وزارة التعليم العالي و البحث العلمي

BADJI MOKHTAR UNIVERSITY- ANNABA
UNIVERSITE BADJI MOKHTAR- ANNABA

جامعة باجي مختار – عنابة

FACULTE DES SCIENCES DE L'INGENIEUR
DEPARTEMENT D'INFORMATIQUE

Année 2009/2010

Mémoire

PRESENTE EN VUE DE L'OBTENTION DU DIPLOME DE MAGISTER

Etude et implémentation d'algorithme de mobilité de groupe et application au P-learnings

OPTION
TEXTE, PAROLE ET IMAGES

PAR
TALAI Meriem

Directeur de Mémoire M^me^ Hassina Seridi MCA Université Badji Mokhtar- Annaba

DEVANT LE JURY

Président	M^r^ Djamel Meslati	MCA	Université Badji Mokhtar- Annaba
Examinateurs	M^r^ Tarek Khadir	MCA	Université Badji Mokhtar- Annaba
	M^me^ Habiba Bellili	MCA	Université Badji Mokhtar- Annaba

Remerciements

Ce travail a été réalisé sous la direction de Mme Hassina Séridi, Maître de conférences à l'université Badji Mokhtar de Annaba et de Mr Mounir Boussedjra, Maître de conférence à IRSEEM-ESIGELEC de Saint Etienne du Rouvray-France. Je tiens à les remercier vivement pour leurs conseils, leur aide, leur disponibilité ainsi que leur soutien tout au long de ce travail de recherche. Qu'ils trouvent ici, ma sincère reconnaissance et ma profonde gratitude.

Je souhaite adresser mes sincères remerciements à Mr Djamel Meslati, maître de conférences à l'université de Badji Mokhtar de Annaba, pour m'avoir fait l'honneur de présider le jury de ma soutenance.

Je tiens aussi à remercier particulièrement les personnes qui ont accepté d'examiner et de juger mon travail et qui sont : Mme Habiba Bellili et Mr Khadir Tarek, maîtres de conférences à l'université Badji Mokhtar de Annaba.

Je remercie également tous les enseignants et responsables du département d'informatique de l'université d'Annaba.

Enfin, je garde une place toute particulière pour ma famille et mes amis proches, pour tout ce qu'ils ont pu (et continuent à) m'apporter tout au long de ces années.

A mes chers parents

A mon petit Islem

ملخـــــــص

من خلال هذا العمل، فإننا نقترح هيكلة إرسال قائمة على جودة الخدمة عبر الشبكات المعلوماتية المتنقلة. نعرض عملية الجدولة الخاصة بـ "بروتوكول الإرسال المحمول بإدارة حسب التيار" إلى تعديلات لتحقيق أفضل إرسال لمختلف أنواع المعلومات على أجهزة التوجيه المتنقلة. الهيكل المقترح يرفق فئة للمعلومات و عتبة خاصة إلى كل تيار معلومات من جلسات البروتوكول المذكور. هذه العتبات تتحدد وفقا لقيود فئة المعلومات من حيث وقت الإرسال. المجدول يرسل المعلومات على تيارات البروتوكول بمراعاة هذه العتبات. اختبار وتقييم الاقتراح يبين أنه يدعم إرسال الملتيميديا حيث أنه يفي بقيود أزمنة الإرسال لهذه الفئة من المعلومات. علاوة على ذلك، فإن المجدول المقترح يحقق نتائج أفضل من اقتراح مسبق حيث أنه يتجنب حرمان باقي فئات المعلومات من الإرسال على تيارات المعلومات للبروتوكول.

الكلمات المفتاح : حركة الشبكات، إدارة التنقل، بروتوكول الإرسال المحمول بإدارة حسب التيار، جودة الخدمات، جدولة.

Abstract

In this work, we propose a QoS based architecture for mobile networks transmissions. The scheduling process of mobile Stream Control Transmission Protocol (mSCTP) is modified in order to optimize transmission several types of data incoming to mobile routers. The proposed architecture attributes both a data class and a specific weight to each stream of an mSCTP association. These weights are determined according to data class constraints in term of transmission time. The scheduler performs transmissions through streams with respects of the defined weights. Tests and evaluation of the proposition show that it supports multimedia traffic since it satisfies transmission time constraints of this data class. Furthermore, the proposed scheduler provides better results than the one proposed in an earlier work since it avoids starvation of the other data type on SCTP streams.

Key words: Network Mobility, Mobility Management, mSCTP, QoS, Scheduling

Résumé

A travers le présent travail, nous proposons une architecture basée QdS pour les transmissions sur les réseaux mobiles. Le processus d'ordonnancement pour le protocole « mobile Stream Control Transmission Protocol- mSCTP » est modifié afin d'optimiser la transmission de différents types de données sur les routeurs mobiles. L'architecture proposée attribue une classe de données et un seuil spécifique à chaque flux d'une association mSCTP. Ces seuils sont déterminés selon les contraintes des classes de données respectives en termes de temps de transmission. L'ordonnanceur effectue les transmissions via les streams en respect aux valeurs des seuils. Les tests et évaluation de la proposition montrent qu'elle supporte le trafic multimédia vu qu'il satisfait les contraintes de temps de transmission de cette classe de données. De plus, l'ordonnanceur proposé fournit de meilleurs résultats qu'une proposition antérieure vu qu'il évite l'affaméité des autres types de données sur les streams de SCTP.

Mots clés : Mobilité Groupée, Gestion de la Mobilité, QdS, mSCTP, Ordonnancement

Table des Matières

Tables des illustrations

Table des figures

Table des tables et listing

Introduction Générale

Les réseaux informatiques sans fil (Wireless Networks) sont des systèmes de communication qui offrent aux utilisateurs une liberté de mouvement en les dispensant de câblage. Ils comprennent des unités mobiles capables d'effectuer des transmissions via les ondes radio sur une étendu équivalente à leur rayon de propagation.

Ces communications radio sont généralement établies sur des réseaux sans fils avec infrastructures qui déploient des équipements mobiles aussi bien que des sites fixes. Ces derniers forment un réseau filaire câblé alimenté par des sources d'énergie illimitée. Les réseaux 802.11, les réseaux UMTS (Universal Mobile Telecommunications System) sont des variantes de réseaux sans fils avec infrastructure.

Néanmoins, les réseaux Ad Hoc ou sans infrastructure présentent une autre alternative de réseaux informatiques sans fil. Un réseau Ad Hoc est définit selon l'IETF-Internet Engineering Task Force, dans le RFC-Request For Comment 2501, comme suit : «Un réseau Ad Hoc comprend des plates-formes mobiles appelées nœuds, libres de se déplacer sans contraintes et utilisant le médium radio pour le transfert d'informations ». Un réseau Ad Hoc est par conséquent un système autonome de nœuds mobiles. Ce système peut fonctionner de manière isolée ou s'interfacer à des réseaux fixes via des passerelles. Dans ce dernier cas, un réseau Ad Hoc est un réseau d'extrémité.

Motivations

Les réseaux Ad Hoc suscitent l'intérêt des recherches vu qu'ils offrent des apports dans différents domaines d'application. En effet, ils permettent l'extension de réseaux filaires avec moindre coût et présentent une solution pour la mise en œuvre de l'informatique embarquée et pour le déploiement rapide des services d'urgences. D'autre part, les réseaux Ad Hoc possèdent des propriétés qui diffèrent des autres réseaux sans fils.

Parmi ces propriétés, la gestion de la mobilité des nœuds est considérée par la communauté scientifique comme un défi majeur pour lequel il faut définir des solutions efficaces et pertinentes. En effet, le caractère dynamique des réseaux Ad Hoc n'est pas soumis à une gestion centralisée et engendre des problèmes de connectivité, tel que la rupture des transmissions en cours lors des déplacements entre les zones de couvertures et les problèmes de handover. De plus, la mobilité des nœuds sur les réseaux Ad Hoc reflète une forme de mobilité dite mobilité groupée. Les nœuds mobiles d'un réseau Ad Hoc peuvent, dans plusieurs cas, constituer un ensemble d'équipements communiquant mobiles. Dans ce cas on parle d'un réseau mobile ou Mobile Ad Hoc Network-MANET

Des solutions sont aujourd'hui standardisées pour assurer la continuité des sessions et minimiser l'impact des handover. Parmi ces solutions, nous citons les protocoles Mobile Internet Protocol-MIP et Network Mobility Basic Support Protocol-NEMO BSP qui opèrent au niveau 3 du modèle TCP/IP, Session Initiation Protocol- SIP [35] au niveau session et Mobile Stream Control Transmission Protocol-mSCTP [25] ainsi que Datagram Congestion Protocol-DCCP [15] qui opèrent sur la couche transport. La gestion de la mobilité est ainsi déployée sur différents niveaux du modèle TCP/IP. Cependant, diverses études encouragent le niveau transport pour effectuer cette gestion [8], [9].

Objectifs et Contribution

Parmi les protocoles de mobilité au niveau transport, nous nous intéressons au protocole mSCTP vu les dispositifs qu'il offre. mSCTP est un nouveau protocole (1er RFC N° 5061 en 2007) étendu du protocole filaire SCTP. mSCTP implémente deux nouveaux mécanismes, le *multihoming* et le *multisteaming* qui le rendent très fiable pour la gestion de la mobilité. Le mécanisme de *multihoming* hérité du protocole SCTP permet à un nœud mobile d'appartenir à plusieurs réseaux et de posséder plusieurs adresses IP, en même temps. Le mécanisme *multisteaming* permet, quant à lui, la séparation des flux de données au niveau transport et une transmission indépendante de ces flux. Il peut répondre ainsi aux besoins spécifiques des différents types de données.

Malgré les avantages indiqués de mSCTP, ce protocole présente quelques lacunes qui réduisent son rendement pour le transfert des données multimédia. Entre autres, il ne propose pas de mécanisme spécifique dédié à la gestion de la qualité de service lors des transmissions à partir des multiples streams. Cette déficience de mSCTP est particulièrement nuisible dans le contexte de la mobilité groupée des réseaux.

Notre étude vise à améliorer cette défaillance de mSCTP sans compromettre ses fonctionnalités de base. Dans nos travaux, nous proposons de nouvelles solutions permettant d'adapter l'ordonnancement des transmissions au type de données à transmettre et d'améliorer les performances de mSCTP. Notre contribution intègre des fonctionnalités intelligentes à la procédure de transmission grâce auxquelles, elle devrait satisfaire les contraintes des données les plus prioritaires en termes de délai de transmission sans négliger les données de moindre priorité.

Organisation du document

Le présent manuscrit est organisé en quatre chapitres. Dans le premier chapitre nous introduisons des concepts liés à la mobilité en spécifiant les principaux éléments qui lui sont associés. Dans le deuxième chapitre nous analysons les solutions protocolaires proposées pour gérer la mobilité dans les réseaux sans fil, avec et sans infrastructure. Nous consacrons le troisième chapitre à l'explication de notre contribution. Nous y argumentons le choix du protocole étudié. Nous présentons les lacunes décelées de ce protocole ainsi que l'algorithme d'ordonnancement proposé. Les résultats obtenus par simulation de la solution proposée sous la plate forme NS2 sont présentés et analysés dans le quatrième chapitre. Nous concluons notre étude en spécifiant l'apport de CQ-mSCTP et les éventuelles perspectives à entreprendre.

CHAPITRE 1

Mobilité sur TCP/IP

Grâce au développement technologique des moyens de transmission radio, les terminaux peuvent communiquer pendant leurs mouvements. Cependant, les protocoles standardisés pour Internet ont montrés leurs limites face à cette mobilité. Il a été nécessaire d'enrichir ces protocoles afin de permettre le support de la mobilité.

Le présent chapitre est une introduction aux éléments de la mobilité. Il comprend en première partie un aperçu de la pile TCP/IP et ses caractéristiques. La deuxième partie est consacrée à l'étude de la mobilité des nœuds sur Internet, nous y présentons le concept de handover et son déroulement. Nous étudions ensuite, les contraintes imposées par la mobilité des nœuds selon le modèle TCP/IP. Nous présentons enfin , les nouveaux concepts de la multidomiciliation et de la mobilité des réseaux.

1. Introduction général à la mobilité

Les transmissions sur les réseaux sans fil, identiquement aux réseaux filaires, reposent sur la pile de protocoles TCP/IP. En effet, l'approche en couche de TCP/IP assure les communications d'un nombre extraordinaire de machines présentes sur Internet. Une telle réussite a encouragé le déploiement de cette approche en couche dans les transmissions sans fil.

Les prochains chapitres et sections font références aux caractéristiques et fonctionnalités des couches du modèle TCP/IP, c'est pourquoi, il convient d'en présenter un bref aperçu.

La couche liaison : Cette couche s'occupe de la connexion physique des machines aux réseaux, de leur accès au médium physique et de l'acheminement des trames de données entre deux équipements voisins.

La couche réseau : Ce niveau est formé d'un ensemble de protocoles – IP (Internet Protocole), ICMP (Internet Control Message Protocol), ARP (Address Resolution Protocol) - qui collaborent pour acheminer les données à destination. Pour ce faire, les sous-réseaux d'Internet possèdent chacun une plage d'adresses avec le même préfixe ce qui permet d'identifier leur position dans la hiérarchie de l'Internet. Tous les nœuds ayant une interface sur un sous-réseau donné ont une adresse IP contenant le préfixe de ce sous-réseau.

La couche transport : Cette couche contrôle les communications de bout en bout. Les protocoles TCP (Transmission Control protocol) et UDP (User Datagram Protocol) de cette couche sont majoritairement déployés. Ils identifient les connexions par les ports et adresses, source et destination des extrémités communicantes. TCP permet un échange fiable de données via le mécanisme d'acquittement des paquets reçus. IL assure également le contrôle de congestion sur le réseau. UDP, quant à lui, ne garanti pas un transfert fiable de données et ne peut pas assurer le contrôle de congestion.

La couche application : Le niveau application génère les programmes d'application et supporte les fonctions nécessaires au déploiement des programmes sur le réseau, telle la conversion d'images ou la compression de texte.

Les objectifs visés par chacune des couches TCP/IP sont similaires pour les transmissions en mode filaire ou sans fil. Toutefois les fonctionnalités et les mécanismes assurés par chaque niveau diffèrent selon les caractéristiques de ces deux modes.

Dans ce qui suit, nous identifions les propriétés et les problèmes engendrés lors de la mobilité des équipements sans fils sur Internet.

2. Mobilité des nœuds

La mobilité des nœuds dans les réseaux informatiques fait référence à la connectivité des mobiles durant leurs déplacements entre zones de couverture. On distingue deux sortes de connectivité : le nomadisme et la mobilité continue.

Le nomadisme indique que la connexion des nœuds mobiles-NMs au réseau sans fil s'effectue de manière identique à celle d'un nœud fixe au réseau filaire. Il se traduit par la possibilité de se connecter aux réseaux sans fil dans les zones de couverture sans la prise en compte des mouvements des nœuds. Autrement dit, le nœud mobile perd l'accès au réseau durant son déplacement entre zones de couverture. Le nomadisme est schématisé, donc, par une séquence de " déplacement -déconnexion - reconnexion". C'est pourquoi, il n'est pas considéré comme un modèle de mobilité propre au réseau sans fil [1]. En effet, le nomadise ne présente aucun mécanisme pour la gestion des données et des sessions durant la déconnection. Il ne peut donc assurer la garantie de service après une reconnexion.

Dans une mobilité continue, le nœud se déplace en préservant sa connexion au réseau. En d'autres termes, le mobile reste accessible durant ses mouvements entre zones de couvertures adjacentes. Le changement des caractéristiques de transmission avec cette forme de mobilité doit être gérer de manière transparente à l'usager et aux applications. Il s'agit de supporter l'opération de handover entre points d'accès des réseaux avec infrastructure, dite également handoff ou roaming [2]. Sur les réseaux Ad Hoc, la mobilité des nœuds est gérée suivant la connectivité des hôtes.

2.1. Qu'est ce qu'un handover ?

Un handover est l'enchaînement des procédures permettant à un nœud mobile, déjà connecté au réseau, de s'associer à un nouveau point d'attache (établir une connexion avec un nouveau réseau) en préservant son ancienne connectivité ainsi que la continuité des éventuelles sessions ouvertes [1].

2.2. Types de Handover

Le handover peut inclure un changement de point d'accès seulement ou le changement de point d'accès et de sous réseau Internet. Le premier cas est dit handover niveau 2, ou intra-routeur, car il ne fait intervenir que les deux premières couches du modèle TCP/IP et ne requiert pas la configuration d'une nouvelle adresse IP. Le deuxième cas est un handover niveau 3 ou inter-routeurs. Il passe d'abord par un handover niveau 2 et implique de plus la couche réseau pour l'obtention d'une nouvelle adresse IP sur le nouveau réseau de connexion. La figure 1 ci-dessous illustre ces types de handover.

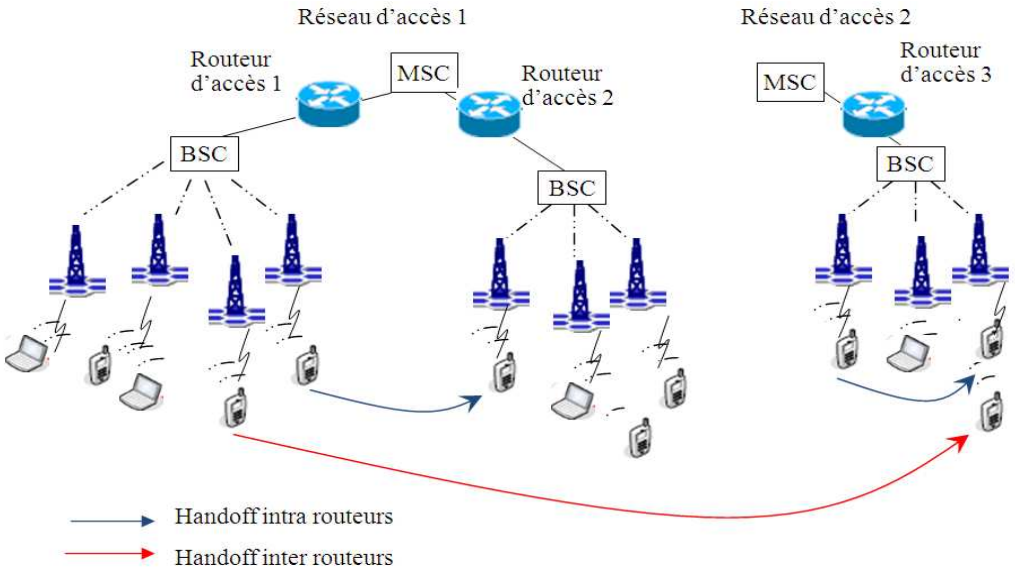


Fig 1 : *Handoff intra et inter routeurs.*

Plusieurs mécanismes sont étudiés pour gérer le handover et le mouvement des nœuds mobiles entre les points d'accès. Ces mécanismes sont évalués en fonction du taux de perte de paquet, engendré par la désinscription du nœud du premier réseau et son inscription au second réseau, et par le temps de transfert de l'association d'un point d'accès à l'autre, également dit temps de latence. Ces caractéristiques permettent de distinguer différents types de handover :

- Smooth handover : ce type de handover a pour objectif de réduire la perte de paquets, sans condition sur le délai de leur transfert.
- Fast handover : un handover est dit rapide lorsqu'il minimise les délais de latence [10].
- Seamless handoff : ou handoff transparent, ce type de handover désigne la définition absolue d'un handoff où il n'y a pas de changement dans la capacité, ni de la sécurité ou la qualité du service [1].

Les handovers sont également classés selon le type du point d'accès auquel le nœud mobile était connecté et celui du nouveau point d'attachement. Un handover est dit horizontale si les points d'accès (ancien et nouveau) sont de même technologie. Dans le cas contraire, on parle de handover vertical (ex : entre 802.11 et GSM). Ce dernier type implique le changement de qualité de service, de bande passante, de règles d'usages et présente une vulnérabilité accrue face aux failles de sécurité [3].

2.3. Processus d'un handover

Dans cette section, nous présentons le déroulement des étapes d'un handover. Nous explicitons essentiellement les étapes d'un handover intra-routeur d'accès, le procédé de handover niveau 3 étant plus détaillé dans le chapitre suivant.

2.3.1. Association au réseau

L'association d'un mobile au réseau est effectuée principalement suite au déplacement du nœud mobile d'une zone de couverture à une autre. Néanmoins, la diminution du signal ou l'encombrement d'un point d'accès peuvent le provoquer également.

Cette première étape concerne tous les types de handover puisqu'elle consiste en l'établissement du lien physique entre le nœud et son nouveau point d'attache. Cette étape présente l'actionneur du processus de handover. Elle dépend fortement de la technologie utilisée. Nous présentons dans ce qui suit un exemple de handover niveau 2 selon la norme IEEE-Institute of Electrical and Electronic Engineers, 802.11 du groupe de travail Internet Engineering Task Force-IETF vu la simplicité de sa mise en œuvre.

L'association au réseau nécessite une superposition en partie des zones de couverture. Elle passe par les étapes suivantes : la découverte des points d'attache, l'authentification et la réassociation :

Afin d'établir une connexion réseau sans fil, le nœud mobile doit d'abord chercher les points d'accès accessibles en effectuant soit un scan actif ou passif [2]. Dans le Scan actif, le nœud mobile initie la recherche par l'envoi de messages "Probe Request" sur les canaux radio. Il attend ensuite pendant un temps limité, sur chacun des canaux, la réception d'éventuels messages "Probe Response". Ces messages indiquent qu'il existe au moins un point d'accès sur le canal et véhiculent, de plus, les paramètres du point d'accès émetteur. A l'inverse du Scan actif, dans le Scan passif, le nœud mobile ne fait qu'écouter les canaux radio et intercepter à chaque fois les trames de signalisations envoyées par les points d'accès à intervalles réguliers [2]. Ces trames, dites "beacons", comportent des informations qui permettent au nœud mobile de s'attacher au point d'accès émetteur.

