250 mètres.
- La taille de paquet SCTP est 1,460 octets.
- Le modèle de mobilité est Random Waypoint. La vitesse de déplacement des noeuds varie de 1 m/s à 20 m/s et nous avons utilisé des temps de pause de 30 s.
- Le trafic de données est de type CBR (Constant Bit Rate) et le débit d'émission est de 447 Kbps ou 266 paquets/s.

Pour comparer les performances relatives des protocoles de transport et routage, nous calculons la métrique suivante :
- **Goodput (unité : Kbps)** : le nombre de bits de données reçus, à l'exclusion du nombre de bits de données retransmis, divisé le temps de simulation.
- **Le gain de goodput (unité : Kbps)** : le goodput de SCTP PAUSE - le goodput de SCTP.
- **Le gain de goodput moyen (unité : Kbps)** : gain de goodput moyenné sur la vitesse.

Nous interprétons la performance de l'optimisation SCTP PAUSE en regard :
- du nombre de connexions,
- de la longueur du chemin,
- du fonctionnement du protocole de routage en cas de rupture de chemin.

3.2.4.1 Influence du nombre de connexions

Dans la mesure où l'optimisation agit sur le processus de gestion de congestion nous étudions l'impact de l'engorgement du réseau au travers du nombre de connexions présent dans le réseau.

Le nombre de connexions varie : 1, 2, 5 et 20 connexions. Les **résultats** que nous avons obtenus sont présentés sur la Figure 3.14 qui indique le gain de goodput entre SCTP PAUSE et SCTP. Il est calculé sur 50 scénarios de mobilité différents. Les résultats sont fonction du nombre de connexions (variant de 1 à 20).

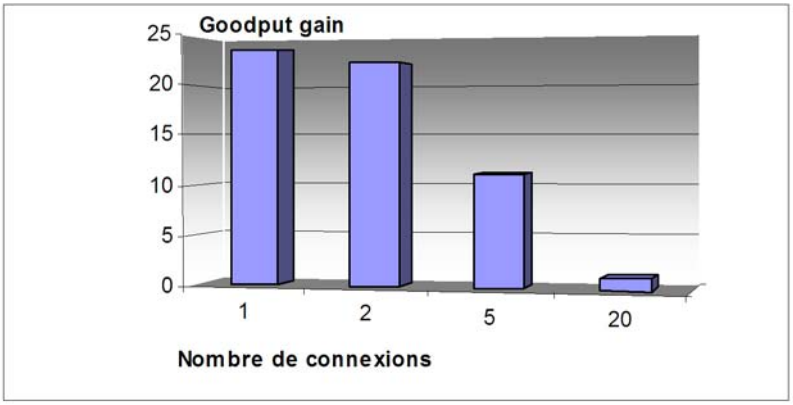


FIG. 3.14 – Gain de goodput SCTP PAUSE / SCTP

Lorsque le nombre de connexions augmente, le gain de performance diminue.

Plus précisément :

Pour 1 connexion (Figure 3.15) le gain de performance est de 23, 59 et 52 Kbps à la vitesse 10, 15 et 20 m/s. Le gain est négatif de 4-11 Kbps pour les vitesses de 1-5 m/s.
Pour 2 connexions (Figure 3.16) le gain est de 127 Kbps à la vitesse de 20 m/s alors que c'est une perte de 18-48 Kbps pour des vitesses de 10-15 m/s.

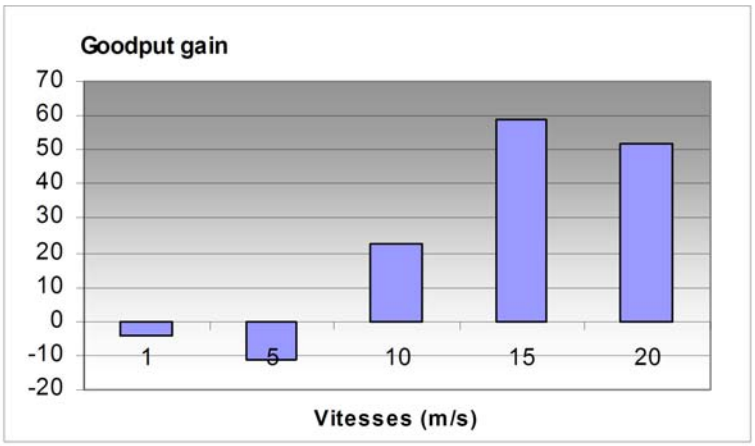


FIG. 3.15 – Gain de goodput SCTP PAUSE / SCTP : 1 connexion

Pour 5 connexions (Figure 3.17) le gain est positif, excepté le point de 10 m/s à moins de 30 Kbps.
Pour 20 connexions (Figure 3.18) nous observons un comportement du même type, un gain ou une perte selon la vitesse.

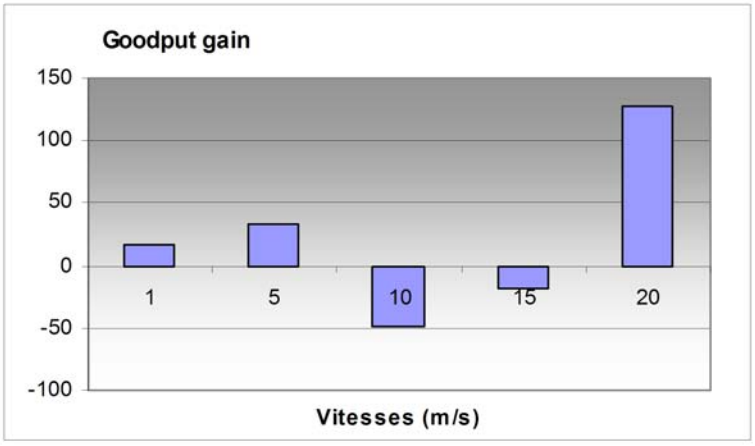


FIG. 3.16 – Gain de goodput SCTP PAUSE / SCTP : 2 connexions

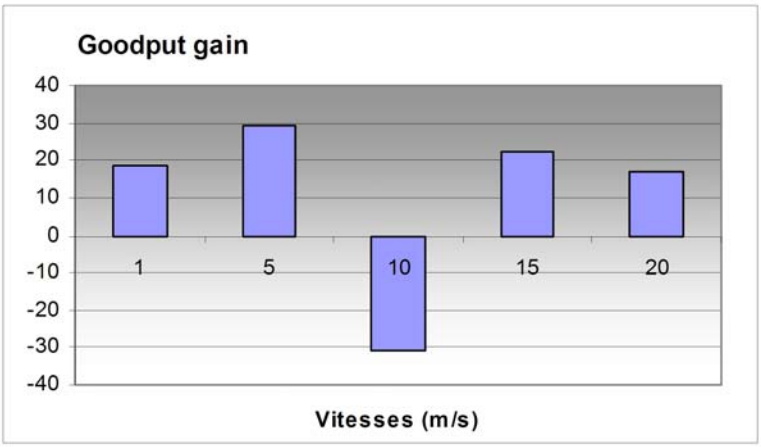


FIG. 3.17 – Gain de goodput SCTP PAUSE / SCTP : 5 connexions

En conclusion, si globalement le gain de performance est avéré, dans de nombreux le cas particuliers, l'optimisation peut s'avérer inefficace. En effet, lorsque le nombre de connexions et donc la congestion augmente, l'intérêt de l'optimisation décroît. Dans les faits, l'optimisation réalisée sur les erreurs de mobilité ne justifie pas la perte de temps occasionnée par l'optimisation qui bloque le timer de retransmission. Notons que nous avons obtenu des résultats similaires avec le protocole de routage DSR [ChPJ07].

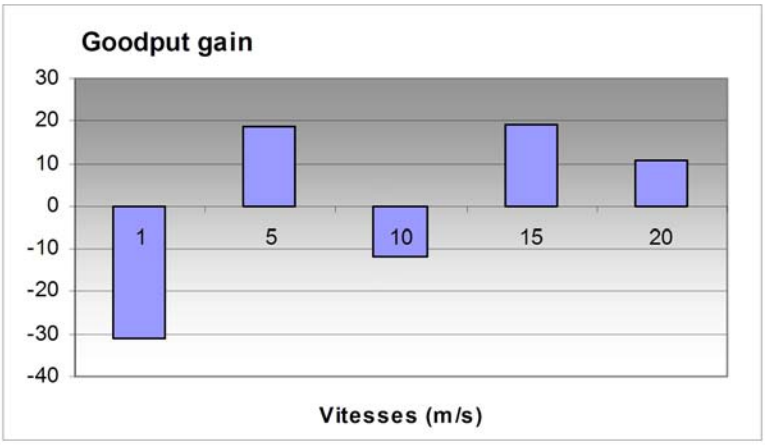


FIG. 3.18 – Gain de goodput SCTP PAUSE / SCTP : 20 connexions

3.2.4.2 Influence de la longueur de chemin

L'incidence de la topologie du réseau sur les performances est prise en compte en considérant la longeur des chemins. Nous avons effectué des simulations pour 50 noeuds. De façon à faire croître la longueur des chemins nous varions l'aire de simulation : 1,000m*1,000m (4 sauts), 2,000m*2,000m (8 sauts) et 2,500m*2,500m (10 sauts). Chaque point calculé représente une moyenne sur 10 scénarios de mobilité. Nous présentons également les performances du réseau sans mobilité.

En plus du goodput, nous calculons le pourcentage d'utilisation de SCTP PAUSE :

– **Pourcentage d'utilisation de SCTP PAUSE (%)** : le nombre de paquets d'erreurs de route aux noeuds sources / (le nombre de paquets d'erreurs de route aux noeuds sources + le nombre de paquets de découverte de route aux noeuds intermédiaires).

Nous observons tout d'abord sur la Figure 3.19 que **SCTP PAUSE obtient de meilleures performances et ce quelle que soit la longueur de chemin.** Plus la longueur du chemin est importante plus le risque de perte de données est important et donc un algorithme qui permet de transmettre plus vite une fois le chemin rétabli par le protocole de routage sera encore plus performant.

Détaillons ce résultat en s'intéressant aux conditions d'activation de SCTP PAUSE. Ayant relevé l'impact de la longueur de chemin sur la rupture de route (cf chapitre 1.3.1) et donc le nombre d'erreurs de routes générées, nous observons le pourcentage d'utilisation de SCTP PAUSE sur la Figure 3.20. **Nous constatons alors un résultat surprenant qui est que le pourcentage d'utilisation diminue avec la longeur du chemin**. La procédure SCTP PAUSE est davantage utilisée pour des chemins de 4 sauts (72 %) que pour des chemins de 10 sauts, (63 %).

Le pourcentage d'utilisation diminue, la différence de gain entre les deux versions diminue (10 Kbps pour 4 sauts et 5 Kbps pour 10 sauts). Posons nous alors la question : **pourquoi l'option SCTP est-elle moins activée lorsque le nombre de relais augmente ?**

Pour répondre à cette question intéressons-nous alors à la procédure de maintenance

de AODV.

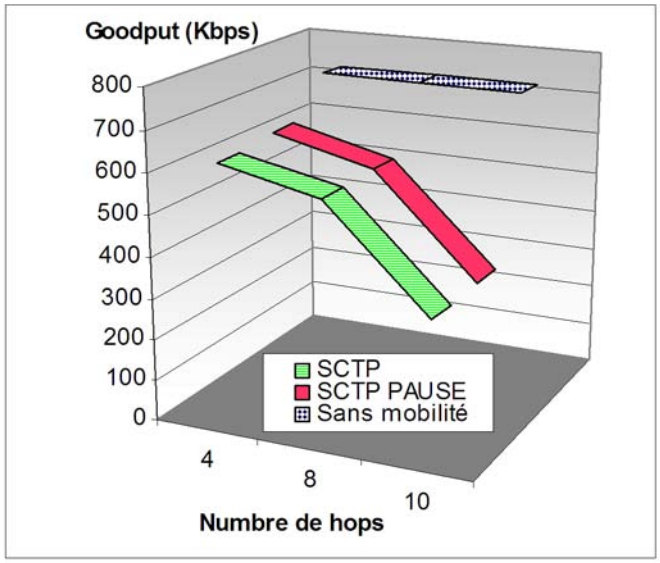


FIG. 3.19 – Comparaison goodput entre SCTP PAUSE et SCTP

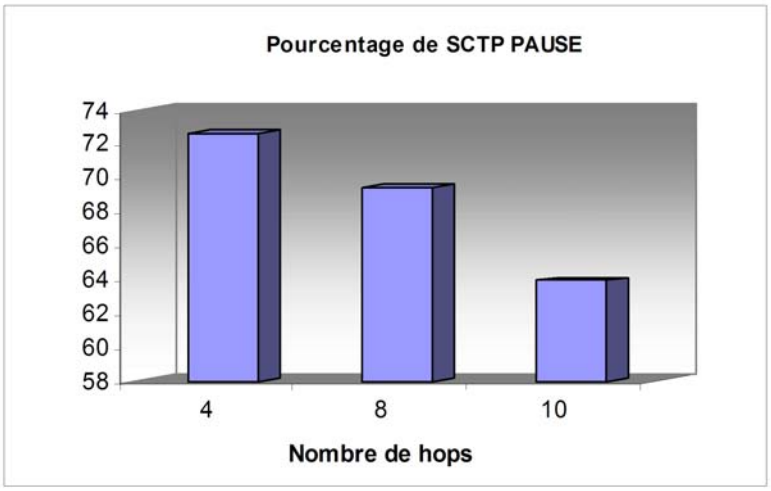


FIG. 3.20 – Pourcentage d'utilisation de SCTP PAUSE

3.2.4.3 Influence des modes de maintenance d'AODV

Nous détaillons le nombre d'erreurs de routes reçues par les sources selon le mode de maintenance : réparation avec découverte locale selon les conditions ou réparation par la source sans conditions.

La Figure 3.21 présente le goodput pour les deux modes de maintenance en fonction de la longueur du chemin.

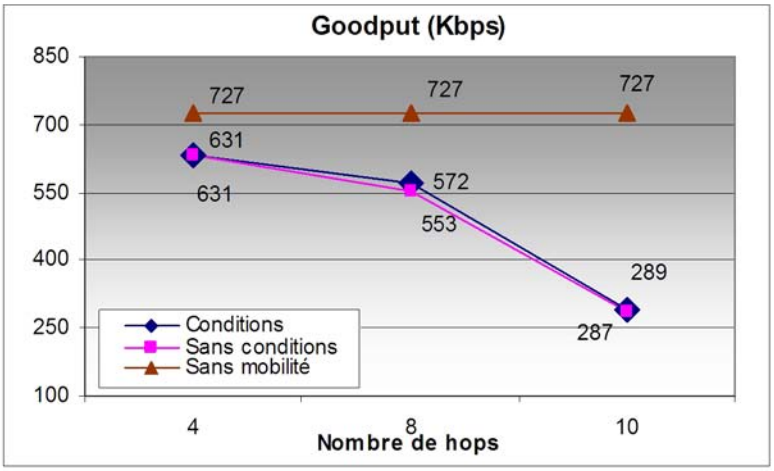


FIG. 3.21 – Influence des modes de maintenance d'AODV

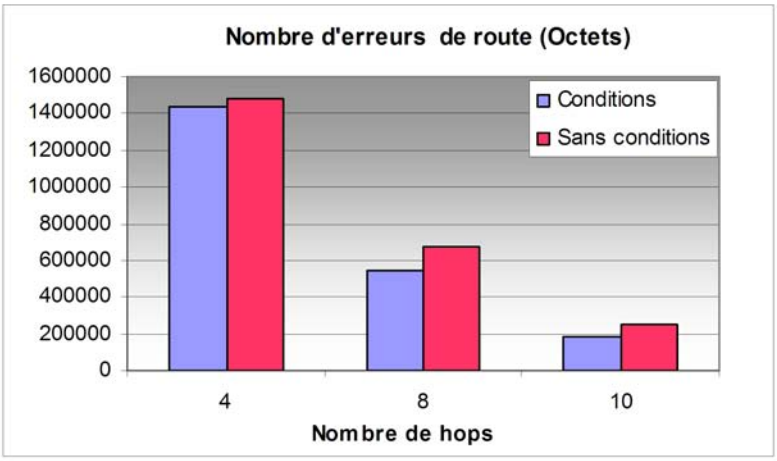


FIG. 3.22 – Nombre d'erreurs de route

Nous constatons une légère différence entre les deux modes : le nombre d'erreurs de routes est logiquement moins important dans le mode de réparation locale (voir la Figure 3.22), et le goodput est alors légèrement moins important. SCTP PAUSE est activé moins souvent.

Cependant, le nombre d'erreurs dans les deux modes décroît avec le nombre de noeuds. Dans les faits, plus le nombre de noeuds est important, plus la possibilité d'utiliser un chemin de remplacement déjà mémorisé dans la table est importante. Les noeuds utilisent une optimisation locale pour trouver un chemin sans avoir besoin de générer des erreurs de route.

En conclusion, le protocole de routage met en oeuvre une optimisation de niveau 3 par auto-adaptation, qui empêche d'utiliser l'optimisation de niveau 4 par communication inter-couches.

3.3 Equité dans les protocoles SCTP PAUSE et SCTP

Nous avons dans la section précédente étudié le gain de performance amené par l'optimisation de SCTP ; considérons à présent son comportement en terme d'équité. Cette optimisation ne doit pas pénaliser les flux émis par un protocole SCTP standard.

Pour cette étude nous nous appuyons sur une topologie de réseau et un modèle de déplacement.

3.3.1 Modèle d'étude de l'équité

La Figure 3.23 montre une topologie à 2 sauts et 2 chemins (un premier chemin (P1) et un deuxième chemin (P2)), pour une même destination.

Le modèle de déplacement est le suivant

- T1 : L'émetteur envoie ses paquets au récepteur, en utilisant un élément relais. L'emetteur transmet deux types de flux : certains passent par le chemin P1 (A-B-D), d'autres passent par le chemin P2 (A-C-D).
- T2 : Le noeud intermédiaire, B, se déplace, provoquant une rupture de route. Ainsi au niveau réseau, l'émission des flux passant par P1 est stoppée, il y a une découverte de route pour chercher un nouveau chemin. Les flux passant par P2 continuent à émettre leurs données.
- T3 : Le chemin est rétabli, la source émet tous les flux sur le chemin A-C-D.

Indice d'équité

Plusieurs métriques d'équité ont été proposées dans la littérature [PaHN04] [Bris06], pour déterminer si les utilisateurs ou les applications reçoivent une partie équitable des ressources. Nous utilisons la métrique présentée dans [JaCH84] l'Indice de Jain : cet indice est défini comme suit :

$$f(x_1, x_2, x_3, ..., x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2}$$

- x_i est le débit obtenu pour la connexion i
- n est le nombre de connexions ou le nombre de flux.

Dans le cas où tous les x_i sont égaux, l'indice d'équité est égal à 1 et l'équité de partage des ressources est maximale. Par contre, plus cet indice s'approche de 0 et plus l'algorithme de partage des ressources est inéquitable.

3.3.2 Résultats et interprétations d'équité

Les scénarios

Nous considérons, selon le nombre de flux empruntant les chemins P1 et P2, trois scénarios (voir Figure 3.24).

- Scenario 1 : P1 = 1 flux et P2 = 1 flux ;
- Scenario 2 : P1 = 1 flux et P2 = 2 flux ;
- Scenario 3 : P1 = 1 flux et P2 = 3 flux.

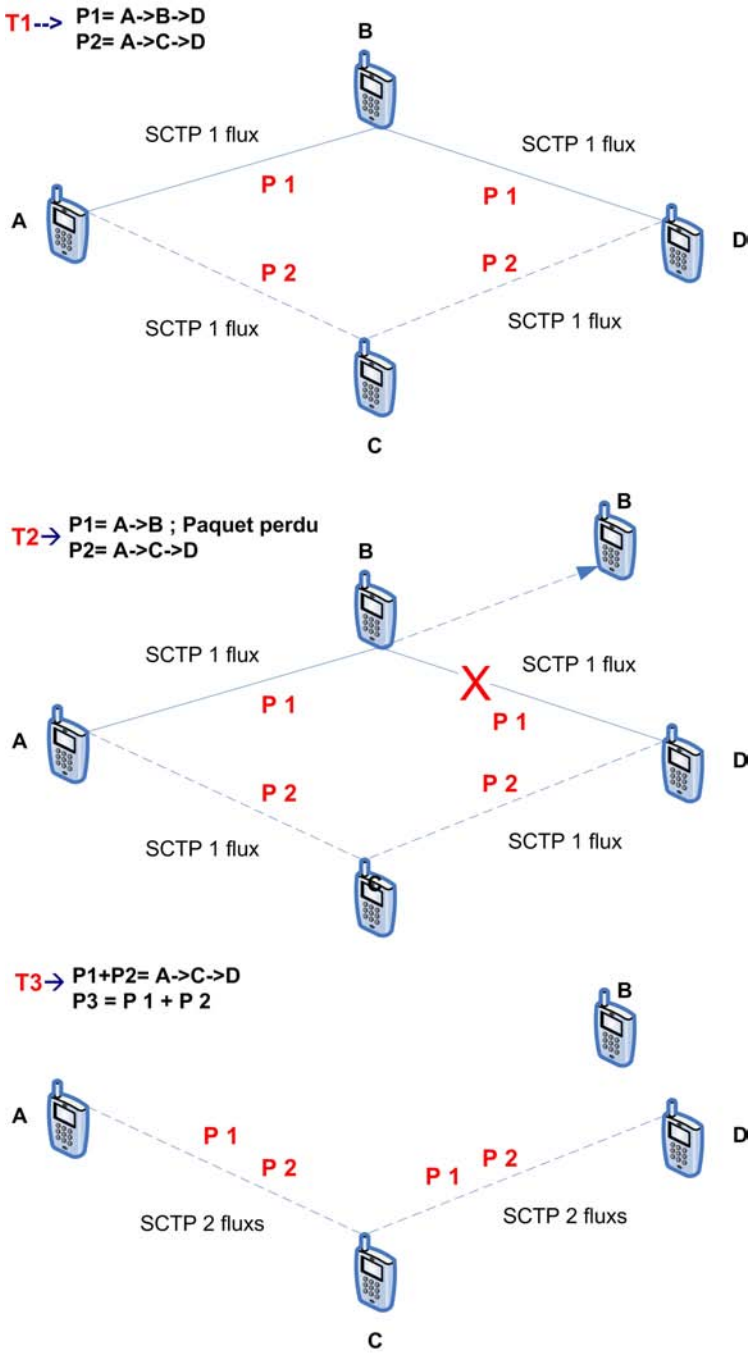


FIG. 3.23 – Modèle d'étude de l'équité

Par exemple, pour le scénario 1 : à T1, P1 a un flux et P2 a un flux. A T2, les paquets sont perdus pour le chemin P1. Enfin à T3, le flux de P1 est distribué sur P2, qui contient à présent deux flux.

Dans un premier temps, nous considérons des flux ayant la même version de protocole, SCTP ou SCTP PAUSE, puis nous mélangeons les types de protocoles qui émettent les flux, selon les chemins.

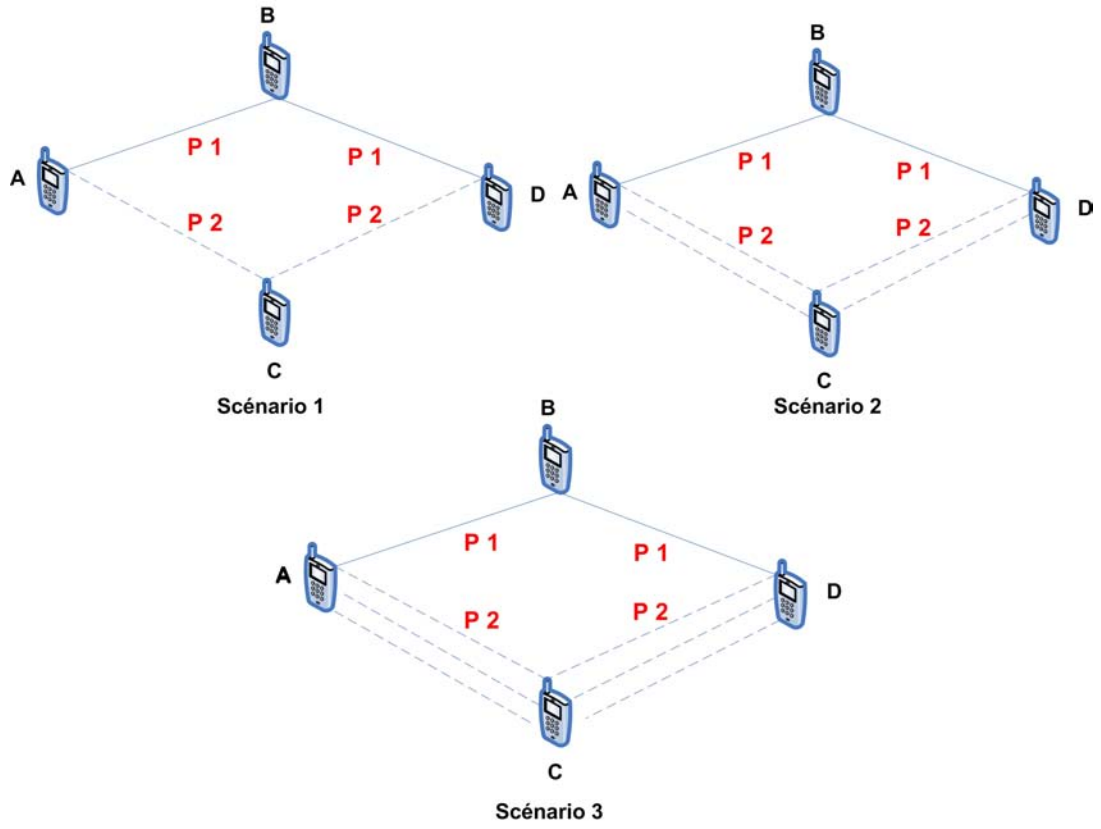


FIG. 3.24 – Scénarios d'études d'équité

3.3.2.1 Chaque flux utilise le même protocole de transport

Nous avons testé l'équité de SCTP et de SCTP PAUSE sur chaque scénario. La durée de simulation a été fixée à 100 secondes et après 5.5 secondes de simulation (T2), le noeud B se déplace.

Les résultats de simulations que nous avons obtenus sont indiqués en Figure 3.25 où sont représentées l'équité, calculée par indice de Jain, de SCTP et SCTP PAUSE.

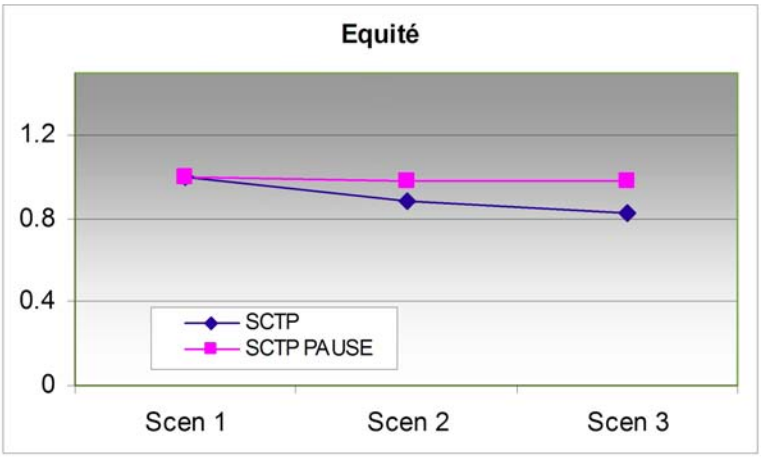


FIG. 3.25 – Etude de l'effet d'équité entre SCTP PAUSE et SCTP dans chaque Scénario

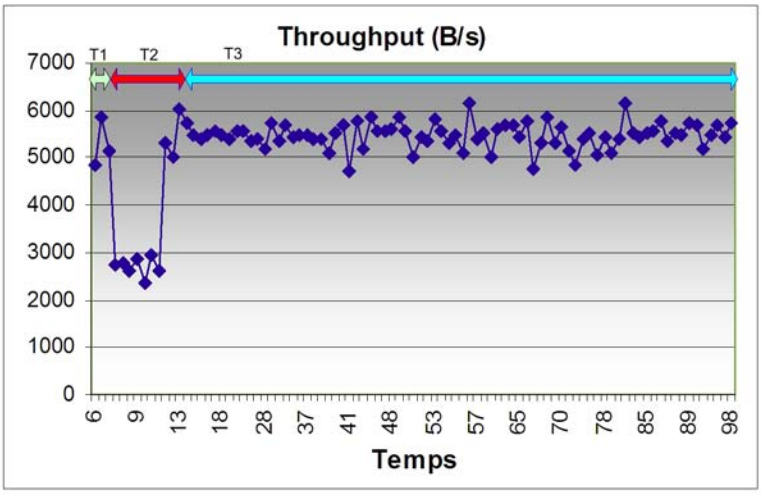


FIG. 3.26 – Throughput global (B/s)

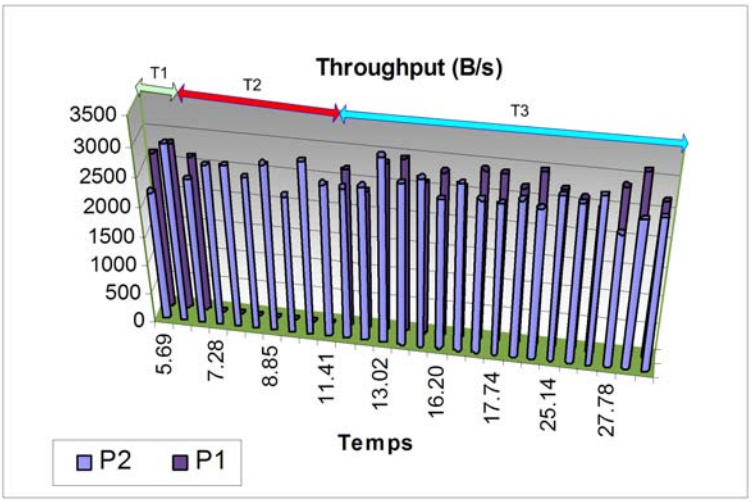


FIG. 3.27 – Etude de l'effet des performances sur chaque chemin (5-30s)

Notons que l'équité des deux protocoles SCTP et SCTP PAUSE, sur les trois scénarios est très proche (l'équité est comprise entre 0.99-0.83). Une explication à la "meilleure" équité présentée par SCTP PAUSE lorsque le nombre de flux augmente est liée au timer RTO, qui dans la version SCTP PAUSE est plus propre à mesurer le temps RTT. La perte de débit due à la mobilité sera alors moins importante avec SCTP PAUSE.

Détaillons le comportement dans le scénario 1. La Figure 3.26 indique le débit de transmission global (throughput relatif aux paquets en émission et en retransmission) de P1 et P2 dans le temps par tranches de 5 secondes. Le débit détaillé par chemin est présenté sur la figure 3.27, l'équilibrage entre les deux flux se trouvant sur le même chemin P2 est illustré sur la Figure 3.28 et 3.29.

Les résultats obtenus pour les scénarios 2 et 3 sont tout à fait similaires. Durant le déplacement du noeud B (phase T2), le débit diminue logiquement (le flux passant par

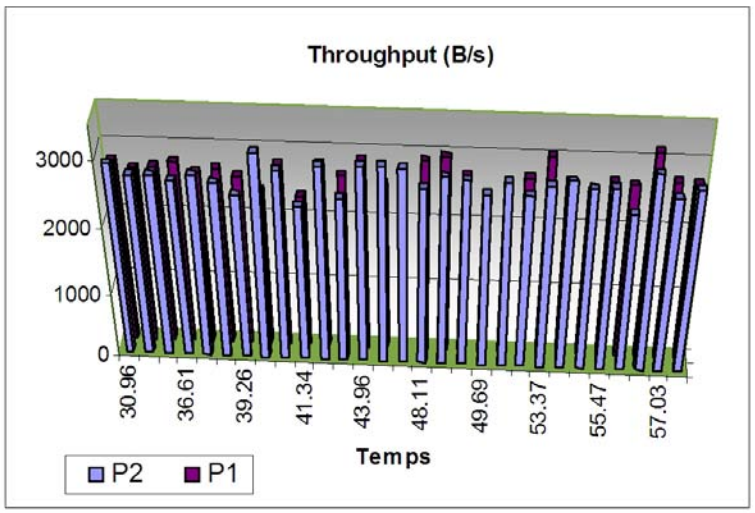


FIG. 3.28 – Etude de l'effet des performances sur chaque chemin (31-60s)

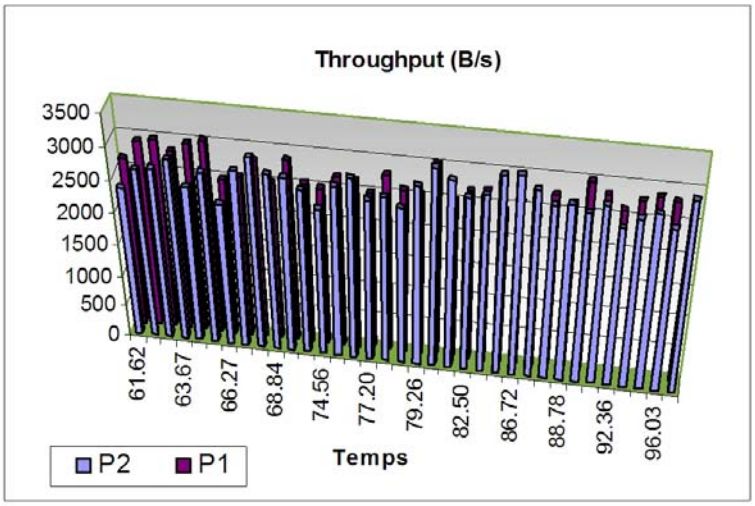


FIG. 3.29 – Etude de l'effet des performances sur chaque chemin (61-100s)

P2 est actif, le flux de P1 est momentanément stoppé) puis, le rétablissement de la route effectué, le débit revient à sa valeur courante avant la rupture.

En conclusion, le protocole SCTP PAUSE est équitable en réseaux ad hoc, non obstant une perte de ressource inévitable due à la mobilité, mais ensuite, une fois la reprise réseau effectuée, un équilibrage de partage de la bande passante s'effectue entre flux de même option.

3.3.2.2 Les flux utilisent des versions différentes de SCTP

Nous positionnons un flux de type SCTP PAUSE et un flux type SCTP sur les chemins P1 et P2 (scénario 1 par la Figure 3.24). Selon le type de flux qui va être affecté par la mobilité, nous définissons deux versions : scénario 1a et scénario 1b (voir Figure 3.30).

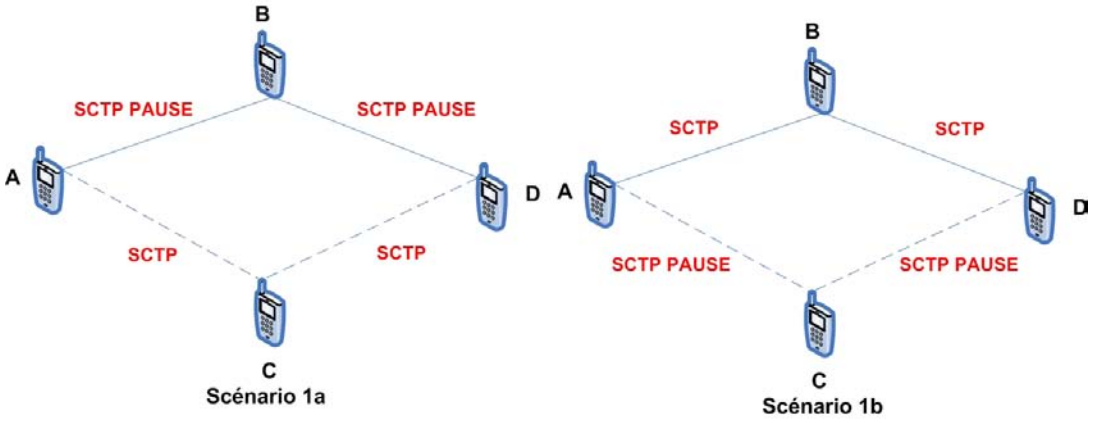


FIG. 3.30 – Scénario d'équité entre flux mixtes

– Scénario 1a : P1 = 1 flux SCTP PAUSE, P2 = 1 flux SCTP.
– Scénario 1b : P1 = 1 flux SCTP, P2 = 1 flux SCTP PAUSE.

Les valeurs d'équité obtenues par simulation sont équivalentes pour les deux scénarios (voir la table 3.3).

Scénarios	Equité	
	P1	P2
1a	0.9900	0.9930
1b	0.9904	0.9918

TAB. 3.3 – Equité de SCTP PAUSE et SCTP dans les scénario 1a et 1b

Ceci s'explique par la durée de simulation prise en compte, qui de par sa grandeur gomme la période d'inactivité due à la mobilité. Détaillons plus avant le comportement des flux au moment de l'interruption de route.

Précisions pour le scénario 1a (SCTP PAUSE sur P1 et SCTP sur P2)

Nous reprenons la même analyse que précédemment en présentant d'une part le débit global des flux puis en détaillant le débit sur chaque chemin.

– La Figure 3.31 indique le throughput global, calculé par tranches de 4 secondes.

– Les Figures 3.32 et 3.33 détaillent le comportement des flux par chemin.

Pour interpréter ces résultats nous devons également disposer des résultats du scénario 1b de façon à voir si SCTP PAUSE et SCTP sont affectés de façon similaire par la rupture de route.

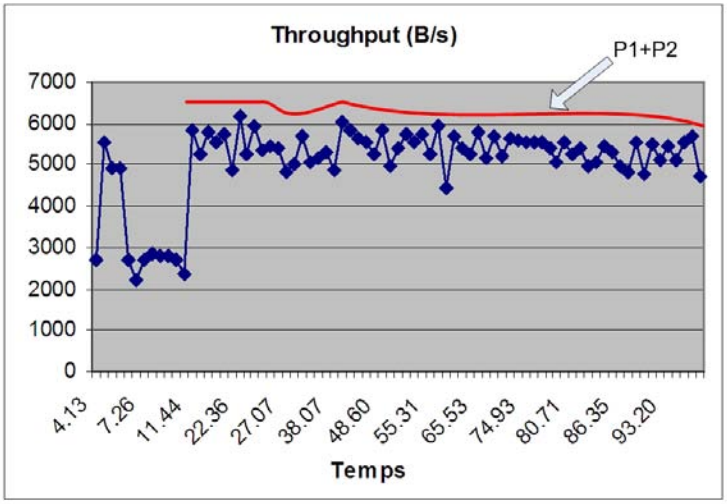


FIG. 3.31 – Throughput global-scénario 1a : SCTP PAUSE sur P1 et SCTP sur P2

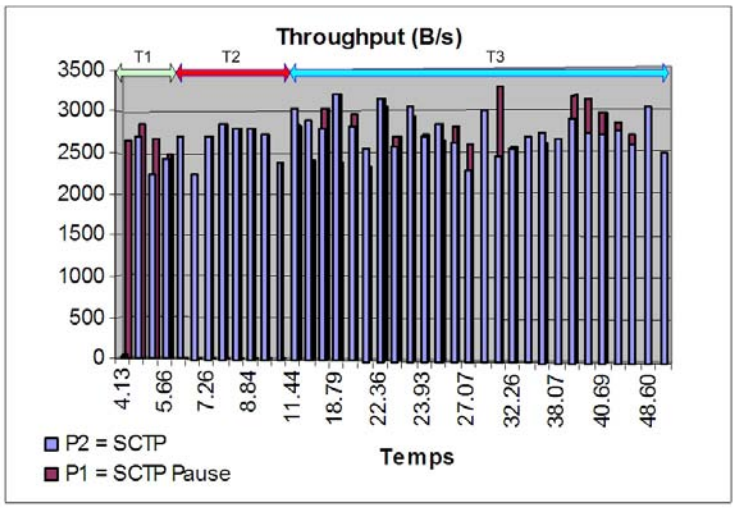


FIG. 3.32 – Détail des throughputs par chemins-scénario 1a de 4 à 50s

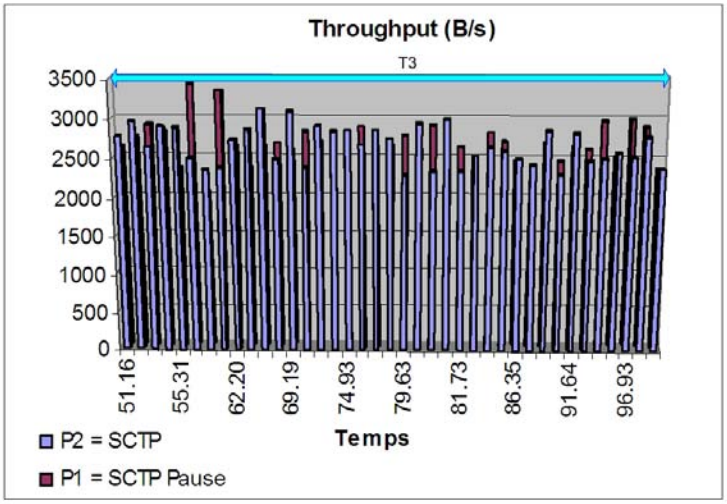


FIG. 3.33 – Détail des throughputs par chemins-scénario 1a de 51 à 100s

Précisions pour le scénario 1b (SCTP sur P1 et SCTP PAUSE sur P2)

– La Figure 3.34 présente les performances globales.
– Les Figures 3.35 et 3.36 détaillent le comportement des flux par chemins.

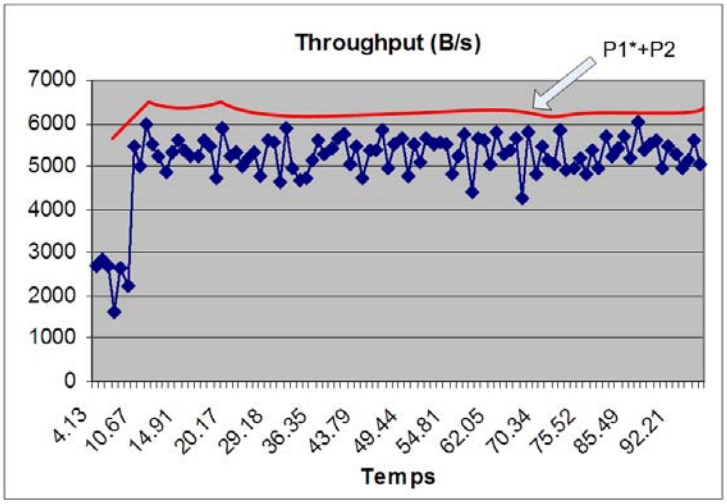


FIG. 3.34 – Throughput global scénario 1b : SCTP sur P1 et SCTP PAUSE sur P2

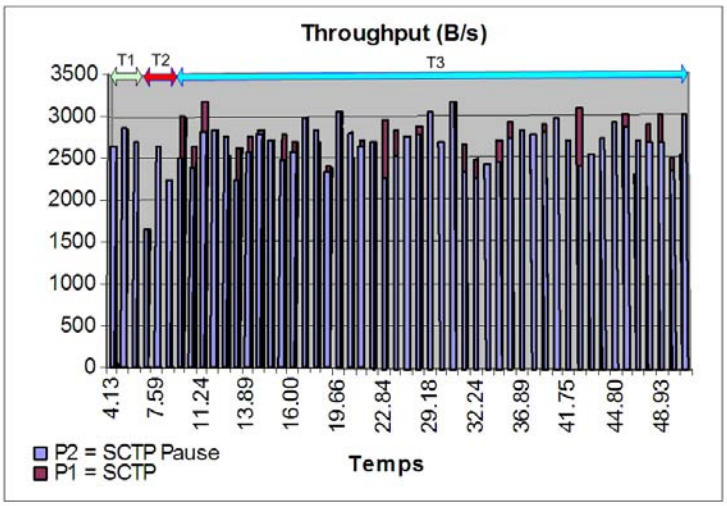


FIG. 3.35 – Détail des throughputs par chemins-scénario 1b de 4 à 50s

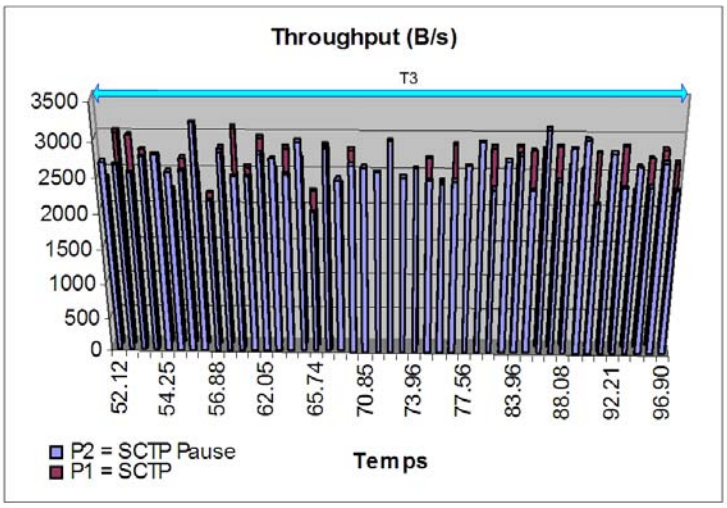


FIG. 3.36 – Détail des throughputs par chemins-scénario 1b de 51 à 100s

Analyse des résultats

Si les deux scénarios sont globalement équitables nous constatons une différence de comportement aux alentours de la phase de mobilité. Dans le scénario “1a” la perte de bande passante de SCTP est plus importante que dans le cas du scénario “1b” avec SCTP PAUSE.

Le mécanisme de partage de la bande passante est synthétisé sur la Figure 3.37.

Pour le scénario 1a, SCTP PAUSE qui n’a pas diminué sa fenêtre se retrouve sur le chemin P2 en concurrence avec le flux SCTP. Il faudra qu’il détecte une congestion avant de diminuer sa fenêtre. Dans le scénario 1b, le flux SCTP a déjà diminué sa fenêtre lorsqu’il se trouve en concurrence avec le flux SCTP PAUSE, il augmente son débit pour arriver à l’équilibre.

Nous retrouvons et illustrons par l’exemple la conclusion précédente : lorsque le réseau est chargé l’intérêt de SCTP PAUSE décroît. Si la reprise de route s’effectue sur une route encombrée, le flux SCTP PAUSE est davantage pénalisé que le flux SCTP.

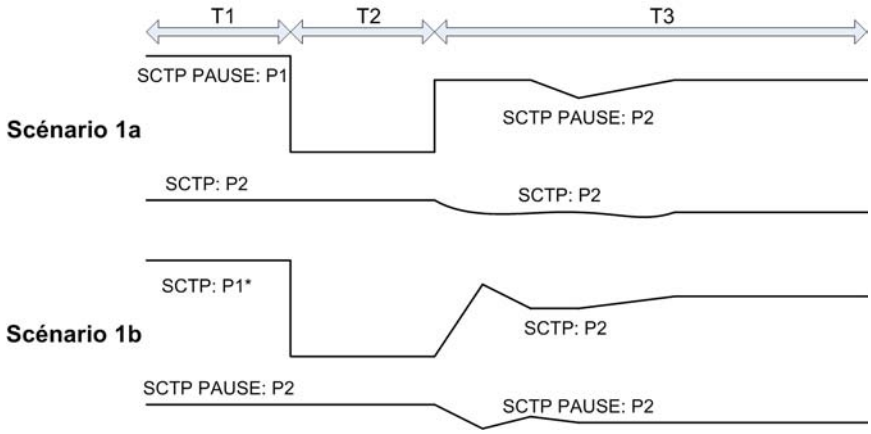


FIG. 3.37 – Augmentation et diminution de débit entre SCTP PAUSE et SCTP

L’optimisation se comporte comme souhaité, après la phase de mobilité les deux flux s’équilibrent. Il y a bien équité : SCTP PAUSE augmente le débit sans pénaliser les autres flux.

3.4 Conclusion

Nous avons évalué l'intérêt d'optimiser le protocole transport fiable par une communication de niveau 3/4, en simulant une optimisation, SCTP PAUSE représentative des optimisations transport avec retour proposées dans la littérature. Nos résultats sont globalement satisfaisants. Cependant, de l'étude approfondie du fonctionnement de l'optimisation face à différents paramètres d'environnement des réseaux ad hoc, nous retiendrons les limitations suivantes.

L'optimisation, dont le but est de détecter les pertes de routage, va s'avérer inefficace et même pénalisante, lorsque la reprise de route provoquée par le protocole de routage conduit à regrouper plusieurs flux sur un même chemin et donc à des phases de congestion qui seront mal traitées par la version dite optimisée.

Un moyen de traiter ce problème est de rajouter à la notion d'adaptation niveau 4 par cross layer niveau 3 que nous avons développée, une adaptation additionnelle à l'encombrement des routes qui pourrait être obtenue via des métriques de qualité de services provenant du routage. Il faut que le transport soit capable d'utiliser ou non l'optimisation. L'adaptation doit être dynamique.

Par ailleurs nous avons mis en avant une limitation importante de l'approche cross layer 3/4 à la source pour optimiser le niveau 4 : l'optimisation interne niveau 3. Un problème important des protocoles de routage est le surcoût de contrôle qu'ils génèrent. L'optimisation de base du routage est alors de minimiser ce surcoût, c'est à dire de limiter la propagation des messages d'erreurs de route, et par là même, la possibilité de mettre en oeuvre l'optimisation de niveau 4, puisque celle-ci est activée sur les reports d'erreurs. La couche 3 s'améliore, et diminue la possibilité d'être améliorée par un mécanisme de niveau supérieur. Toutefois dans la mesure où le gain de performance obtenu en agissant au niveau 3 est équivalent si ce n'est meilleur, il n'y a pas de dyfonctionnement réel si ce n'est un gaspillage inutile de ressources d'implantation.

Une solution à ce problème d'incohérence entre une auto-adaptation et une adaptation inter-couches est de définir un module global d'optimisation capable de négocier les optimisations à installer dans la pile de protocoles.

Un autre moyen de traiter ce problème est d'avoir une vue globale du réseau ad hoc, en utilisant, par exemple, une méthode de communication répartie sur les noeuds du réseau, telle celle présentée dans le chapitre suivant.