Le mobile choisit ensuite parmi les points d'accès trouvés celui auquel il va s'associer en se basant généralement sur le meilleur indicateur Signal to Noise Ratio- SNR qui définit la puissance du signal reçu à la réception d'une trame, rapporté au bruit présent sur le canal de communication [1].

Après choix du point d'accès au réseau, le mobile doit s'y authentifier afin d'établir la liaison des données et de bénéficier des services réseau. Pour cela, deux méthodes sont possibles : soit la méthode ouverte ou celle à clé partagé. Pour plus d'informations sur ces deux méthodes nous référons aux travaux présentés par [2]. La phase d'authentification peut être établie durant la procédure de scan. On parle dans ce cas de procédure de pré-authentification qui a l'avantage de réduire le temps de latence lors de handover.

Finalement, le processus de transfert entre les points d'accès passe à l'opération d'association/réassociation. En effet, afin de pouvoir communiquer sur le réseau, le nœud mobile doit "se mettre d'accord" avec le point d'accès sur les paramètres de communication. Cette opération est effectuée selon deux schémas de connexion:

- Si le nœud mobile s'attache pour la première fois au réseau, il initie la procédure d'association. Le point d'accès accepte ou refuse cette nouvelle association en fonction des ressources disponibles nécessaires pour assurer une bonne qualité de service au nœud mobile. Par exemple, la place requise pour la mise en tampon des trames du nœud mobile.
- Si le nœud mobile est déjà connecté à un point d'accès et qu'un handover devient nécessaire, une procédure de réassociation est établie et se déroule comme suit : lorsque le nœud mobile envoi une requête d'authentification au nouveau point d'accès incluant l'adresse de l'ancien point d'accès, le nouveau point d'accès demande à l'ancien de vérifier si le nœud mobile est vraiment authentifié chez lui. Si ce n'est pas le cas, la réassociation est refusée sinon le nouveau point d'accès envoi un message de réassociation positive au nœud mobile. L'ancien point d'accès qui recevait les trames, en attendant la réassociation du nœud mobile, les redirige au nouveau point d'accès et supprime son association au nœud mobile. Le processus de réassociation est ainsi terminé. Notons que durant toute cette phase le nœud mobile n'est connecté qu'à un seul point d'accès à la fois.

2.3.2. Configuration de l'interface réseau

Cette étape concerne les handovers niveau 3 ou inter-routeurs, et implique une gestion de l'adressage des nœuds mobiles. Les protocoles de gestion de ces handovers doivent attribuer au nœud mobile une interface réseau qui permet de l'identifier et de lui acheminer les paquets quelque soit son emplacement durant ces mouvements.

Parmi les solutions déployées pour router les paquets à destination, celles qui reposent sur le protocole IP attribuent au nœud mobile plusieurs adresses selon sa localisation sur le réseau. D'autres solutions reposent sur les nœuds mobiles ayant la capacité d'avoir plus d'une interface réseau, dits nœuds multidomiciliés (Chapitre 1, section 4).

Nous consacrons le chapitre 2 à l'étude de quelques solutions déployées pour la configuration d'une interface réseau qui répond à la mobilité des nœuds (Mobile Internet Protocole- MIP [37] et sa version 6- MIPv6 [38] et Network Mobility Basic Support Protocol- NEMO BSP [39]).

2.3.3. Transfert de sessions

Cette étape du handover est indispensable dans le cas où le nœud mobile possède des sessions ouvertes, lors de ses déplacements, qu'il doit maintenir. Elle représente un complément à l'étape précédente pour la gestion de la mobilité continue. Il s'agit de transférer les connexions actives du mobile vers la nouvelle interface réseau obtenue. Les protocoles de transport : Stream Control Transmission Protocol-SCTP [25], Datagram Congestion Protocol-DCCP [15], TCP Multi-Home Options- TCP-MH [36], permettent le transfert et le maintien des sessions ouvertes [8]. Nous analysons les propriétés et l'apport des deux premiers protocoles cités dans le chapitre suivant.

3. Contraintes d'une mobilité continue sur TCP/IP

Les divers protocoles qui assurent les handovers sont déployés selon le modèle en couches de TCP\IP. Quelques protocoles sont directement étendus des solutions filaires, tel que MIP, TCP-MH alors que d'autres tels que DCCP ou SCTP, n'en utilisent que des mécanismes essentiels tel que le contrôle de congestion. Néanmoins, les protocoles capables de gérer les handover intègrent de nouvelles fonctionnalités afin d'adapter les spécifications du modèle TCP/IP à la mobilité des nœuds.

Dans cette section, nous relevons quelques contraintes engendrées par le déploiement du modèle TCP/IP pour gérer la mobilité des hôtes.

3.1. Identification des nœuds

Selon le modèle TCP/IP, le protocole IP identifie chaque nœud avec une adresse unique. De plus, IP décompose cette adresse en deux parties : il identifie dans la première partie le sous-réseau d'appartenance du nœud (préfixe du réseau). La seconde partie détermine le nœud dans ce sous-réseau. L'adresse IP joue alors un double rôle : elle identifie le nœud et définie sa position sur Internet également. Cet adressage assure la cohérence et la sécurité des transmissions, par exemple, si un paquet provenant d'un sous-réseau indique l'adresse d'un autre sous-réseau, il sera détruit.

Ce principe appliqué aux réseaux filaires ne pose pas de problème puisque les nœuds sont reliés au même réseau et n'utilisent qu'une seule adresse pour la transmission des paquets. Cependant, le déploiement de ce principe pour identifier les nœuds mobiles implique que ces nœuds mobiles doivent configurer leur adresse sur les nouveaux réseaux avec les anciens préfix. La cohérence du protocole IP ne permet pas l'identification d'un nœud localisé dans un réseau 'A' avec un préfixe correspondant à un réseau 'B' ! C'est pourquoi, un nœud mobile doit obtenir une nouvelle adresse à chaque changement de réseau.

D'autre part, les protocoles de transport de TCP/IP identifient les connexions via un quadruplet (port source, adresse IP source, port destination, adresse IP destination). Toute modification sur l'un de ces paramètres est interprétée par une réinitialisation de la connexion.

De ce fait, l'application des protocoles transport et le protocole IP filaires pour supporter la mobilité continue des hôtes impose à la fois d'obtenir de nouvelles adresse IP sur chaque nouveau sous-réseau et de continuer à utiliser la même adresse initiale d'un nœud durant la vie d'une connexion transport. Les solutions qui permettent aujourd'hui de gérer le handover selon le modèle TCP/IP doivent répondre à cette contrainte.

C'est pourquoi, les protocoles proposées au niveau de la couche réseau préconisent de gérer une correspondance entre une adresse initiale des nœuds et des adresses obtenues lors des déplacements. Les solutions au niveau transport suggèrent des mécanismes pour permettre la modification des adresses transport en maintenant les paramètres identifiant les sessions.

3.2. Détection et contrôle de la congestion associés à la mobilité des hôtes

Les problèmes de la mobilité sur TCP/IP ne se limitent pas aux contraintes d'adressage des hôtes. Le contrôle de congestion effectué au niveau transport doit, également, être adapté à la mobilité des nœuds. En effet, contrairement aux réseaux filaires où les pertes de paquets sont dues majoritairement à la congestion du réseau, les pertes de paquets sur les réseaux sans fils sont plus importantes et sont dues, en plus, aux problèmes du signal, à une mauvaise qualité de la liaison radio ou suite au handover. De ce fait, l'application des protocoles de transport filaires aux réseaux sans fils ne fait qu'aggraver le problème de congestion. Ces protocoles interprètent les pertes de paquets comme congestion du réseau et réagissent par la réduction des délais et des débits des transmissions au lieu de réémettre le plus rapidement possible, comme il se doit sur les réseaux sans fil.

Les solutions transport déployés sur les réseaux sans fils ne devraient réduire le taux de transmission qu'après confirmation d'une réelle congestion du réseau. Il s'agit là d'une autre contrainte imposée par l'application du modèle TCP/IP pour la gestion de la mobilité. [6] présente de nombreuses méthodes pour détecter d'éventuelles congestion sur les réseaux sans-fil ce qui permet d'optimiser les performances des protocoles de transport. Les auteurs de [6] classe ces solutions en trois types :

Le premier type nécessite la mise en place d'un agent intermédiaire entre la source et la destination localisé sur les stations de base. L'agent enregistre une copie de chaque paquet transmis et examine chaque acquittement. Il réalise des retransmissions locales sur le canal sans-fil quand un paquet est perdu. Il informe ainsi l'émetteur si la perte s'est produite sur le réseau sans-fil ou bien sur le réseau filaire. Ce type d'approche nécessite d'apporter des modifications sur les équipements du réseau, et affecte plus de traitement aux stations de base.

Le deuxième type de solutions regroupe les mécanismes de bout-en-bout qui ne nécessitent aucune modification des équipements du réseau. [6] subdivise les solutions de cette classe en deux catégories : celles basées sur le temps d'arrivée entre deux paquets (*Inter Arrival Time- IAT*) et celles basées sur le temps d'aller des paquets (*Relative One-way Trip Time- ROTT*). L'évaluation de performance faite, montre que les méthodes basées ROTT sont meilleurs que celles basées sur IAT car dans la plupart des cas les pertes de congestion arrivent autour d'un pic de ROTT.

Dans le troisième type de méthodes, l'émetteur fait appel au mécanisme Explicit congestion notification-ECN, comme un simple mécanisme permettant de différencier entre causes des pertes de paquets. Le principe est de se référer au dernier intervalle de temps dans lequel une perte de paquets est signalée. Si dans cet intervalle, la source a reçu un ECN elle détecte la présence d'une congestion. Dans le cas contraire, la source détecte la perte de paquets suite à un problème de transmission sans fils.

4. Le multihoming

[8] définit le concept de multihoming comme suit : un nœud est dit multidomicilié lorsqu'il possède plusieurs interfaces réseau, étant configuré sur chaque interface avec une adresse IP différente. Les interfaces peuvent être de différente technologie sans fil. Le nœud peut utiliser ces interfaces simultanément ou alternativement [1]. La figure 2 montre deux nœuds multi-interfacés : à un instant (t), le nœud NM1 s'attache à Internet via deux adresses A1 et A2. Le nœud NM2, lui possède les adresses B1, B2 et B3.

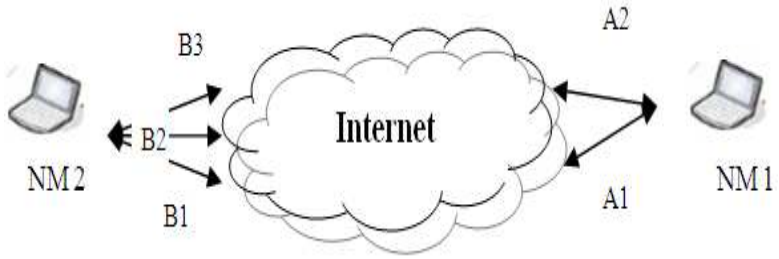


Fig 2: *Le Multihoming des nœuds*

Le multihoming est théoriquement utilisé pour augmenter la robustesse et le débit des connexions [1]. En effet, une telle configuration offre plusieurs avantages tels que :

- Une tolérance aux pannes de transmission physiques ; lorsque l'un des liens de transmission n'est plus disponible, les transmissions continuent sur les autres interfaces [8]. L'intérêt du multihoming est majeur lors d'un échec d'accès au réseau qui peut isoler tout un système [9].
- Un support à la mobilité lors des déplacements entre réseaux car les nœuds peuvent maintenir leurs connexions en utilisant différentes adresses. Le multihoming présente une solution aux problèmes de handover.
- Le partage des flux. En effet, les nœuds peuvent répartir leur téléchargement à travers les différentes interfaces [9].
- Mise en pratique des préférences. Un nœud choisit l'interface de transmission en fonction de différents critères comme la qualité de services et le coût d'accès au fournisseur de service selon l'interface [10], [9]. i.e : garantir un meilleur accès à l'Internet en faisant appel à différentes technologies [3].

Notons cependant que le déploiement du multihoming simultané entraîne quelques problèmes qui réduisent la performance au lieu de l'augmenter dont on cite :

- Conflits de routage des paquets sur des chemins concurrents ce qui engendre souvent un taux élevé de désordre dans l'arrivée de paquets [1].
- Surconsommation de l'énergie des équipements suite à l'ajout d'interfaces [2].
- La nécessité de considérer les aspects de changement d'interface afin de répondre aux contraintes de la mobilité sur TCP/IP [3].

5. Mobilité groupée

La gestion de la mobilité tend aujourd'hui à résoudre des problèmes liés au déplacement de tout un réseau et non d'une station unique [7]. On parle alors de la gestion de la mobilité groupée. Dans ce contexte, [7] définit un réseau mobile comme un ensemble de nœuds mobiles qui se déplacent collectivement vers une même destination. Un routeur mobile-RM du réseau assure les transmissions des données des nœuds intérieurs au réseau. La taille des réseaux mobiles varie de quelques nœuds mobiles dans le cas d'un réseau personnel jusqu'à plusieurs centaines de stations interconnectées par plusieurs routeurs et sous-réseaux dans le cas d'un train.

La gestion d'une mobilité groupée trouve son intérêt devant l'émergence des réseaux de capteurs embarqués dans les véhicules, les réseaux d'accès déployés dans les transports, les réseaux ambiants ou encore les réseaux personnels sans fil. Le besoin de connecter des réseaux à l'Internet est suscité soit par les fabricants de véhicules, soit par les compagnies de transport ou alors par les usagers eux-mêmes [3]. Ceci favorise alors le déploiement d'applications multimédia en tout lieu et à tout instant.

Le tableau suivant résume les liens entre le type de mobilité, les équipements utilisés et les caractéristiques de chaque réseau mobile [3].

Type	Mobilité	Equipement	Besoins Particuliers
Véhicule personnel ou bus interurbain	Forte Locale et globale (Zone urbaine ou faiblement urbanisée, franchissement des frontières)	Petit nombre, Plusieurs technologies d'accès	Handovers verticaux multidomiciliation
Train ou bus urbain	Moyenne Locale (à l'intérieur d'un réseau d'accès propriétaire)	Quelques dizaines ou centaines en majorité appartenant aux passagers, 1 ou 2 technologie d'accès au maximum	Contrôle d'accès Handovers horizontaux Multidomiciliation Mobilité à plusieurs niveaux
Avion	Faible, Locale (réseau appartenant aux opérateurs de satellite et aéroports)	Quelques dizaines ou centaines, en majorité appartenant aux passagers, 2 ou 3 technologie d'accès (satellites, radio, wifi)	Sécurité renforcée Contrôle d'accès Handovers verticaux Multidomiciliation
PAN (Personal Area Network)	Faible (selon l'usage) Zone urbaine	Petit nombre Plusieurs technologies d'accès	Connectivité globale Handovers verticaux

Tab 1 : *Caractéristiques des réseaux mobiles selon leur type*

Notons que la gestion d'une mobilité groupée diffère de la mobilité des réseaux Ad Hoc. Cependant une interaction entre les réseaux Ad Hoc et les réseaux mobiles peut être relevée. En effet, vu que les hôtes d'un réseau Ad Hoc sont également routeurs de paquets, il est possible qu'un sous réseau Ad Hoc constitue un réseau mobile. [3] cite l'exemple des passagers d'un train qui peuvent former un réseau Ad Hoc. Si la connectivité à Internet est offerte par le biais d'une passerelle à bord du train changeant son point d'ancrage (i.e. un RM), il s'agit alors d'un réseau mobile constitué de nœuds Ad Hoc. En revanche, lorsque des réseaux sont embarqués dans des véhicules, et qu'une flotte de véhicules de ce type forme un réseau Ad Hoc, nous avons un réseau Ad Hoc constitué de réseaux mobiles.

La mobilité groupée présente des caractéristiques et des problèmes spécifiques. Entre autres, la quantité du trafic qui transite via un routeur mobile et qui correspond aux sessions ouvertes simultanément est d'autant plus significative que le nombre de correspondants ou de réseaux servis par le routeur mobile. Les protocoles de gestion de la mobilité groupée doivent assurer un passage à l'échelle, comparé à la gestion de la mobilité des stations. De plus, en conséquence à l'hétérogénéité des nœuds intérieurs aux réseaux mobiles, on identifie une mobilité enchaînée à plusieurs niveaux. En effet, un réseau mobile peut accueillir soit une station mobile, soit un routeur mobile servant lui-même de passerelle à un autre mobile ce qui peut rendre la mobilité des réseaux récursive. Tel est le cas lorsqu'un passager utilisant des équipements communicants personnels dans un bus servi par un routeur mobile, permet à son tour l'ancrage d'autres passagers.

D'autres parts, le changement du point d'ancrage sur les réseaux mobiles est plus au moins prévisible. Les handovers d'un train par exemple qui suit une trajectoire déterminée sont prévisibles contrairement à une hôte (piéton ou une automobile) en milieu urbain qui change de trajectoire, de vitesse, et de réseau d'accès si bien que ses handovers soient très difficilement prévisibles.

Par ailleurs, un réseau mobile est multidomicilié lorsqu'il est simultanément connecté à l'Internet via plusieurs routeurs mobiles ou lorsque l'un des routeurs mobiles a plusieurs interfaces externes. Cette possibilité de se connecter par l'intermédiaire d'un ou plusieurs routeurs mobiles disposant au total de plusieurs interfaces externes nécessite de considérer les aspects de changement d'interface et de changement de routeur mobile.

D'autres particularités de la mobilité groupée sont relevées dans [3] tels que le contrôle d'accès aux ressources des réseaux, et l'adaptation des applications à la bande passante disponible, la sécurité des transmissions. Ces caractéristiques spécifiques aux réseaux mobiles ont orienté les recherches vers le développement de protocoles qui en tiennent compte. Il s'agit particulièrement du protocole NEMO BSP et de ses extensions.

6. Conclusion

Dans ce chapitre, nous avons identifié les concepts essentiels à la gestion de la mobilité. Nous avons présenté les étapes nécessaires à l'accomplissement de handover, chacune étant effectuée à un niveau précis du modèle TCP/IP. Nous avons vu que ce modèle, bien qu'il soit intéressant, impose des contraintes à prendre en considération pour la gestion de la mobilité des nœuds. De nouveaux concepts tels que le multihoming et la gestion d'une mobilité de réseaux sont aujourd'hui implémentés pour permettre de gérer la mobilité et d'en améliorer le fonctionnement.

Dans ce qui suit, nous présentons les solutions effectives, nécessaires à la gestion de la mobilité sur Internet, déployées sur différents niveaux du modèle TCP/IP. Ces solutions répondent aux contraintes vues dans ce chapitre.

CHAPITRE 2

Gestion de la Mobilité aux Niveaux Réseau et Transport

Dans ce qui suit, nous explorons des solutions pour la gestion de la mobilité sur les couches supérieures du modèle TCP/IP. Nous nous intéressons d'abord aux extensions du protocole IP pour le support de la mobilité. Nous présentons ensuite d'autres protocoles déployés au niveau transport.

1. Gestion de la mobilité au niveau IP

1.1. Mobile Internet Protocol-MIP

Pour gérer la mobilité des équipements sur Internet, Les premières solutions au niveau 3 du modèle TCP/IP ont visé l'adaptation du protocole filaire IP. L'IETF propose le 1er RFC pour traiter la mobilité sur IP en 1996 [37]. Mobile Internet Protocol- MIP a ensuite été étendu par de nouvelles fonctionnalités qui ont donné jour aux extensions MIPv4 et MIPv6, aujourd'hui déployés pour gérer la mobilité.

1.1.1. Mécanisme de MIP

Le protocole MIP intègre essentiellement les fonctionnalités d'identification des nœuds mobiles et de transmission des données sur Internet.

1.1.1.1. Identification des nœuds mobiles

MIP définit comment un NM doit s'attacher à Internet de manière à permettre son identification sur le réseau durant ses déplacements. Nous présentons d'abord les grandes lignes de cette procédure. Nous décrivons ensuite les phases et les messages qui permettent de l'accomplir.

Lorsque les NMs s'attachent à Internet, MIP leur attribue deux types d'adresses selon leurs positions sur la hiérarchie Internet :

Lors du premier ancrage du NM à un sous réseau d'Internet, il obtient une adresse permanente sur 32 bits constituée du préfixe de ce réseau comme pour IP standard. Ce premier réseau est dit "réseau mère" du nœud mobile. L'adresse obtenue joue le rôle d'identifiant propre au mobile quelque soit le sous réseau auquel il s'attache durant ses déplacements. Un routeur du réseau mère qui intègre MIP, dit "agent mère" (Home Agent –HA) maintient la liste des NMs appartenant au réseau, i.e : de même préfixe. L'agent mère associe à chaque NM de son réseau une entrée dans une table d'association qui servira, de plus, à garder trace de la localisation du NM.

Lorsque le NM se déplace, un routeur dédié sur le nouveau réseau dit "agent visité" (Foreing Agent-FA) lui attribue une nouvelle adresse temporaire, ou Care-of-Adress-CoA. Cette adresse permet d'identifier le NM sur ce réseau visité uniquement. Le NM informe son agent mère de l'obtention de chaque adresse temporaire. Le HA peut ainsi connaître la CoA des nœuds issus de son réseau. L'agent mère maintient dans le cache d'association la correspondance entre l'adresse principale d'un NM et son adresse temporaire courante.