Chapitre 4

Optimisation de SCTP par une communication inter-couches répartie

Sommaire

Nous proposons une méthode d'optimisation en réseaux ad hoc pour un protocole de transport fiable qui repose sur une augmentation de la robustesse du réseau face à la mobilité des éléments, grâce à l'établissement de plusieurs routes pour un même flux d'information. Cette méthode, que nous nommons, SCTP ChangePath (SCTP CP), est mise en oeuvre par le protocole SCTP sur un réseau AODV, et une communication inter-couches répartie sur les noeuds du réseau ad hoc. L'intérêt de la méthode est évalué par simulation.

4.1 Multihoming dans SCTP

4.1.1 Caractéristiques et applications

Le multihoming est la possibilité pour un protocole internet d'utiliser plusieurs identifiants pour un même échange. L'identifiant internet, l'adresse IP étant, par définition la localisation d'une machine sur internet, cette situation se produit lorsqu'un noeud dispose de plusieurs interfaces de communications, ou bien parce que le réseau dans lequel il est localisé est raccordé à l'internet par plusieurs routeurs ou encore par un routeur multicartes.

Les principales applications du multihoming sont :

1. L'accélération des changements de réseau par obtention anticipée d'adresse.
2. Le partage de charge par une répartition du flux sur plusieurs chemins.
3. L'augmentation de la robustesse du réseau, par un basculement du flux d'information d'un chemin en panne vers un chemin en fonctionnement.

La première application a été proposée pour les réseaux avec infrastructure (voir par exemple [KoCL04]).Lorsqu'un mobile change de réseau de rattachement il doit obtenir une nouvelle adresse IP. Pour garantir la continuité du transfert d'information mobile IP établit alors une correspondance entre l'identificateur de la machine (son adresse IP dans son réseau de rattachement) et sa localisation (son adresse dans le réseau visité). La machine capable de multihoming accélère la mise en place de cette correspondance, en obtenant sa nouvelle adresse alors que son changement de réseau n'est pas encore effectué, (grâce à un mécanisme de communication inter-couches), tout en continuant à recevoir des informations a son adresse courante.

L'application du partage de charge en réseaux ad hoc a été étudiée par [LiXG03]. Alors que le transfert de flux CBR par le protocole UDP via des chemins disjoints permet d'améliorer le débit du transfert [LeGe01], les auteurs constatent de mauvaises performances pour un transfert TCP. Ceci est dû pour partie à un problème de fiabilité de l'estimation du temps d'aller retour car le temps moyen de RTT sur plusieurs chemins peut être plus court que la valeur maximale (sur le chemin le plus long) du RTT. Dans ce cas de figure l'émetteur TCP peut avoir une expiration de timer prématurée sur le paquet qui emprunte le chemin le plus long. Par ailleurs une autre raison à la dégradation de performances provient du fait que les paquets qui empruntent différents chemins peuvent arriver dans le désordre déclenchant alors l'émission d'acquittements dupliqués qui pousseront l'émetteur à réduire inutilement sa fenêtre.

Dans ce chapitre nous nous intéressons à la troisième application du multihoming : l'amélioration de la robustesse réseau. Dans les réseaux MANETs les pannes de chemins sont gérées par le protocole de routage, nous nous proposons d'améliorer cette gestion, et donc les performances du transport, en introduisant une gestion de panne au niveau transport. L'objectif est qu'en cas de panne sur un chemin, le protocole transport bascule le flux sur un autre chemin le temps que le protocole de routage reétablisse le chemin de routage [ChPa07].

L'utilisation de chemins multiples pour améliorer les performances en réseaux ad hoc a fait l'objet de nombreux travaux au niveau routage. L'utilisation de routes de secours dans AODV (AODV BR : Backup Route) a été proposée par [LeGe00]. Le réseau contient des noeuds primaires et des noeuds de secours. Les noeuds primaires sont ceux qui reçoivent la réponse de route après avoir diffusé la requête. Les noeuds de secours sont les voisins des noeuds primaires qui n'appartiennent pas à la route mais qui entendent la réponse

de route (même si elle ne leur est pas destinée, ils peuvent se mettre en état de tout recevoir). Un noeud secondaire mémorise dans une table dite alternative, le prochain noeud de la route qu'il a entendu dans la réponse de route. Quand un noeud primaire se déplace et provoque ainsi une rupture de route, qu'il détecte par la perte de sa liaison, il diffuse sa donnée pour un relayage par noeud de secours. Le noeud de secours qui entend la donnée,(l'entête indique qu'un routage de secours est demandé) regarde dans sa table alternative, et relaye la donnée. Le problème de cette approche est qu'elle ne considère que des routes de secours à un noeud de distance ce qui augmente les collisions lorsque les paquets circulent à la fois sur des routes primaires et des routes secondaires. Des améliorations pour simplifier la gestion des routes secondaires ont été proposées en [HLCH07] et [LaHL07]. MNH (Multiple Next Hop) [JiJa01] crée plusieurs routes AODV, un noeud mémorise lors de la réception d'une requête de route plusieurs entrées dans sa table, une seule est mémorisée dans AODV, vers lesquelles il émettra la réponse de route, au prix d'un surcoût de routage important. Des travaux ont également été effectués sur le protocole DSR (Multipath DSR) [NaCD01], ou encore sur des protocoles utilisant des mécanismes hybrides, AODV et DSR dans [SaKa04], ou des mécanismes hybrides OLSR-AODV dans [MtKa06].

L'originalité de notre démarche est de gérer l'établissement des chemins de secours au niveau des flux ce qui permet de définir ceux-ci uniquement si l'application le souhaite. Le routage n'a pas à se préoccuper de la gestion des routes secondaires, il est concerné par l'établissement, et s'il contient une fonction pour des routes disjointes, c'est comme nous le verrons par la suite un avantage.

4.1.2 Ajout dynamique d'adresses SCTP ADDIP

SCTP ADDIP est une extension de SCTP définie dans la RFC 5061 [SXTM07] qui permet la mise à jour dynamique de la liste des adresses de l'association (ADDIP). Ainsi un mobile qui est en train de communiquer par une association SCTP et qui se déplacant obtient une nouvelle adresse, est capable d'intéger cette dernière dans l'association et de poursuivre son transfert sur cette nouvelle adresse. Le protocole SCTP enrichi par cette extension est également connu sous le nom mSCTP (Mobile SCTP) [RiTu07].

Identifiant	**Nom de chunk**
0xC1	Address Configuration Change Chunk (ASCONF)
0x80	Address Configuration Acknowledgment (ASCONF-ACK)

TAB. 4.1 – Les nouveaux types de chunks pour SCTP ADDIP

Parameter Type	**Address Configuration Parameters**
0xC001	Add IP Address
0xC002	Delete IP Address
0xC004	Set Primary Address

TAB. 4.2 – Les paramètres de ASCONF

ADDIP ajoute deux nouveaux types de messages de contrôle : Address Configuration Change (ASCONF) et Address Configuration Acknowledgment (ASCONF-ACK) (la table 4.1). Ces messages permettent d'ajouter, d'enlever ou de changer une adresse dans l'ensemble des adresses IP des deux extrémités de l'association (voir les table 4.2 et 4.3).

Ensuite, l'émetteur peut notifier à l'hôte correspondant un changement de l'adresse utilisée dans le chemin primaire, pour utiliser une des adresses rajoutées par ADDIP.

Les autres chunks, tels le chunk INIT et le chunk INIT-ACK, contiennent de nouveaux paramètres comme indiqué par la table 4.4.

Parameter Type	Address Configuration Parameters
0xC003	Error Cause Indication
0xC005	Success Indication

TAB. 4.3 – Les paramètre de ASCONF-ACK

Parameter Type	Address Configuration Parameters
0xC004	Set Primary Address
0xC006	Adaptation Layer Indication
0x8008	Supported Extensions

TAB. 4.4 – Les nouveaux types de INIT/INIT-ACK pour SCTP ADDIP

4.1.3 Gestion des retransmissions en multihoming

SCTP définit le multihoming comme étant la capacité d'un point terminal SCTP à contenir des interfaces multiples avec différentes adresses IP. Dans une association mono-mono domiciliée (single-single-homed), un point terminal contient seulement une interface de réseau et une adresse IP. Un seul chemin de communication est établi. Dans le cas de connexions multi-multi domiciliées de nombreux chemins peuvent être établis pour chaque interface source (Figure 4.1).

SCTP choisit une adresse comme adresse "primaire" et, en fonctionnement normal, l'utilise comme destination pour tous ses transferts de données. Toutes les autres adresses sont considérées en tant qu'adresses alternatives. SCTP utilise ces adresses alternatives en cas de panne sur l'adresse primaire, pour retransmettre les données perdues [IyAS04]. Toutes les données seront transmises à l'adresse alternative jusqu'à ce que la communication avec l'adresse primaire soit rétablie.

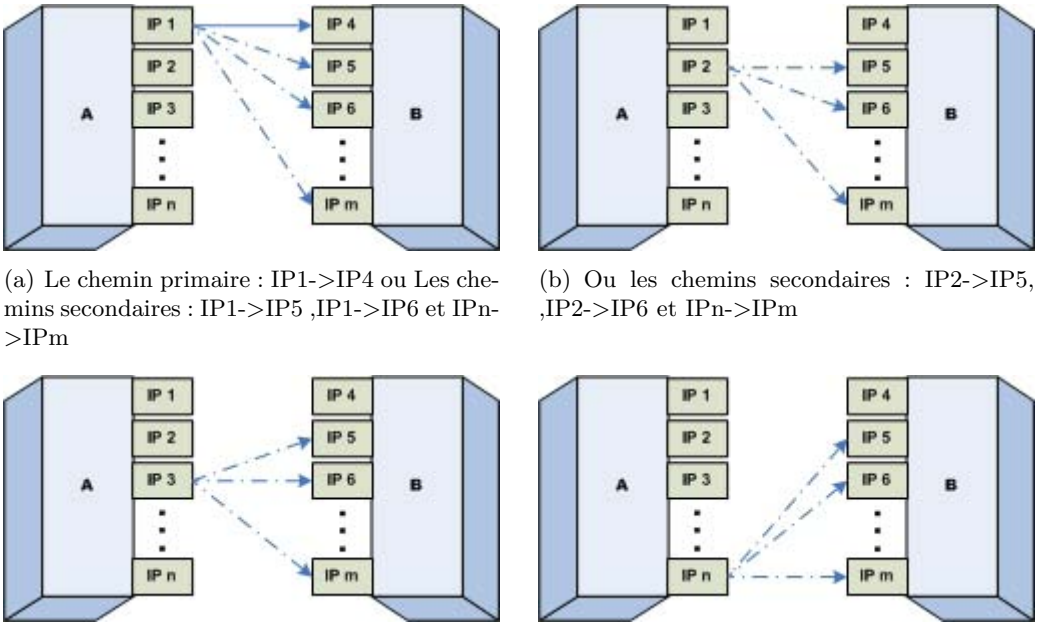
4.1.3.1 Politique de retransmission MFR (Multiple Fast Retransmit)

[CaAS06] montre que l'algorithme MFR améliore les performances de SCTP en réseau filaires [CaAS06] en utilisant plusieurs retransmissions d'un même paquet.

Par exemple, si le paquet x est perdu, un SACK sera renvoyé à la source indiquant le paquet manquant :

- L'émetteur attend trois SACKs de paquet x, il initie le *fast retransmit*, et retransmet le paquet immédiatement.
- Si la retransmission de paquet x est encore perdue, l'émetteur détecte la perte avec trois autres SACKs. Le paquet x est retransmis une autre fois.

Nous avons choisi d'utiliser cette technique pour les réseaux ad hoc, de façon à ne pas inutilement attendre un timer pour retransmettre les données alors que le chemin est déjà rétabli. Comme nous l'avons analysé dans le chapitre précédent, dans les réseaux ad hoc, lorsque le nombre de noeuds est suffisant, le protocole de routage AODV est capable de



(a) Le chemin primaire : IP1->IP4 ou Les chemins secondaires : IP1->IP5 ,IP1->IP6 et IPn->IPm

(b) Ou les chemins secondaires : IP2->IP5, ,IP2->IP6 et IPn->IPm

(c) Ou les chemins secondaires : IP3->IP5, ,IP3->IP6 et IPn->IPm

(d) Ou les chemins secondaires : IPn->IP5, ,IPn->IP6 et IPn->IPm

FIG. 4.1 – Interfaces et chemins multiples dans SCTP multihoming

rétablir rapidement le chemin (par une optimisation locale), il est alors inutile d'attendre un timer avant de tester le fonctionnement de la nouvelle route. Cette technique permet également de réagir plus rapidement à des pics de signalisations de routage.

Un paquet perdu sera retransmis deux fois en MFR sur le chemin primaire, puis après expiration de timer sera retransmis sur le chemin secondaire.

4.1.3.2 Gestion du timer par politique TS (Timestamps)

Le mécanisme TS est une option de SCTP qui a pour but de calculer précisément le timer de retransmission en se basant sur la mesure du temps aller/retour. Elle est utile pour retransmettre le plus rapidement possible, en cas d'échec des retransmissions multiples sur le chemin primaire, sur le chemin secondaire (l'émission sur le chemin secondaire a lieu sur expiration du timer) (Fast retransmit to same and RTO to alternate : FrSameRtoAlt) [CaAS06].

Le mécanisme d'étiquetage temporel (TimeStamps : TS) de SCTP est similaire à celui de TCP. Chaque paquet contient une estampille temporelle, ce qui permet de mieux estimer le temps d'aller retour en levant l'ambiguité de l'algorithme de Karn [StAP99] ; celui ci estime le temps d'aller retour à partir des acquittements reçus sans distinguer ceux correspondants à des paquets originaux de ceux associés à des paquets retransmis et obtient donc une mesure moins précise.

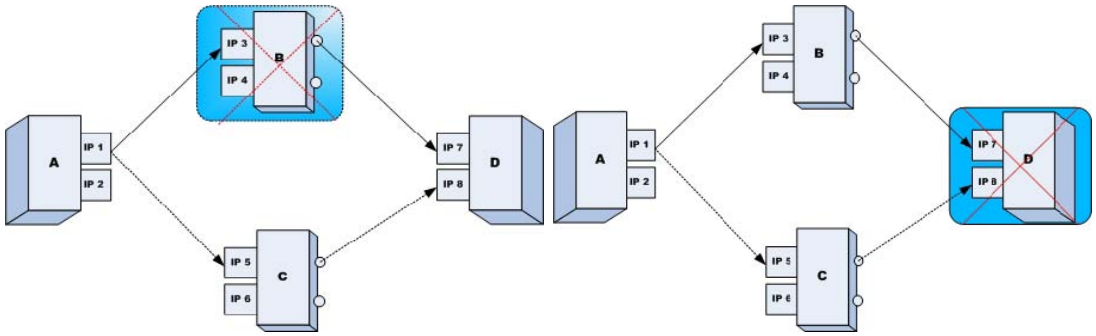
Notons que si le mécanisme d'estampillage permet d'accélérer les performances de la retransmission, dans la mesure où il nécessite un surcoût d'entête pour contenir l'estampille, il augmente néanmoins le coût de transmission. [CaAS06] et [IyAS06], proposent une solution alternative, qui diminue l'en-tête, en utilisant les drapeaux dans les entêtes des paquets de données et de SACK pour signaler s'ils sont relatifs à une transmission

originale ou à une retransmission.

4.1.4 Méthode de détection d'erreurs de SCTP multihoming

SCTP multihoming intégre une méthode de surveillance par émission de chunk HEARTBEAT. Le HEARTBEAT est envoyé cycliquement (30 secondes par défaut) par un des correspondants pour vérifier si une adresse terminale SCTP est active. Il est envoyé aux différentes adresses négociées durant la phase d'établissement de l'association. Les requêtes Heartbeat sont acquittées par des HEARTBEAT-ACK.

Le fonctionnement sur détection de panne dépend du type de panne détectée : soit une rupture de chemin soit une panne sur l'équipement terminal destinataire. Dans le contexte MANET ces pannes sont dues à des déplacements d'un équipement intermédiaire ou d'un équipement terminal (voir figure 4.2).



(a) Intermediaire (b) Terminal

FIG. 4.2 – Reprise sur erreur en cas de mobilité d'équipement

Précisons les algorithmes de Heartbeat ;

- Rupture de chemin : lorsque un HEARTBEAT est envoyé à une adresse inoccupée (idle address) ; il n'y a pas acquittement (HEARTBEAT-ACK), le paramètre "error counter" de cette adresse de destination est incrémenté. Quand la valeur compteur d'erreur dépasse un seuil paramètre ("Path.Max.Retrans"), l'émetteur doit marquer l'adresse de transport de destination comme inactive, et une notification est envoyée à la couche supérieure. Quand le chemin primaire est marqué inactif, l'émetteur transmet automatiquement de nouveaux paquets à une adresse alternative si elle existe et qu'elle est active. Le paramètre "error counter" est remis à jour chaque fois qu'un chunk de données a été acquitté par SACK ou qu'un HEARTBEAT-ACK est reçu.
- Rupture du noeud de destination, si la valeur du paramètre "error counter" est supérieure à celle du paramètre "Association.Max.Retrans", l'émetteur considère que le destinataire n'est pas joignable et arrête de transmettre (il entre en phase de fermeture d'association).

4.2 SCTP et autres protocoles de gestion d'identifiants multiples

Le multihoming SCTP propose une séparation entre les fonctions d'identification et de localisation. L'identification du transfert de l'information s'effectue au niveau de l'as-

sociation alors que la localisation s'effectue au niveau du chemin emprunté. Ceci permet de faire correspondre comme nous l'avons vu précédemment à une association plusieurs chemins.

Plus généralement, cette séparation permet d'avoir un identificateur stable, généralement appelé identifiant, qui facilite la gestion de la mobilité et de la sécurité, et un identificateur, appelé localisateur (locator), qui lié à la localisation de la machine garantit l'efficacité du routage.

La définition de plusieurs identificateurs est actuellement étudiée à l'IETF au travers plusieurs approches [Nika07]. Elles se différencient selon les éléments concernés par le nouvel espace d'adressage qui est introduit (l'espace d'adressage traditionnel est celui des adresses IP). Celui-ci peut être pris en charge par les machines d'extrémité, l'approche est alors dite au dessus de IP, ou bien être transparent aux machines et géré par le réseau, approche dite sous-IP. Dans l'approche sous-IP, la correspondance entre les deux éléments (identifiants/localisateurs) est gérée par le premier routeur d'entrée dans le réseau (voir Figure 4.3).

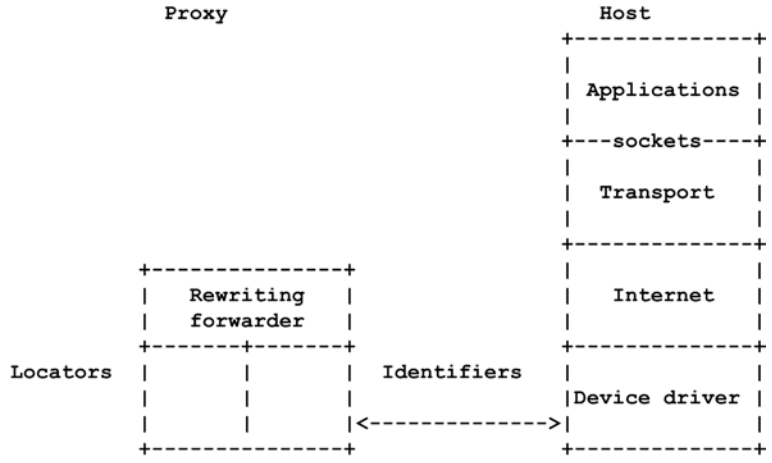
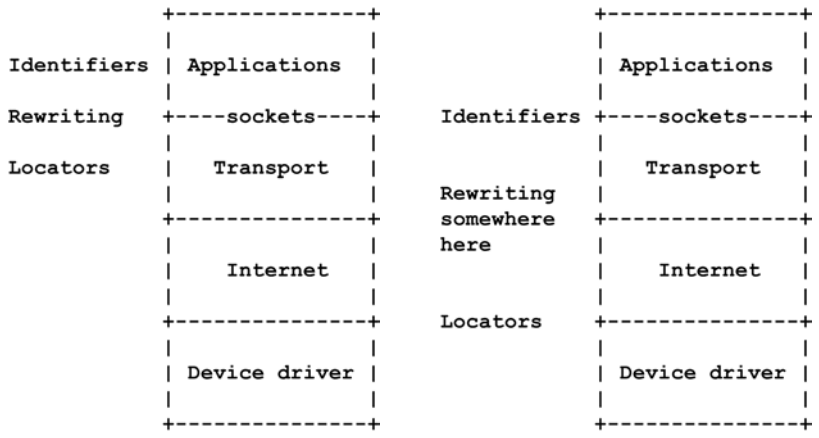


FIG. 4.3 – Gestion de la correspondance identifiant/localisateur dans le réseau

Parmi les approches sur-IP, effectuées au site terminal, nous séparons les approches applicatives des approches TCP/IP (Figure 4.4). La migration de connexion relève de la première approche, elle préconise d'utiliser le localisateur au niveau transport et l'identifiant au niveau application. Son inconvénient majeur est sa complexité en cas de changement de localisation. Il est alors nécessaire de fermer la connexion en cours sur l'ancien localisateur, et de reouvrir une nouvelle connexion associée au localisateur courant tout en assurant la synchronisation des échanges. Un exemple de synchronisation est présenté en [SSIM02]. La deuxième approche utilise l'identifiant au niveau du transport et le localisateur au niveau "TCP/IP". La correspondance localisateur-identifiant est faite dans une sous-couche au niveau transport ou au niveau routage.

Les technologies SHIM6 (Site Multi-homing by IPv6 Intermediation), HIP (Host Identity Protocol), SCTP ADDIP sont des approches terminales de niveau TCP/IP. Elles ont comme inconvénient de modifier la pile de protocoles traditionnelle, car elles nécessitent de reécrire des API adaptées. Cependant, notons que le protocole SCTP qui utilise ses propres sockets, est actuellement parfaitement intégré dans les piles protocolaires, (versions linux, drivers SCTP pour windows XP/vista ; des implantations de SCTP sont recensées sur le site SCTP.org)



(a) Localisation au niveau transport (b) Identification au niveau transport

FIG. 4.4 – Gestion de la correspondance identifiants/localisateurs aux extrémités

Le protocole LISP (Locator/ID Separation Protocol) relève d'une approche sous-IP, gérée au niveau du réseau et non des machines terminales, il utilise une syntaxe compatible avec les adresses traditionnelles (IPv4 ou IPv6) grâce a un mécanisme de tunnel qui a pour inconvénient d'augmenter l'entête de routage.

Dans cette section nous nous intéressons à l'approche, terminale. Nous avons vu que SCTP était une solution pour gérer la séparation identification/localisation au niveau transport nous présentons à présent une solution au niveau sous-transport (HIP niveau 3.5) et une solution au niveau réseau (SHIM6 niveau 3).

4.2.1 SHIM 6 (Site multihoming by IPv6 intermediation)

Le protocole SHIM6 est implanté sur les machines terminales pour supporter le multihoming dans IPv6. Il est actuellement étudié à l'IETF mais n'est pas encore à l'état de RFC (Figure 4.5). Son objectif est de pallier les problèmes de pannes de chemin et de proposer de la répartition de charge [NoBa07]. Dans cette technique l'identifiant du point terminal est une adresse machine utilisée à l'initialisation pour la communication entre deux machines d'extrémités, (ULID - upper layer identifier), géré dans une sous-couche IP. Les autres adresses que peut avoir la machine sont considérées comme des localisateurs et gérés dans la sous-couche routage.

Les paquets sont transmis dans le réseau en utilisant des localisateurs. Les paquets reçus avec un localisateur sont associés à l'identifiant du point terminal et démultiplexés vers le point transport associé à ce dernier. De cette façon, du point de vue du transport et des applications, un paquet possède toujours un identificateur statique (ULID), même si finalement, pour transmettre le paquet, il est traduit en un localisateur.

Le document [ArBe06], indique comment détecter des échecs de communication entre deux hôtes puis changer de localisateurs. De même que pour SCTP le document utilise la terminologie chemin primaire, chemin secondaire, et le protocole utilise deux messages (KEEP ALIVE and PROBE) pour gérer les changements de chemin. La spécification d'une API adaptée à SHIM6 permettant une interaction entre applications et multihoming est étudiée dans [KBSS07].

L'utilisation conjointe de SCTP et de SHIM6 risque de fournir des résultats inattendus dans la mesure où les deux protocoles gèrent le multihoming à des échelles de

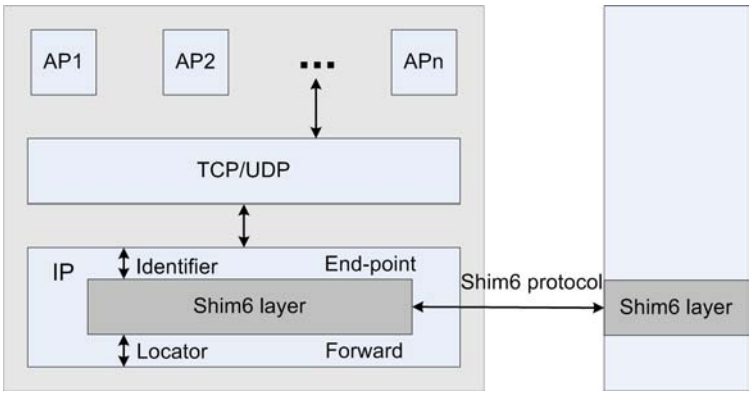


FIG. 4.5 – Le processus SHIM 6

temps différentes (Figure 4.6). Il est recommandé de ne pas utiliser les deux protocoles en même temps [AbBa07]. Pour ce faire soit SCTP est au courant de l'existence de SHIM6 et il désactive alors le multihoming de niveau réseau, soit c'est l'entité SHIM6 qui est au courant de l'existence d'une socket SCTP multidomiciliée (multihomed) et qui alors ne crée pas de contexte SHIM pour celle-ci.

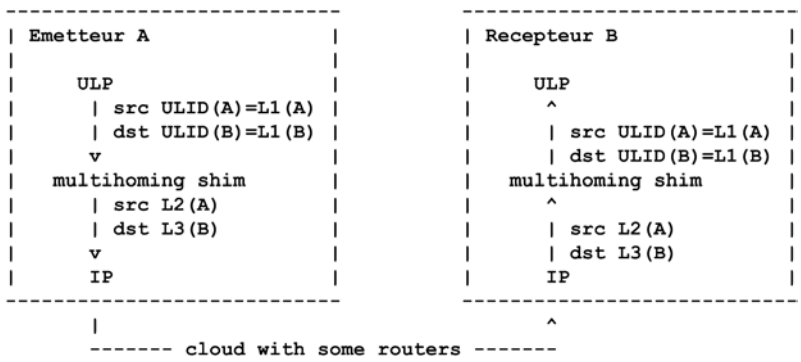


FIG. 4.6 – Plusieurs adresses différentes (locateurs)

4.2.2 HIP (Host Identity Protocol)

SHIM6 et Host Identity Protocol (HIP) proposent tous deux une architecture dont le but est d'utiliser plusieurs localisateurs pour communiquer entre machines terminales tout en conservant un identifiant terminal stable. L'échange de signalisation pour établir le contexte de démultiplexage est très similaire. Le point principal de divergence est la création d'un nouvel espace d'adressage pour HIP qui permet de sécuriser l'association identifiant terminal/localisateur, ainsi que l'échange des données (voir la Figure 4.7). SHIM6 est défini pour répondre à un problème spécifique en minimisant les coûts de déploiement alors que HIP est une solution générale de séparation davantage expérimentale, qui peut être considérée comme une évolution à long terme de SHIM6.

Le protocole HIP est spécifié dans le RFC 4423 [MoNi06], il se situe entre les couches réseau et transport dans le modèle OSI. HIP utilise des paires clés public/privées, au lieu d'adresses IP, pour identifier les machines terminales (Host identifier : HI). Les clés sont allouées par une architecture de sécurité PKI, ou encore crées aléatoirement (comme les

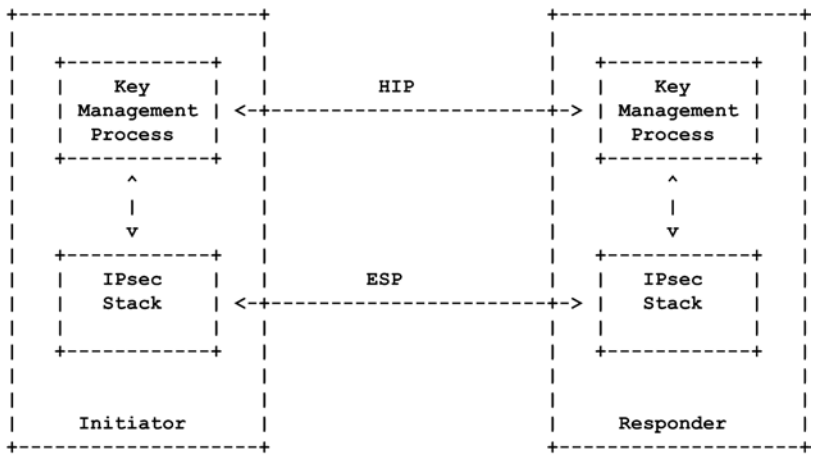


FIG. 4.7 – Le modèle de HIP

clés SSH). A partir de sa clé, une machine crée (par hashage) un nombre de 128 bits, le HIT (Host Identity Tag), qui sera transmis dans les paquets (la clé reste sur la machine terminale). Les HIT devraient être publiés par le DNS, des travaux sont en cours sur des extensions de DNS, [Komu08], voir la Figure 4.8. Un mécanisme de rendez-vous permet à une machine de mettre à jour sa correspondance HIT/localisateur.

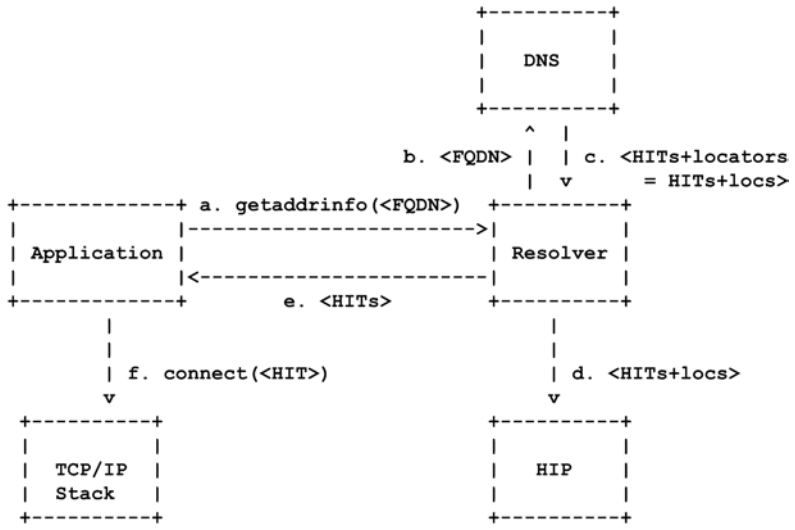


FIG. 4.8 – Translation nom de machine-identifiant par DNS

Le fonctionnement de HIP en concordance avec SCTP ADDIP peut amener à certaines simplifications, ainsi il ne serait plus nécessaire à l'établissement de l'association SCTP d'obtenir la liste des adresses, et augmente la sécurisation du transfert. L'utilisation de TCP avec HIP peut également être envisagée. En l'état actuel, la solution de gestion du multihoming par SCTP nous apparaît comme étant la plus simple, car déjà disponible, à déployer en réseaux ad hoc. Notons que la solution HIP, repose sur un système DNS qui s'avère difficile à mettre en oeuvre pour un réseau sans infrastructure ni administration centralisée. Elle nécessite des travaux additionnels.

4.3 SCTP ChangePath une optimisation par communication inter-couches répartie

Afin d'améliorer les performances de retransmission en cas de rupture de route nous souhaitons pouvoir utiliser un chemin déjà préétabli : nous souhaitons utiliser SCTP multihoming. Nous allons voir dans cette section les problèmes que nous avons détectés à cette utilisation et pourquoi des besoins de communications inter-couches sont nécessaires. Nous présenterons ensuite le fonctionnement de la version SCTP que nous proposons avant de s'intéresser à ses performances et à son aptitude à coexiter avec un protocole SCTP de base, mesurée par l'équité.

4.3.1 Problème d'utilisation de SCTP multihoming en ad hoc

Nous détaillons plus précisément le multihoming sur des scénarios de topologie.

4.3.1.1 Différents scénarios pour SCTP multihoming en MANET

Analysons les corrélations entre chemins de transport et routes de réseau. Selon la topologie du réseau il se peut qu'il ne soit pas possible d'établir des routes disjointes pour les différents chemins SCTP ; de plus même si les routes sont disjointes d'un point de vue adresse IP, il peut cependant y avoir des noeuds communs. Un noeud qui a plusieurs interfaces, sera considéré par le routage comme deux noeuds distincts, alors que s'il se déplace, les deux routes construites à partir de chacune de ses interfaces seront rompues. Nous envisageons trois scénarios différents.

- Le premier considère deux noeuds multidomiciliés qui peuvent communiquer directement, sans relais.
- Le deuxième utilise un relais, les chemins ont des routes non disjointes.
- Le troisième utilise un relais et les chemins ont des routes différentes.

Cas 1

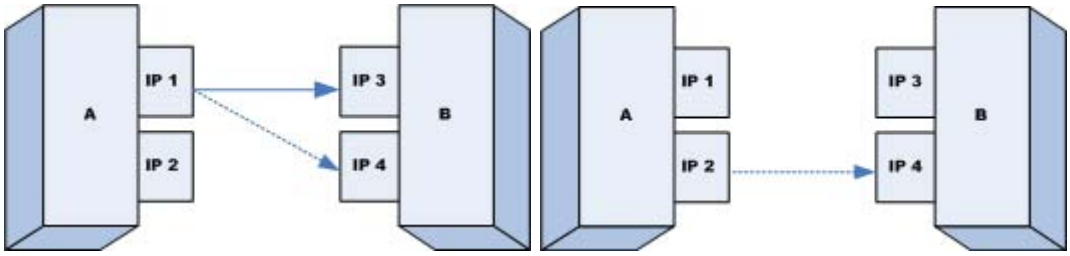
Le noeud source transmet directement à la destination (Figure 4.9(a)). Le chemin primaire est IP1->IP3, le chemin secondaire est soit IP1->IP4 soit IP2->IP4 (Figure 4.9(b)).

Les différentes étapes :

- Étape 1 : Le noeud A (IP1) initie un route request (RREQ) à l'adresse primaire (IP3) du noeud destinataire B et diffuse à ses voisins.
- Étape 2 : À la réception d'un RREQ, rediffusion par les voisins (non destinataire) éventuels.
- Étape 3 : Le noeud B (IP3) envoie un RREP au noeud source A (IP1)
- Étape 4 : Le noeud A (IP1) re-lance une RREQ à l'adresse alternative (IP4) du noeud B et diffuse à ses voisins.
- Étape 5 : À la réception d'un RREQ, rediffusion par les voisins (non destinataire) éventuels.
- Étape 6 : Le noeud B (IP4) envoie un route reply (RREP) au noeud source A (IP1).

Cas 2

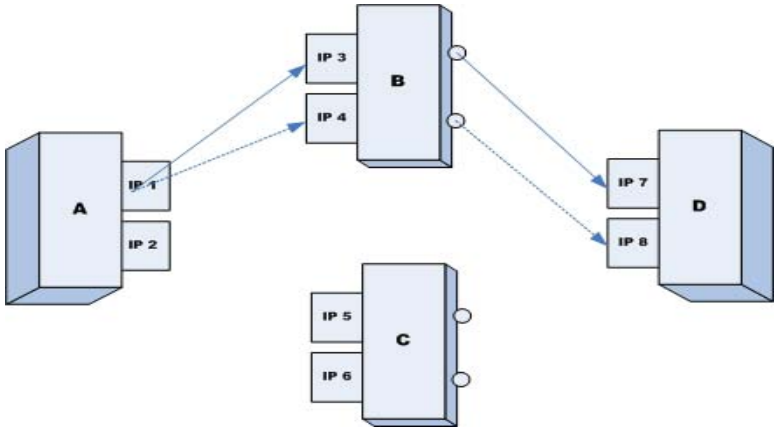
Le noeud A source ne peut pas directement communiquer avec la destination il utilise un ou plusieurs relais (Figure 4.10(a)). Le processus est similaire au cas 1. Il y a deux



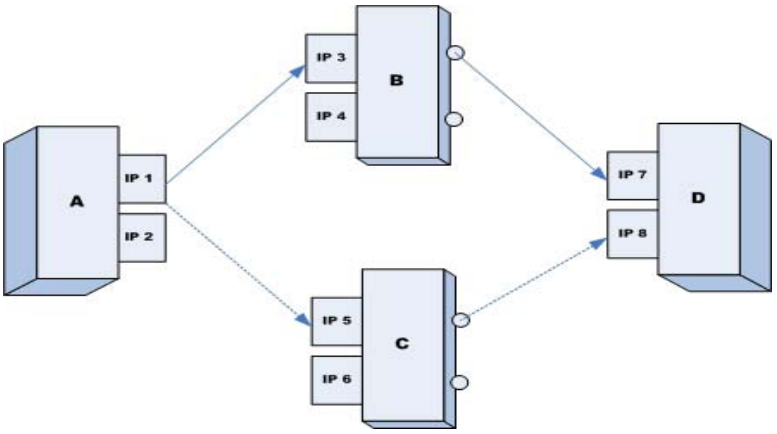
(a) IP1->IP3 ou IP1->IP4 (b) IP2->IP4

FIG. 4.9 – Topologie de multihoming en transmission directe

chemins de transport, sur deux itinéraires différents de réseau. Le chemin primaire est IP1->IP3->IP7 et le chemin secondaire est IP1->IP5->IP8.



(a) Deux routes disjointes



(b) Deux routes non disjointes

FIG. 4.10 – Topologie de multihoming avec des noeuds intermédiaires

Cas 3

Deux chemins de transport avec un noeud intermédiaire commun sont créés (Figure 4.10(b)), le chemin primaire est IP1->IP3->IP7 et le chemin secondaire est IP1->IP4->IP8.

4.3.1.2 Problèmes et solutions pour la mise en oeuvre et la simulation

A partir de l'étude des scénarios nous avons mis en avant d'une part un problème de conception que nous résolvons par une communication entre les couches 3 et 4, et d'autre part, un problème de simulation lié à la modélisation du multihoming sous NS2 que nous avons dû corriger.

1. Problème "discontinuité du transfert de données". Ce problème est commun aux trois cas de topologies : quand une erreur de route arrive le noeud source ne peut pas envoyer les paquets à la route existante alternative, car il ne sait pas que cette route arrive à la même destination, ainsi, le routage arrête d'envoyer ses paquets et rejette selon sa capacité mémoire les paquets de SCTP jusqu'à ce que la route soit rétablie. Comme l'adresse de destination IP du chemin alternatif diffère de celle du chemin primaire, AODV ne l'utilise pas, il essaie de rétablir la route.

2. Problème "requête de route superflue". Pendant que nous simulions avec NS2 les cas de topologies, nous avons constaté qu'en cas d'erreur de route sur le chemin primaire, la source émettait deux RREQ : un pour rétablir une route pour le chemin primaire, et un pour établir une route pour le chemin alternatif même si cette route était encore vivante. Un autre exemple de "requête de route superflue" est qu'en cas d'arrivée d'un RREQ, le noeud le diffuse trois fois. Ceci provient d'une erreur de conception du modèle de simulation disponible dans l'outil NS2.

   Après avoir examiné plus avant le code de simulation, nous avons constaté un problème d'implémentation du multihoming. AODV travaille au niveau noeud sans considérer les interfaces. Dans NS, un noeud de transport multidomicilié est vu comme un groupe de noeuds : il est composé d'un noeud central (core node), tel que le noeud A, et de plusieurs noeuds 'interfaces" tels qu'IP1 et IP2. C'est-à-dire que le noeud central dans la couche de routage agit comme un noeud d'interface. Quand un paquet est envoyé au noeud de destination, les noeuds intermédiaires "multidomiciliés" utilisent le noeud central comme un noeud pour expédier des données ou retransmettre la requête de route au noeud de destination. Nous avons alors 3 requêtes de routes : une pour chaque interface et une pour le noeud central. Notons que ce modèle fausse également la procédure de routage car le noeud central incrémente également le compteur de noeuds de AODV (le hop count).

Solution

Nous souhaitons accélérer la reprise du routage en utilisant la route du chemin secondaire au niveau 3 au lieu d'attendre que le protocole de routage ne fasse une découverte de route pour reétablir le chemin primaire. Pour cela nous proposons d'utiliser des informations de niveau 4 au niveau 3. L'information de niveau 4 sera utilisée sur tous les noeuds de routage associé au flux de transport : la communication inter couches n'est plus limitée à la source comme dans le chapitre précédent, elle est répartie sur les noeuds. Nous nommons cette optimisation SCTP ChangePath (SCTP CP).

4.3.2 Principe de fonctionnement de SCTP ChangePath

La communication entre couches que nous avons mise en oeuvre suit un modèle de type interactions vers une entité intermédiaire (cf chapitre 2).

Cette architecture effectue le partage de données par un service de stockage/récupération /positionnement d'informations entre deux couches (couches transport et routage). Elle est illustrée sur la Figure 4.11. Les informations échangées entre AODV et SCTP multi-homing sont gérées par le module "Distributed Cross Layer Interface" (DCLI). Ce module n'est pas uniquement un mécanisme implanté sur un noeud comme le Source Cross Layer Mechanism (SCLM) décrit dans le chapitre 3, c'est un module contenant ses données et ses processus, qui propose une interface de service.

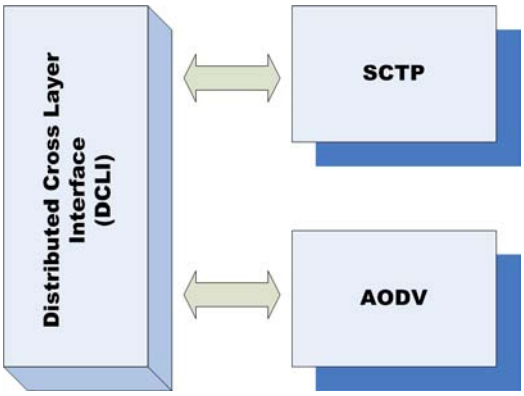


FIG. 4.11 – L'architecture de Distributed Cross Layer Interface

4.3.2.1 Fonctionnement inter-couches pour l'initialisation

Considérons un réseau MANET où la source A, avec une adresse IPA1, est en association avec un noeud B, sur une adresse primaire IPB1, et, sur une adresse secondaire IPB2. Le cas idéal pour une reprise d'erreur efficace se produit lorsque le réseau offre des routes distinctes entre le chemin primaire et le chemin secondaire [Kim07]. Ceci peut être obtenu par une modification du protocole de routage qui repose sur l'utilisation des informations d'adressage connues par SCTP. Le protocole de routage doit savoir que les chemins primaires et secondaires doivent être disjoints et donc les identifier en tant que tel.

Un algorithme de routage pour construire des routes qui soient le plus disjointes possible est présenté en [LeGe01]. C'est un routage à la demande par la source inspiré de DSR. La solution que nous proposons pour le protocole AODV dans le cas d'un multihoming SCTP, repose sur l'ajout de paramètres dans les messages route request et route reply. Nous l'explicitons avec une topologie de deux routes distinctes pour deux chemins.

1. Dans un premier temps, la source crée une route vers l'adresse primaire puis elle recoit le route reply avec un paramètre supplémentaire pour indiquer l'adresse (ou les adresses supplémentaires) du mobile destinataire.
2. La source utilise ensuite l'extension SCTP ADDIP pour inclure ces adresses dans l'association.
3. Lance une découverte de route du chemin secondaire avec un message de requête auquel est rajouté l'adresse du chemin primaire, de sorte que le traitement des noeuds

intermédiaires soit différent de celui effectué sur le route request antérieur correspondant au chemin primaire. S'il n'existe pas de route disjointe, le noeud intermédiaire incrémente un compteur de similitude. Le route reply transmis à la source contiendra la valeur de ce compteur.

4. Le paramètre de similitude pourra être utilisé pour décider ou non de l'intérêt d'un chemin secondaire.

4.3.2.2 Fonctionnement inter-couches pour la retransmission

En cas de panne sur le chemin primaire le protocole de routage n'est pas au courant de l'existence du chemin secondaire dans la version de base de SCTP alors que cette information est disponible dans SCTP CP.

Le comportement de SCTP CP en regard de la version standard SCTP est alors le suivant (Figure 4.12) :

1. La version SCTP lance une procédure de découverte de route pour chercher une nouvelle route, pour le chemin primaire, tandis que SCTP ChangePath peut envoyer les paquets sur la route associée au chemin secondaire directement et lancer une procédure de découverte de route pour la reprise de la route associée au chemin primaire.
2. Lors de la découverte de route, les noeuds intermédiaires étant au courant de l'association entre l'adresse IP primaire et l'adresse IP secondaire peuvent répondre à la requête de route par un route reply et diminuer ainsi la signalisation de routage.

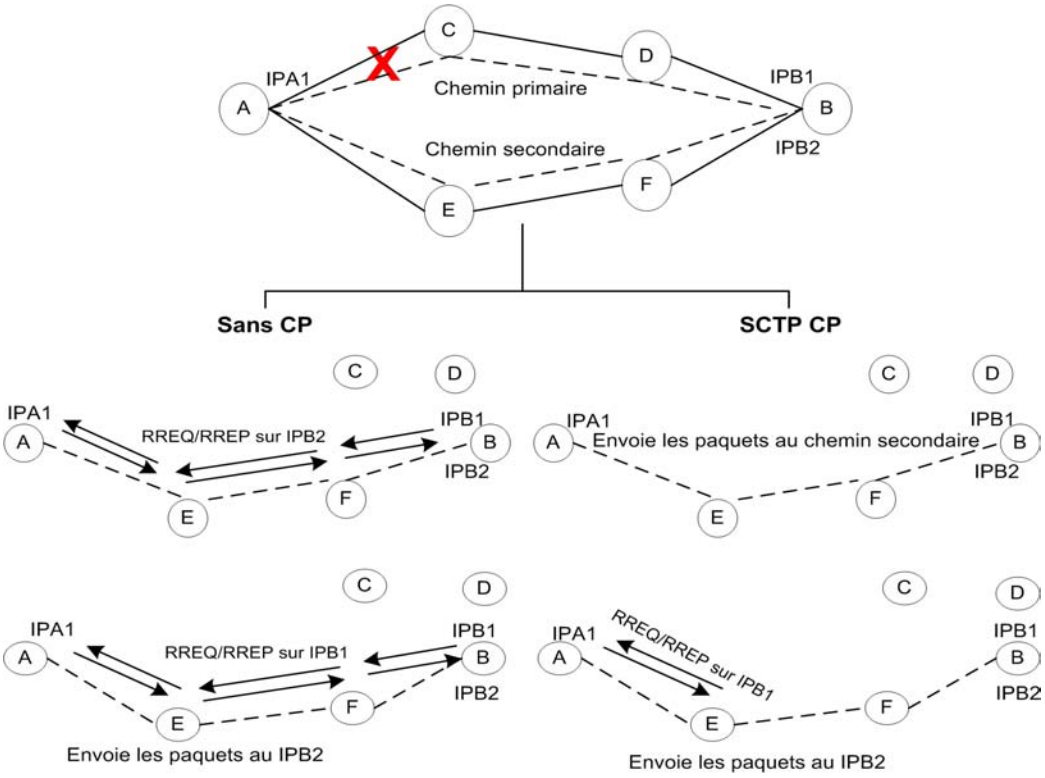


FIG. 4.12 – Idée de base pour SCTP CP

4.3.3 Mise en oeuvre avec le protocole AODV

4.3.3.1 Etablissement d'une association de SCTP multihoming avec AODV

SCTP établit l'association par les 4 messages (INIT, INIT-ACK, COOKIE-ECHO et COOKIE-ACK), AODV utilise 2 messages (Route Request et Route Reply).

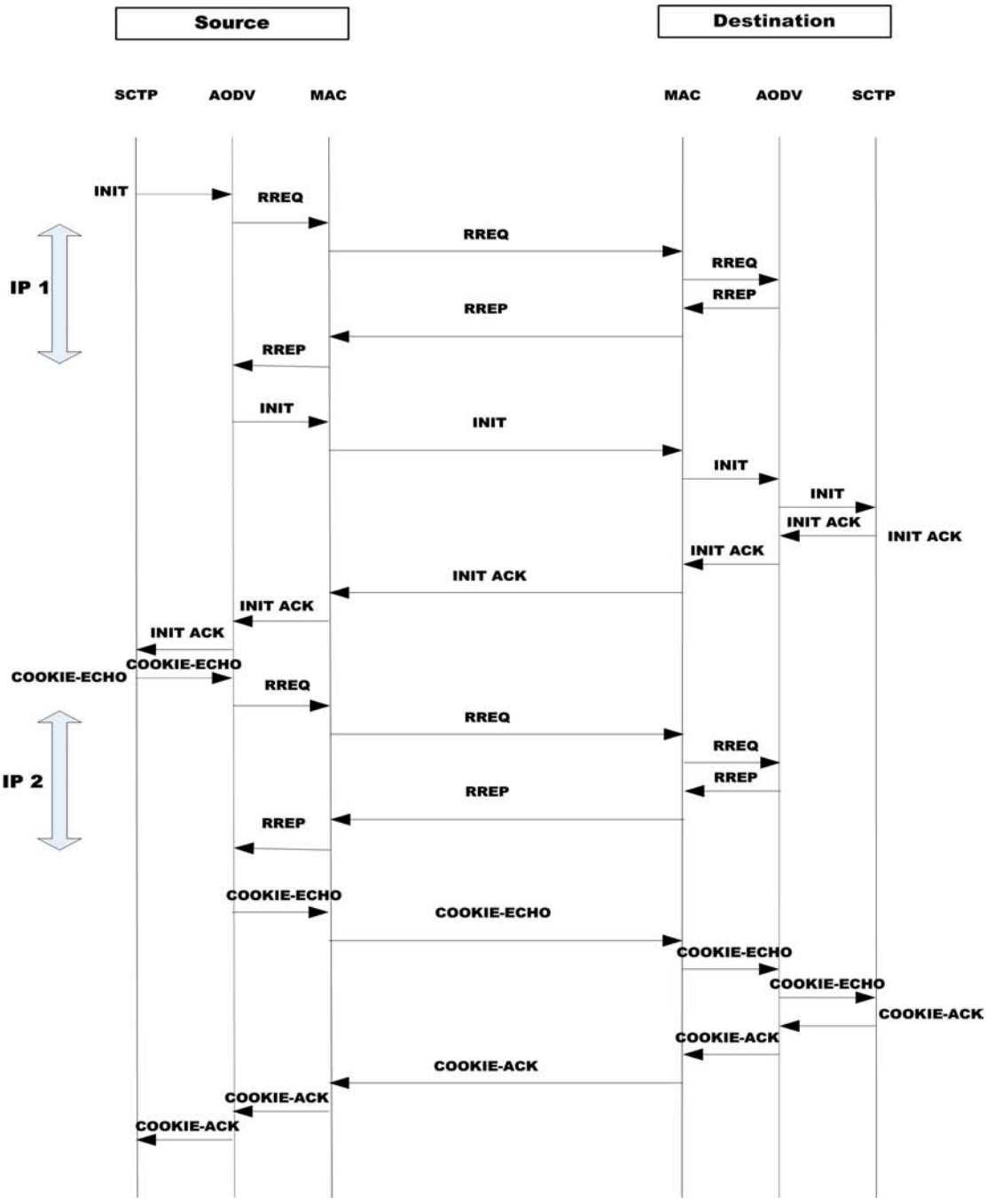


FIG. 4.13 – SCTP ChangePath dans le protocole AODV

La Figure 4.13 illustre le mécanisme de création de connexion par SCTP avec création de routes AODV.