Nous détaillons dans ce qui suit, la mise en œuvre de cet adressage qui passe par les phases de découverte d'agents, d'obtention d'adresse temporaire et d'enregistrement auprès du HA. Durant ces phases, les NMs, HA et FA s'échangent différents types de messages :

i) Découverte d'agents

Dès l'accès d'un NM à Internet, il établit une première connexion au réseau mère et d'éventuelles connexions aux réseaux visités ou reconnexions au réseau mère. Afin de permettre aux NMs de détecter ces changements de réseaux, les agents HA ou FA diffusent périodiquement des paquets "Agent Advertisement". Ces paquets présentent l'identité du réseau : des informations sur la disponibilité des agents, sur le préfixe du réseau et la durée de vie des messages diffusés [1]. Toutefois, les NMs peuvent anticiper cette découverte et demander ces informations explicitement via diffusion de messages "Agent Solicitation"

ii) Obtention d'adresse temporaire

Si le NM détecte un changement du préfix réseau dans les messages "Agent Advertisement" reçus, il "déduit" qu'il a migré vers un nouveau réseau auquel il doit s'attacher. Dans le cas ou le NM se trouve sur un nouveau réseau visité, c'est le FA de ce réseau qui attribue au NM une nouvelle adresse temporaire CoA. Si le NM détecte le même préfix réseau que son réseau d'origine, il réutilise son adresse permanente.

iii) Enregistrement auprès du HA

Lorsque le NM obtient la CoA, il envoi une requête de prise en charge à son HA où il indique la correspondance entre une adresse temporaire acquise et son adresse permanente. Cette demande d'enregistrement est transférée par le FA dans un paquet UDP [11].

Le HA doit répondre à la réception de cette requête par un message "Registration Reply" qui est également transféré par le FA. Si le HA accorde cet enregistrement, il met à jour la correspondance entre adresse fixe et la CoA du NM dans le cache d'association. Ce processus est sécurisé grâce à Hashed Message Authentication Code with Message Digest version 5 - HMAC-MD5 [1]. La figure suivante présente un diagramme de séquence de ces transmissions.

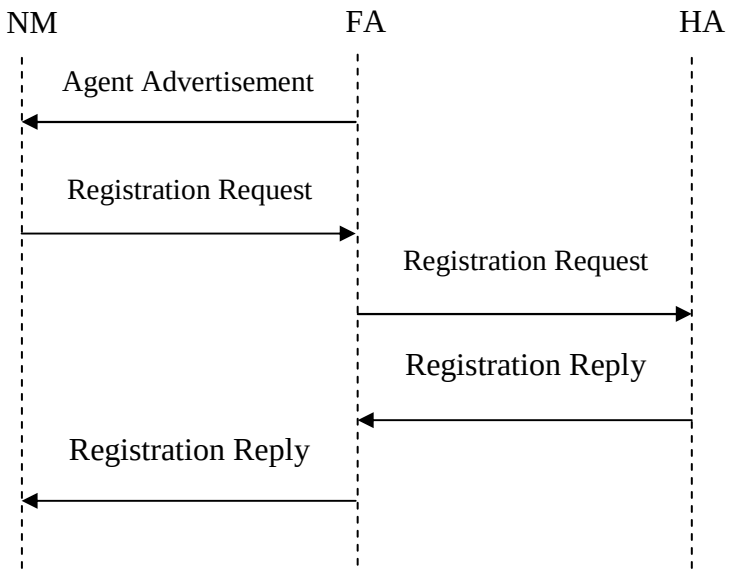


Fig 3 : *Diagramme de séquence pour enregistrement des adresses identifiants un NMs.*

Le NM doit renouveler son enregistrement auprès de l'agent mère périodiquement afin que son entrée dans le cache d'association n'expire pas. D'autre part, lorsque le nœud revient au réseau mère, il doit annuler son entrée du cache d'association via l'envoi/réception de messages "Dé-registration Request" et "Dé-registration Reply".

1.1.1.2. Transmission des données

Le procédé d'adressage décrit, permet d'identifier un NM via son adresse permanente, mais de plus, il permet de lui router les paquets après déplacement via son adresse temporaire. Dans cette section, nous présentons comment MIP utilise cet adressage dans la transmission des données.

Lorsque les correspondants envoient leurs paquets aux NMs, ils indiquent son adresse permanente. Cette destination fait référence au préfixe du réseau mère du NM. Le HA du NM capture ces datagrammes grâce au protocole Adress Resolution Protocol-ARP [1]. Le HA consulte l'entrée du NM dans son cache d'association pour déterminer sa position sur Internet. Si le nœud mobile est toujours attaché au réseau mère, les paquets lui seront livrés comme pour un nœud du réseau fixe. Par contre, si le NM se trouve hors son réseau mère, le HA ajoute à ces paquets interceptés, une entête correspondant à la CoA actuelle du NM et les lui transfert sur cette adresse. Les nouveaux datagrammes ainsi générés sont routés vers le FA selon le préfix réseau de la CoA. Le FA décapsule ces datagrammes et les remet au NM.

Cette technique d'encapsulation des datagrammes par un entête supplémentaire est dite "Tunneling". La figure 4 montre que cet encapsulation ne modifie pas les paquets et préserve l'entête original.

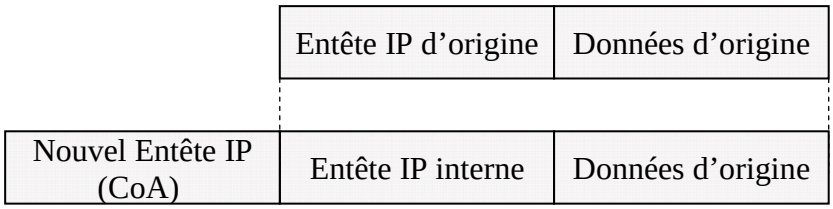


Fig 4 : *Encapsulation IP dans IP*

Dans le sens inverse de transmission, les datagrammes en provenance du NM sont routés selon le mécanisme standard de IP. Le NM spécifie son adresse permanente comme adresse source des paquets qu'il envoi vers ses correspondants sans passer par les agents FA et HA.

On note que le mécanisme de routage de MIP spécifie deux chemins différents dans l'émission et la réception des paquets, d'où son appellation "routage triangulaire". La figure ci-après montre le routage des datagrammes sur MIP.

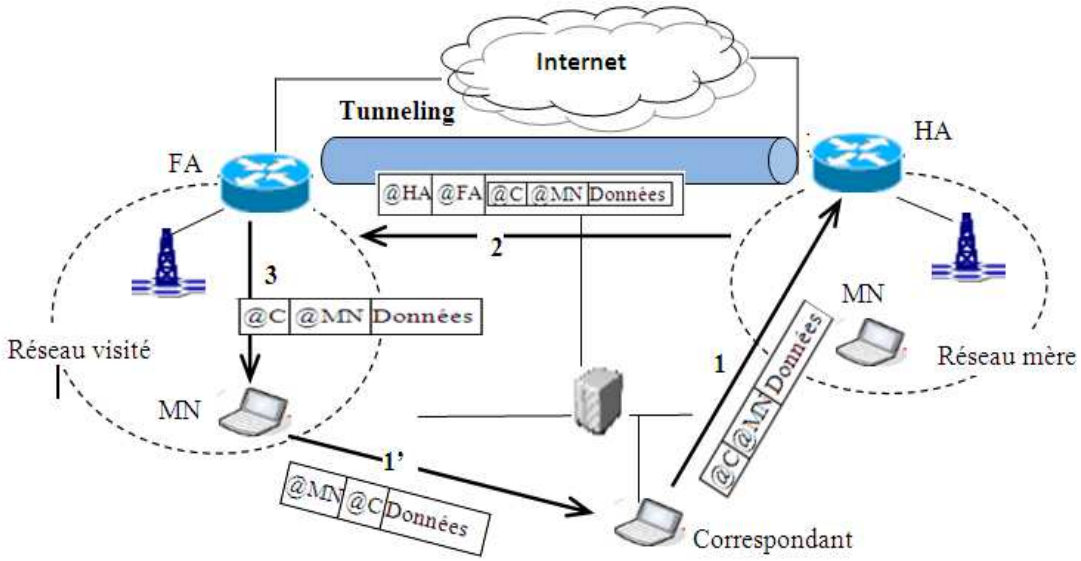


Fig 5 : *Principe du routage triangulaire de MIP*

1.1.2. Discussion

L'approche d'adressage avec deux types d'adresses est la plus évidente et la plus simple car elle a l'avantage de ne pas remettre en cause l'exploitation des différents protocoles de la couche réseau et ne nécessite l'ajout de fonctionnalités qu'aux entités mobiles en terme IP ainsi qu'aux HAs [3]. De plus, cette approche permet d'acheminer les datagrammes aux NMs selon leurs localisations et assure la continuité des transmissions lors des handovers.

En effet, lorsque le NM est en mouvement entre deux réseaux adjacents, les paquets des correspondants lui sont toujours envoyés en utilisant son adresse permanente. Le HA se charge alors de recevoir ces paquets en attendant que le NM obtienne une nouvelle CoA et mette à jour son entrée dans le cache d'association du HA et termine ainsi le handover. Les paquets en attente sur le HA peuvent alors être routés vers la nouvelle CoA du NM. Notons que MIP maximise le rendement des handovers lorsque les cellules des routeurs se recouvrent fortement, i.e : le NM détecte le nouveau réseau et déclenche l'opération d'obtention d'une nouvelle CoA suffisamment à temps avant que l'ancienne ne soit plus utilisable, ce qui réduit les paquets à stocker au niveau de l'ancien FA.

MIP répond ainsi à la contrainte d'une mobilité continue sur Internet, identifiée au chapitre 1- section 3.1, via les mécanismes d'adressage et de transmission des donnés implémentés. Cependant, l'utilisation d'une adresse unique pour effectuer les transmissions quelque soit la position du mobile dénature le sens de l'adresse IP qui est de permettre aux correspondants, à la fois, d'identifier et de localiser une machine grâce à son adresse. Par ailleurs, MIP présentent quelques lacunes qui ont bloquées son déploiement sur Internet. Nous citons entre autres :

- MIP surcharge le réseau par le volume des données dû à l'encapsulation IP dans IP et aux quantités de messages de signalisation, envoyés périodiquement, pour permettre la détection et l'enregistrement auprès des agents.

- Le routage des paquets via le HA est indirect et n'optimise pas le chemin de transmission. Il devient particulièrement peu performant lorsque le nœud mobile en déplacement et son correspondant sont sur le même réseau IP. En outre, ce routage triangulaire engendre des problèmes de congestion au niveau du HA même.

- MIP ne spécifie aucune solution pour faire face aux mécanismes de filtrage de paquets (ingress et egress filtering) implémentés au niveau des routeurs pour situer les attaquants Denial of Service-DoS dans les sous réseau [1]. Le « ingress filtering » préconise la destruction de tout paquet sortant d'un réseau et qui ne porte pas son préfixe. Quant au « egress filtering », il préconise la destruction des paquets dont le champ source indique le préfixe du réseau auquel ils vont s'introduire. Ce filtrage empêche l'envoi direct des paquets du NM vers ses correspondants lorsqu'il est dans le réseau visité et détruisent les paquets qu'un NM envoi à ses correspondants situés dans son réseau mère.

- Le temps de handover de MIP est potentiellement long dû aux mécanismes de détection et d'enregistrement auprès des agents ajoutés aux temps de handover de niveau 2. De plus, la détection d'un changement de réseau se base sur la réception d'un message "Agent Advertisement" qui indique un nouveau préfix réseau ou sur l'écoulement du temps d'attente d'un "Agent Advertisement" de l'ancien réseau. Cependant, ces messages ne sont diffusés que périodiquement et non pas en réponse aux mouvements des NMs.

- MIP souffre de pertes de paquets. Cette déficience est une conséquence directe au routage triangulaire et au temps de latence imposé par les mécanismes de détection et enregistrement auprès des agents.

- Le changement fréquent d'adresse temporaire (CoA) rend le support de QoS difficile. Avec le protocole Resource ReSerVation Protocol- RSVP classique, par exemple, les réservations de services doivent être rétablies le long du chemin chaque fois que le NM se déplace, même si une partie importante du réseau reste inchangée [11].

1.1.3. Extensions de MIP

Pour permettre la mise en œuvre de MIP, l'IETF propose d'y intégrer de nouveaux mécanismes qui remédient aux problèmes précédemment cités. Des extensions ont finalement pu être déployées dans les versions ultérieures de MIP : MIPv4 et MIPv6. Dans ce qui suit nous en spécifions quelques unes.

1.1.3.1. MIP avec encapsulation au retour

Pour résoudre le problème du filtrage sur les routeurs, deux éventuelles solutions se présentent : soit de permettre à un NM d'envoyer des paquets à partir du réseau visité avec la CoA comme adresse source ou de faire passer tous les datagrammes du NM destinés aux correspondants par le HA. Vu que les correspondants ne connaissent que l'adresse permanente des nœuds mobiles dans MIPv4, c'est la deuxième solution qui est implémentée dans cette extension.

Bien que ce routage avec encapsulation au retour évite la destruction des paquets par le mécanisme de filtrage, il impose cependant un passage par le HA dans les deux sens de transmission. L'emploie du tunneling dans les deux sens de transmissions ne fait qu'accroître le problème du routage de MIP, de congestion au niveau du HA et de perte de paquets.

1.1.3.2. MIP avec optimisation du routage (Routing Header)

Pour optimiser le routage triangulaire de MIP, l'IETF propose une autre extension. Il s'agit d'acheminer les paquets directement entre correspondants et NMs sans passer par le HA. Ce routage direct implique les correspondants dans le traitement des adresses temporaires.

Un correspondant de cette extension dispose d'un cache d'association identiquement à l'agent mère. Grâce à ce cache, le correspondant est capable d'encapsuler les datagrammes qu'il émet en destination du NM avec son adresse temporaire. Ainsi, le HA aura moins de tâches d'interception des paquets en destination du NM.

La disposition du cache au niveau des correspondants implique que d'une part, les NMs doivent informer les correspondants aussi bien que le routeur mère dès qu'ils changent de réseau. Pour ce faire chaque NM maintient la liste des correspondants auxquels il doit envoyer ces messages de mise à jour. D'autre part, les correspondants doivent être capables de traiter ces messages et mettre à jour les entrées du cache par les adresses temporaires des NMs.

Le problème se pose avec cette optimisation lorsque les informations de mise à jour des adresses temporaires ne parviennent pas à temps aux correspondants. En effet, le cas se présente lorsque le NM se déplace et devient hors portée de l'ancien réseau mais que les traitements des MAJ entre le HA et les correspondants ne sont pas encore achevés. Les paquets seront routés vers l'ancien réseau visité alors que le NM n'y est plus. Pour éviter ces pertes de paquets, le NM de cette extension informe son dernier FA de sa nouvelle position. Ce FA participe au handover en stockant puis en transférant les datagrammes vers la nouvelle position du NM.

Cette solution permet un routage direct (Routing Header), cependant elle impose la reconfiguration des hôtes des correspondants de manière à supporter l'encapsulation/décapsulation des paquets et le traitement des notifications de changement d'adresse. Cette extension a été ajoutée aux fonctionnalités de MIPv4 et constitue, une partie intégrante de MIPv6 dès son élaboration.

1.1.3.3. Obtention d'adresses temporaires via DHCP

MIPv4 a fait objet d'une autre modification, intégrée ensuite à MIPv6. Il s'agit de permettre au NM de s'auto configurer une adresse temporaire via le protocole Dynamic Host Configuration Protocol- DHCP. L'utilisation de ce protocole nécessite la présence d'un serveur DHCP qui se charge de définir l'adresse d'un nœud réutilisable dès que le poste est éteint. Les RFCs 2131 et 2132 présentent de plus ample informations sur le fonctionnement du protocole. Le déploiement de DHCP dans MIPv6 permet de router les paquets interceptés par le HA directement vers le NM. En effet, il n'est plus nécessaire de passer par le FA pour décapsuler les paquets puisqu'il n'est pas à l'origine des adresses temporaires. Ce fonctionnement plus simple pour les clients, est plus complexe pour le serveur puisqu'il requière la réservation d'une plage d'adresses pour la gestion des mobiles dans un réseau en excluant les adresses des serveurs et routeurs.

1.2. Mobile Internet Protocol version 6 (MIPv6)

MIPv6 est le standard défini par l'IETF pour gérer la mobilité dans les réseaux IPv6. Il intègre des extensions qui permettent de pallier aux problèmes décelés avec MIP et MIPv4. Nous présentons dans ce qui suit les modifications apportées par MIPv6 par rapport à ses antécédents.

1.2.1. Identification des NMs

MIPv6 se base sur le même principe de MIP pour identifier les NMs. Il utilise des adresses temporaires pour identifier les NMs durant leurs déplacements et affecte une adresse permanente à chaque NM pour permettre la continuité des sessions. MIPv6 ne déploie plus d'agent mère mais plutôt un routeur d'accès mère ou Home Router-HR. Ce HR, comme pour le HA de MIP, attribue les adresses permanentes aux NMs et intercepte les paquets en destination de ces NMs.

Néanmoins, MIPv6 intègre des modifications au processus d'obtention d'adresses temporaires rendues possibles grâce aux fonctionnalités implémentées dans IPv6. Entre autres, MIPv6 ne déploie plus de FA. En effet, l'intérêt du FA dans MIP est de pouvoir réutiliser la même CoA pour différents NMs [1]. Mais puisque l'espace d'adressage de IPv6 est de 128 bits, le NM peut acquérir son adresse temporaire, par le mécanisme de DHCP. Il en vérifie l'unicité par le protocole de détection de duplication d'adresse.

La phase de découverte d'agents de MIP, qui permet l'identification des NMs, se traduit par un processus de découverte de routeurs d'accès semblable à celui de MIPv4. Les routeurs mère émettent périodiquement des messages "Router Advertisement-RA" pour indiquer au NM une information lui permettant de déterminer s'il se situe dans un nouveau sous-réseau. Le NM utilise cette information pour s'auto-configurer son adresse temporaire. Les NMs peuvent eux même solliciter un routeur via un message "Router Sollicitation".

MIPv6 spécifie l'optimisation du routage en tant que mécanisme intégré via le "Routing Header". C'est pourquoi les correspondants détiennent un cache d'association qu'ils doivent mettre à jour.

Par ailleurs, les NMs émettent des requêtes pour la mise à jour du cache d'association aux correspondants aussi bien qu'aux HRs. Ces requêtes sont véhiculées périodiquement par des messages "Binding Update-BU". Les BUs sont des paquets spéciaux contenant deux extensions d'en-tête IPv6 supplémentaires. L'adresse permanente est contenue dans l'option "Home Address Option". Le message de mise-à-jour instruisant le destinataire d'ajouter ou de mettre à jour l'entrée correspondante dans son cache (Binding Cache) est contenu dans l'entête d'extension "Mobility Header" [3]. Toutefois, le HA doit répondre par un "Binding Ackowledgement" pour confirmer s'il peut mettre à jour l'entrée du NM et donc l'établissement du tunnel de transmission. Par ailleurs, les correspondants peuvent demander à un NM d'envoyer des "Binding Update" pour rafraichir son entrée dans le cache d'association.

La protection des mises à jour envoyées par le NM aux correspondants avec MIPv6 ne demande ni d'avoir établi une association de sécurité auparavant, ni l'existence d'une infrastructure d'authentification. MIPv6 inclut un entête d'authentification aux paquets et implémente la méthode de "Return Routability" pour vérifier que l'adresse temporaire et l'adresse principale référencent le même terminal [2].

1.2.2. Transmission de données

Grâce au déploiement du "Routing Header", les transmissions directes entre correspondants et NMs sont possibles. Les correspondants envoient les datagrammes en spécifiant l'adresse temporaire du NM dans l'entête du paquet. L'adresse permanente du NM est également incluse dans le nouvel entête de routage IPv6 (Routing Extension Header) ce qui permet d'identifier le NM destinataire des paquets. En recevant les paquets, un NM de MIPv6 est capable de décapsuler lui-même l'entête IP contenant son adresse temporaire. Il substitue ensuite l'adresse temporaire par l'adresse permanente et passe les paquets à la couche supérieure.

MIPv6 établit, selon la figure 6 un tunnel bidirectionnel entre correspondants et NMs permettant le maintien des sessions ouvertes [3].

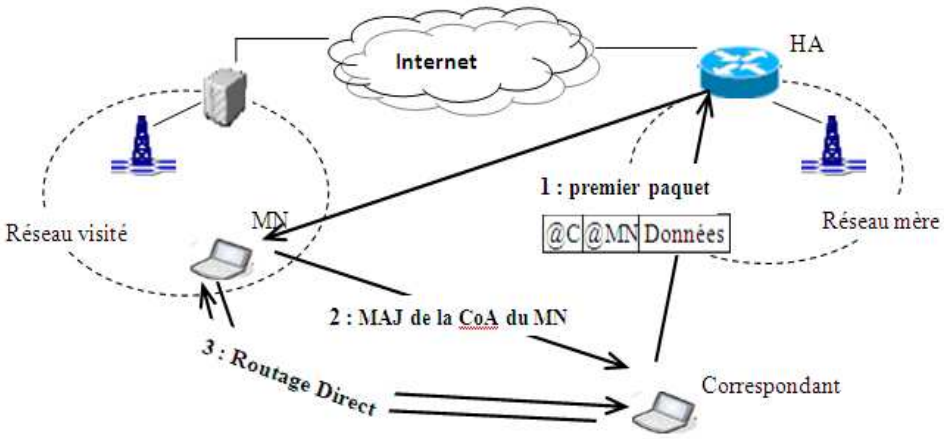


Fig 6 : *Routage directe dans MIPv6*

Dans le cas où le NM reçoit un paquet d'un correspondant qui ne contient pas de "Routing Header", le NM détermine que le paquet a été tunnelé par l'agent mère. Ce cas de figure se présente lorsque le correspondant n'a pas d'entrée dans son cache d'association pour le NM. Le NM envoie un "Binding Update" à l'émetteur pour l'amener à exécuter le tunnel bidirectionnel.