Nous retrouvons un fonctionnement similaire à celui présenté dans le chapitre précé-

dent, exception faite du choix des chemins par lesquels passent les messages d'établissement de l'association :

- La source émet le message INIT et reçoit le message INIT-ACK sur le chemin primaire d'adresse IP1.
- Les messages COOKIE, COOKIE-ECHO sont transmis sur le chemin secondaire d'adresse IP2.
- AODV établit deux routes sur deux adresses destinataires distinctes.

4.3.3.2 Processus inter-couches

Dans notre proposition, c'est la couche de routage qui exploite les données de transport par l'interface de Distributed Cross Layer Interface (DCLI). C'est une vue locale qui contient l'information spécifique de noeud. DCLI contient des adresses IP, des numéros de ports et le statut des adresses IP (voir Figure 4.14).

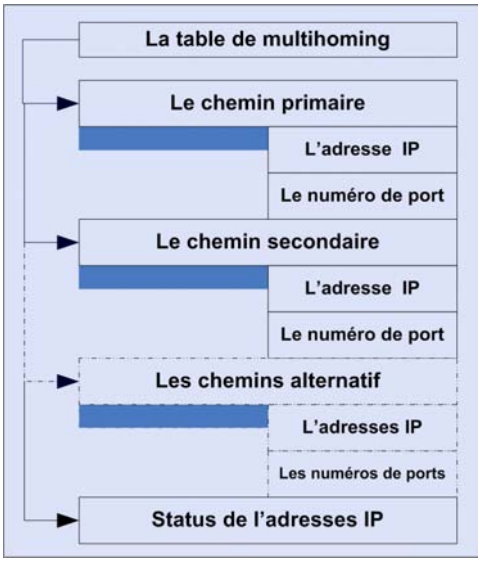


FIG. 4.14 – Informations dans le bloc DCLI : Distributed Cross Layer Interface

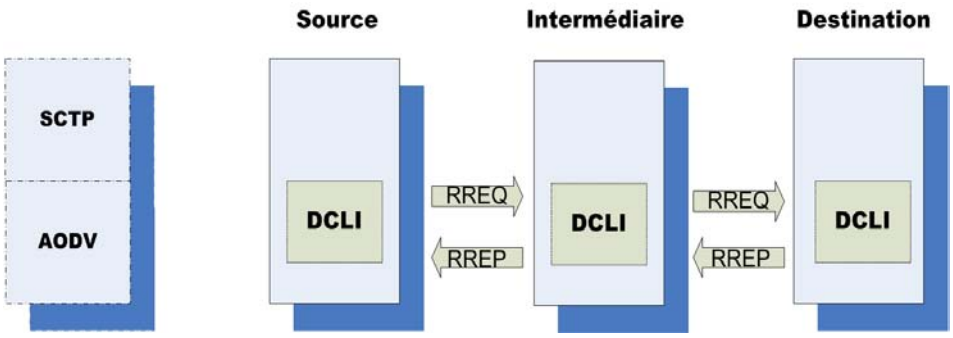


FIG. 4.15 – Cross layer dans chaque noeud

DCLI définit l'interaction entre les différentes couches de la pile protocolaire. Afin de résoudre le problème de discontinuité du transfert de données, le processus DCLI associe

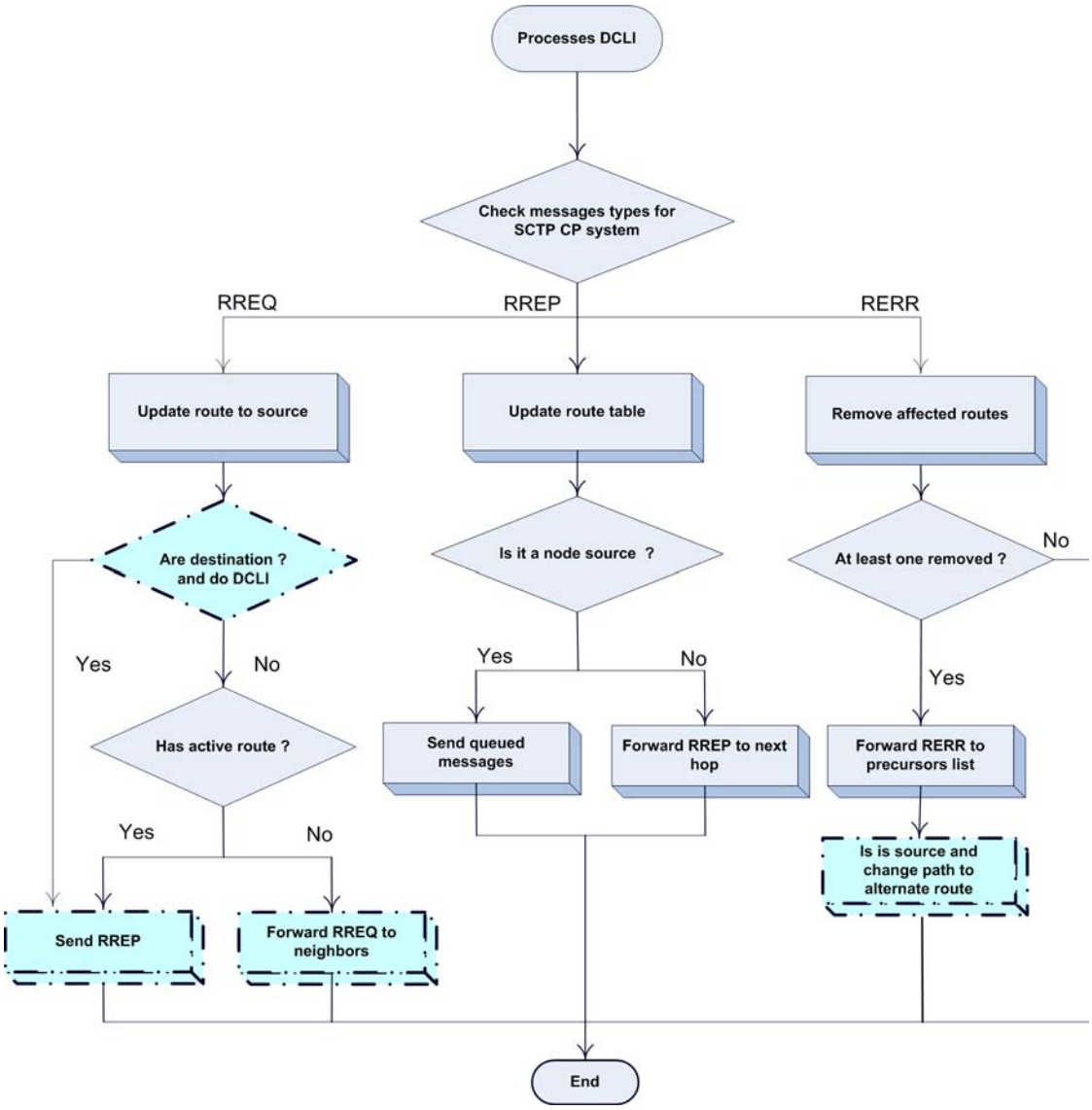


FIG. 4.16 – Les processus inter-couches

les données transport au chemin secondaire quand la route du chemin primaire a été rompue puis associe les données à nouveau au chemin primaire quand celui-ci redevient actif.

Les messages de routage utilisés par DCLI sont les messages route request et route reply pour obtenir les informations inter-couches et route error pour activer l'optimisation. Chaque noeud accède au DCLI quand il reçoit un route request (voir Figure 4.15).

Le détail des opérations est présenté en Figure 4.16.

Le processus de "route request"

Un noeud intermédiaire qui reçoit une requête de route utilise son bloc DCLI pour détecter sa situation de noeud multidomicilié avant de poursuivre la diffusion de la requête (nous n'avons pas représenté sur ce schéma le traitement de requêtes pour des routes disjointes qui n'entraine pas de modifications sur les processus inter-couches).

Le processus de “route error”

Quand un noeud intermédiaire reçoit une erreur de route, il marque son itinéraire à la destination comme invalide, et retransmet l'erreur de route à la source (sauf dans les conditions étudiées au chapitre 3). Le noeud source qui reçoit l'erreur infirme l'itinéraire primaire à la destination et choisit un chemin valide par la table de routage pour continuer à expédier ses paquets de données. S'il est la source, il modifie la route associée au chemin.

4.4 Evaluation de SCTP ChangePath

Nous évaluons le gain apporté par le mécanisme de changement de route géré par le processus inter-couches 3/4 en comparant notre solution avec les protocoles de base de transport tels que TCP Reno, TCP Sack et TCP Westwood et SCTP puis en la comparant avec l'optimisation SCTP PAUSE. L'environnement de simulation est analogue à celui du chapitre 3. Dans un premier temps nous déterminons l'option de retransmission la plus adaptée.

4.4.1 Evaluation des options MFR et TS

Fonctionnement de base de la retransmission

Dans SCTP ChangePath, quand le chemin primaire ne peut pas être utilisé, le flux est transféré sur un chemin alternatif. La retransmission de paquet sur le chemin alternatif est déclenchée par une expiration du timer (Timeout (TO)) ou une retransmission rapide (Fast Retransmission (FR)). Le fonctionnement de base consiste à utiliser une retransmission rapide sur le chemin primaire et en cas d'expiration du timer à émettre les paquets sur le chemin alternatif.

Les options

Deux options sont associées au fonctionnement de base : Multiple Fast Retransmision (MFR) et Timestamps (TS).

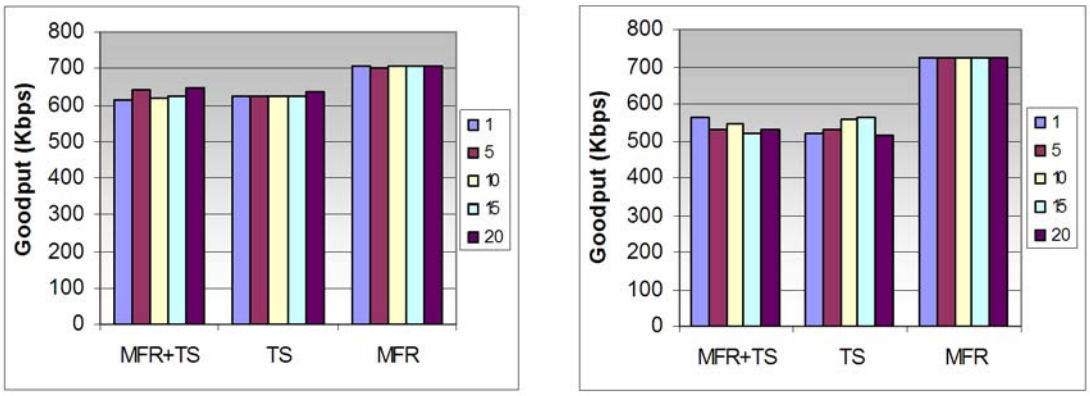
- MFR : sur un évènement de fast recovery (FR) retransmission rapide multiple sur le chemin primaire ; sur expiration timer (TO), émission des paquets sur destination alternative.
- TS : sur FR, fast retransmsion et fast recovery sur le chemin primaire. Sur TO, émission à la destination alternative.

Résultats d'évaluation pour le choix d'une option

Les Figures 4.17 (a) et (b) comparent sur un réseau de 10 noeuds et un réseau de 50 noeuds, le goodput de trois politiques de gestion de retransmission, à savoir : MFR, TS, MFR plus TS.

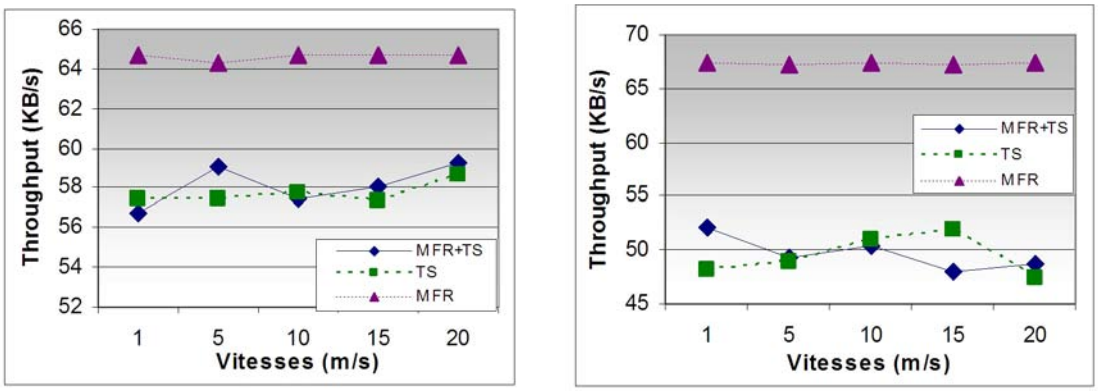
Nous constatons dans les deux configurations l'intérêt de ne pas utiliser l'option TS.

Précisons cette observation par deux autres résultats de simulations prenant en compte les retransmissions.



(a) Goodput par options sur 10 noeuds (b) Goodput par options sur 50 noeuds

FIG. 4.17 – Comparaison des options de multihoming pour SCTP CP



(a) 10 noeuds (b) 50 noeuds

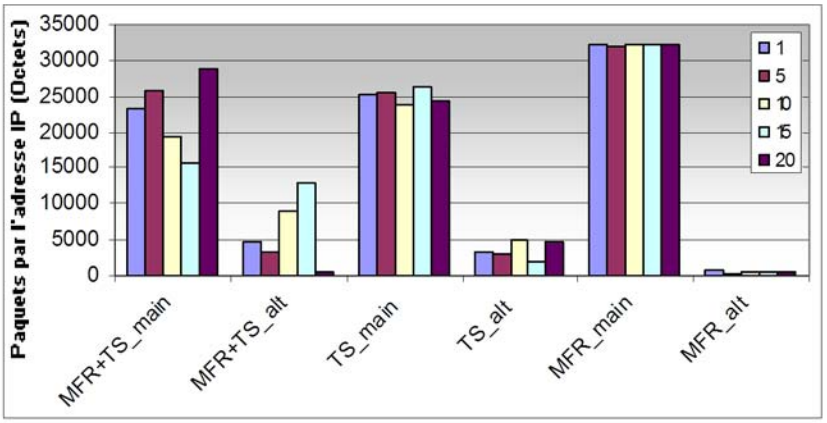
FIG. 4.18 – Throughput des options de multihoming pour SCTP CP (KB/s)

1. Le throughput qui indique les transmissions et les retransmissions (Figure 4.18). Le calcul s'effectue en octets pour une meilleure lisibilité des figures.

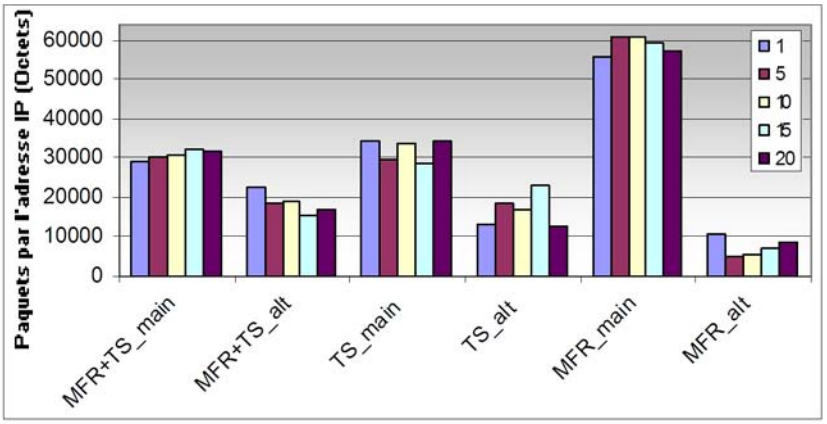
2. Les taux d'utilisation des chemins primaires et secondaires (Figure 4.19).

Nous constatons que l'option TS transmet moins d'informations qu'elle pourrait le faire : il semblerait que le calcul du timer ne soit pas particulièrement adapté à la mobilité des réseaux ad hoc, et si nous regardons le taux d'occupation des chemins (sur la Figure 4.19 ; *main* correspond au chemin principal et *alt* au chemin alternatif) nous constatons que lorsque le timer est calculé en TS, des informations sont transmises sur le chemin alternatif alors qu'elles auraient pu être transmises sur le chemin principal (comparaison MFR_main et MFR+TS_main).

Conclusion : nous choisissons l'option de retransmission MFR



(a) MFR+TS, TS et MFR pour 10 noeuds



(b) MFR+TS, TS et MFR pour 50 noeuds

FIG. 4.19 – Occupation des chemins selon les options

4.4.2 SCTP ChangePath et les autres optimisations transport

Nous comparons les goodput de TCP Reno, TCP Westwood (TCP W), TCP Sack, SCTP NewReno puis de SCTP PAUSE avec celui de SCTP ChangePath (SCTP CP).

Optimisation par multihoming et optimisation du contrôle de congestion

Nous retrouvons les résultats obtenus dans le premier chapitre que nous mettons en perspective avec ceux de notre proposition.

Comme indiqué en Figure 4.20, les performances de TCP Sack, TCP W et SCTP NewReno sont du même ordre. Notons que TCP Sack et SCTP NewReno ont des performances équivalentes en raison des choix de paramètres effectués dans le chapitre 2. **L'optimisation SCTP CP a un meilleur goodput. Nous confirmons l'intérêt d'utiliser des chemins multiples, montré dans la littérature pour les réseaux filaires et pour les protocoles de routage ad hoc**.

Comparaison : Optimisation par communication inter-couches à la source ou répartie ?

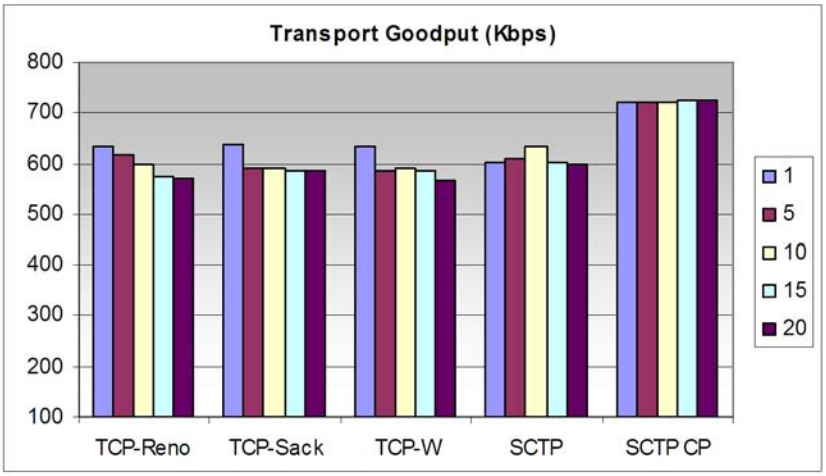


FIG. 4.20 – Comparaison des Goodput (Kbps) des protocoles transport

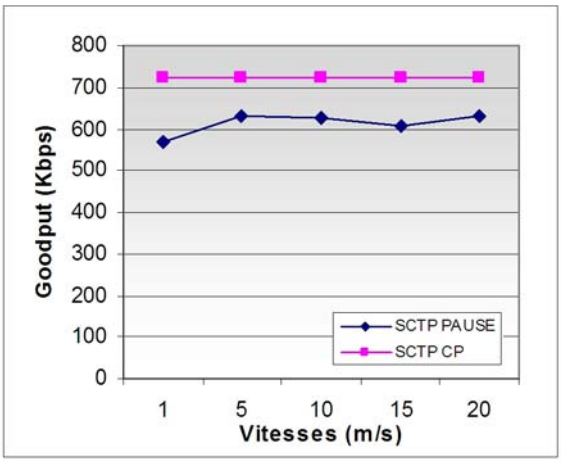


FIG. 4.21 – Comparaison des Goodput des protocoles SCTP PAUSE et SCTP CP

La comparaison de performance entre SCTP PAUSE et SCTP CP est indiquée en Figure 4.21. Nous observons que SCTP CP obtient de meilleures performances.

En conclusion. **Globalement nous obtenons un gain de performance, avec SCTP CP devant les versions sans communications inter-couches et devant la version de SCTP PAUSE.** Néanmoins, certaines topologies doivent être plus favorables que d'autres à l'utilisation de SCTP CP. Intuitivement, si le chemin primaire et le chemin secondaire sont similaires l'optimisation n'a pas raison d'être.

4.4.3 Influence de la topologie sur SCTP ChangePath

Analysons l'intérêt d'utiliser SCTP CP par l'examen de ses performances sur des topologies présentant des degrés de similitudes différents. Le degré de similitude est la mesure que nous définissons pour prendre en compte le fait que les chemins primaires et

secondaires soient disjoints.

4.4.3.1 Modèle d'étude

Degré de similitude

N	noeud de réseau
IPP	adresse IP Primaire
IPA	adresse IP Alternative
] S,D [	ensemble ordonné de noeuds permettant de connecter S à D, (S, D exclus)

R(P) = [$N \setminus N \in$]$S, D.IPP$[ ] : Route Primaire entre S (source) et D (destination) sur l'adresse IPP (IP Primaire).

R(A) = [$N \setminus N \in$]$S, D.IPA$[ ] : Route Alternative entre S (source) et D (destination) sur l'adresse IPA (IP Alternatif).

$Ds = card[N \setminus N \in R(P) \wedge N \in R(A)]$: Degré de similitude.

Nous normalisons le degré de similitude par le nombre de noeuds moyen composant les routes.

$\overline{Ds} = Ds \div ((cardR(P) + cardR(A)) \div 2)$

Les topologies de test que nous utilisons sont indiquées en Figure 4.22. Nous obtenons :

- T1 : Source = A ; Destination = F ; **R(P)** = B,D ; **R(A)** = C,E et $\overline{Ds} = 0 \div 2 = 0$,
- T2 : Source = A ; Destination = E ; **R(P)** = B,D ; **R(A)** = C,D et $\overline{Ds} = 1 \div 2 = 0.5$,
- T3 : Source = A ; Destination = D ; **R(P)** = B,C = **R(A)** et $\overline{Ds} = 2 \div 2 = 1$.

Modèle de déplacement

Les hypothèses sont :
- Un déplacement d'un élément de la route entre la source et la destination qui provoque une rupture de route.
- La probabilité de rupture est équitablement répartie sur les noeuds.
- Il y a card (R (P)) scénario de mobilité.

Exemple : sur la topologie T1, nous avons 2 scénarios de mobilité : le mobile B se déplace, le mobile D se déplace.

Sur chaque topologie et chaque scénario, nous mesurons le goodput d'un flux SCTP CP sur le chemin primaire. Le goodput G obtenu est la moyenne des goodputs des différents scénarios.

$G = (\sum Gi) \div card(R(P))$

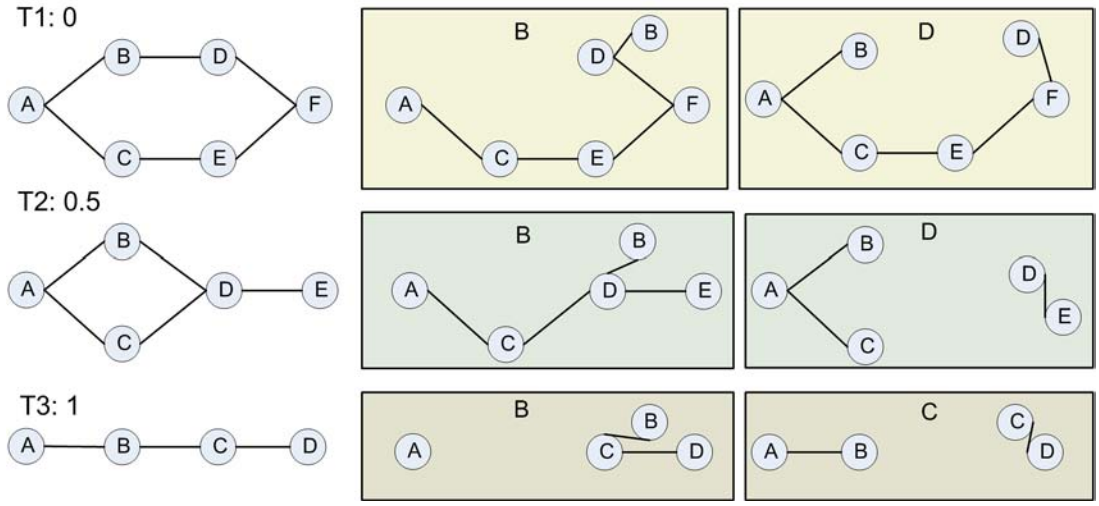
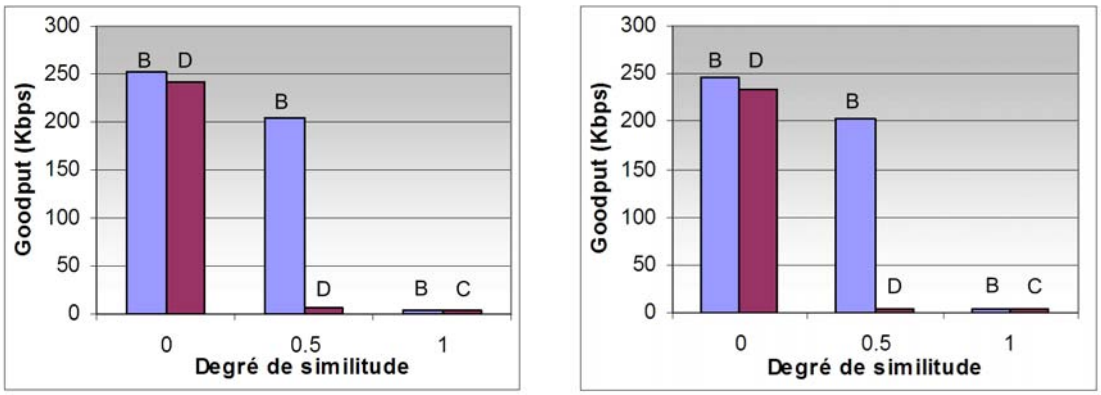


FIG. 4.22 – Trois topologies avec un degré de similitude différent

4.4.3.2 Résultats et interprétations

Les résultats de simulations sont indiqués en fonction du degré de similitude de la topologie. Lorsque le degré est de 1 le goodput est la quantité d'information qui a été émise avant que la panne ne se produise. Il est ensuite nul puisque il n'y a pas de chemin secondaire ni de route AODV de secours. Plus précisément :

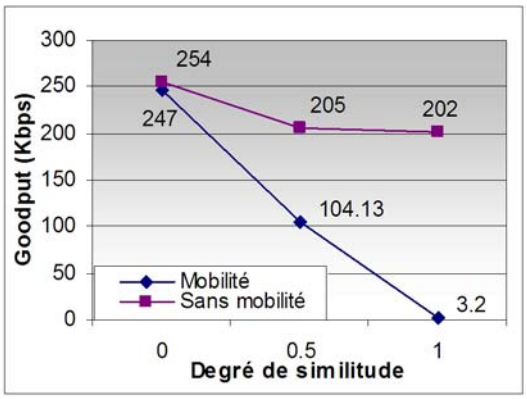
1. Figure 4.23 : précise les valeurs de goodput par scénario. Le goodput est fonction de la position de la rupture de route. Plus la rupture de route est éloignée de la source plus celle-ci met de temps à s'en apercevoir et donc à basculer de route,
2. Figure 4.24 : indique le goodput moyen en référence au goodput qui aurait été obtenu sans mobilité. La décroissance du goodput est directement liée au degré de similitude,
3. Figure 4.25 : le gain de goodput entre SCTP CP/SCTP. Le gain de goodput normalisé est satisfaisant.



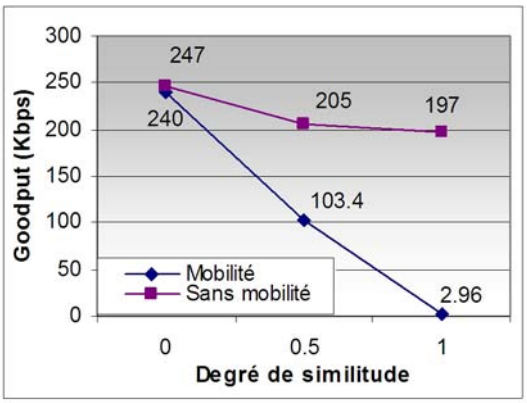
(a) SCTP CP

(b) SCTP

FIG. 4.23 – Goodput par scénario de mobilité

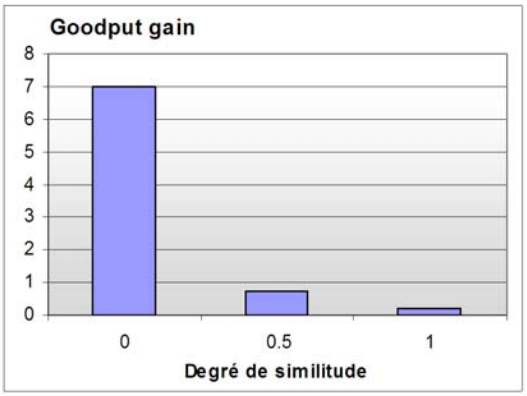


(a) SCTP CP

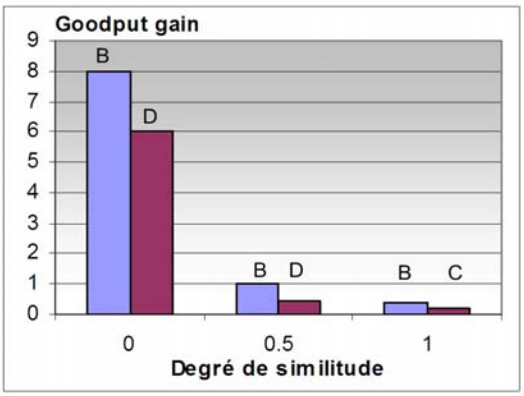


(b) SCTP

FIG. 4.24 – Goodput moyen



(a) Gain de goodput total



(b) Gain de goodput pour chaque noeud

FIG. 4.25 – Gain de goodput SCTP CP/SCTP

En conclusion l'intérêt de l'optimisation dépend fortement de la topologie et de sa capacité à supporter des routes disjointes entre une source et une destination.

4.4.4 Equité dans les protocoles SCTP ChangePath et SCTP

Nous examinons l'impact de l'optimisation sur des flux transmis en version non optimisée avec une méthode similaire à celle du chapitre précédent, nous calculons l'équité par l'Indice de Jain (cf chapitre 3.3).

La topologie du scénario que nous étudions est indiquée sur la Figure 4.26.

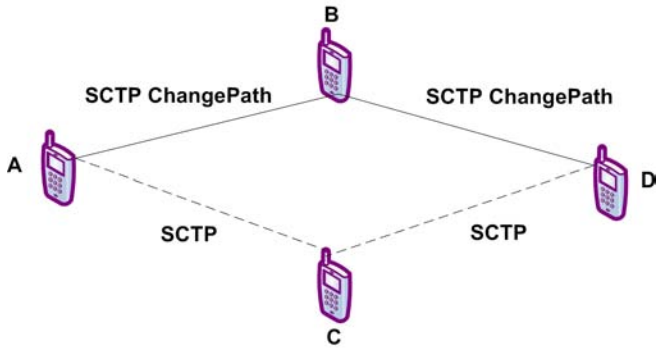


FIG. 4.26 – Scénario d'étude de l'équité SCTP ChangePath/SCTP

- A l'instant T0, un flux transmis en SCTP ChangePath (SCTP CP) circule sur le chemin primaire de route A-B-D et un flux émis en SCTP circule sur la route A-C-D.
- A l'instant T1, le mobile B se déplace.
- A l'instant T2 le flux SCTP CP est basculé sur son chemin secondaire passant par A-C-D et doit partager sa bande passante avec le flux SCTP de base.

Les résultats de simulations indiqués ci dessous sont prévisibles, dans la mesure où le protocole SCTP multihoming est équitable et que l'optimisation ne change pas le comportement de transmission. Ils montrent bien l'équité des deux versions (table 4.5) sauf durant la phase de rupture de route ou naturellement le flux brisé ne peut obtenir de bande passante (Figure 4.27). Ce résultat nous permet néanmoins de comparer les deux optimisations que nous proposons.

Scénario	Equité	
	SCTP CP	SCTP
Figure 4.26	0.9935	0.9923

TAB. 4.5 – Equité de SCTP CP et SCTP

Si nous comparons ces résultats avec ceux obtenus au chapitre 3, nous constatons que la diminution de débit globale est moins importante avec SCTP CP que SCTP PAUSE (Figure 4.28), confortant ainsi les résultats obtenus dans les paragraphes précédents sur l'avantage d'utiliser SCTP CP.

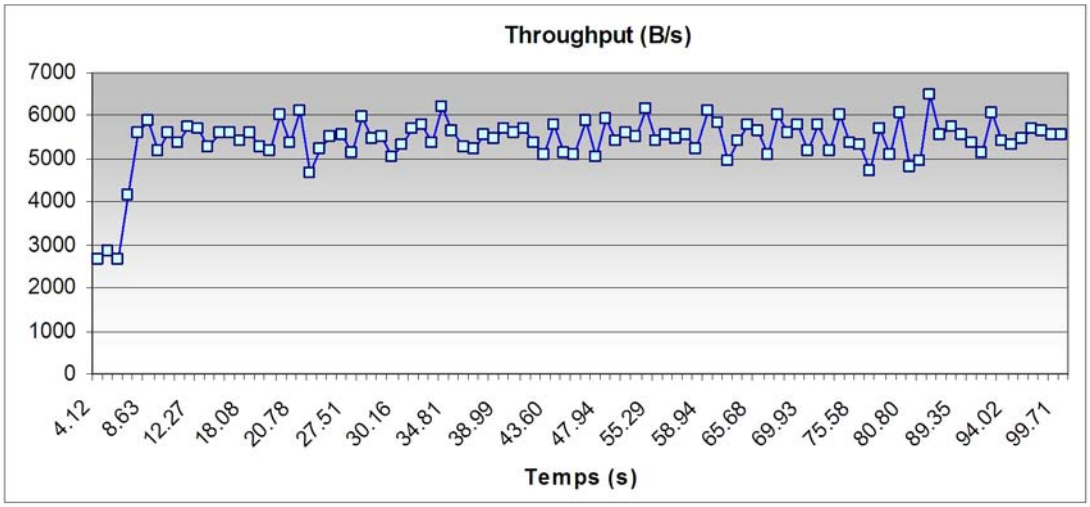


FIG. 4.27 – Throughput global des deux flux

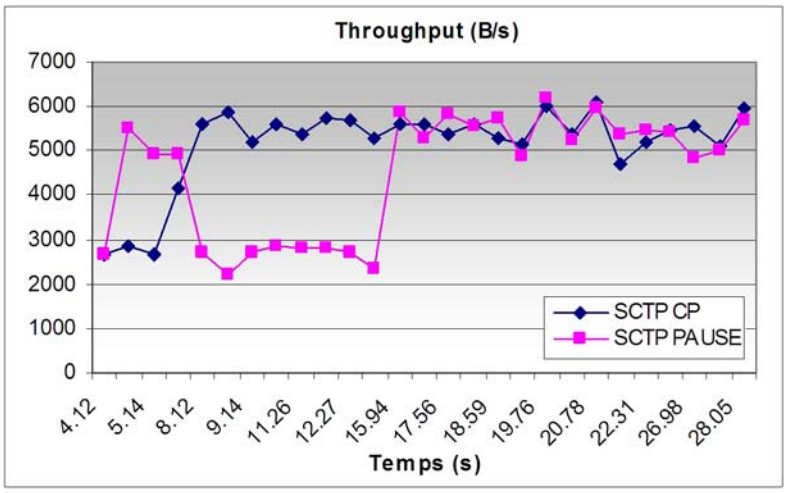


FIG. 4.28 – Etude de l'effet des performances (4-30s)

4.4.5 Comparaison SCTP ChangePath et SCTP PAUSE

Nous synthétisons les avantages de la solution SCTP ChangePath (CP) en regard de SCTP PAUSE comparativement aux caractéristiques de SCTP NewReno, par la table ci-dessous.

- **La performance** mesurée en goodput est globalement, c'est à dire en moyenne pour touts les flux avec un modèle de mobilité RWP, la meilleure (+++) pour SCTP CP ; la moins bonne pour SCTP NewReno (+).
- **L'équité** est vérifiée pour toutes les versions (+++).
- **Le passage à l'échelle**, c'est à dire les résultats de performance pour 10 noeuds et 50 noeuds : SCTP CP est moins sensible que SCTP PAUSE (++), dont l'activation depend du nombre d'erreurs de routes reçues (celui-ci diminue avec la longueur du chemin en raison du mode de maintenance optimisé de AODV). Les performances de SCTP PAUSE (++) pour 50 noeuds sont meilleures que celles de SCTP mais SCTP de base a des performances qui se dégradent de manière équivalente à celles de SCTP PAUSE.
- **La signalisation**, les informations de surcharge rajoutées pour faire fonctionner l'optimisation, dans notre réalisation : les informations rajoutées dans les messages AODV. SCTP NewReno et SCTP PAUSE ne rajoutent aucune information (+++), SCTP CP rajoute des informations dans des messages existants (++).
- **La complexité** : SCTP NewReno ne demande aucune modification d'implantation (+++), SCTP PAUSE demande l'ajout d'une interface pour recevoir le message de niveau 3 et une modification minime du contrôle de congestion (++) ; SCTP CP demande des modifications plus importantes.

Caractéristique	Protocole SCTP		
	CP	PAUSE	NewReno
Performance : goodput	+++	++	+
Equité	+++	+++	+++
Passage à l'échelle	+++	++	++
Signalisation	++	+++	+++
Complexité	+	++	+++

TAB. 4.6 – Comparaison des caractéristiques de SCTP CP, SCTP PAUSE et SCTP NewReno

4.5 Conclusion

Nous avons évalué une optimisation de SCTP par une communication inter-couches qui augmente la robustesse du transfert à la mobilité en s'appuyant sur un transfert multihoming. L'optimisation grâce à la communication inter-couches accélère la reprise du routage.

Les résultats obtenus montrent l'intérêt de cette optimisation, ils doivent cependant être complétés par la prise en compte de la topologie du réseau.

Pour ce faire nous avons introduit la notion de degré de similitude qui indique le nombre de noeuds communs entre le chemin primaire et le chemin secondaire sur lequel le flux sera basculé en cas de panne du chemin primaire. Nous avons normalisé ce degré en fonction du nombre de noeuds du chemin. L'intérêt de l'optimisation est fortement corrélé au degré de similitude normalisé.

L'optimisation devient inutile voire coûteuse si le chemin alternatif à la même route que le chemin primaire.

Un autre facteur influence les performances : la longueur du chemin. Comme le taux de pertes est corrélé à la longueur du chemin, lorsque celle-ci est importante, le gain de l'optimisation diminue. Le temps de retransmission devient prépondérant devant les délais réseau. L'optimisation ou plutôt le fonctionnement de base du multihoming peut même s'avérer pénalisant si après une panne de chemin primaire le protocole de transport rebascule sur un chemin primaire qui a été rétabli par une route plus longue que celle du secondaire. Ceci peut se produire si le protocole de routage cherche à établir des routes disjointes. Nous proposons alors une auto-adaptation du niveau 4, fonction du RTT mesuré sur le chemin secondaire et le chemin primaire rétabli. Si celui du chemin rétabli est plus important alors les chemins primaires et secondaires sont inversés (L'auto-adaptation devra prendre garde à éviter les problèmes d'oscillations de chemins).

Chapitre 5

Synthèse et perspectives d'utilisation des résultats

Sommaire

5.1 Synthèse des résultats

Dans cette thèse nous avons étudié l'adaptation inter-couches du transfert fiable SCTP dans les réseaux ad hoc. Notre objectif était d'améliorer les performances du transport de façon à utiliser au mieux les ressources limitées des réseaux ad hoc.

Les travaux que nous avons exposés montrent qu'il n'existe pas de solution qui soit adaptée à toutes les conditions d'erreurs du réseau. De ce fait l'adaptation du mode de transport à l'état du système de communication par une communication inter-couches qui évite de consommer de la bande passante avec une signalisation supplémentaire, tout en permettant de réutiliser simplement un protocole standard, nous a paru une solution intéressante.

L'étude menée sur les protocoles de transport traditionnels nous a montré l'importance du routage ad hoc sur les performances de transport et nous a conduit à considérer plus particulièrement les interactions entre réseau et transport. Nous avons défini deux modes d'adaptation inter-couches prenant en compte le routage, un mode local et un mode réparti sur les noeuds.

Le mode d'adaptation local adapte sur la source le protocole de transport à la mobilité du réseau perçue par le protocole de routage. La métrique de mobilité est définie par une rupture de chemin, connue via un message existant du routage (le route error dans le cas de AODV) qui est émis vers la source. Cette métrique peut être étendue à d'autres méthodes de mesures, telle la durée de liaison, le nombre de voisins ou encore la vitesse de déplacement [YaPD08]. L'adaptation consiste en cas de mobilité à inhiber à la source le contrôle de congestion du transport de façon à laisser le protocole de routage faire son travail en rétablissant le chemin, et non pas à réduire inutilement le débit d'émission parce que de par son adaptation interne, le transport perçoit une congestion. Lorsque le chemin se rétablit, le transport transmet à un débit correspondant à celui qu'il avait lors

de la panne de mobilité. L'étude détaillée de cette proposition a montré son domaine de validité, c'est à dire les cas ou ses performances sont meilleures que celles de la solution standard. Le domaine de validité se définit par la bande passante disponible sur le chemin de secours. Lorsque sur le chemin de secours des flux se partagent la bande passante, l'arrivée d'un nouveau flux va provoquer une congestion, ce nouveau flux, de par son algorithme de congestion qui a été modifié, va ralentir plus lentement son débit et donc l'arrivée à un état d'équilibre sur la nouvelle route sera plus lente que s'il n'y avait pas eu de modifications. Le débit global s'en trouve ralenti. Un fonctionnement adaptatif efficace de l'optimisation serait alors de désactiver l'optimisation en fonction du degré d'utilisation de la route (la bande passante disponible). Cette information actuellement non disponible avec les protocoles de routage MANET standardisés, devrait être accessible dès lors que l'on utilise un protocole de routage à qualité de service ([BMAP03] [MuFo07]).

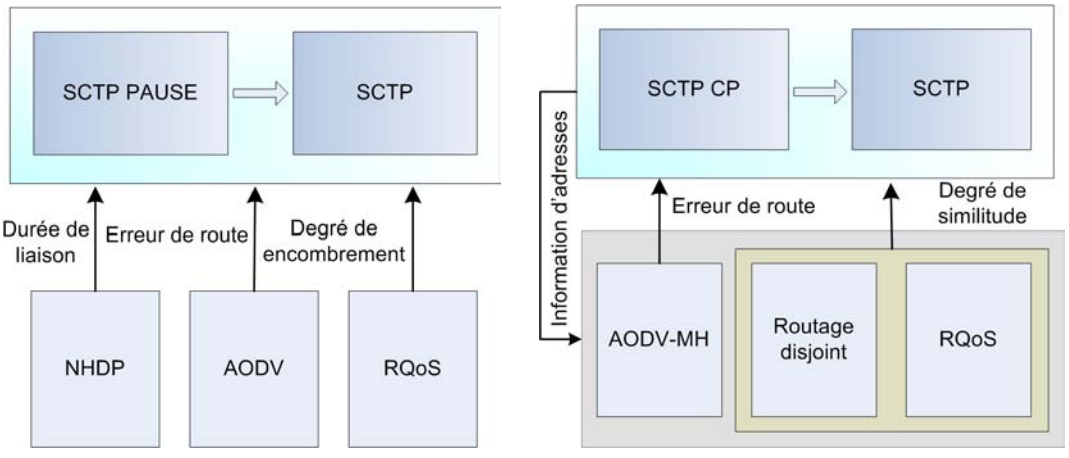
Nous avons également montré que selon le fonctionnement du protocole de routage l'optimisation peut n'être activée que rarement. En effet celle-ci est activée sur réception de messages d'erreurs de routes et certaines optimisations du routage tendent à diminuer ces messages. Un agent ayant la connaissance du fonctionnement de routage peut alors décider soit de changer de métrique de mobilité soit de ne pas activer l'optimisation pour minimiser les ressources consommées sur le poste source. Dans le premier cas l'optimisation SCTP CP sera activée par exemple sur une métrique de durée de liaison et non plus d'erreur de route. Un mobile qui se déplace et rompt sa route détectera localement la rupture, grâce à cette métrique, alors qu'il ne l'aurait pas forcément fait avec la métrique précédente. Le protocole Neighborhood Discovery Protocol (NHDP) [ClDD08] actuellement à l'étude par le groupe MANET, est un protocole indépendant du routage qui peut être utilisé pour après calcul de la métrique durée de liaison, activer SCTP PAUSE. Un agent qui a une vue globale de l'état du réseau peut également décider de ne pas activer SCTP PAUSE si le réseau est trop chargé (pas uniquement la route de secours), ou trop mobile. S'il est trop chargé, il ne faut pas toucher au contrôle de congestion, et intuitivement s'il est trop mobile il risque d'y avoir un très grand nombre de pertes qui tarderont a être détectées (arrêt du timer).

La figure 5.1(a) indique les informations inter-couches nécessaires à l'activation de l'optimisation. Nous retrouvons *l'erreur de route* expérimenté dans le chapitre 3 avec AODV, qui peut être amélioré et remplacé (ou combiné) par un paramètre de *durée de liaison* fourni par un protocole de voisins type NHDP, le *degré d'encombrement* d'une route, obtenu grâce à un routage à Qualité de service. Ces paramètres sont également utilisables pour passer dans un mode SCTP normal.

Le mode d'adaptation réparti adapte le protocole de routage à la topologie de flux multi domiciliés mis en oeuvre de façon dynamique par le protocole SCTP ADDIP. Connaissant l'existence d'un chemin alternatif géré par le niveau transport, le protocole de routage cherche, en cas de rupture de route, à utiliser ce dernier le temps de rétablir une route pour le chemin primaire. Il s'agit pour le niveau réseau de connaître l'information de lien entre identifiant et localisateurs qui est connue du protocole transport. Nous proposons d'obtenir cette information à l'établissement de l'association transport en rajoutant un minimum de signalisation : ce sont les messages de routage traditionnels qui véhiculent l'information. Lors de l'établissement d'une route les noeuds intermédiaires mémorisent les localisateurs d'une même route. En cas de rupture de route connue sur un localisateur, le protocole de routage sera alors capable d'utiliser une route existante sur un autre localisateur. Dans les faits nous obtenons un double routage. La mise en oeuvre que nous avons effectuée repose, au niveau réseau, sur un routage réactif, AODV, complété par

un routage proactif de secours, alors que le routage de niveau transport qui établit, un chemin alternatif, s'effectue à la demande, en mode réactif. Notre proposition doit pouvoir également s'appliquer à un protocole réseau proactif tel OLSR. Les basculements entre route de transport et de routage peuvent être améliorés en choisissant en chemin primaire celui qui a la route la plus courte, ceci étant mesuré grâce à la variable temps d'aller/retour (RTT) du niveau transport. Cette optimisation doit être contrôlée par un mécanisme de seuil de façon à éviter des oscillations de chemins préjudiciables aux performances.

La connaissance des données du transport, améliore les performances du protocole de routage en accélérant la reprise de la transmission et donc les performances transport. Le fait que l'utilisation du chemin secondaire ne puisse avoir lieu que si celui-ci n'est pas en panne influe naturellement sur les performances transport, mais cette influence n'est pas propre à l'optimisation proposée, car elle est déjà présente au niveau du protocole de routage. Si la route secondaire est en panne le protocole de routage par nature cherche une autre route. Par contre, la connaissance de la correspondance identifiant-localisateur peut accélérer la reprise, en cas de panne sur le chemin primaire et sur le chemin secondaire. Le protocole de routage peut chercher deux routes sur deux localisateurs avec un même identifiant, en une même requête. Grâce à leur connaissance les noeuds seront à même de traiter la requête. Remarquons que si le degré de mobilité qui peut être mesuré par le nombre d'erreurs de routes reçues, ou autre métrique, est trop important, il y a de fortes chances que le double routage soit inefficace. Le fonctionnement transport multi domiciliés au niveau routage peut alors être désactivé en cas de trop grande mobilité, il peut cependant être maintenu au niveau transport de façon à prendre en compte des interfaces multiples sur un équipement. Une autre limitation à la solution que nous avons mise en avant concerne la différentiation des routes associées à un même identifiant. Nous avons montré que le domaine de validité est déterminé par le degré de similitude des routes. Une façon d'augmenter ce degré est de modifier le protocole de routage standard en utilisant un paramètre supplémentaire, le degré acceptable, lors de la phase d'établissement de route. Le degré de similitude peut être interprété comme un paramètre de la qualité de service obtenu au niveau routage.