MIPv6 réduit la surcharge au niveau du routeur mère observée dans Mobile IPv4 puisque les paquets des correspondants ne sont plus contraints de passer par le réseau mère. De plus ce mécanisme permet aux niveaux supérieurs de continuer à fonctionner de manière transparente.

Toutefois, MIPv6 apporte une modification importante au principe d'optimisation du routage. En effet, cette version définit une solution alternative au déploiement des FA pour éviter la perte de paquet lors de handovers. Il se sert du fait qu'une machine peut supporter simultanément plusieurs adresses IPv6 (i.e. multi-interfaces).

[2] présente le processus d'utilisation du multihoming pour accomplir un handover selon MIPv6. A un instant donné, l'une des interfaces du NM est associée à un point d'accès et est utilisée pour communiquer normalement. Tant que la qualité du signal entre cette interface et le point d'accès est suffisante, les autres interfaces sont en mode veille. Lorsque le terminal s'éloigne du point d'accès, la qualité du signal décroît. De ce fait, le terminal active une seconde interface et l'utilise pour rechercher un nouveau point d'accès en vue de s'y associer. Dès l'obtention d'une nouvelle adresse sur la seconde interface, le terminal peut envoyer un BU à son agent mère lui indiquant d'envoyer simultanément les paquets de données vers ses deux interfaces. Lorsque le terminal n'utilise plus sa première interface, il la passe en mode veille.

Ce processus permet de réaliser des handovers complètement transparents pour l'utilisateur, étant donné que le terminal est continuellement associé à un point d'accès. Grâce aux interfaces supplémentaires, le terminal peut réaliser toutes les procédures définies dans le protocole MIPv6 sans perturber ses communications. Un terminal peut ainsi maintenir une connexion ouverte et continuer à recevoir des paquets et obtenir en parallèle une nouvelle adresse temporaire avant son enregistrement.

1.2.3. Discussion

Le protocole MIPv6 est aujourd'hui une solution déployée pour gérer la mobilité au niveau IP. Il présente un avantage majeur par rapport à MIPv4 puisqu'il tient compte "nativement" des optimisations développées et permet de les exploiter directement sur les NMs d'Internet.

En outre, grâce à l'intégration du routage direct entre un NM et ses correspondants et sans faire appel au FA, MIPv6 offre une meilleure résistance aux facteurs d'échelle et fiabilité. La communication entre NMs et correspondants engendre moins de charge sur le réseau et devient plus rapide. Comme l'agent mère est peu sollicité pour la retransmission des paquets, il y a beaucoup moins de risque de congestion à son niveau. Une déficience sur l'agent mère aura un effet moindre [10].

Les protocoles d'auto-configuration présents dans les spécifications d'IPv6 permettent une gestion simplifiée de l'attribution d'adresses temporaires. Toutefois, le déploiement de cette extension dépend également des équipements et des débits offerts par les fabricants d'équipements sans fil.

1.3. NEtwork MObility Basic Support Protocol- NEMO BSP

1.3.1. Introduction

NEtwork MObility Basic Support Protocol- NEMO BSP est un protocole standardisé par l'IETF pour la gestion de la mobilité des réseaux au niveau IP. Le groupe de travail NEMO est crée en 2002 en réponse aux caractéristiques et besoins de la mobilité groupée. Le premier RFC de NEMO [39] qui porte sur les spécifications de base, est publié en 2005.

En fait, L'IETF discute initialement le déploiement de MIP et ses extensions pour la gestion de la mobilité des réseaux. Cependant, ces protocoles montrent leurs limites face à cette forme de mobilité. [3] identifie ces limites, relevant certes de la sécurité et du partage des ressources mais le problème majeur étant lié à l'adressage et le routage des paquets :

D'une part, selon le principe de la mobilité des réseaux, les nœuds de chaque réseau mobiles représentent une seule entité dont la mobilité est gérée au niveau des routeurs mobiles RMs. C'est pourquoi, les entrées dans le cache d'associations des HAs correspondent aux adresses des RMs uniquement. Les HAs ne maintiennent pas d'entrées pour les nœuds intérieurs aux réseaux mobiles. Ils ne traitent que les paquets ayant pour destination finale ces RMs. Les paquets en destination des nœuds internes aux réseaux mobiles (Mobile Network Node-MNNs) seront perdus puisque aucune spécification de MIP et ses extensions n'indique aux HAs des RMs d'intercepter et d'encapsuler les paquets destinés aux nœuds situés derrière ses RMs.

D'autres part, lorsqu'un réseau mobile se déplace, le RM préserve son adresse permanente et lui associe une nouvelle CoA dans son entrée au niveau du HA. Le mécanisme d'optimisation du routage utilise ensuite les BUs pour mettre à jour ces associations entre adresse permanente et CoA. Cette gestion ne peut être directement appliquée aux NMs intérieurs au réseau mobile. En effet, bien que les MNNs soient toujours attachés au même RM, ils changent d'adresses du point de vue des correspondants. Selon le principe

d'optimisation du routage, ces NMs doivent également envoyer des BUs aux correspondants et périodiquement pour maintenir à jour leur cache d'association. Une telle gestion conduit, comme indiqué sur la figure 7, à une explosion de BUs (Binding Update Explosion ou BU Storm) et à des problèmes de congestion sur les RMs.

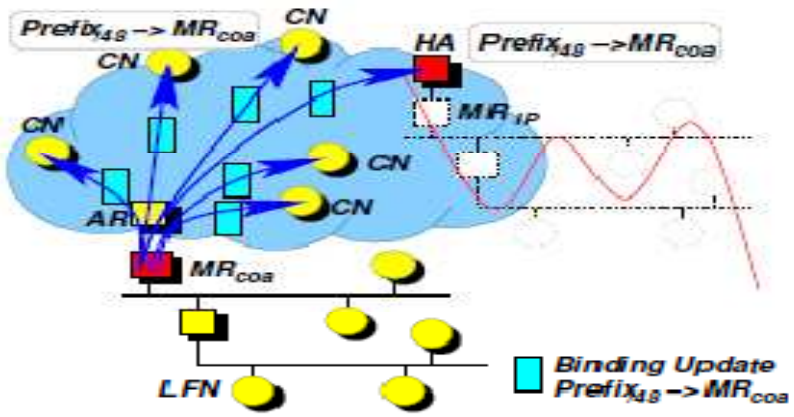


Fig 7: *Explosion des messages de contrôle Binding Updates[3]*

La résolution de ces problèmes et bien d'autres, impose des modifications majeures dans MIPv4 ou v6. C'est pourquoi, l'IETF propose le protocole NEMO, étendu de MIP mais qui intègre des spécifications lui permettant de gérer la mobilité des réseaux. Notons que MIP et NEMO se distinguent essentiellement par le niveau de gestion de la mobilité. En effet, MIP charge les NMs de la gestion de leur mobilité, tandis que NEMO place la complexité de la gestion des déplacements sur Internet au niveau des RMs et n'apporte aucune modification d'adressage aux nœuds intérieurs.

1.3.2. Principe de NEMO

Dans ce qui suit nous présentons les spécifications de NEMO et comment il gère la mobilité des nœuds et des réseaux.

1.3.2.1. Types de nœuds dans les réseaux mobiles

Bien que les MNNs intérieurs à un RM se déplacent comme une seule entité, cependant NEMO en distingue trois types classés selon leur ancrage [2]. La figure 8 illustre ces types de nœuds:

- Les "Local Fixed Node- LFN" sont des hôtes ou des routeurs qui appartiennent au réseau mobile mais qui ne peuvent pas gérer la mobilité. Ils sont incapables de se déplacer sans interrompre leurs communications. Ces équipements sont similaires à des nœuds fixes dans un réseau filaire (ex : un capteur de pression des pneus ou de température).
- Les "Visited Mobile Node- VMN" sont des équipements mobiles (terminal ou routeur) qui sont capables de se déplacer tout en maintenant leurs communications. De tels équipements supportent donc soit le protocole MIPv6 (dans le cas d'un terminal), soit le protocole NEMO Basic Support (dans le cas d'un routeur). Les VMNs considèrent le réseau mobile comme un réseau visité qu'ils utilisent de manière temporaire.
- Les "Local Mobile Node- LMN" sont capables de gérer la mobilité identiquement aux VMNs. Néanmoins, ils appartiennent au réseau mobile car leur adresse possède le préfixe du RM du réseau. Ce cas se présente pour un équipement appartenant aux passagers tel un ordinateur portable.

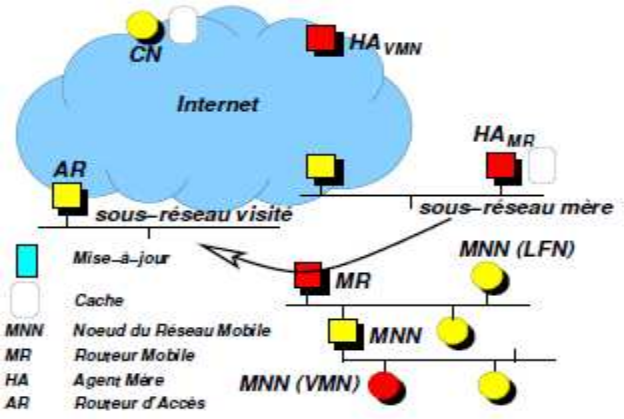


Fig 8 : *Terminologie pour les réseaux mobiles [3]*

1.3.2.2. Adressage et identification des nœuds

L'IETF préconise le même principe d'adressage pour NEMO que celui du standard MIP. Il fait usage de deux adresses pour identifier un routeur mobile et nécessairement un mécanisme pour mettre à jour la relation entre ces deux identifiants.

Par conséquent, Comme pour MIPv6, NEMO BSP opère un adressage dynamique mais qu'il déploie au niveau des routeurs mobiles uniquement. NEMO assigne au RM une adresse permanente, obtenu via le HA du réseau mère et qui identifie le RM quelque soit sa position. Cette adresse identifie à la fois l'interface externe et l'interface interne du RM

[3] : Elle présente un préfixe MNP (Mobile Network Prefix) correspondant au bloc d'adresses assignée au HA. L'adresse de chacun des NMs du même réseau mobile aura ce même MNP. Quant au second type d'adresse du RM, il correspond aux adresses temporaires obtenues selon l'ancrage des RMs dans les réseaux visités de la hiérarchie d'Internet ce qui permet le routage des paquets.

Cet adressage dynamique au niveau des RMs est transparent aux MNNs. Ces nœuds implémentent le protocole IPv6 et ne possèdent pas d'adresse temporaire. Ils préservent leurs adresses avec le préfixe interne du RM puisqu'ils sont toujours ancrés à ce même routeur mobile. Le RM informe ses nœuds de son préfixe MNP grâce à des messages de type Router Advertisement de MIPv6, [2].

Les HAs selon NEMO maintiennent également un cache d'association dont chaque entrée établit la relation entre le préfixe MNP spécifique au RM et son adresse temporaire. De plus, pour mettre à jour son entrée, un RM envoi des messages de mise à jour de préfixes PBUs à son HA dans des paquets contenant l'entête d'extension "Mobility Header" de IPv6. Lorsque le HA reçoit un PBU valide (i.e. obéissant aux tests de conformité liés à la sécurité, particulièrement l'authentification de l'émetteur), il met à jour l'entrée correspondante au MNP du RM avec sa nouvelle adresse temporaire. Après quoi, le HA doit encapsuler tous les paquets à destination d'une adresse ayant un préfixe correspondant au MNP (i.e. l'ensemble des stations résidant dans le réseau mobile) vers la destination effective du RM (i.e. adresse temporaire).

1.3.2.3. Transmission des paquets

NEMO BSP permet le maintien des sessions en procédant à la redirection des paquets destinés aux MNNs vers la position courante de leur RM, comme indiqué sur la figure 9, ci-après.

Lorsque le RM se trouve dans son réseau mère, le problème de transmission ne se pose pas. Les paquets du correspondant indiquent le MNP du RM dans l'adresse de destination ce qui permet de les acheminer vers ce RM dans son réseau mère.

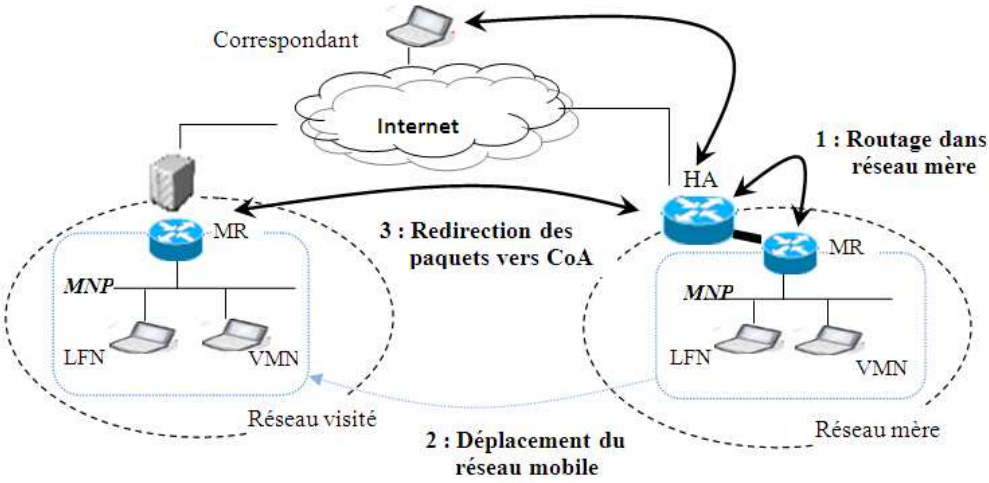


Fig 9 : *Routage des paquets d'un réseau mobile avec NEMO*

Cependant, quand le RM change de réseau, ses MNNs préservent leur adresse avec le même MNP et les correspondants continuent à utiliser ces adresses pour envoyer les paquets aux MNNs. Le HA doit capturer les paquets transmis par les correspondants et dont la destination fait référence au MNP de l'un de ses RMs. Le HA encapsule les paquets en utilisant l'adresse temporaire actuelle du RM, indiquée dans son cache d'association. Il retransmet ensuite les paquets au RM dans sa nouvelle localisation. Le RM décapsule les paquets et les transferts aux nœuds mobiles intérieurs en utilisant leur adresse avec le préfixe permanent MNP.

Dans le sens contraire, lorsqu'un nœud envoi un paquet vers son correspondant, il le route d'abord vers son RM qui encapsule le paquet avec son adresse temporaire. Ce paquet encapsulé est transmis au HA du réseau mobile qui le décapsule et l'envoi au correspondant avec l'adresse identifiant le nœud dans son sous réseau mère (i.e avec le MNP). Ce procédé d'encapsulation au retour permet d'éviter la perte de paquets due aux mécanismes de filtrage implémentée sur les routeurs.

NEMO déploie ainsi un tunnel bidirectionnel mais entre le HA et le RM du réseau mobile uniquement. De plus, il préserve la transparence de la mobilité et de l'adressage aux MNNs.

1.3.3. Discussion

Bien que NEMO BSP permette la gestion de la mobilité groupée, il présente cependant des lacunes héritées de MIP dont il est étendu. Ces problèmes se voient accentuer par les caractéristiques de la mobilité des réseaux.

On note particulièrement, les problèmes de routage inefficace, des longs temps de handover qui engendrent la perte de paquets : En effet, le transit des paquets par le HA dans les deux sens constitue un routage non optimal ce qui augmente les délais de transmission et de handover en particulier lorsque correspondants et MNNs sont proches ou sur le même réseau. De plus, le problème de routage via le HA de tous les paquets, étudié et résolu avec MIPv6, est plus complexe avec NEMO à cause des nombreuses configurations possibles de la mobilité. Dans le cas d'une configuration enchaînée par exemple, vu que les paquets transitent par le HA de chaque routeur ou station mobile de la hiérarchie, le routage des paquets s'effectue sur plusieurs niveaux de mobilité. L'optimisation du routage devient vraiment cruciale lorsque les paquets des VMNs situés dans le même réseau mobile sont dirigés vers le HA du RM du réseau actuel ensuite vers leurs HAs respectifs.

Un autre problème aussi important et auquel NEMO doit faire face concerne la gestion de la multidomiciliation dans le contexte de la mobilité groupée. En effet, avec le support d'interfaces multiples, un RM peut posséder simultanément plusieurs adresses temporaires, mais la spécification de NEMO ne permet d'en enregistrer qu'une seule dans le cache du HA pour un MNP donné. Les solutions destinées pour la gestion des interfaces multiples dans Mobile IPv6, peuvent s'appliquer au cas des réseaux mobiles mais doivent traiter les problèmes liés à la mobilité enchaînée tel que comment découvrir la profondeur du réseau ou comment déterminer le RM qui connecte le réseau agrégé à Internet [3].

Suite aux spécifications du protocole NEMO BSP et à l'étude des problèmes qu'il présente, le groupe de travail doit présenter le "NEMO Extended Support" afin de standardiser des solutions pour l'optimisation du routage. L'IETF discute une autre approche de renumérotation qui consiste à changer les adresses de l'ensemble des nœuds lors des déplacements du réseau de manière à disposer d'une adresse routable topologiquement correcte. Le routage sera ainsi fondamentalement optimisé. En revanche, il faut un mécanisme complémentaire pour ne pas rompre les sessions ouvertes ainsi qu'un mécanisme permettant aux correspondants de déterminer l'adresse courante d'un MNN.

1.3.4. En Conclusion

Le support des réseaux mobiles intéresse de nombreux industriels, allant des fournisseurs d'équipement réseau ou d'électronique grand public jusqu'aux fabricants d'automobiles, en passant par les opérateurs de téléphone et de transport public [7]. Cet intérêt est dû à l'apport de la gestion d'une mobilité groupée. En effet, elle permet de développer l'idée d'un Internet omniprésent, à tout instant, à tout endroit et avec n'importe qui, ce qui implique aussi que toute application doit être en mesure de fonctionner dans un environnement mobile.

Ceci nécessite l'implémentation de mécanismes de changement d'interface et de routeur mobile, et la prise en considération d'un certain nombre de paramètres, tel que le changement de qualité de service ou de bande passante, en fonction des technologies accessibles à un instant donné et du réseau d'accès. De plus, la gestion de la mobilité groupée au niveau IP doit supporter les réseaux mobiles en nombre et en taille importantes ainsi que le nombre élevé de correspondants, en considérant divers types de configurations (un seul sous-réseau, la multidomiciliation, la mobilité enchaînée). Ceci impose de minimiser la quantité de messages de contrôle relatifs à la gestion de la mobilité tout en optimisant le routage.

C'est pourquoi, le déploiement des mécanismes fondamentaux de MIP doit être revu pour permettre un routage optimal tout en considérant la question de la mobilité enchaînée et de la multidomiciliation qui accentuent les difficultés du passage à l'échelle. La question de l'optimisation de routage n'est pour l'instant pas officiellement abordée par le groupe de travail NEMO [3].

2. Gestion de la mobilité au niveau Transport

De nombreux travaux proposent de gérer la mobilité au niveau de la couche transport en alternative à la couche réseau. Les principaux protocoles de transport déployés dans ce contexte sont SCTP [25], DCCP [15] et TCP-MH [8].

SCTP et DCCP sont des protocoles nouvellement développés pour enrichir les fonctionnalités et/ou les services des protocoles UDP et TCP. Les deux sections suivantes présentent ces protocoles ainsi que les mécanismes qu'ils offrent pour la gestion de la mobilité des nœuds.

2.1. Le protocole Datagram Congestion Control Protocol - DCCP

2.1.1. Introduction

Le protocole DCCP [15] est relativement un nouveau protocole de transport de datagrammes défini par l'IETF depuis Mars 2006. DCCP est une alternative à TCP et UDP puisqu'il emploi des mécanismes intermédiaires entre ces deux protocoles. Il permet à la fois un service de transport non fiable comme pour UDP mais dans un mode connecté avec un contrôle de congestion sur le réseau, identiquement à TCP.

L'objectif de la conception de DCCP est de fournir un compromis entre la rigidité de TCP et la simplicité d'UDP pour le transport des données multimédia. En effet, DCCP doit faire face au risque de congestion du réseau (congestion collapse) dû à l'usage exponentiel des streaming, des jeux vidéo et de la téléphonie sur IP. Il doit de plus, permettre aux applications de streaming, qui peuvent tolérer des pertes de paquets, une souplesse des transmissions sans dégrader la qualité des vidéos.

DCCP simplifie l'écriture d'applications et évite les pertes de paquets puisqu'il prend en charge le contrôle de congestion. D'autre part, il est capable de s'adapter aux fluctuations de débit imposées par les mécanismes de contrôle de congestion [2]. DCCP est adapté aux applications qui requièrent des transmissions basées flux, comme pour TCP mais qui ne nécessitent ni la fiabilité ni une livraison de paquets dans l'ordre [8].

Dans ce qui suit, nous présentons d'abord les principaux éléments pour définir une connexion DCCP. De plus amples détails sont fournit dans [14]. Nous explicitons, ensuite, le déploiement du multihoming pour la gestion de la mobilité avec DCCP.