(a) SCTP PAUSE/SCTP (b) SCTP CP/SCTP

FIG. 5.1 – Activation des optimisations

Nous proposons d'utiliser ce paramètre pour décider ou non de l'intérêt d'utiliser des

ressources à exécuter une optimisation qui ne sera pas efficace. Un contexte multidomicilié est alors crée, au niveau routage, uniquement pour les identifiants dont le degré de similitude sur les routes associées à leurs localisateurs est inférieur à une borne.

La Figure 5.1(b) indique les informations inter-couches pour l'activation de SCTP CP. Le routage multihoming, c'est à dire celui ou les noeuds ont la connaissance de l'association identifiants-localisateurs, active sur une erreur de route SCTP CP (Nous le notons AODV-MH pour le différentier de la version de base utilisée pour SCTP PAUSE, il correspond à celui expérimenté dans le chapitre 4). Le degré de similitude peut être généré par un routage à qualité de service ou par un routage spécifique qui établit des routes disjointes.

Comme nous le voyons, les propositions d'améliorations que nous avons faites, suivies de leur évaluation et donc de leur limitations, nous amène à proposer différentes règles d'optimisations. Ces règles reposent sur des métriques qui sont obtenues à partir d'une connaissance multi-niveaux, et permettent d'adapter la pile protocolaire. Avant de définir une liste de toutes les règles d'optimisations et métriques il nous paraît important de poursuivre ces travaux par l'étude de l'architecture de mise en oeuvre des différentes optimisations. Cette architecture doit permettre de définir dynamiquement des règles, des métriques et des optimisations.

5.2 Une architecture protocolaire adaptative par communication inter-couches

Nous proposons d'utiliser nos résultats pour spécifier les caractéristiques d'une architecture protocolaire adaptative à même d'optimiser les performances du transfert de données. Dans un premier temps nous identifions les composants du processus d'adaptation, puis nous proposons un modèle d'architecture pour le mettre en oeuvre dans une pile protocolaire standard.

5.2.1 Processus d'adaptation

A partir des règles d'optimisations que nous avons énoncées dans nos résultats, nous pouvons différencier deux niveaux d'adaptation. Premièrement une adaptation de configuration protocolaire qui activerait le fonctionnement transport le plus adapté à la pile protocolaire existante, et à l'état du système de communication. Deuxièmement, une adaptation interne au protocole, de l'auto-adaptation, qui permettrait de régler de façon optimale le fonctionnement.

Le processus d'adaption (Figure 5.2) est alors exécuté par l'intermédiaire de deux types d'agents :

- un agent de configuration, le **Configuration Agent** ;
- un agent de mise au point, le **Tuning Agent**.

Le premier agent utilise des données de configurations disponibles dans la **MIB** (Management Information Base) alors que le second est en relation avec des données gérées par les protocoles, les données opérationnelles générées par des protocoles en activité qui servent à l'optimisation, disponibles dans la **COB** (Cross layer Operational Base).

Plus précisément, les agents vont agir sur les protocoles en fonction d'une politique d'optimisation, **Optimization Policy**, définie par des règles, telles la règle R1 "si degré de similitude supérieur à x alors désactiver SCTP CP" ou encore la règle R2 "si SCTP CP activé alors activer Routage-Disjoint". Pour appliquer ces règles un traitement (metric

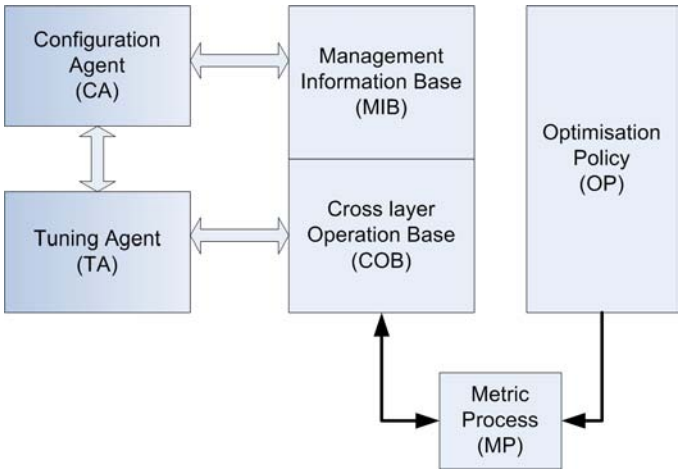


FIG. 5.2 – Processus d'adaptation

process) peut être nécessaire pour calculer des métriques élaborées à partir des données disponibles dans la COB.

Les règles d'optimisation sont déduites des domaines de validité des optimisations telle la règle R1 et des configurations de la pile de protocole disponibles telle la règle R2. L'agent de configuration a une vision globale du système, il peut gérer la cohérence des optimisations. L'agent de mise au point dispose quand à lui d'une vue locale.

5.2.2 Modèle d'architecture d'adaptation

Nous proposons une architecture en 4 plans (Figure 5.3).

- Nous retrouvons deux plans classiques des architectures de réseaux ; le plan de données utilisateur (User Plane) et le plan de gestion (Management Plane) auquel nous rajoutons,
- un plan de contrôle (Control Plane) chargé d'optimiser le plan de données et,
- un plan de communication inter-couches (Cross Layer Plane).

5.2.2.1 Plan de données et Tuning Interface

Le plan de données est étendu pour communiquer avec les deux plans que nous avons rajoutés grâce à une interface de mise au point (TI : Tuning Interface). Cette interface est définie pour chaque protocole adaptable, elle contient les paramètres qui seront utilisés pour l'adaptation ainsi que les processus d'adaptation, telle l'interface DCLI proposée dans le chapitre 4. Les processus sont obtenus par l'agent de mise au point responsable du niveau du protocole dans le plan de contrôle. Pour une couche il y a un seul agent mais plusieurs interfaces, une par protocole à optimiser (Figure 5.4).

5.2.2.2 Plan de communication inter-couches et processus de calcul de métrique

Le plan de communication inter-couches est chargé de communiquer les données nécessaires à une interface, TI. Les données sont élaborées par le processus de calcul de métrique à partir des informations exportées par les protocoles. Le plan obtient la connaissance des informations qu'il doit communiquer et recevoir ainsi que les interfaces origine/destination

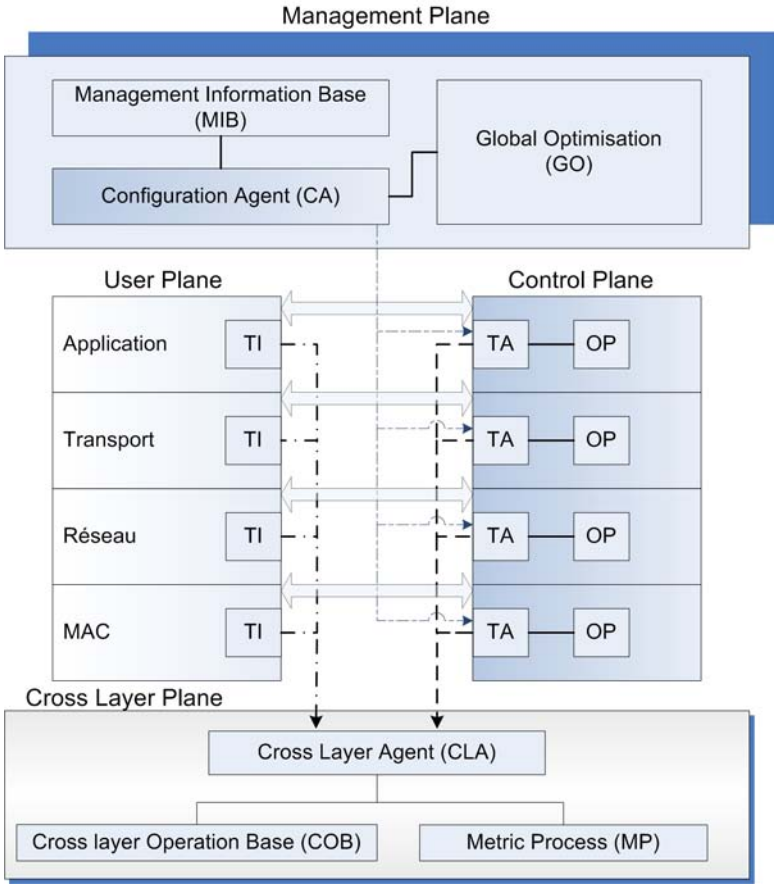


FIG. 5.3 – Architecture d'adaptation

de celles-ci par l'intermédiaire des agents de mise au point contenus dans le plan de contrôle. La communication plan de données, plan de communication inter-couches est constituée d'échanges de données, celle entre le plan inter-couches et le plan de contrôle, de commandes.

Ces commandes sont de type :
- <get, Id data, Id Tuning Interface> pour créer les variables correspondant aux données qui seront reçues ;
- <set, Id métrique, calcul> pour calculer des métriques complexes ;
- <report, Id data / Id métrique, Id Tuning Interface, event> pour envoyer des données sur un événement à une interface.

5.2.2.3 Plan de contrôle, Tuning Agent et Cross layer Operational Base

Le Plan de contrôle contient les règles d'optimisations et les agents de mise au point qui sont en charge de l'adaptation locale. Il interagit avec le Plan de communication inter-couches comme nous venons de le voir ainsi qu'avec le Plan de données et le Plan de gestion.

La communication avec le plan de données a pour but de définir les variables qui doivent être importées et exportées par les interfaces TI, ainsi que les processus d'adaptation. Ceux-ci sont déduits des règles d'optimisation par l'agent de tuning et transmis aux

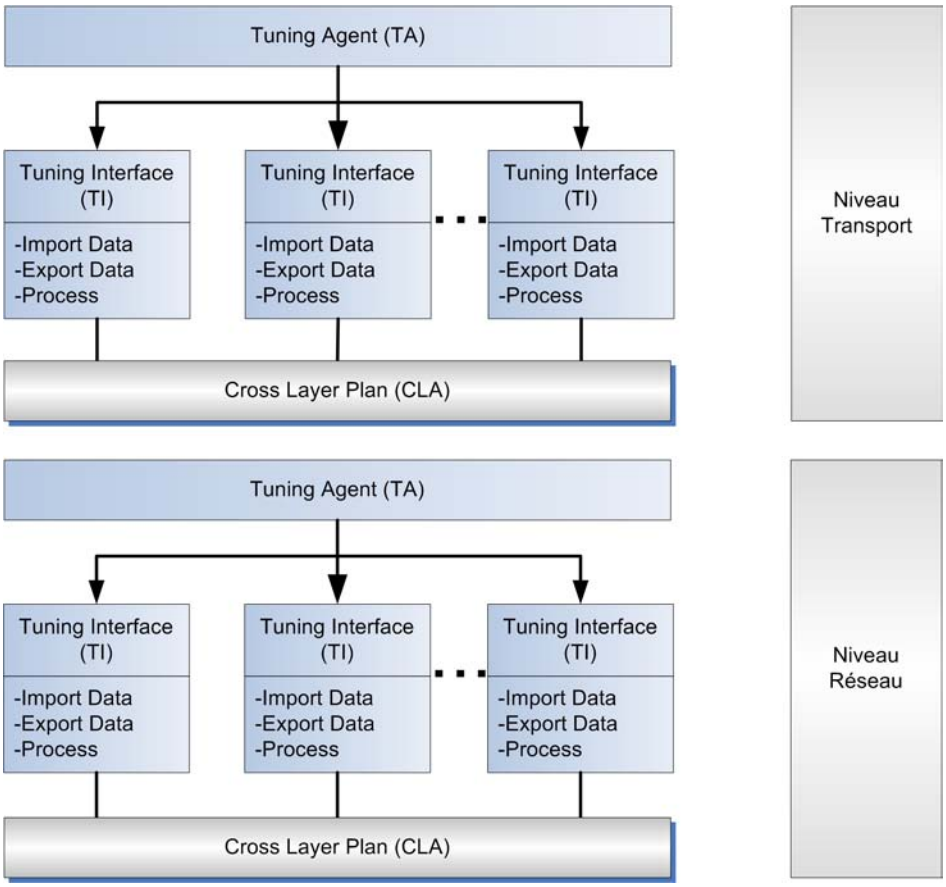


FIG. 5.4 – Tuning Agent

interfaces TI. L'objectif est de pouvoir définir dynamiquement l'adaptation.

Les commandes sont :

- <set, status = active, Id Tuning Interface = TI,i> ;
- <import, Id data> pour créer les données qui seront reçues du plan inter-couches ;
- <export, Id data, event> pour créer les données qui seront transmises au plan inter-couches ;
- <set, process> pour créer dynamiquement un processus d'adaptation interne.

La communication avec le plan de gestion participe à l'adaptation globale. C'est le plan de gestion qui ayant connaissance d'un agent de tuning de niveau N l'active. Il lui indique les optimisations qu'il souhaite activer en fonction de la configuration, par exemple pour une machine multicartes une activation du multihoming transport "SCTP CP".

Les commandes échangées sont de type :

- <set, status, Id Tuning Interface> pour activer, désactiver une interface, l'agent de tuning agit comme mandataire de l'agent de configuration. Ceci permet de faire des adaptations de niveau configuration.

5.2.2.4 Plan de gestion, Configuration Agent et Optimisation Policy

Le plan de gestion est traditionnellement chargé de la configuration des équipements. Il est accessible par toutes les couches du plan de données, via les fonctions classiques des agents SNMP (Simple Network Management Protocol) [HaPW02], et du plan de contrôle via les agents de mise au point. Dans ce modèle il délègue une partie de la configuration au plan de contrôle. L'agent de configuration configure les agents de mise au point qui à leur tour configurent les protocoles. De façon classique, l'agent de configuration utilise l'information de sa base de gestion (la MIB) mais dispose également d'une base d'optimisation (OP : Optimisation Policy). Elle est utilisée à l'initialisation et si nécessaire pour vérifier la cohérence des optimisations locales demandées par les agents de mise au point pour leurs adaptations locales.

Exemple :
Une machine multicartes dispose du protocole SCTP, du protocole SCTP CP du protocole AODV et du protocole AODV optimisé pour le multihoming AODV-MH.

1. L'agent de configuration consulte sa MIB puis sa base d'optimisation :
   - Au niveau transport : si la machine est multicartes et si SCTP CP présent alors activer SCTP CP, Id Tuning Interface = TI.T1 ;
   - Au niveau réseau : si SCTP CP présent et actif alors activer AODV-MH, Id Tuning Interface = TI.R1.
2. L'agent de configuration active l'agent de mise au point transport et l'agent de mise au point réseau en les informant de l'interface qu'ils vont devoir gérer :
   - <set, status = active, Id Tuning Interface = TI.T1> ;
   - <set, status = active, Id Tuning Interface = TI.R1>.
3. L'agent de mise au point transport consulte ses règles pour l'optimisation SCTP CP :
   - Import : route error, degré de similitude (les données inter-couches à utiliser) ;
   - Export : adresse IP Primaire, adresse IP Alternatif, TI.R1 (les données inter-couches qui seront utilisées par le routage) ;
   - Process : tuning interface : si degré de similitude > 4 alors activer SCTP (le chiffre 4 n'est pas significatif mais illustratif).
4. L'agent de mise au point transport configure :
   (a) L'interface de l'optimisation SCTP CP
      - <set, status = active, Id Tuning Interface = TI.T1> ;
      - <import, route error, degré de similitude> ;
      - <export, adresse IP Primaire, adresse IP Alternatif> ;
      - <set, process = degré de similitude > 4 activer SCTP>.

   (b) La communication inter-couches
      - <get, adresse IP Primaire, adresse IP Alternatif, TI.T1> ;
      - <report, adresse IP Primaire, adresse IP Alternatif, TI.R1>.
5. De même l'agent de mise au point réseau consulte ses règles pour l'optimisation AODV-MH :
   - Import : adresse IP Primaire, adresse IP Alternatif (les données inter-couches à utiliser) ;

- Export : route error, degré de similitude, TI.T1 (les données inter-couches qui seront utilisées par le transport) ;
- Process : compute metric : degré de similitude (adresse IP Primaire) = champ degré (route reply adresse IP Alternatif) (la valeur du champ degré contenu dans le message de route reply pour l'établissement de la route sur l'adresse IP du chemin secondaire).

6. L'agent de mise au point réseau configure :

   (a) L'interface de l'optimisation AODV-MH
   - <set, status = active, Id Tuning Interface = TI.R1> ;
   - <export, route error, degré de similitude, TI.T1> ;
   - <set, process = degré de similitude (adresse IP Primaire) = champ degré (route reply adresse IP Alternatif)>.

   (b) La communication inter-couches
   - <get, route error, degré de similitude> ;
   - <report, adresse IP Primaire, adresse IP Alternatif, TI.R1>.

Notons que dans cet exemple, il n'y a pas besoin de calculer de métriques à partir de données de plusieurs couches, le processus de calcul de métriques du plan inter-couches n'est pas utilisé. Nous avons défini le calcul de la métrique au niveau de l'interface de tuning de façon à pouvoir le modifier facilement, mais il peut être directement inclus dans le protocole.

Le modèle d'architecture que nous proposons doit être approfondi par des études ultérieures que nous développons en conclusion de cette thèse.

Conclusion

Dans cette thèse nous avons montré l'intérêt du protocole SCTP adapté par une communication inter-couches, et en particulier, du multihoming, pour les réseaux ad hoc en développant une approche expérimentale dont nous avons synthétisé les résultats dans le chapitre précédent. Notre approche qui repose sur la simulation a mis en oeuvre deux niveaux d'expériences, un premier niveau global utilise un modèle de mobilité couramment utilisé par la communauté de recherche, le Random Waypoint, qui fournit des indications de comportement, le deuxième niveau approfondi les résultats généraux par des scénarios. Avec cette approche nous avons relevé des limites aux propositions d'optimisations qui ne sont généralement pas relevées dans la littérature.

Par nature, une optimisation qui repose sur une adaptation à l'état du réseau a un domaine de validité fonction de cet état. Par ailleurs les réseaux ad hoc ont des ressources limitées qui justifient un besoin d'optimisation, même au prix d'une complexité accrue, aussi proposons nous une architecture protocolaire d'adaptation qui puisse gérer au mieux le fonctionnement de la communication.

Les travaux effectués dans cette thèse pourraient être approfondis par des travaux ultérieurs, en particulier concernant les aspects :

- Routage : les expérimentations ont été menées sur le protocole réactif AODV, il serait intéressant d'examiner comment un protocole proactif comme OLSR, ainsi que la version OLSR avec qualité de services peuvent être intégrés pour optimiser SCTP.
- Multichemins : la gestion de chemins multiples améliore le transfert de données en augmentant la robustesse du réseau. Nous avons traité cette gestion au niveau transport, ce qui offre l'avantage de réutiliser des protocoles standardisés existants. Une perspective de recherche est d'étudier plus avant l'architecture de multihoming en réseaux ad hoc, en prenant en compte un protocole de routage capable de fournir des chemins multiples disjoints.
- Evènements de non congestion : nous avons traité les erreurs de mobilité, les erreurs de niveau liaison détectables au niveau MAC par une interface 802.11 sont également sources d'optimisations.

L'intérêt principal de nos travaux est d'étendre l'utilisation des réseaux ad hoc à des communications nécessitant une grande robustesse. Une application que nous envisageons

se situe dans le cas d'une communication de robots mobiles en réseaux ad hoc. Les robots auraient à transmettre des flux image temps réel, des informations de contrôle critique, et données non critiques par des interfaces WiFi. Pour chaque application les besoins de communication sont exprimés en terme de qualité de service et de robustesse. Le multihoming serait utilisé pour augmenter la robustesse du transfert de données critiques. Nous avons montré l'intérêt d'utiliser des optimisations et comment les mettre en oeuvre, l'application pratique devrait permettre d'étudier plus précisément de nombreux paramètres concernant tant l'optimisation que l'architecture adaptative tels les règles d'optimisations ou encore l'interface d'adaptation, le calcul des métriques.

Acronymes

A

ACK : Acknowlegment
Airmail : Asymmetric reliable mobile access
AODV : Ad hoc On-Demand Distance Vector
ASCONF : Address Configuration Change
ASCONF-ACK : Configuration Acknowledgment
ATCP : Ad hoc Transmission Control Protocol

B

BER : Bit Error Rate

C

CA : Configuration Agent
CATS : Cross-layer Approach to Self-healing
CBR : Constant Bit Rate
CCID : Contol Congestion Identifier
COB : Cross layer Operational Base
CSMA/CA : Carrier Sense Multiple Access with Collision Avoidance
CTS : Clear to send
CWND : Congestion Window
CWR : Congestion Window Reduced

D

DCCP : Datagram Congestion Control Protocol
DCLI : Distributed Cross Layer Interface
DNS : Domain Name System
DSR : Dynamic Source Routing
DUPACKs : Duplication Acknowlegments

E

ECLAIR : Efficient Cross Layer Architecture
ECN : Explicit Congestion Notification
ELFN : Explicit Link Failure Notification
ERDN : Explicit Route Disconnection Notification
ERSN : Explicit Route Successful Notification
ESP : Encapsulating Security Payload

F
FEC : Forward Error Correction
FR : Fast Retransmission

G
GRACE : Global Resource Adaptation through CoopEration

H
HIP : Host Identity Protocol

I
ICMP : Internet Control Message Protocol
IETF : Internet Engineering Task Force
INIT : Initiation
INIT-ACK : Initiation-Acknowlegment
IP : Internet Protocol
I-TCP : Indirect-TCP

L
LLC : Logical Link Control
LQSR : Link Quality Source Routing

M
MAC : Medium Access Control
MANETs : Mobile Ad hoc Networks
MFR : Multiple Fast Retransmit
MIB : Management Information Base
MobileMAN : Mobile Metropolitan Ad hoc Network
MP : Metric Process
mSCTP : Mobile Stream Control Transmission Protocol
M-TCP : Mobile-TCP

N
NHDP : Neighborhood Discovery Protocol
NS : Network Simulator

O
OLSR : Optimized Link State Routing
OOO : Out-of-order
OP : Optimization Policy
OSI : Open Systems Interconnection

P
POEM : Performance-Oriented Model

R
RERR : Route ERRor

RFC : Request For Comments
RFN : Route Failure Notification
RREP : Route REPly
RREQ : Route REQuest
RRN : Route Re-establishment Notification
RTO : Retransmission Timeout
RTS : Request to send
RTTVAR : Round Trip Time Variance
RVS : Rendez Vous Server

S
SACK : Selective Acknowledgement
SCTP : Stream Control Transmission Protocol
SCTP ADDIP : Stream Control Transmission Protocol Dynamic Internet Protocol
SCTP CP : SCTP ChangePath
SHIM6 : Site multihoming by IPv6 intermediation
SI : Stream Identifier
SNMP : Simple Network Management Protocol
SRTT : Smoothed round trip time
SSthresh : Slow Start threshold
SSN : Stream Sequence Number

T
TA : Tuning Agent
TCP : Transmission Control Protocol
TCP Bus : TCP “BUffering capability and Sequence information
TCP DOOR : TCP Detection of Out-Of-Order and Response
TCP-F : TCP-Feedback
TCP-Sack : TCP Selective Acknowlegement
TCPTL : TCP tuning layer
TCP-W : TCP Westwood
TFRC : TCP Friendly Rate Control
TI : Tuning Interface
TO : Timeout
TPSN : TCP Packet Sequence Number
TSN : Transmission Sequence Number
TS : Timestamp
T3-rtx : Timer of Retransmission

U
UDP : User Datagram Protocol
ULID : Upper-layer IDentifiers

W
WIDENS : Wireless Deployable Network System

Bibliographie

[AASS00] A. Ahuja, S. Agarwal, J. Singh and R. Shorey, “Performance of TCP over Different Routing Protocols in Mobile Ad-Hoc Networks”, Vehicular Technology Conference (VTC), Tokyo Japan, May 2000.

[AbBa07] J. Abley and M. Bagnulo, “Applicability Statement for the Level 3 Multihoming Shim Protocol (Shim6)”, Internet-Draft (work in progress), draft-ietf-shim6-applicability-03, 2007.

[ACLL06] H. Aiache, V. Conan, J. Leguay and M. Levy, “XIAN : Cross-Layer Interface for Wireless Ad hoc Networks”, Mediterranean Ad Hoc Networking Workshop (MedHoc), Lipari Italy, June 2006.

[ACLL08] H. Aiache, V. Conan, L. Lebrun, J. Leguay, S. Rousseau and D. Thoumin, “XIAN Automated Management and Nano-Protocol to Design Cross-Layer Metrics for Ad hoc Networking”, The International Federation for Information Processing Networking (IFIP Networking), Singapore, May 2008.

[AlAI02] R. Alamgir, M. Atiquzzaman and W. Ivancic, “Effect of Congestion Control on the Performance of TCP and SCTP over Satellite Networks”, NASA Earth Science Technology Conference, Pasadena California USA, June 2002.

[APLS95] E. Ayanoglu, S. Paul, T. F. LaPorta, K. K. Sabnani and R. D. Gitlin, “AIRMAIL : A link-layer protocol for wireless networks”, ACM Wireless Networks, 1995.

[ArBe06] Arkko, J. and I. Beijnum, “Failure Detection and Locator Pair Exploration Protocol for IPv6 Multihoming”, draft-ietf-shim6-failure-detection-07 (work in progress), December 2006.

[BaBa95] A. Bakre and B. R. Badrinath, “I-TCP : Indirect TCP for Mobile Hosts”, Proceeding 15th International Conference on Distributed Computing Systems (ICDCS), May 1995.