2.1.2. Paquets DCCP

Les paquets DCCP sont constitués d'un entête de taille générique égale à 12 ou 16 octets, suivi de champs fixes et d'éventuels champs options selon le type de paquet ainsi que les données à transporter. La figure suivante illustre le format des paquets DCCP.

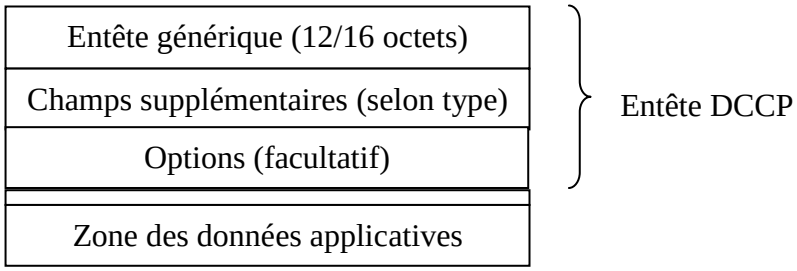


Fig 10 : *Champs d'un paquet DCCP*

2.1.2.1. Entête générique DCCP

L'entête DCCP comporte les champs port source et port destination similaires à ceux de TCP, un identificateur du type de paquet et un champ pour identifier le numéro de séquence du paquet. DCCP utilise un champ particulier qui indique si le numéro de séquence est de forme étendue ou simple "Extended Sequence Numbers bit". L'entête inclut d'autres champs détaillés dans [15].

2.1.2.2. Types de paquets DCCP

DCCP utilise dix types de paquets pour assurer ses fonctionnalités, définit ci-dessus. Les huit premiers types interviennent au cours de l'évolution typique d'une connexion transport, (établissement de connexion, transfert de données et fermeture de connexion) alors que les deux derniers types sont utilisés pour resynchroniser les connexions suites à un grand taux de perte de paquets.

- DCCP-Request : envoyé par le client pour initier une connexion (Première des trois étapes nécessaires pour initialiser une connexion DCCP).
- DCCP-Response : envoyé par le serveur en réponse à un DCCP-Request.
- DCCP-Data : utilisé pour le transport de données.
- DCCP-Ack : utilisé lors d'acquittements simples.
- DCCP-DataAck : utilisé pour transmettre des données et acquitter celles reçues.
- DCCP-CloseReq : utilisé par le serveur pour demander au client de clore la connexion.
- DCCP-Close : utilisé par le client ou le serveur pour clore la connexion.
- DCCP-Reset : utilisé pour terminer la connexion et libérer les ressources.
- DCCP-Sync / DCCP-SyncAck : utilisé pour resynchroniser les numéros de séquence des datagrammes transmis après de nombreuses pertes.

2.1.2.3. Les options et dispositifs de DCCP

Tout type de paquet DCCP peut contenir une ou plusieurs options. Il s'agit de données complémentaires, placées en fin des entêtes des paquets DCCP. Les options sont distinguées par un numéro qui varie de 0 à 225, utilisées selon le type de paquet.

Parmi ces options, celles allant de 32 à 35 : "Change L", "Confirm L", "Change R" et "Confirm R" sont utilisés pour négocier les valeurs des dispositifs fournis par les extrémités d'une connexion DCCP. Les options avec le préfixe "Change" sont utilisées pour initier les négociations, tandis que le préfixe "Confirm" indique l'accord de fin de négociation. Quant aux "L" / "R", ils indiquent que l'option est envoyée par l'extrémité qui déploie le dispositif (Local) ou l'extrémité distante (Remote).

Les dispositifs existants sont également identifiés de 0 à 255 (identification du mécanisme de contrôle de congestion, le bourrage, utilisation d'ECN, utilisation du numéro de séquence court, …).

2.1.2.4. Numéros de séquence et acquittement

DCCP attribue les numéros de séquence aux paquets transmis, qu'ils portent des données utilisateurs ou des informations de contrôle uniquement. DCCP emploi différentes variables de numéro de séquence:

- ISS: (Initial Sequence Number Sent), le premier numéro de séquence envoyé par une extrémité.
- ISR: (Initial Sequence Number Received), premier numéro de séquence reçu de l'autre extrémité.
- Les numéros ISS et ISR, sont déterminés dès l'initialisation d'une connexion DCCP (DCCP-Request et DCCP-Response). Ils sont choisit afin d'éviter deux problèmes : la réception d'anciens paquets et les attaques sur les numéros de séquence [14].
- GSS: (Greatest Sequence Number Sent), le plus grand numéro de séquence envoyé à l'autre extrémité.
- GSR: (Greatest Sequence Number Received), le plus grand numéro de séquence de paquets reçu.
- GAR: (Greatest Acknowledgement Number received), le plus grand numéro d'acquittement reçu de l'autre extrémité (le paquet acquitté, n'étant pas un DCCP-Sync).

- SWL / SWH: (Sequence Number Window low/high), les bornes valides d'une fenêtre de numéros de séquence des paquets reçus.
- AWL / AWH: (Acknowledgement Number Window low / high), les bornes valides d'une fenêtre de numéros de séquence des acquittements reçus.

Comme pour TCP, les numéros de séquences sont utilisés pour détecter les paquets invalides dont le numéro d'acquittement ou de séquence n'appartiennent pas à l'intervalle des numéros de séquence valides.

2.1.3. Etats d'une connexion DCCP

Une connexion DCCP est réalisée entre deux points terminaux selon une architecture de client-serveur. La connexion entre ces deux terminaux se réalise en mode half-connection. Notons que les connexions DCCP sont bidirectionnelles simultanées ; i.e. les paquets peuvent emprunter les deux directions en même temps mais sur des ports différents [15].

L'établissement d'une connexion DCCP est fiable et passe par trois étapes, illustrées sur la figure 11, au cours desquelles, les deux extrémités négocient les options qui vont être utilisées durant la transmission. Les négociations portent, entre autres, sur l'algorithme de contrôle de congestion employé et le mécanisme d'acquittement choisi par les deux extrémités.

Fig 11 : *Procédure d'établissement d'une connexion DCCP, [14]*

Un client étant dans l'état « REQUEST », envoi pas moins d'un paquet DCCP-Request au serveur, chaque 64 secondes et durant un temps limité. Dans le cas ou le client ne reçoit pas de réponse, il abandonne la requête [14]. De plus, chaque requête de connexion DCCP contient un code de service sur 32 bits correspondant aux services et protocoles pour choisir le mécanisme de contrôle de congestion approprié par le serveur.

Une fois que la connexion DCCP est établit, ses deux bouts peuvent commencer la transmission de leurs données. DCCP assure un acquittement systématique des paquets bien reçus, mais ne retransmet pas les paquets perdus.

La fin d'une association peut avoir lieu de différentes manières : la première étant une procédure générique de fin de connexion ou l'une des extrémités émet une requête de fermeture et le correspondant y répond en basculant vers l'état de fermeture. La seconde permet à une extrémité de mettre fin à la connexion directement sans envoyer de requête.

2.1.4. Contrôle de congestion sur DCCP

DCCP permet aux deux semi-connexions entre les extrémités d'utiliser chacune un mécanisme de contrôle de congestion différents. En fait, DCCP permet la diversité des mécanismes de contrôle de congestion que les deux bouts négocient au moyen d'un identifiant appelé Congestion Control Identifier- CCID [4]. Le CCID représente un nombre compris entre 0 et 255 associé à chaque mécanisme de contrôle de congestion. Néanmoins, DCCP n'utilise que les valeurs de CCID égales à 2 et 3 pour respectivement un contrôle de congestion avec TCP-like et avec TCP-Friendly Rate Control.

DCCP prend en considération également l'ECN (Explicit Congestion notification) qui permet au nœud du réseau de signaler explicitement une congestion au protocole de transport. Ceci permet de différencier les types de perte que les paquets peuvent subir (suppression dans un routeur ou corruption des données) [5].

Le contrôle de congestion selon « TCP-like » adopte le même principe que celui de TCP. Il utilise une fenêtre de congestion, un seuil du démarrage lent, …. Le CCID 2 est approprié au flux qui peut tolérer les variations causées par AIMD (Additive-Increase / Multiplicative-Decrease. Les applications utilisent CCID 2 si elles visent l'utilisation d'un maximum de bande passante, dans le cas des jeux en ligne par exemple [13].

Le contrôle de congestion avec CCID 3 ou TFRC-TCP Friendly Rate Control est basé sur un mécanisme de calcul qui détermine le taux auquel les paquets peuvent être envoyés en fonction du taux de pertes reportée par le receveur. L'usage de TFRC génère un flux dont le débit varie relativement lentement. Ainsi, TFRC permet un meilleur lissage du débit ce qui l'adapte bien à un transfert de médias de type streaming. Une variante existe également pour permettre une faible latence sur les petits paquets (p.ex. VoIP) [13].

2.1.5. Gestion de la mobilité avec DCCP

En plus des fonctionnalités de DCCP qui favorisent la transmission des données multimédias, DCCP permet également la gestion de la mobilité des nœuds via le déploiement du multihoming. Néanmoins, la multidomiciliation n'est pas activée par défaut avec DCCP.

La conception de la mobilité avec DCCP spécifie les points suivants :

- DCCP gère la mobilité via le transfert d'un bout de connexion d'une adresse à une autre [8].
- le protocole ne supporte pas les mouvements simultanés des deux extrémités à la fois. Lorsqu'une extrémité est mobile, l'autre extrémité est considérée comme un serveur (nœud stationnaire).
- Avant d'effectuer toute migration d'adresse, les extrémités d'une connexion DCCP doivent négocier l'utilisation des dispositifs de la mobilité.
- Chaque extrémité de connexion qui supporte la mobilité, maintient un identificateur de mobilité, dit Mobility ID sur 128 bits.
- DCCP utilise un autre dispositif dit Mobility Capable Feature-MCF utilisé pour informer le serveur que le client pourrait changer son adresse durant la vie d'une connexion [9].
- Les extrémités partage une clé secrète de mobilité qui doit être modifiée à chaque mouvement.
- Un numéro de séquence relatif à la mobilité augmente durant les mouvements lié au secret de mobilité du paquet.

Le diagramme suivant (figure 12) illustre le flux des paquets DCCP transmis entre les extrémités afin de gérer la mobilité.

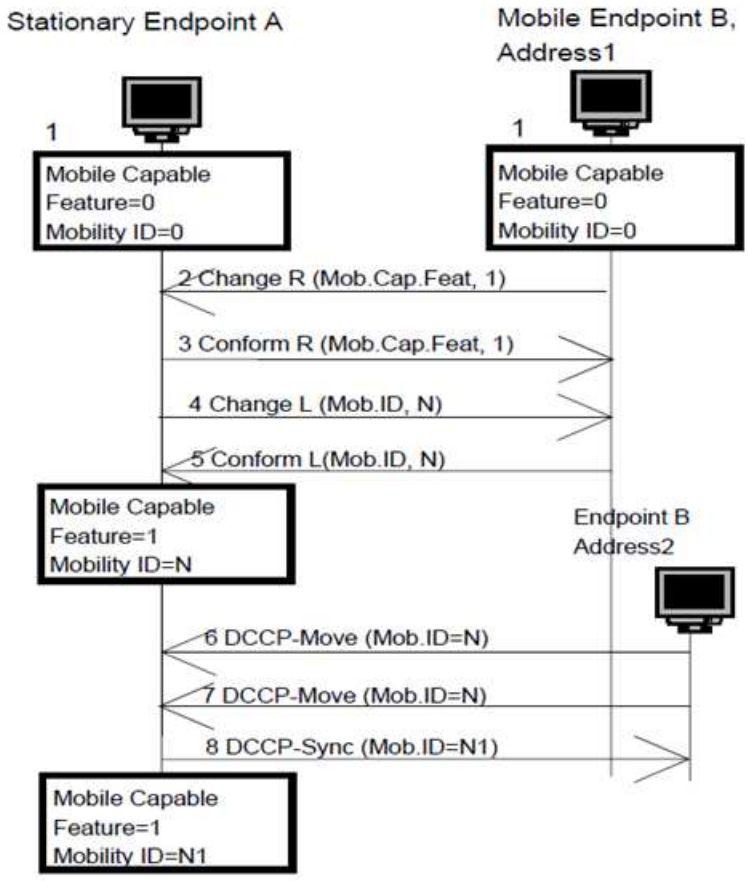


Fig 12 : *Flux de paquets DCCP pour gérer la mobilité, [14]*

1. Avant toute communication, la valeur de l'identificateur de mobilité des deux extrémités est à "0". De même, les dispositifs de mobilité MFC ont une valeur égale à "0" sur les extrémités des connexions.

2. Le mobile, B sur le diagramme, envoi l'option "change R" avec une valeur "1" pour négocier le déploiement du dispositif MFC de mobilité au niveau du correspondant.

3. L'extrémité non mobile confirme la valeur de MFC par envoi de l'option "ConfirmR".

4. Ensuite, le serveur envoi l'option "Change L" signifiant que B négocie les options disponibles à son niveau. Il envoi la valeur de l'identificateur de mobilité que les deux extrémités utilisent pour identifier la connexion en cours. La valeur de cette ID est sélectionnée aléatoirement pour assurer la sécurité. Notons qu'une nouvelle valeur est choisie après chaque mouvement du NM (parfois même plus fréquemment) [12].

5. L'extrémité B confirme la valeur reçue de l'identificateur de mobilité en envoyant l'option "Confirm L".

6. Une fois que le mobile B se déplace vers le nouveau réseau ou change de port, il envoi un paquet "DCCP-Move" contenant la valeur ID de l'identificateur de mobilité précédemment fixé (en étape 4.) pour que A puisse identifier la connexion.

7. Le mobile B doit ré-envoyer le paquet "DCCP-Move" jusqu'à réception d'un paquet "DCCP-Sync".

8. Le serveur A retrouve la nouvelle adresse et port de B à partir de l'adresse source du paquet "DCCP-Move" reçu. L'extrémité A envoi un message "DCCP-Sync" contenant la nouvelle valeur de l'identificateur de mobilité et confirme ainsi le mouvement de B.

Notons que A peut refuser ce mouvement en envoyant l'option "DCCP-Reset". Ceci peut avoir lieu en raison d'une erreur d'adressage, par exemple. Dans ce cas, l'ancienne adresse de B ne peut être utilisée, et doit être réémise via un message DCCP-Move.

2.1.6. Discussion

DCCP offre une nouvelle conception de protocole de transport, intéressante vu qu'elle simplifie l'écriture des applications par la prise en charge du contrôle de congestion et qu'elle introduit moins de variations de délais que TCP.

Cependant, DCCP souffre du même problème que le protocole TCP dans les réseaux sans-fil parce que son contrôle de congestion considère également toute perte de données comme un signe de congestion [6]. Le transfert de connexion avec DCCP ne se fait pas automatiquement vu que le serveur n'obtient la nouvelle adresse du mobile qu'après fin du déplacement et la réception d'un paquet "DCCP-Move". Le serveur peut alors commencer les transmissions en utilisant la nouvelle adresse. De plus, l'implémentation de DCCP est complexe (en-tête variable, options, négociations, …), c'est pourquoi, il est peu déployé à ce jour bien que disponible sur Linux [5].

2.2. Le protocole Stream Control Transmission Protocol- SCTP

2.2.1. Introduction

Le protocole SCTP - Stream Control Transmission Protocol [25], est un protocole de Transport proposé par l'IETF. SCTP se veut l'équivalent de IPv6 au niveau transport et on prévoit qu'à terme il va remplacer les protocoles TCP et UDP [1], [9].

SCTP est un protocole fiable qui opère en mode connecté [4], unicast et orienté session. Lorsqu'une session, dite association selon la terminologie de SCTP, est établie entre deux hôtes, elle est maintenue jusqu'à la transmission effective de toutes les données [9], [8]. SCTP est déployé sur un service paquet sans connexion, offert au niveau Réseau, tel que le protocole IP [4]. C'est pourquoi, SCTP diffère de TCP par une communication basée datagrammes plutôt que par flux d'octets [12].

SCTP fut initialement conçu pour le transfert de signalisation dans un environnement VoIP (Voice over Internet Protocol) [16]. En effet, Les réseaux de télécommunications modernes dépendent fortement de l'échange rapide et fiable de message de contrôle des systèmes de signalisation 7- SS7 déployés sur les réseaux IP. TCP et ses extensions présentent des caractéristiques rigides telles que l'utilisation d'un flux unique ou le séquencement strict des données transmises et qui ne répondent pas aux besoins des systèmes SS7 [5]. SCTP intègre des mécanismes qui en font une alternative à TCP et ses extensions. De plus, comme on peut le constater dans les diverses études faites sur ce protocole, son application ne se limitera pas uniquement au transport de la signalisation sur IP, mais aussi à la gestion de la mobilité et au transport d'applications multimédias [16].

Dans ce qui suit, nous présentons un aperçu sur SCTP et ses mécanismes avant d'étudier la gestion de la mobilité via ce protocole.

2.2.2. Paquet SCTP

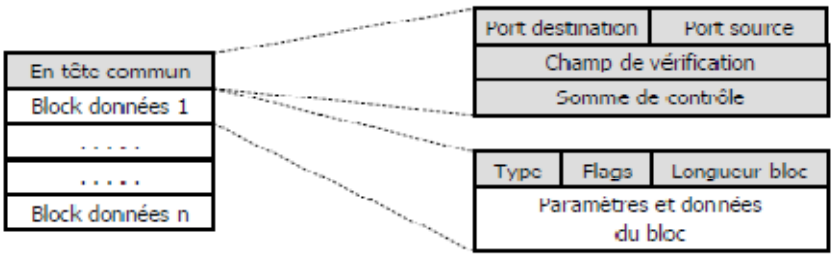


Fig 13 : *Format d'un paquet SCTP*

Contrairement à TCP où un flux est une suite d'octets, un flux SCTP (stream) représente une suite de messages de différentes tailles [12]. Ainsi un paquet SCTP contient en plus d'un en-tête commun, comme indiqué sur la figure 13, une partie donnée constituée d'un nombre de messages qui dépend de la taille maximale permise du paquet.

L'entête commun identifie les ports source et destination de l'association que ce soit des données utilisateurs ou des données de contrôle. Il contient de plus, une étiquette pour vérifier si le message appartient à l'instance courante de l'association ainsi qu'une somme sur 32 bits pour contrôler un transfert sans erreurs des données.

La partie donnée est constituée d'un ou plusieurs Chunks. Un chunk est un ensemble de bits de tailles variables qui définissent soit des données utilisateurs provenant de la couche application ou des informations de contrôle de différents types pour gérer les associations (initialisation, fermeture, test d'état de l'association…). La figure 14 présente les types de chunks de SCTP. Les chunks sont constitués de champs pour identifier leur type, leur taille ainsi que les paramètres associés au type de chunk. Le RFC 2960 [25] liste ces chunks et détaille leurs paramètres.

```
ID Value    Chunk Type
-----       ----------
0           - Payload Data (DATA)
1           - Initiation (INIT)
2           - Initiation Acknowledgement (INIT ACK)
3           - Selective Acknowledgement (SACK)
4           - Heartbeat Request (HEARTBEAT)
5           - Heartbeat Acknowledgement (HEARTBEAT ACK)
6           - Abort (ABORT)
7           - Shutdown (SHUTDOWN)
8           - Shutdown Acknowledgement (SHUTDOWN ACK)
9           - Operation Error (ERROR)
10          - State Cookie (COOKIE ECHO)
11          - Cookie Acknowledgement (COOKIE ACK)
12          - Reserved for Explicit Congestion Notification Echo (ECNE)
13          - Reserved for Congestion Window Reduced (CWR)
14          - Shutdown Complete (SHUTDOWN COMPLETE)
15 to 62    - reserved by IETF
63          - IETF-defined Chunk Extensions
64 to 126   - reserved by IETF
127         - IETF-defined Chunk Extensions
0x80        - Address Configuration Acknowledgment (ASCONF-ACK)
128 to 190  - reserved by IETF
191         - IETF-defined Chunk Extensions
0xC1        - Address Configuration Change Chunk (ASCONF)
192 to 254  - reserved by IETF
255         - IETF-defined Chunk Extensions
```

Fig 14 : *Liste des types de chunks selon le RFC 2960*

2.2.3. Etats d'une association SCTP

2.2.3.1. Etablissement d'une association

Une association dans SCTP s'établi en quatre étapes afin d'éviter le problème du "SYN Attacks" de TCP. De plus, le serveur ne doit pas réserver le moindre espace mémoire sur les extrémités avant que l'association ne soit complètement établie. Les chunks utilisés dans cette procédure sont "INIT Chunk" pour initialiser l'association et son acquittement "INIT-ACK Chunk" ainsi que "COOKIE-ECHO chunk" et son acquittement "COOKIE-ACK chunk" pour sécuriser l'association. Les deux premiers chunks d'initialisation d'une association ne contiennent pas de données application.

2.2.3.2. Transmission des paquets

Contrairement à TCP, une association SCTP peut transmettre ces paquets sur plusieurs flux distingués par des identifiants. L'identifiant du flux et le numéro de séquence des messages à l'intérieur des flux sont inclus dans l'en-tête de chaque chunk. Les messages appartenant au même flux sont livrés en séquence, indépendamment des messages des autres streams. Ainsi, la perte de l'un des messages d'un stream n'affecte pas la livraison des messages sur les autres streams. Grâce à ce mécanisme de multistreaming, SCTP réduit le risque du "head-of-line-blocking" sur l'unique flux de TCP induit par la nécessité d'une livraison par ordre [8]. De plus, le multistreaming permet un service de multiplexage/démultiplexage entre streams. En effet, les données d'une application peuvent être découpées sur plusieurs flux SCTP pouvant chacun remettre les messages à la couche supérieure de manières différentes, alors que TCP devrait établir plusieurs sessions en parallèle pour gérer ce type de transmission. SCTP est donc un protocole d'ordre total optionnel au sein d'un flux et n'offrant aucune garantie sur l'ordre entre les flux. Ce qui lui permet de délivrer les données d'un flux même si des pertes ou des hors-séquences sont détectées sur un autre flux [4].

Pour gérer les transmissions, SCTP utilise deux numéros de séquence différents, le Stream Sequence Number- SSN et le Transmission Sequence Number- TSN :

Le SSN précise le numéro de séquence des chunks sur chaque stream. Il est utilisé optionnellement à la demande de la couche application. En d'autres termes, si l'application requière la réception des chunks de l'autre bout de l'association dans le même ordre de leur

envoi, elle doit activer l'option d'ordre de transmission. Lorsque cette option n'est pas activée pour un chunk, il est livré à l'application correspondante dès son arrivée.

Le TSN, quant à lui, identifie le numéro de séquence de chaque chunk transmis, indépendamment de son numéro de stream. Il est principalement utilisé pour les acquittements sélectifs des chunks, pour la signalisation des duplications, la détection et la retransmission des paquets perdus. Un récepteur acquitte tous les chunks reçu même si des TSNs sont hors-séquence. SCTP sépare ainsi la fiabilité des transmissions de l'ordre de livraison des paquets à la couche application.

La figure suivante définit le format d'un chunk de données utilisateurs de SCTP comme suit :

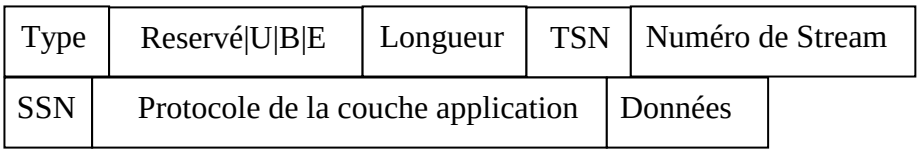


Fig 15 : *Champs d'un Data Chunk de SCTP*

Le champ Reserved|U|B|E identifie essentiellement les informations suivantes :
Reservé : un ensemble de 5 bit mis à 0 et ignoré à la réception du chunk.
U: un bit pour indiquer lorsqu'il est mis à 1 que le chunk ne suit pas un ordre de transmission sur le stream et le récepteur doit ignorer le champ SSN.
B: un bit mis à 1 pour indiquer que c'est le premier fragment d'un chunk.
E: un bit mis à 1 pour indiquer que c'est le dernier fragment d'un chunk.
Le récepteur de ces fragments utilise les TSNs pour regrouper ces fragments dans l'ordre et les remettre à la couche application.

Le champ *Longueur* est sur 16 bits et indique la longueur du chunk en octet.
TSN représente sur 32 bits la valeur de TSN de ce chunk de données.
NS indique sur 16 bits le numéro de stream auquel appartient le chunk en question.
SSN est le champ qui représente le numéro de séquence du chunk sur le stream 'NS'.
La valeur du champ *Protocole de la couche application* sur 32 bits, identifie le protocole utilisé de la couche application. Cette valeur n'est pas utilisée par SCTP mais c'est l'application réceptrice qui en fait usage.

2.2.3.3. Fermeture d'une association

Comme tout protocole de communication fiable, SCTP spécifie la procédure pour mettre fin à une association. SCTP implémente deux méthodes :
Une fermeture similaire à celle de TCP sur trois étapes, activée par les protocoles de la couche supérieure. Cette méthode assure que les queues de transmissions sont vides et que les acquittements sont reçus.
Un abandon d'association : correspond à une association annulée sans se préoccuper de l'acquittement des messages envoyés.

2.2.4. Autres mécanismes de SCTP

SCTP assure, comme pour DCCP ou autre protocole de transport fiable, le contrôle de congestion et le contrôle de flux. Néanmoins, il se distingue par de nouvelles fonctionnalités décrites ci-dessous.

2.2.4.1. Fragmentation/ Regroupement des messages

SCTP offre un mécanisme de fragmentation/réassemblage des messages de l'utilisateur en fonction de l'estimation du MTU- Maximum Transmission Unit d'un chemin. De plus, STCP permet de regrouper plusieurs messages de l'utilisateur dans un seul paquet SCTP. En effet, durant les périodes de congestion, SCTP peut exécuter un empaquetage des paquets même si le niveau application ne le demande pas.

2.2.4.2. Contrôle de flux et de congestion

Par ailleurs, les extrémités des connexions négocient les mécanismes de contrôle de flux et de congestion dès l'établissement de la connexion. Ces mécanismes sont construits sur la base des algorithmes de TCP : le récepteur informe l'émetteur de sa taille de buffer et la taille de la fenêtre de congestion qui est contrôlée au cours de la connexion SCTP.

2.2.4.3. Contrôle d'erreur

Un mécanisme de contrôle d'erreur est implémenté dans SCTP et permet de détecter les pertes, la rupture de séquences, la duplication ou la corruption de paquets. Un schéma de retransmission est utilisé pour corriger ces erreurs. SCTP utilise le principe de Selective

ACKnowledgement- SACK pour la confirmation de la réception des données. Les retransmissions sont faites après expiration d'un timer ou sur interprétation du SACK [4].

2.2.4.4. Multihoming

Cette fonctionnalité est l'une des points forts de SCTP puisqu'elle offre au protocole la capacité de supporter la multidomiciliation des hôtes [9]. Elle permet de définir des associations entre les extrémités en identifiant les ports source et destination, certes mais en tenant compte de plus, des éventuelles multiples adresses IP de chaque bout. Dans le cas d'une association "multihomed", les différentes adresses IP utilisées partagent un port SCTP unique. Un terminal est identifié par [adresse IP1, ... , adresse IPn : port SCTP].

Dans le cas de la figure ci dessous, le nœud A est l'équipement terminal pour deux associations SCTP décrites de la manière suivante: le point terminal desservant l'application 1 est [@IP1, @IP2: Port 1] et celui desservant l'application 2 est [@IP3: Port 2]. Dans l'exemple, l'application 1 du nœud A communique avec l'application 1 du nœud B. L'association établie sera décrite de la manière suivante: Association = {[@IP1, @IP2: Port 1]: [@IP: Port 1]}

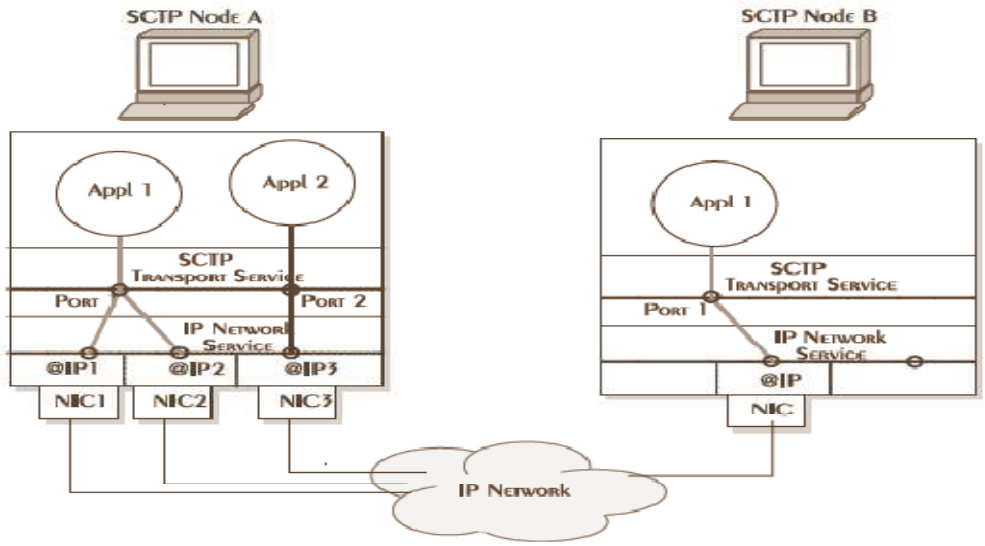


Fig 16 : *Exemple d'une association SCTP utilisant le mécanisme de multihoming, [16]*

En effet, dès l'initialisation des associations, SCTP permet aux deux bouts d'échanger leur liste d'adresses IP. En fait, les Chunks d'initialisation INIT et INIT-ACK définissent les paramètres "IPv4Parameter" et "IPv6Parameter" qui contiennent ces listes d'adresses selon leur type. Dans le cas où une extrémité ne possède qu'une seule adresse IP, c'est le champ source du paquet contenant le chunk INIT qui est considéré comme adresse de l'autre extrémité.

SCTP définit les instances de transmission entre toute extrémité de l'association et l'ensemble des éventuelles adresses de l'autre extrémité [9]. Le protocole utilise les chunks "HERATBEAT Chunk" pour sonder l'état des chemins non utilisés. Lorsqu'une extrémité envoi ce type de chunk, la paire correspondante répond par un chunk d'acquittement "HEARTBEAT-ACK Chunk". A défaut d'acquittements après une émission d'un nombre prédéfini de chunks "HEARTBEAT Chunk", le chemin est considéré inaccessible [9]. Une instance de SCTP peut ainsi confirmer l'état inactif du destinataire lorsqu'elle détecte une absence d'acquittement des chunks DATA émis.

Cependant, même si chaque bout de l'association peut détenir optionnellement une liste d'adresses IPs de ses correspondants [1], l'une de ces adresses est considérée comme adresse primaire et utilisée lors des transmissions tandis les autres adresses sont utilisées dans les retransmissions pour gérer d'éventuelles pertes de paquets ou la défaillance de l'adresse primaire. En effet, Une seule paire d'adresses définit le chemin principal utilisé à un instant donné. Toutes les autres adresses forment des chemins de réserve [16].

C'est pourquoi, même si SCTP supporte le multihoming, il ne peut exécuter une répartition de transmission d'une même application entre les adresses. Le déploiement d'un nouveau chemin de communication ne se fait que lorsque l'ancien chemin n'est plus disponible. Des études telle que [9] proposent une solution ou le chemin primaire n'est pas utilisé durant la vie d'une connexion mais peut être choisi périodiquement selon l'état de congestion du réseau [9].

2.2.5. Mobilité via SCTP

La fonctionnalité de multihoming offerte par SCTP permet d'envisager une solution pour gérer la mobilité, puisqu'elle permet aux deux hôtes d'utiliser plus d'une adresse IP. Néanmoins, les spécifications de SCTP ne répondent pas aux besoins des applications lors de la mobilité des stations ou réseaux. En effet, les deux points terminaux se communiquent les listes d'adresses secondaires lors de l'initialisation des associations avant tout déplacement ou l'obtention même de nouvelles adresses au niveau IP.

Afin d'adapter SCTP à un environnement mobile, une extension de ce protocole permettant la mobilité des nœuds est publiée, baptisée Mobile SCTP ou mSCTP. Cette version supporte la mobilité via le dispositif de multihoming et en offrant aux hôtes le moyen d'attribuer une nouvelle adresse de transmission à une extrémité même en cours de connexion. mSCTP permet un ajout dynamique d'une nouvelle adresse de l'autre bout et de créer ainsi un nouveau chemin de transmission [8].

En effet, mSCTP permet l'ajout d'une nouvelle adresse IP, obtenue lors des déplacements du nœud, à la liste d'adresses détenue par un correspondant même si les transmissions continuent à utiliser l'ancienne adresse. Pour ce faire, mSCTP définit deux nouveaux chunks de contrôle "Address Configuration Change Chunk- ASCONF Chunk" et "Address Configuration Acknowledgment Chunk- ASCONF-ACK Chunk" et utilise, entre autres, les paramètres "AddIP Address" et "DeleteIP Address" [27]. Ces messages permettent d'ajouter, d'enlever ou de changer une adresse dans l'ensemble des adresses IP des deux extrémités de la connexion.

L'idée de base de mSCTP est l'intégration des mécanismes permettant la reconfiguration de la liste d'adresses des extrémités d'une association établie, en la maintenant active. Lorsqu'un nœud mobile se déplace et obtient une nouvelle adresse IP, son mSCTP envoi un chunk "ASCONF" à l'autre extrémité de l'association. Ce chunk informe le correspondant de la nouvelle adresse via le paramètre "AddIPAdress". Dès que le correspondant reçoit ce chunk, il doit rajouter la nouvelle adresse à sa liste d'adresses. Il doit également acquitter cet ajout par l'envoi d'un chunk "ASCONF-ACK" avec comme paramètre le même "AddIPAdress". Néanmoins, le correspondant continue à envoyer les

paquets sur l'ancienne adresse du nœud mobile jusqu'à ce qu'elle ne soit plus utilisée. Le correspondant est informé que l'ancienne adresse IP n'est plus valide via un chunk "ASCONF" avec comme paramètre "DeleteIPAdress". Il doit alors supprimer cette adresse de sa liste et répondre par un chunk "ASCONF-ACK". Dés lors, le correspondant peut immédiatement basculer les transmissions sur la nouvelle adresse enregistrée du nœud mobile. La figure 17 montre l'interaction entre les extrémités SCTP pour assurer une reconfiguration dynamique des adresses.

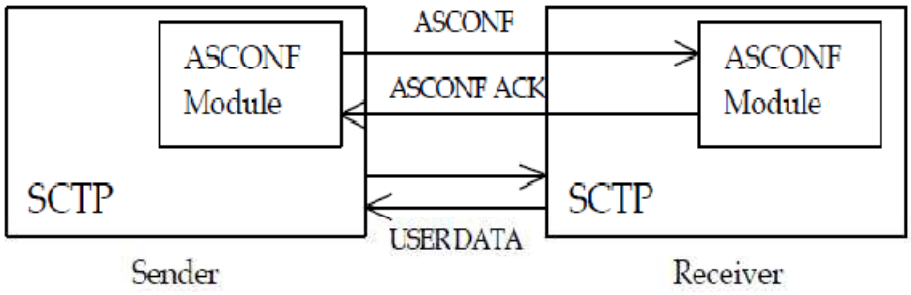


Fig 17 : *Interaction entre instance de mSCTP pour gérer la mobilité*

2.2.6. Discussion

SCTP est une solution Transport intéressante vu les dispositifs qu'elle offre. En effet, outre le multihoming et le multistreaming qui font de SCTP un éventuel remplaçant de TCP et UDP pour les transmissions filaires, son extension mSCTP favorise son déploiement pour gérer la mobilité des nœuds au niveau Transport de TCP/IP.

Le multihoming de SCTP permet d'augmenter la fiabilité des associations au cas où une adresse devient inaccessible. Il fournit une tolérance aux défaillances au niveau Réseau par le support des multiples interfaces sur les extrémités [9]. De plus, en associant les transmissions à partir des streams aux adresses disponibles selon l'état des interfaces, le multihoming peut être utilisé pour fournir des services différents sur les streams.

SCTP offre aux applications un service de transport à ordre partiel. Cependant, le fait que SCTP offre un service totalement fiable entraîne une incompatibilité avec les applications multimédias ayant des contraintes en termes de débit, de délai ou de gigue [4].

3. Etude comparatif des protocoles de gestion de la mobilité

Chacun des protocoles présentés dans ce chapitre, reflète les fonctionnalités de la couche TCP/IP qui l'exploite. Entre autres, la mobilité des nœuds peut être gérée aussi bien sur la couche réseau que sur la couche transport. Notons que jusqu'à aujourd'hui, aucune étude n'identifie l'une ou l'autre des couches TCP/IP comme meilleure couche pour la gestion de la mobilité. Bien au contraire, les mécanismes adoptés par chacun de ces niveaux se complètent pour assurer des handovers transparent aux utilisateurs.

La gestion de la mobilité au niveau réseau, par exemple, vise la transparence de la mobilité sur la couche transport et l'évittement de toute perturbation au niveau application. Ces objectifs sont réalisés en gardant l'adresse IP des nœuds inchangée. Cependant cette gestion est effectuée au prix d'un routage inefficient, la nécessité de configurer et d'intégrer de nouveaux agents (agents mère et agents visités) pour MIP ainsi que des modifications sur les nœuds pour supporter la multidomiciliation pour MIPv6.

Par ailleurs, la gestion de la mobilité au niveau transport via mSCTP ou DCCP s'effectue sans modification du protocole de la couche réseau puisque ces protocoles transport continuent à utiliser IPv4 ou IPv6 et ne requièrent l'intégration de nouvelles entités au réseau. Toutefois, l'utilisation des spécifications de SCTP pour le transport de données tel que le multistreaming nécessite que la couche application puisse en tenir compte.

Notons cependant que les solutions pour gérer la mobilité qui nécessite moins de modifications matérielles aux niveaux des nœuds du réseau, offrant ainsi un déploiement simpliste, sont les plus utilisées. En effet, cette contrainte fait obstacle devant l'émergence du protocole MIP bien qu'il soit identifié depuis plus de 10 ans [24] et favorise le déploiement de DCCP ou mSCTP.

Le tableau suivant présente un résumé comparatif du déploiement de la mobilité sur le niveau réseau ou transport via des protocoles représentant chacune de ces couches : MIPv6, NEMO, SCTP et DCCP.

	MIPv6	**NEMO**	**SCTP**	**DCCP**
Adresse IP des MNs	@permanente et temporaire	@permanente et temporaire sur RM et permanente sur MNN	Plusieurs adresses en même temps	Deux adresses au plus
Routage	Direct	Indirect	Direct sur plusieurs chemins	Direct
Modification des datagrammes	Selon les options de IPv6	Encapsulation	Non	Non
Déroulement de handover	MN ←→ Corresp.	MNN ←→ RM← → HA	MN ←→ Corresp.	MN ←→ Corresp.
Entités additionnelles	HR, partiellement	HA et FA	Non	Non
Avantages	Routage directe entre MN et correspondants	Transparence de la mobilité aux MNNs du réseau mobile ainsi qu'aux correspondants	Gestion du multihoming	Contrôle de congestion
Inconvénients	N'est toujours pas déployé à grande échelle	Routage triangulaire et long temps de handover, nécessite de nouvelles entités	Nécessite définition de nouvelles applications tenant compte de ses spécificités	Le second chemin n'est utilisée qu'après fin du déplacement du MN

Tab 2 : *Comparaison des caractéristiques des protocoles MIPv6, NEMO, SCTP et DCCP*

Bien que DCCP ainsi que mSCTP gèrent la mobilité des nœuds via le mécanisme de multihoming, cependant on peut noter des différents qui favorisent mSCTP.

En effet, DCCP identifie le mécanisme de multihoming pour gérer la mobilité d'un unique nœud sans plus. Il ne supporte pas la mobilité simultanée des extrémités. De même, la sécurité des transmissions n'est assurée que via l'identificateur de mobilité 'Mobility ID'. C'est aux applications qui déploient DCCP de se charger de mieux sécuriser les connexions. DCCP est conçu avec MIP au niveau réseau, c'est pourquoi sa gestion de la mobilité dans le contexte de IPv6 n'est pas intéressante [9].

Quant au mSCTP, il présente certes des lacunes telles que l'utilisation des multiples chemins uniquement pour répondre à un échec de transmission sur le chemin courant (chemins alternatifs et non redondants). De plus il ne permet d'établir que des connexions entre deux extrémités via leurs ports et éventuelles adresses [9]. Cependant, mSCTP permet un support efficace et sécurisé de la mobilité et du multihoming qui encourage son déploiement.

CHAPITRE 3

CQ-mSCTP, Extension de mSCTP Basée QoS

Introduction

Après étude et analyse des différents protocoles déployés pour gérer la mobilité des hôtes, nous consacrons le chapitre suivant à la présentation de notre extension au protocole mSCTP. La première partie de ce chapitre identifie les contextes de déploiement de mSCTP qui suscitent sa modification. Dans la seconde partie, nous décelons les disfonctionnements de SCTP qui empêchent son déploiement selon le cadre spécifié. Cette étude permet de cibler les extensions nécessaires et de définir leur intégration aux mécanismes originels du protocole. La troisième partie présente les améliorations apportées à SCTP et qui constituent l'architecture de notre contribution.

1. Gestion de la mobilité groupée via mSCTP

mSCTP est un nouveau protocole de transport qui opère des smooth handover [40] via le mécanisme de multihoming et la technique de reconfiguration dynamique d'adresses. Ces caractéristiques favorisent le déploiement de mSCTP en tant que protocole de gestion de la mobilité des stations aussi bien que celles des réseaux.

Les recherches aujourd'hui, telles que [32] et [33] proposent l'utilisation de mSCTP pour gérer la mobilité groupée, ce qui revient à implémenter ce protocole au niveau des routeurs mobiles. Deux sortes d'associations SCTP sont alors établies [40], comme indiqué sur la figure 18. La première étant entre les RMs et leurs correspondants, elle peut gérer la mobilité des réseaux. La seconde connecte les RMs aux nœuds intérieurs au réseau.