[BaSK95] H. Balakrishnan, S. Seshan, and R. H. Katz, “Improving Reliable Transport and Handoff Performance in Cellular Wireless Networks”, ACM Wireless Networks, 1995.

[BMAP03] H. Badis, A. Munaretto, K. A. Agha and G. Pujolle, “QoS for ad hoc networking based on multiple-metric : Bandwidth and delay”, The 5th IFIP International Workshop On Mobile and Wireless Communications Networks (IFIP MWCN), Singapore, October 2003.

[BMJH98] J. Broch, D. A. Maltz, D.B. Johnson, Yih-Chun Hu and J. Jetcheva, “A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing

Protocols", The ACM International Conference on Mobile Computing and Networking (ACM MobiCom), pp. 85-97, October 1998.

[BoCD06_1] E. Borgia, M. Conti and F. Delmastro, "MobileMAN : Design, Integration and Experimentation of Cross-layer Mobile Multi-hop Ad Hoc Networks", IEEE Communication Magazine, Ad Hoc sensor Network Series, vol. 44, no. 7, July 2006.

[BoCD06_2] E. Borgia, M. Conti and F. Delmastro, "MobileMAN : Integration and Experimentation of legacy Mobile Multihop Ad Hoc Networks", IEEE communications Magazine, July 2006.

[BoCo04] A. Boukalov and V. Conan, "Wireless Deployable Network System for Future Public Safety", WIDENS project, "http ://www.comlab.hut.fi/projects/WIDENS/".

[BPSK97] H. Balakrishnan, V. N. Padmanabhan, S. Seshan and R. H. Katz, "A comparison of Mechanisms for Improving TCP Performance over Wireless Links", IEEE/ACM Transactions on Networking (TON), vol. 5, no. 6, pp. 756-769, December 1997.

[Bris06] B. Briscoe, "Flow Rate Fairness : Dismantling a Religion", British Telecommunication Research, October 2006.

[BrSi97] K. Brown and S. Singh, "M-TCP : TCP for Mobile Cellular Networks", ACM Computer Communication Review, vol. 27, no. 5, 1997.

[CaAS06] A. Caro, P. Amer and R. Stewart, "Retransmission policies for multihomed transport protocols", Computer Communications, vol. 29, no. 10, 2006.

[CaBD02] T. Camp, J. Boleng and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research", Wireless Communication and Mobile Computing (WCMC), vol. 2, no. 5, pp. 483-502, 2002.

[ChPa07] S. Charoenpanyasak and B. Paillassa, "SCTP multihoming with Cross Layer Interface in ad hoc Multihomed Networks", IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), New York USA, Octobre 2007.

[ChPJ07] S. Charoenpanyasak, B. Paillassa and F. Jaddi, "Experimental Study on TCP Enhancement Interest in Ad Hoc Networks", International Conference on Wireless and Mobile Communications (ICWMC), Guadeloupe, March 2007.

[ClDD08] T. Clausen, C. Dearlove and J. Dean, "MANET Neighborhood Discovery Protocol (NHDP)", Internet-Draft : draft-ietf-manet-nhdp-06, Expires : September 2008.

[CMTG04] M. Conti, G. Maselli, G. Turi and S. Giordano, "Cross layering in mobile Ad hoc network design", IEEE Computer Society, pp. 48-51, February 2004.

[CoMa99] S. Corson and J. Macker, "Mobile Ad hoc Networking (MANET) : Routing Protocol Performance Issues and Evaluation Considerations", RFC 2501, IETF, January 1999.

[CRVP98] K. Chandran, S. Ragbunathan, S. Venkatesan and R.Prakash, "A feedback based scheme for improving TCP performance in ad-hoc wireless networks", Proceedings 18th International Conference on Distributed Computing Systems, May 1998.

[CSIA03] A. L. Caro, K. Shah, J. R. Iyengar, P. D. Amer and R. R. Stewart, “SCTP and TCP Variants : Congestion Control Under Multiple Losses”, Tech Report TR2003-04, CIS Dept, University of Delaware, February 2003.

[DBEB06] M. Duke, R. Braden, W. Eddy and E. Blanton, “A Roadmap for Transmission Control Protocol (TCP) Specification Documents”, RFC 4614, IETF, September 2006.

[DyBo01] T. Dyer and R. Boppana, “A Comparison of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks”, The ACM International Symposium on Mobile Ad Hoc Networking and Computing (ACM MOBIHoc), Long Beach California USA, pp. 56-66, October 2001.

[Elaa02] H. Elaarag, “Improving TCP performance over mobile networks”, ACM Computing Surveys, vol. 34, no. 3, September 2002.

[FaFl96] K. Fall and S. Floyd, “Simulation-based comparisons of Tahoe, Reno, and SACK TCP”, ACM Computer Communications Review, vol. 26, no. 3, pp. 5-21, July 1996.

[FlHG04] S. Floyd, T. Henderson and A. Gurtov, “The NewReno Modification to TCP’s Fast Recovery Algorithm”, RFC 3782, IETF, April 2004.

[FlHK06] S. Floyd, M. Handley and E. Kohler, “Problem Statement for the Datagram Congestion Control Protocol (DCCP)”, RFC 4336, IETF, March 2006.

[FlKo06] S. Floyd and E. Kohler, “Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 2 : TCP-like Congestion Control”, RFC 4341, IETF, March 2006.

[FlKP06] S. Floyd, E. Kohler and J. Padhye, “Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3 : TCP-Friendly Rate Control (TFRC)”, RFC 4342, IETF, March 2006.

[FMMP00] S. Floyd, J.Mahdavi, M. Mathis and M. Podolsky, “An Extension to the Selective Acknowledgement (SACK) Option for TCP”, RFC 2883, IETF, July 2000.

[GBZT99] M.Gerla, R. Bagrodia, L. Zhang, K. Tang and L. Wang, “TCP over Wireless Multi-hop Protocols : Simulation and Experiments”, Proceedings of IEEE International Conference on Communication (ICC), 1999.

[GFTW05] X. Gu, X. Fu, H. Tshofenig and L. Wolf, “Towards self-optimizing protocol stack for autonomic communication : initial experience”, Proceedings of the 2nd IFIP International Workshop on Autonomic Communication, Springer Lecture Notes in Computer Science, vol. 3854, pp. 183-201, October 2005. “http ://www.ibr.cs.tu-bs.de/projects/poem/”.

[GMPG00] T. Goff, J. Moronski, D.S. Patzk and V. Guptal, “Freeze-TCP : A true end to end TCP enhancement mechanism for mobile environments”, The nineteenth annual joint conference of the IEEE Computer and Communications Societies (IEEE INFOCOM), Telaviv Israel, March 2000.

[GrMa04] L. A. Grieco and S. Mascolo, “Performance evaluation and comparison of Westwood+, New Reno and Vegas TCP congestion control”, ACM Computer Communication Review, April 2004.

[GuWo07] X. Gu and L. Wolf, “To Layer or Not To Layer : Architectural Considerations on Autonomic Communications”, International Journal of Inter-

net Protocol Technology (IJIPT), vol. 2, no. 1, pp. 66-76, January 2007. “http ://www.ibr.cs.tu-bs.de/projects/poem/”.

[HaPW02] D. Harrington, R. Presuhn and B. Wijnen, “An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks”, RFC 3411, IETF, December 2002.

[HLCH07] C. J. Huang, W. K. Lai, Y. T. Chuang and S. Y. Hsiao, “A dynamic Alternate Path QoS Enabled Routing Scheme in Mobile Ad hoc”, International Journal of Wireless Information Networks, Springer Netherlands, vol. 14, no. 1, March 2007.

[HoVa99] G. Holland and N. Vaidya, “Analysis of TCP Performance over Mobile Ad Hoc Networks”, The ACM International Conference on Mobile Computing and Networking (ACM MobiCOM), pp. 219-230, August 1999.

[HoVa02] G. Holland and N. Vaidya, “Analysis of TCP Performance over Mobile Ad Hoc Networks”, Kluwer Academic Publishers, pp. 275-288, 2002.

[HuJo00] Y. C. Yu and D. Johnson, “Caching strategies in on-demand routing protocols for wireless ad hoc networks”, Proceedings of the 6th Annual IEEE/ACM International Conference on Mobile Computing and Networking (MobiCom), Boston Massachusetts, pp. 231-242, August 2000.

[IyAS04] J. Iyengar, P. Amer and R. Stewart, “Retransmission policies for concurrent multipath transfer using SCTP multihoming”, Proceedings of the 12th International Conference on Networks (ICON), Singapore, November 2004.

[IyAS06] J. Iyengar, P. Amer and R. Stewart, “Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths”, The IEEE/ACM Transactions on Networking, vol. 14, no. 5, October 2006.

[JaCH84] R. Jain, D.M Chiu and W. Hawe, “A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems”, DEC Research Report TR-301, 1984.

[Jaco88] V. Jacobson, “Congestion avoidance and Control”, The symposium on Communications Architectures and Protocols (SIGCOMM), pp. 314-329, 1988.

[Jaco90] V. Jacobson, “Berkeley TCP Evolution from 4.3 Tahoe to 4.3 Reno”, Proceedings of the 18th Internet Engineering Task Force, University of British Colombia, Vancouver BC, September 1990.

[JiJa01] M. Jiang and R. Jan, “An Efficient Multiple Paths Routing Protocol for Ad-Hoc Networks”, The 27th Annual Conference of the IEEE of Industrial Electronics Society (IEEE ICON), Februry 2001.

[JoHM07] D. Johnson, Y. Hu and D.Maltz, “The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4”, RFC 4728, IETF, February 2007.

[KaKu05] V. Kawadia and P. R. Kumar, “A cautionary perspective on cross-layer design”, IEEE Wireless Communications, pp. 3-11, February 2005.

[KBSS07] M. Komu, M. Bagnulo, K. Slavov and S. Sugimoto, “Socket Application Program Interface (API) for Multihoming Shim”, Internet-Draft (work in progress), draft-ietf-shim6-multihome-shim-api-03, July 2007.

[Kim07] K. I. Kim, “A Cross Layered Approach for Multihoming on SCTP in mobile Ad Hoc Networks”, TENCON (Taipei International Convention Center), IEEE Region 10 Conference, October 2007.

[KiTC01] D. Kim, C.K Toh and Y. Choi, “TCP-Bus : Improving TCP Performance in Wireless Ad-Hoc Networks”, Journal of communications and networks, vol. 3, no. 2, June 2001.

[KNBA04] R. Knopp, N. Nikaein, C. Bonnet, H. Aiache, V. Conan and S. Masson, “Overview of the WIDENS Architecture, A Wireless Ad Hoc Network for Public Safety”, The 1st IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON), Santa Clara USA, October 2004.

[KoCL04] S. J. Koh, M. J. Chang and M. Lee, “mSCTP for Soft Handover in Transport Layer”, IEEE communications letters, vol. 8, no. 3, March 2004.

[KoHF06] E. Kohler, M. Handley and S. Floyd. “Datagram Congestion Control Protocol (DCCP)”, RFC 4340, IETF, March 2006.

[Komu08] M. Komu, “Basic Socket Interface Extensions for Host Identity Protocol (HIP)”, Internet-Draft, draft-ietf-hip-native-api-04, Expires : August 2008.

[Kood05] R. Koodli, “Fast Handovers for Mobile IPv6”, RFC 4068, IETF, July 2005.

[KuJa04] A. Kumar and L. Jacob, “SCTP vs TCP : Performance Comparison in MANETs”, Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN), 2004.

[LaHL07] W. K. Lai, S. Y. Hsiao and Y. C. Lin, “Adaptive backup routing for ad-hoc networks”, Computer Communications 30, January 2007.

[LDPJ04] L. A. Larzon, M. Degermark, S. Pink, L. E. Jonsson and G. Fairhurst, “The Lightweight User Datagram Protocol (UDP-Lite)”, RFC 3828, IETF, July 2004.

[LeGe00] S. J. Lee and M. Gerla, “AODV-BR Backup Routing in Ad hoc Networks”, Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), September 2000.

[LeGe01] S.J. Lee and M. Gerla, “Spilt mutipath routing with maximally disjoint paths in ad hoc networks”, Proceedings of IEEE International Conference Communications (ICC), June 2001.

[LiSi01] J. Liu and S. Singh, “ATCP : TCP for mobile ad hoc networks”, IEEE Journal Selected Areas in Communications, vol. 19, pp. 1300-1315, July 2001.

[LiXG03] H. Lim, K. Xu and M. Gerla, “TCP performance over multipath routing in mobile ad hoc networks”, Proceedings of IEEE International Conference Communications (ICC), May 2003.

[LoDP07] S. Lohier, Y. G. Doudane and G. Pujolle, “Cross-Layer Loss Differentiation Algorithms to improve TCP Performance in WLANs”, Telecommunication Systems (Springer Verlag), DOI : 10.1007/s11235-007-9054-0, pp. 61-72, November 2007.

[MaMu04] B.S. Manoj and C. Siva Ram Murthy, “Ad Hoc Wireless Networks : Architectures and Protocols”, Prentice Hall PTR, May 2004.

[MCGL00] S. Mascolo, C. Casetti, M.Gerla, S.S. Lee and M. Sanadidi, “TCP Westwood : congestion control with faster recovery”, The 21st Century Military Communications Conference (MILCOM), Los Angeles California USA, October 2000.

[MeGo05] D. E. Meddour and Y. Gourhant, “Routing Based Feedback Towards Applications in Mobile Ad Hoc Networks”, Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS), Big Island Hawaii, January 2005.

[MGFC04] S. Mascolo, L. A. Grieco, R. Ferorelli, P. Camarda and G. piscitelli, “Performance evaluation of Westwood+ TCP congestion control”, Internet Performance Symposium (IPS), Elsevier, North-Holland, vol. 55, pp. 93-101, January 2004.

[Micr05] Microsoft Version of DSR with Link Quality Metrics 2005, http ://research.microsoft.com/mesh/.

[MoNi06] R. Moskowitz and P. Nikander, “Host Identity Protocol (HIP) Architecture”, RFC 4423, IETF, May 2006.

[MtKa06] A. Mtibaa and F. Kamoun, “MMDV : Multipath and MPR based AODV routing protocol”, The IFIP Fifth Annual Mediterranean Ad Hoc Networking Workshop (MedHoc), June 2006.

[MuFo07] A. Munaretto and M. Fonseca, “Routing and quality of service support for mobile ad hoc networks”, The International Journal of Computer and Telecommunications Networking, Elsevier North-Holland, New York USA, August 2007.

[NaCD01] A. Nasipuri, R. Castaneda, and S. R. Das, “Performance of Multipath Routing for On-Demand Protocols in Mobile Ad Hoc Networks”, Mobile Networks and Applications, pp. 339-349, August 2001.

[Nika07] P. Nikander, “Generic Proxying as a Deployment Tool (GEPROD)”, Internet-Draft, draft-nikander-ram-generix-proxying-00, Expires, July 2007.

[NJMS01] A. Natani, J.Jakilnki, M. Mohsin and V. Sharma, “TCP for Wireless Networks”, Technical report university of texas at dallas, 2001. “http ://www.utdallas.edu/ sudha/WirelessTCP.pdf”.

[NoBa07] E. Nordmark and M. Bagnulo, “Shim6 : Level 3 Multihoming Shim Protocol for IPv6 Internet-Draft”, (work in progress), draft-ietf-shim6-proto-08, 2007.

[NS] NS - Le simulateur (ns-2) : http ://www.isi.edu/nsnam/ns/.

[PaHN04] M. Park, R. W. Heath and S. M. Nettles, “Improving Throughput and Fairness for MIMO Ad Hoc Networks Using Antenna Selection Diversity”, IEEE Communications Society, Globecom, 2004.

[PeBD03] C. Perkins, E. Belding-Royer and S. Das, “Ad hoc on-demand distance vector (AODV) routing”, RFC 3561, IETF, July 2003.

[PeCB07] C. Perkins, P. Calhoun and J. Bharatia, “Mobile IPv4 Challenge/Response Extensions (Revised)”, RFC 4721, IETF, January 2007.

[Pent00] K. Pentikousis, “TCP in wired-cum-wireless environments”, IEEE Communications Surveys Tutorials, vol. 3, no. 4, pp. 2-14, 2000.

[PeRD01] C. Perkins, E. Royer, S. Das and M. Marina, “Performance Comparison of Two On-Demand Routing Protocols for Ad Hoc Networks”, IEEE Personal Communications, February 2001.

[Perk02] C. Perkins, “IP Mobility Support for IPv4”, RFC 3344, IETF, August 2002.

[Post80] J. Postel, “User Datagram Protocol”, RFC 768, IETF, August 1980.

[Post81] J. Postel, “Transmission Control Protocol”, RFC 793, IETF, September 1981.

[RaDN07] M.A. Razzaque, S. Dobson and P. Nixon, “Cross-layer architectures for autonomic communications”, Journal of Network and Systems Management, pp. 13-27. March 2007.

[RaFB01] K. Ramakrishnan, S. Floyd, and D. Black, “The addition of explicit congestion notification (ECN) to IP”, RFC 3168, IETF, September 2001.

[RaFl99] K. Ramakrishnan and S. Floyd, “A Proposal to add Explicit Congestion Notification (ECN) to IP”, RFC 2481, January 1999.

[RaIy04] V. T. Raisinghani and S. Iyer, “ECLAIR : An effécient cross layer architecture for wireless protocol stacks”, The 5th World Wireless Congress (WWWC), San Francisco USA, May 2004.

[RaIy06] V.T. Raisinghani and S. Iyer, “Cross Layer Feedback Architecture for Mobile Device Protocol Stacks”, IEEE Communications Magazine, Special Issue of Cross-Layer Protocol Engineering, January 2006.

[RiTu07] M. Riegel and M. Tuexen, “Mobile SCTP”, IETF Internet Draft, draft-riegel-tuexen-mobile-sctp-09.txt, November 2007.

[Sach06] D. G. Sachs, “A New Framework for Hierarchical Cross-layer Adaptation” , Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2006.

[SaKa04] Y. Sakurai and J. Katto, “AODV Multipath Extension using Source Route Lists with Optimized Route Establishment”, International workshop on wireless ad-hoc networks (IWWAN), May-June 2004.

[SaKC05] C. M. Sadler, L. Kant and W. Chen, “Cross-Layer Self-Healing Mechanisms in Wireless Networks”, World Wireless Congress (WWC), May 2005.

[SAPC06] R. Stewart, I. Aria-Rodriguez, K. Poon, A. Caro and M. Tuexen, “Stream Control Transmission Protocol (SCTP) Specification Errata and Issues”, RFC 4460, IETF, April 2006.

[SrMo05] V. Srivastava and M. Motani, “Cross-layer design : a survey and the road ahead”, IEEE Communications Magazine, vol. 43, no. 12, pp. 112-119, December 2005.

[SSIM02] F. Sultan, K. Srinivasan, D. Iyer, and L. I. Migratory, “TCP : Highly available Internet services using connection migration”, IEEE International Conference on Distributed Computing Systems (ICDCS), Vienna Austria, 2002.

[StAP99] W. Stevens, M. Allman, and V. Paxson, “TCP Congestion Control”, RFC 2581, IETF, April 1999.

[Stew07] R. Stewart Ed, “Stream Control Transmission Protocol”, RFC 4960, IETF, September 2007.

[SuHu05] H. Sun and H. D. Hughes, “Adaptive QoS routing by cross-layer cooperation in ad hoc networks”, EURASIP Journal on Wireless Communications and Networking, vol. 5, pp. 661-671, October 2005.

[SXMS00] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang and V. Paxson, “Stream Control Transmission Protocol”, RFC 2960, IETF, October 2000.

[SXTM07] R. Stewart, Q. Xie, M. Tuexen, S. Maruyama and M. Kozuka, “Stream Control Transmission Protocol (SCTP) Dynamic Addres Reconfiguration”, RFC 5061, IETF, September 2007.

[SYHH03] D. G. Sachs, W. Yuan, C. J. Hughes, A. Harris, S. V. Adve, D. L. Jones, R. H. Kravets, and K. Nahrstedt, “GRACE : A hierarchical adaptation framework for saving energy”, IEEE Computer, Special issue on Power-Aware Computing, December 2003.

[SYQH04] S. Sesay, Z. Yang, B. Qi and J. He, “Simulation Comparison of Four Wireless Ad hoc Routing Protocols”, Information Technology Journal 3, 2004.

[SYZG05] E. Setton, T. Yoo, X. Zhu, A. Goldsmith and B. Girod, “Cross-layer Design of Ad Hoc Networks for Real-Time Video Streaming”, IEEE Wireless Communications Magazine, vol. 12, no. 4, pp. 59-65, August 2005.

[VMPM99] N. Vaidya, M. Mehta, C. Perkins and G. Montenegro, “Delayed Duplicate Acknowledgements : A TCP-Unaware Approach to Improve Performance of TCP over Wireless”, Technical Report, Computer Science Dept, Texas A&M University, 1999.

[WaZh02] F. Wang and Y. Zhang, “Improving TCP performance over mobile ad hoc networks with out-of-order detection and response”, The ACM International Symposium on Mobile Ad Hoc Networking and Computing (ACM MOBIHOC), Lausanne Switzerland, pp. 217-225, June 2002.

[WSNB05] R. Winter, S. Schiller, N. Nikaein, and C. Bonnet, “CrossTalk : A Data Dissemination-based Cross-layer Architecture for Mobile Ad-hoc Networks”, IEEE Workshop on Applications and Services in Wireless Networks (ASWN) Paris, June 2005.

[WVSG02] R. Wang, M. Valla, M.Y. Sanadidi and M. Gerla, “Adaptive Bandwith Share Estimation in TCP Westwood”, Proceedings of IEEE Global Telecommunications Conference (Globecom), 2002.

[YaPD08] C. Yawut, B. Paillassa and R. Dhaou, “Mobility Metrics Evaluation for Self-Adaptive Protocols”, Journal of Networks Academy Publisher, Finland, Issue 1, ISSN : 1796-2056, vol. 3, pp. 53-64, January 2008.

[YNAJ03] W. Yuan, K. Nahrstedt, S. Adve, D. Jones and R. Kravets, “Design and Evaluation of A Cross-Layer Adaptation Framework for Mobile Multimedia Systems”, Proceedings of the SPIE/ACM Multimedia Computing and Networking Conference (MMCN), Santa Clara - California USA, January 2003.

[ZPGS01] A. Zanella, G. Procissi, M. Gerla and M.Y. Sanadidi, “TCP Westwood : Analytic Model and Performance Evaluation”, Proceedings of IEEE Global Telecommunications Conference (Globecom), pp. 1703-1707, 2001.

RESUME :

Les réseaux ad hoc utilisent une transmission sans fil directe entre les équipements sans passer par des stations de base : ils n'ont pas d'architecture. La transmission entre les noeuds adjacents est gérée par un protocole d'accès et un protocole de routage lorsque la source doit utiliser des relais pour joindre la destination. Les protocoles d'accès et de routage ont fait l'objet de nombreuses recherches qui ont abouti à la standardisation de solutions spécifiques. Cependant l'utilisation de ces solutions ne résout pas les problèmes de performances au niveau des protocoles supérieurs. Les réseaux ad hoc, outre la transmission sans fil et la capacité de routage, sont également caractérisés par la mobilité de leurs éléments. Les protocoles de transport qui sont chargés de fiabiliser la communication sur des réseaux à pertes sont affectés par ce type de comportement. Alors qu'ils ont été conçus pour s'adapter aux problèmes de congestion dans les réseaux filaires, il s'agit à présent pour eux de s'adapter à la mobilité en utilisant au mieux les ressources limitées du système.

L'adaptation du transport par une communication inter-couches est l'approche utilisée dans cette thèse, elle évite de consommer de la bande passante par une signalisation supplémentaire tout en réutilisant un protocole standard. Nous étudions les possibilités d'améliorer les performances du protocole de transport SCTP. Très semblable au protocole TCP il définit un fonctionnement multihoming qui en environnement filaire améliore les performances du transfert. Nous montrons que cette option est également intéressante en réseaux ad hoc.

Dans ce document nous montrons l'importance du protocole de routage sur les performances du transport, puis considérons l'adaptation par interactions entre les niveaux réseau et transport. Nous définissons deux modes d'adaptation : un mode local et un mode réparti. Dans un premier temps nous proposons une adaptation locale du transport selon l'état du réseau perçu par le protocole de routage. Cette adaptation optimise le protocole en améliorant ses performances en terme de nombre de paquets transmis, sans avoir besoin de modifier celui-ci, mais en conditionnant l'activation du processus de congestion à la détection des ruptures de routes. Nous avons implanté ce fonctionnement sur un simulateur NS2 et examiné le domaine de validité de l'optimisation. Dans un second temps, nous proposons une optimisation inverse où il s'agit d'adapter le routage au fonctionnement du protocole transport, de façon à pouvoir bénéficier du traitement multihoming de SCTP. Grâce à un mode d'adaptation réparti, le protocole de routage prend en compte la topologie des flux de transmissions pour accélérer sa reprise de route en cas de rupture. Pour étudier le domaine de validité de l'optimisation nous introduisons la métrique de degré de similitude qui reflète le nombre d'éléments communs entre deux routes. Ce travail se conclut par la proposition d'une architecture d'adaptation permettant d'utiliser selon leurs domaines de validité les optimisations proposées.

MOTS CLES : SCTP, multihoming, inter-couches, réseaux ad hoc, cross layer.