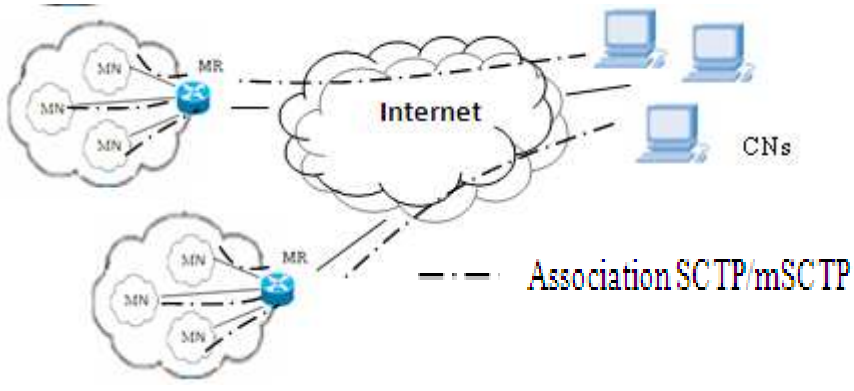


Fig 18 : *Déploiement de mSCTP dans le contexte d'une mobilité groupée*

mSCTP doit alors gérer au niveau transport, les contraintes relatives aux RMs. Il s'agit, en plus de la gestion de la mobilité, de transférer différents types de paquets, provenant de tous les nœuds intérieurs au réseau mobile en destination des correspondants. Le taux de trafic sur les RMs étant ainsi élevé, le risque de perte de paquets augmente. Ces pertes peuvent survenir, suite à une congestion sur les RMs même ou en raison de mauvais ordonnancement des transmissions. En effet, dans le cas où des paquets multimédia arrivent aux RMs suite à des paquets d'autres types moins contraignants mais plus volumineux, les paquets multimédia sont mis en attente et risque d'être perdus en fin de leur durée de vie. mSCTP doit contrôler le flux de données pour éviter des problèmes de congestion et doit supporter un ordonnancement des transmissions selon le type des données.

De plus, pour déployer mSCTP pour la gestion de la mobilité groupée, il doit assurer la qualité de service requise pour transmettre les paquets multimédia. En effet, parmi les intérêts de la gestion d'une mobilité groupée est qu'elle permet de développer l'idée de l'Internet omniprésent [7]. La mobilité des réseaux rend en pratique possible la mobilité de tout équipement, ce qui implique aussi que toute application multimédia devra pouvoir tourner dans un environnement mobile, et prendre en considération un certain nombre de paramètres, en particulier le support de la qualité de service.

Notre contribution vient en réponse au besoin de déployer le protocole mSCTP pour la gestion de la mobilité groupée. Elle consiste en l'extension du protocole par des mécanismes qui favorisent son implémentation sur les RMs. Cependant, ces extensions n'interviennent pas sur les fonctionnalités de mobilité de mSCTP. En effet, les modifications que nous apportons à mSCTP consistent plutôt à le munir de mécanismes lui permettant d'assurer la qualité de service requise pour le transfert des données multimédia. L'objectif de notre contribution étant de supporter le principal intérêt de la gestion d'une mobilité groupée qui est de mettre en œuvre un Internet omniprésent.

Bien que mSCTP soit relativement, un nouveau protocole de transport, sa QoS a déjà fait l'objet d'une multitude d'études. Les solutions proposées agissent sur divers mécanismes de SCTP. Il convient à ce stade de notre travail, d'avoir une vue sur l'avancée des recherches ayant des contextes similaires à ceux de notre contribution. Nous citons entre autres : [28], [29], [30], [34], [41].

[41] propose une extension de SCTP pour supporter le trafic sensible aux contraintes de temps. Ce type de trafic requière des transmissions non fiables qui ne permettent au récepteur de détruire les paquets 'morts' que suite à la réception d'un message d'autorisation. Le problème est que SCTP met en attente tous les chunks qui requièrent l'ordre de transmission jusqu'à envoi et traitement de tous les chunks non ordonnés d'un même stream. Les auteurs proposent une solution pour réduire le temps de la procédure de destruction des chunks de données reçus, dont la durée de vie est dépassée. Les tests et analyses de cette extension montrent une amélioration de la qualité de service en termes de temps de transmission des chunks. Notons cependant que cette proposition opère les transmissions sur un seule stream et n'est pas testée avec le mécanisme de multistreaming de SCTP.

[28] propose une plate forme pour fournir une meilleure qualité de service aux applications multimédia. L'idée de base de cette proposition est de choisir, au niveau de la couche transport, le meilleur chemin parmi ceux créés par les routeurs, pour la transmission de paquets. Pour ce faire, une première procédure sonde et calcul les informations sur les chemins en utilisant les chunks de control 'heartbeat' de SCTP. Cette procédure est effectuée périodiquement ou lorsque des nouvelles données sont soumises à SCTP pour transmission. Elles a pour objectif d'attribuer un état à chaque chemin. La seconde procédure utilise des métriques spécifiées pour associer chacun des paquets des utilisateurs à l'un des chemins évalués. Cette approche permet de répondre à la qualité de service des données utilisateurs, cependant elle doit assurer que les différentes associations SCTP représentent effectivement des chemins de routage différents. De plus, elle nécessite que la couche application identifie le type des données soumises pour transmission SCTP.

[34] est une extension récente qui fournit une fiabilité partielle à la transmission des données avec SCTP. Les services de cette extension se basent sur la définition d'un nouveau chunk « Forward TSN ». Une extrémité de l'association envoi ce chunk à l'autre extrémité pour incrémenter l'indicateur d'acquittement cumulative TSN du récepteur. En d'autres termes, l'émetteur de ce chunk l'utilise pour forcer le récepteur à considérer tous les chunks, ayant un numéro de séquence TSN plus petit que celui indiqué dans le chunk émis, comme étant bien reçu. Cette extension répond aux applications sensibles aux contraintes de temps puisqu'elle permet de générer les paquets à nouveau au lieu de perdre du temps à attendre la notification de leur perte et de continuer à les retransmettre entre temps.

Les auteurs de [29], [30] proposent d'implémenter la qualité de service de SCTP en procédant, entre autres, à l'ordonnancement des transmissions selon la priorité des données. [30] utilise le mécanisme de multistreaming de SCTP pour recevoir les données provenant aussi bien de la couche application que de la couche réseau. Grâce à ce mécanisme, la séparation des données qui présentent les mêmes propriétés est rendue possible. Cette modification de SCTP assure que lorsque les transmissions sont actives à partir de l'un des streams, aucun des streams plus prioritaire ne contient de chunks à transmettre. L'implémentation et les tests de l'architecture proposée montrent qu'elle réduit les temps de transmissions des données multimédia. Cependant, elle présente un risque de perte de paquets sur les streams moins prioritaires.

2. Adaptation de la QoS de mSCTP aux contraintes d'une mobilité groupée

Notre contribution vient en complément aux solutions étudiées pour en améliorer le rendement. Elle est établie avec comme perspective de supporter la QoS de mSCTP dans le contexte d'une mobilité groupée. Nous visons comme pour [28] et [30], d'assurer que les transmissions des chunks de données SCTP s'effectuent par rapport à leur type. Cette approche a pour objectif de répondre aux contraintes et aux besoins des différentes classes de données, en particulier les données multimédia, lors de leur transmission.

En effet, le mécanisme de multistreaming originel de SCTP ne fait aucune distinction entre les classes de données. SCTP reçoit les différentes données générées par la couche application sur ses streams de manière identique. Il effectue ensuite leur transmission sans aucun traitement pour le support de la QoS. Le seul apport lors des transmissions natives de SCTP s'effectue en dissociant l'ordonnancement des transmissions de leur fiabilité pour assurer une livraison des paquets multimédia dès leur réception. Mais cet aspect de transmission relève de la fiabilité du protocole plutôt que de sa QoS.

De plus, les extensions de [28] et [30] pour supporter la QoS des transmissions présentent des lacunes soit au niveau des critères de différentiation des données, soit au niveau de l'algorithme d'ordonnancement de leur transmission. En effet, [28] utilise les mêmes clases de données que celle de l'UMTS (Universal Mobile Telecommunications System). Cependant le mappage de cette classification au niveau de SCTP se fait par rapport

aux caractéristiques des éventuels chemins de transmissions. Hors, les auteurs n'explicitent aucun mécanisme pour assurer que ces chemins sondés au niveau SCTP représentent réellement des chemins différents. De même, [30] propose une classification intéressante des données provenant de la couche application qui repose sur l'utilisation de multiples streams sur SCTP. Cependant l'affectation des TSNs aux chunks prêts à la transmission s'effectue selon un ordre de priorité qui impose de transmettre tous les chunks de haute priorité avant de passer aux moins prioritaires. Cet ordonnancement offre certes de la QoS requise aux données multimédia mais au dépend de la privation des données moins contraignantes. Cet ordonnancement présente un conflit de cohérence des transmissions.

Pour mettre en œuvre notre approche, nous devons en premier, définir les classes de données à traiter. Ce processus est délicat vu qu'il impose de déterminer les critères d'une différenciation sans recouvrement entre les données. Nous devons également préciser, entre autres, à quel niveau s'effectue cette distinction, est ce au niveau applicative ou celui de SCTP. De plus, il est nécessaire d'étudier comment le mécanisme de multistreaming permet de formuler cette distinction au niveau de SCTP.

Une fois que SCTP opère cette classification des données, il doit procéder à leur transmission vers l'autre extrémité. Cette procédure doit, autant que la procédure de classification des données, répondre aux critères de QoS. En d'autres termes, l'algorithme d'ordonnancement des transmissions doit sélectionner les chunks de données à transmettre selon les contraintes de leur type. Ceci impose de déterminer le ou les critères d'ordonnancement qui représentent et répondent à ces contraintes. Il faudra également examiner l'ordonnancement du multistreaming de SCTP, pour assurer un support cohérent de nos modifications sans nuire au fonctionnement natif du protocole.

3. Contribution

Pour développer notre contribution, nous étudions les éléments identifiés dans la section précédente. Cette étude a pour objective de déceler les lacunes de chaque élément et d'y intégrer nos extensions. Elle permet de plus de fixer les différents paramètres pour assurer le fonctionnement de notre extension SCTP selon la qualité de service requise par les données. Suite à cette étude, nous établissons les architectures descriptive et fonctionnelle qui regroupent les éléments de notre proposition et identifient leurs liens.

3.1. Classification des données

3.1.1. Distinction des données sur SCTP

La distinction des données favorise un traitement spécifique à chaque type. La procédure de distinction des différents types de données lors de leur transmission a déjà été soulevée en utilisant les protocoles TCP/UDP. En effet, lorsqu'un nœud A transmet au nœud B des types distincts de données, trois approches sont possibles. La première alternative consiste à ouvrir autant de connexion que de types de données. Cependant cette approche nuit au contrôle de congestion puisqu'elle permet à une application d'augmenter sa bande passante au dépend des autres flux de données sur le réseau [29]. Dans la seconde approche, l'émetteur A transmet les différents types de données sur une unique connexion. Les applications utilisant cette approche doivent assurer un multiplexage/démultiplexage complexe et efficace lors des transmissions. La dernière alternative consiste à transmettre les données multimédia en utilisant UDP. Cependant, elle impose que l'application gère d'elle-même la fiabilité des transmissions.

L'utilisation de SCTP offre une quatrième alternative pour effectuer ce type de transmission. Cette nouvelle possibilité combine les avantages des multiples connexions entre les deux extrémités communicantes et ceux du multiplexage/démultiplexage des différents types de données. En effet, l'utilisation de multiples streams de SCTP permet la distinction des différents types de données dans une même association et assure un contrôle de congestion au niveau de SCTP même, sans impliquer des contrôles complexes au niveau des applications. De plus, aucune application ne consomme un surplus de bande passante au dépend des autres applications.

Ainsi, une extrémité SCTP peut requérir autant de streams que de types de données à transmettre. Chaque stream SCTP, étant un canal de communication logique unidirectionnel, il lui sera attribué un buffer indépendant en émission et un autre buffer en réception sur les deux bouts d'une association SCTP. Ces tampons de streams existeront durant la vie de leur association SCTP. La figure 19 montre un exemple d'une association SCTP entre les extrémités A et B. Le nœud A transmet trois types de données sur trois streams libellés de '0' à '2'. De l'autre bout, le nœud B ne transmet qu'un seul type de données. C'est pourquoi, il n'utilise qu'un seul stream libellé '0'.

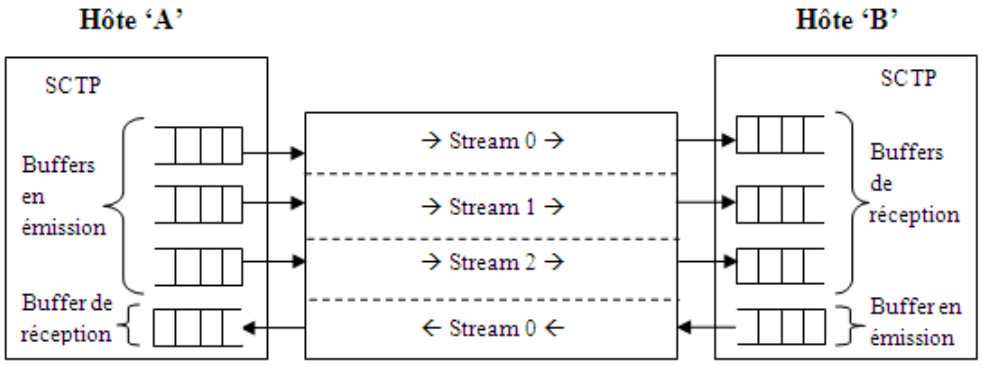


Fig 19 : *Association à multiple flux entre deux extrémités SCTP*

3.1.2. Interface entre multistreaming et le niveau application

Lorsque l'application génère des chunks à soumettre à SCTP, elle doit identifier s'ils sont des chunks de contrôle ou de données utilisateurs. Elle doit indiquer également, leur taille et leur durée de vie. SCTP préconise que l'application doit déterminer de plus, le numéro du stream qui recevra chaque chunk de données ainsi qu'une indication sur le respect de l'ordre de transmission à l'intérieure de ce stream.

Notre extension nécessite que l'application fournisse de plus, une indication sur le type de données transportées par un 'data chunk'. En effet, bien que la distinction des types de données soit utilisée au niveau de SCTP, c'est la couche application qui doit les déterminer. Ceci revient à la définition d'une interface entre la couche application et SCTP pour maintenir cette distinction.

Pour simplifier la mise en place de cette interface et permettre son intégration à SCTP, le protocole de l'application ainsi que SCTP se mettre d'accord dès l'établissement de l'association pour affecter toujours le même type de données au même stream en émission et en réception. La figure 20 indique que lorsque l'application génère des chunks, elle indique dans le champ 'numéro du Stream' un numéro correspondant au type de données du chunk en question. De cette manière, notre extension ne nécessite aucune modification sur le format originel des chunks de données de SCTP. De même, les applications qui utilisent SCTP tiennent compte de son mécanisme de multistreaming. Ainsi, l'association du type de données au stream utilisée ne nécessite pas des implémentations complexes au niveau des applications.

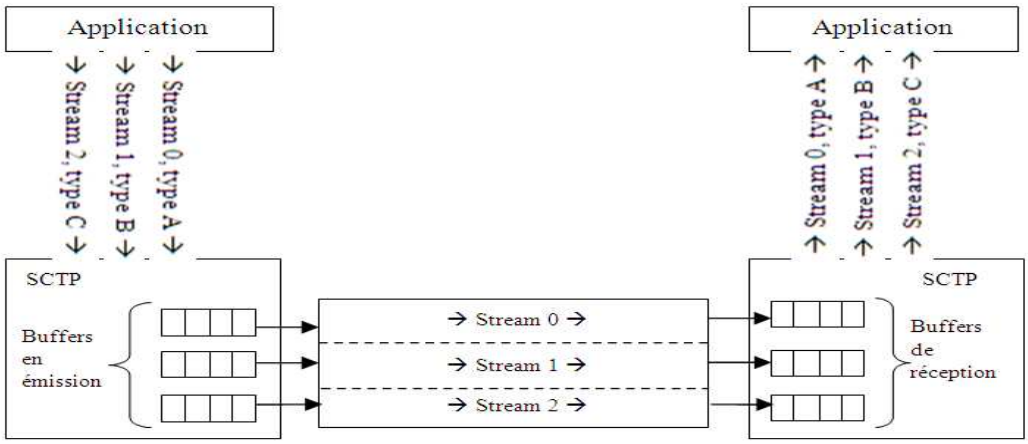


Fig 20 : *Correspondance entre types de données et numéros de stream*

3.1.3. Classes des données

Les applications utilisées aujourd'hui sur Internet vont de celles basées sur l'échange de données essentiellement textuelles, dites applications classiques, aux applications multimédias, impliquant la manipulation coordonnée et le transfert de plusieurs types de média texte, graphisme, audio, vidéo, …etc. L'identification des différents types de données au niveau des applications doit être établie selon des critères efficients pour éviter des recouvrements entre les classes définies et répondre aux caractéristiques et contraintes de chaque classe. Des études antérieures tel que [12], [28] et [40] ont classé ces données selon les services requis par chaque type. Il s'agit essentiellement de leur dépendance du temps : les contraintes de délai de bout en bout pour le transfert des données et la gigue.

[28] spécifie quatre types de flux utilisée également pour les transmissions sur UMTS qui sont : les applications conversationnelles, données de type streaming, la classe interactive et les données textes.

[12], [30], [40] proposent une autre classification qui spécifie cinq types de données. Il s'agit des données audio, vidéo, non temps réel mais dépendant du temps, indépendant du temps et les données textes.

Notre proposition opère selon cette dernière classification, vu que [40] l'utilise également dans le contexte des transmissions sur les réseaux mobiles avec le protocole NEMO. De plus, cette distinction présente mieux les différences en termes de dépendance du temps de diffusion de bout en bout.

En effet, selon [40] et comme le montre le tableau ci-après :

- La première classe requière une présentation du média strictement au fur et à mesure de son arrivé. Elle implique également la possibilité des utilisateurs d'interagir entre eux via ces médias. De plus, ce type ne supporte pas l'irrégularité dans l'arrivée des données. Un exemple type de cette classe est les donnés VoIP (Voice over Internet Protocol) ou la téléphonie sur IP.
- La seconde classe concerne les données qui nécessitent un quasi temps réel de transmission mais une interaction élevée entre utilisateurs tel que le vidéo streaming. Cette classe supporte la perte de certaines informations selon le codage des médias.
- Les réservations on line et la visioconférence sont un exemple des données non temps réel mais dépendant du temps. Cette classe est également caractérisée par l'interaction des utilisateurs.
- Le quatrième type de données n'est pas interactif et est faiblement dépendant du temps. Il nécessite selon [12] de haut débit de transmission. Il s'agit par exemple du web browsing.
- Le dernier type de cette classification ne nécessite aucune contrainte de temps. Il n'est ni interactif ni dépendant du temps tel que les applications FTP.

Type de trafic	Caractéristiques de transmissions	Exemple
Type 1	Strictement temps réel, un patern conversationnel, sensible à la gigue	VoIP
Type 2	Temps réel et interactive	Video Streaming
Type 3	Non temps réel mais dépendant du temps	Réservations on line
Type 4	Faiblement dépendant du temps et non interactive	Web browsing
Type 5	Ni interactive ni temps réel backgrouund	Service FTP

Tab 3 : *Classes des données applicatives selon [40]*

3.2. Ordonnancement des transmissions

Notre extension définit un nouveau principe pour la sélection des chunks à partir d'un même stream et entre streams. L'algorithme d'ordonnancement que nous proposons repose sur le nombre de streams utilisés ainsi que sur les caractéristiques des chunks de données à transmettre. Il doit assurer en définitive un ordre d'affectation des TSNs aux 'Data Chunks' selon les contraintes de leur type.

3.2.1. Algorithme de SCTP

On peut identifier deux types de transmission selon le nombre de streams déployés. Les associations à un seul flux sont similaires à celle de TCP vu que les applications déposent systématiquement leurs chunks sur ce flux. De ce fait, SCTP ne peut traiter ces chunks que selon leur ordre d'arrivée. Une telle gestion ne répond pas à la qualité de service requise au niveau des RMs qui reçoivent en parallèle différents types de données des nœuds intérieurs au réseau.

Lorsqu'une application fait appel à une association SCTP qui déploie plus d'un stream, elle doit préciser le numéro du stream 'NS' pour chaque chunk qu'elle génère. SCTP n'aura qu'à affecter à chaque chunk reçu un numéro de séquence 'SSN' sur son stream. Ce SSN est pris en considération si l'application exige le respect de l'ordre de transmission pour le chunk en question. Autrement, le chunk est livrée à l'application de l'autre extrémité dès sa réception sur le stream SCTP.

Cependant, il est entièrement du ressort de SCTP d'attribuer obligatoirement un numéro TSN aux chunks reçus avant de les transmettre à l'autre extrémité. Dans le cadre de flux multiples, cette opération revient à sélectionner un chunk de l'un des stream et l'insérer en queue d'une séquence de chunk à transmettre. Le RFC 2960 [25] qui constitue une référence du fonctionnement de SCTP ne fournit aucune précision sur le procédé de sélection des chunks à transmettre à partir des streams. Néanmoins, le guide d'implémentation de SCTP [26] vient en complément qui détaille quelques fonctionnalités de SCTP dont l'ordonnancement des transmissions de chunks à partir des streams. [26] recommande que toute éventuelle implémentation de SCTP doit choisir un algorithme d'ordonnancement qui évite l'affaméité des streams. Ce guide suggère deux algorithmes : le Round Robin-RR et le First In First Serve-FIFS.

En utilisant l'algorithme du Round Robin [29] avec trois streams, selon la figure 21, un nœud 'A' transmet un premier chunk du premier stream '0'. Il transmet le second chunk à partir du stream suivant, libellé '1'. Le troisième chunk sera transmis à partir du stream '2'. Ensuite, le nœud reprend les transmissions à partir du stream '0'. En utilisant FIFS, le nœud 'A' transmet les chunks de données dans leur ordre de réception de la couche application sans considérer leur stream.

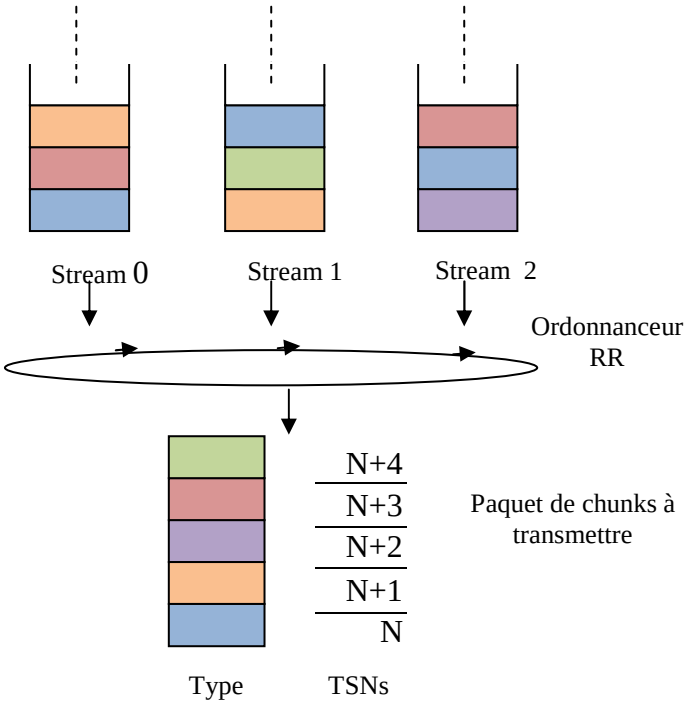


Fig 21 : O*rdonnancement des chunks selon Round Robinson*

L'ordonnancement des transmissions entre multiple flux de SCTP via RR ou FIFS, assure que les transmissions s'effectuent à partir de tous les streams. Ces procédés réduisent certes le temps d'attente des chunks sur les streams ce qui peut minimiser le taux de pertes de chunks causées par de long temps d'attente. Cependant, ces spécifications ne répondent qu'à un objectif unique qui est d'éviter l'affaméité de n'importe quel stream de l'association. En effet, ces spécifications, identiquement à celles des associations SCTP avec un flux unique, ne garantissent aucun traitement propre selon le type de données à transmettre. Hors, pour permettre le support de la qualité de service lors des transmissions de données sur des RMs, il est nécessaire, entre autre, d'assurer un traitement selon le type de données.

3.2.2. Extension de l'ordonnancement

Avant de définir notre algorithme d'ordonnancement, nous rappelons que lors de la classification des données dans la première procédure de notre extension, nous avons définit cinq types (audio, vidéo, non temps réel mais dépendant du temps, indépendant du temps et données textes). Cette distinction repose sur les contraintes de temps de transmission de chaque type. Ainsi, il va de soit d'établir un algorithme de transmission de la deuxième procédure à base des mêmes contraintes.

Notre extension ordonnance les transmissions à partir des différents streams en favorisant les données les plus dépendantes du temps de transmission. Ainsi, les données audio seront transmises en priorité, les données vidéo viennent en seconde position….etc. En d'autres termes, l'algorithme d'ordonnancement accorde la plus haute priorité de transmission aux streams qui portent les données les plus contraignantes, comme indiqué sur la figure 22. C'est pourquoi, la disposition des données sur les streams devra correspondre à leur priorité de transmission. A titre d'exemple, les données audio ne doivent pas être mises sur le stream libellé '2', alors que les données textes sont sur le stream libellé '0'. Une telle attribution des données aux streams ne représente pas correctement l'ordre des transmissions mais implique de plus une difficulté de gestion et d'implémentation tant au niveau application qu'au niveau de SCTP.

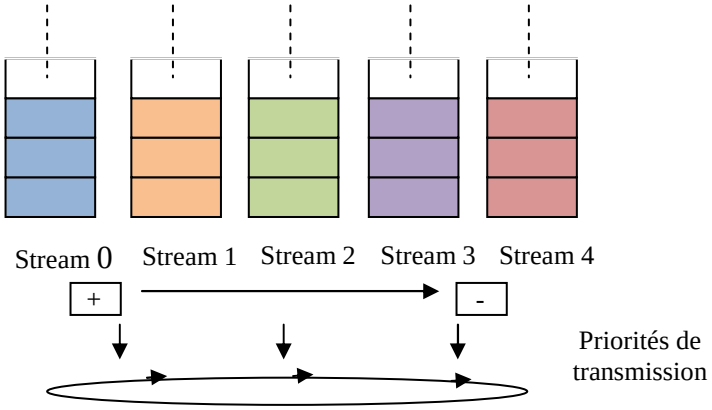


Fig 22 : O*rdonnancement entre streams selon leur priorité*

Notre extension utilise cinq streams libellés de '0' à '4' et qui portent respectivement les données audio, vidéo, non temps réel mais dépendant du temps, indépendant du temps et données textes. L'ordonnanceur passe par ces streams selon l'ordre croissant de leur label.

Cependant, il est insuffisant d'exprimer la priorité des différents types de données via l'ordre de transmission uniquement. En effet, en utilisant l'algorithme RR, l'intérêt de cet ordonnancement n'apparait qu'au premier passage du scheduler à travers les streams vu qu'il commence par les steams les plus prioritaires. Ensuite, lorsque RR effectue plusieurs itérations, les priorités seront confondues et n'auront plus d'intérêts. Ceci est une conséquence directe au traitement effectué par l'algorithme RR qui transmet un chunk par stream et passe au stream suivant, quelque soit le stream en cours. Néanmoins, le déploiement de l'algorithme RR est recommandé vu qu'il assure un passage par tous les streams de l'association.

C'est pourquoi, nous proposons d'utiliser l'algorithme RR en respectant un ordre de service des streams basé sur leur priorité. Mais de plus, l'ordonnanceur accorde un traitement différent selon le stream. Une première alternative consiste à transmettre un nombre variable de chunks à partir des streams selon leur type. Le nombre de chunks transmis à partir des streams audio à chaque itération du RR sera plus grand que tout autre streams. Cette conception simplifie l'implémentation de l'ordonnanceur et privilégie les streams prioritaires mais elle ne garantit pas le traitement des streams selon leurs contraintes temporelles. En effet, si un chunk de type texte est volumineux, sa transmission prendra plus de temps que celle de plusieurs chunks consécutifs de type audio. [30] propose une autre alternative qui suspend toute transmission à partir des streams moins prioritaires jusqu'à transmettre tous les chunks de plus haute priorité. Cette conception risque de priver les streams moins contraignants.

La solution que nous préconisons pour accorder un traitement spécifique par stream se base, comme pour la classification des données, sur les contraintes de temps de transmission de chaque type. Elle consiste à opérer une version préemptive de l'algorithme du Round Robinson. La préemption repose sur le temps de transmission acceptable pour chaque classe de données. La transmission à partir de chaque stream se déroule périodiquement pour une durée égale à la valeur du seuil de transmission- VST du stream. Lorsque cette durée est consommée, l'ordonnanceur arrête les transmissions à partir du stream en cours et c'est le stream suivant qui préempte le droit de transmission même s'il reste des chunks sur le stream actif. Pour déterminer les valeurs des VSTs, nous nous référons à [4] ainsi que [42] qui définissent les contraintes de transmissions des données selon leur type.

[4] présente le tableau suivant (Tab 3) extrait de l'International Telecommunications Union- ITU, 2001 qui identifie les principales applications ainsi que leur contrainte de délai de transmission. Notons que les variations indiquées dépendent du codage de données utilisé.

Type de médias	Applications	Délai
Donnée	Images	De < 15 s À < 60 s
Donnée	Jeux interactifs, Telnet	De < 200 ms À < 250 ms
Donnée	Accès eMail, Web, e-commerce	De < 2 s À < 4 s
Donnée	FTP	De < 15 s À < 60 s
Donnée	Fax	< 30 s/page
Audio	Conversation	De < 150 ms À < 400 ms
Audio	Audio streaming	< 10 s
Vidéo	Visioconférence	De < 150 ms À < 400 ms
Vidéo	Vidéo streaming	< 10 s

Tab 4 : *Principales applications sur Internet et leur paramètre de Délai de transmission*

De même, [42] définit le temps de transmission acceptable pour les données audio et vidéo à des valeurs inférieures à 150 ms. Ceci implique que l'intervalle de temps maximum entre deux passages successifs sur les streams audio et vidéo ne devrait pas excéder 150 ms. Ce qui assure que les chunks sur les streams audio et vidéo n'attendent pas plus de 150 ms pour être transmit. En d'autres termes, lorsque l'ordonnanceur est activé, il doit assurer que le passage sur les chunks moins prioritaires (trois derniers types) ne dépasse pas cette limite de temps. De plus, les VST associées aux streams multimédia ne doivent pas réquisitionner les droits de transmission. Ces valeurs doivent permettre d'effectuer des transmissions à partir des streams moins prioritaires.

Les combinaisons possibles de VST sont innombrables et doivent être soumises à des tests pour validation. Néanmoins, les premières valeurs utilisées sont fixées à respectivement (60, 50, 20, 10, 5) ms pour les streams (0, 1, 2, 3, 4). Ainsi, la durée d'une itération pour revenir au stream Audio et servir le premier chunk en attente sur ce stream est inférieure à 150 ms (égale à 85 ms).

L'algorithme d'ordonnancement ainsi définit, présente deux issues pour chaque stream:

- le VST est supérieur au temps nécessaire pour transmettre tous les chunks présents sur le stream. Dans ce cas, l'ordonnanceur passe directement au stream suivant, avant la fin du VST.
- le VST ne suffit pas pour transmettre tous ces chunks. L'ordonnanceur arrête le processus de transmission à partir du stream actuel et reprend les transmissions à partir du stream moins prioritaire suivant.

3.3. Architecture descriptive du CQ-mSCTP

Notre proposition s'inscrit dans le cadre de l'amélioration des transmissions du protocole mSCTP. L'architecture de notre extension permet d'adapter l'ordonnancement des transmissions de mSCTP selon les types de données, d'où son intitulé "*Custom Queuing based mSCTP"* ou *CQ-mSCTP*. Selon la figure 23, CQ-mSCTP effectue deux procédures : la première consiste à distinguer les données applicatives et les remettre aux streams CQ-mSCTP. Quant à la seconde procédure, elle accomplie l'algorithme d'ordonnancement entre ces types de données.

Les données proviennent à CQ-mSCTP via la couche application ou via la couche réseau. Dans les deux cas, chaque 'Data Chunk' indique le numéro du stream qui correspond à la fois à son type et à sa priorité de service.

Notre extension, CQ-mSCTP utilise cinq flux de données libellés de '0' à '4'. Ces labels référencent le type de données reçu au niveau de chaque stream et leur ordre de transmission. Les données Audio étant plus prioritaires sont mises sur le stream '0', les données vidéo en deuxième priorité sont mises sur le stream '1'…etc.

Il en résulte de cette classification, cinq buffers de chunk en émission de CQ-mSCTP vers la couche inférieure ou en réception de la couche application vers CQ-mSCTP. Ces buffers représentent des entrées à la deuxième procédure de notre extension.

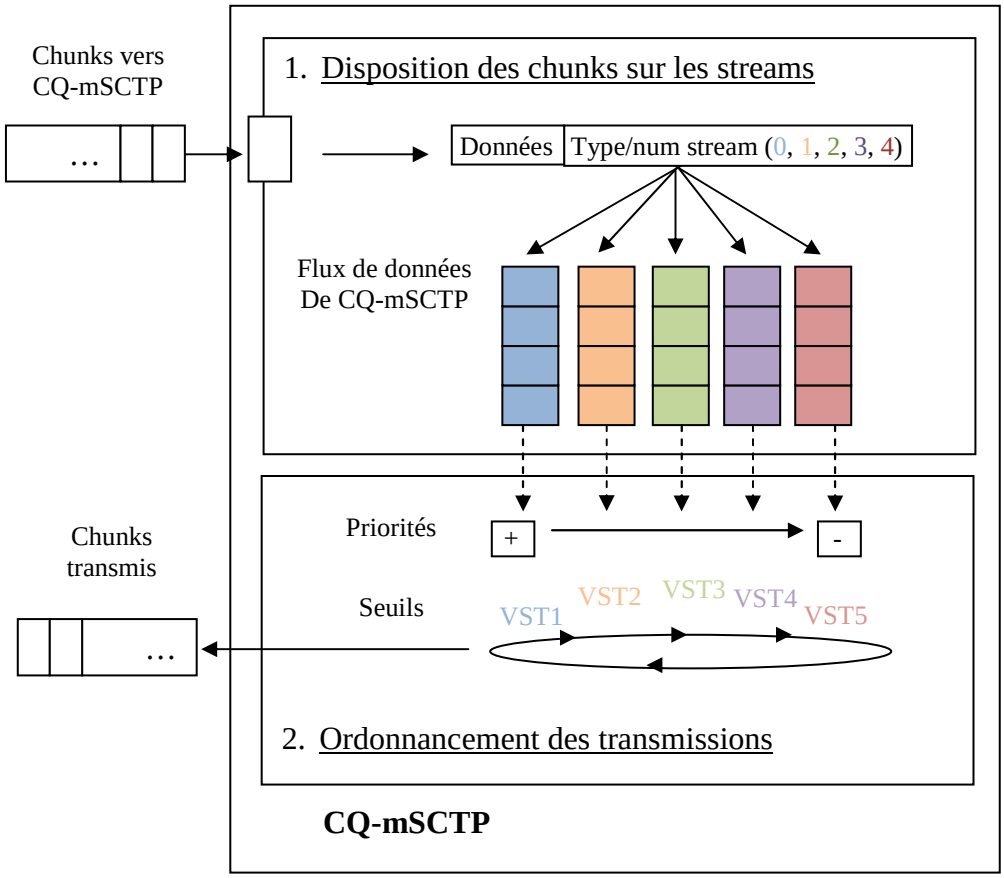


Fig 23 : Schéma de conception du multistreaming de CQ-mSCTP

L'ordonnanceur récupère les chunks en tête de chacun de ces buffers pendant un seuil VST propre à chaque type de buffer. Cette opération s'effectue dans un ordre séquentiel du stream '0' au stream '4'. Lorsque l'ordonnanceur effectue une itération sur les cinq buffers, il reprend les transmissions du stream'0'. Le scheduler peut regrouper ces chunks en paquets et les remettre à la couche inférieure. Il peut également les transférer au fur et à mesure de leur réception à la couche application.

3.4. Architecture fonctionnel

Dans cette section, nous présentons l'intégration de notre extension dans l'accomplissement de handover dans le contexte d'une mobilité groupée.

Etape 1 : établissement de l'association

Les extrémités des associations fixent le nombre de streams et s'échangent les listes d'adresses.

Etape 2 : échange des données

Les nœuds internes envoient leurs paquets en destination de leur RM en utilisant cinq streams pour chaque association. Le RM reçoit cinq flux logiques des différents nœuds intérieurs qu'il transite vers ses correspondants. Ces paquets portent comme adresse source l'adresse du RM correspondant à sa localisation sur Internet.

Etape 3 : Obtention d'une nouvelle adresse

Lorsque le réseau mobile se déplace vers l'intersection de deux zones de couvertures, il déclenche la procédure d'obtention d'une nouvelle adresse sur la deuxième zone. Néanmoins, les transmissions en utilisant cinq steams continue sans modification ni au niveau des routeurs mobiles ni à celui des nœuds intérieurs.

Etape 4 : Ajout de l'adresse obtenu à la liste des correspondants

Après que le RM ait obtenu une nouvelle adresse, il notifie ses correspondants pour qu'ils l'ajoute à leur liste d'adresses. Le RM met en œuvre le mécanisme de configuration dynamique d'adresse de mSCTP qui n'affecte en rien les transmissions des nœuds internes au réseau mobile. De plus, les paquets du/vers le RM indique la même identification de l'association.

Etape 5 : Redirection des paquets

Lorsque le réseau mobile arrive à la limite de la première zone de couverture et qu'il s'enfonce plus dans la deuxième, ses paquets lui sont dirigés en utilisant la nouvelle adresse obtenue. Cependant le processus d'ordonnancement de CQ-mSCTP continue sans être modifié.

Etape 6 : suppression de l'adresse initiale

Cette adresse du RM ne peut être utilisée dans la nouvelle zone de couverture, c'est pourquoi, elle est supprimée de la liste d'adresse des correspondants.

Le diagramme de la figure 24 montre l'enchainement de ces étapes de transmission via CQ-mSCTP.

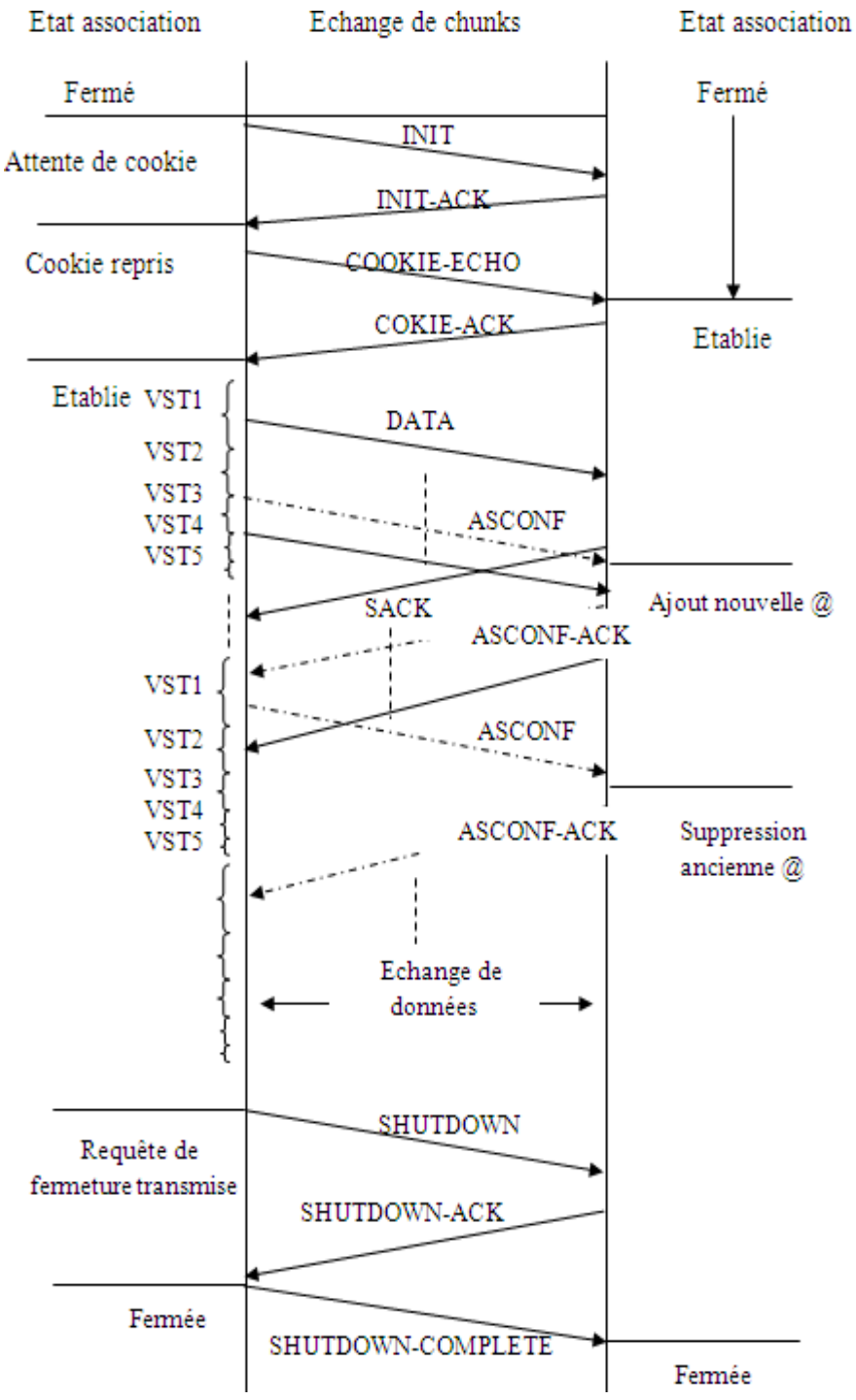


Fig 24 : *Schéma fonctionnel de CQ-mSCTP*

Conclusion

Ce troisième chapitre, décline la suite d'idées qui nous a menés vers notre contribution. Nous avons d'abord présenté le contexte de cette contribution. Ensuite nous nous sommes intéressés de plus près au SCTP via son mécanisme de multistreaming. Nous y avons décelé le problème de transmission soumis à l'un des algorithmes FIFS ou RR sans traitement spécifique par rapport au type de données. Enfin, nous y avons intégrer notre conception d'un ordonnancement basée QoS, le CQ-mSCTP.

Notre contribution consiste à améliorer la QoS des transmissions de mSCTP via l'extension du mécanisme de multistreaming. Nous y apportons un nouveau principe d'ordonnancement basé sur le type de données sur les streams. En effet, nous classons les données sur les streams de CQ-mSCTP selon leurs caractéristiques temporels. Nous définissons un seuil de transmission pour chaque stream selon les contraintes de sa classe de données. Nous procédons ensuite à la transmission des chunks entre stream et à l'intérieure des streams selon les seuils définis.

Cette conception devrait offrir à SCTP différents avantages dont principalement un privilège aux données multimédia en termes de taux de transmission sans négliger les données moins contraignants :

- Offre un ordonnancement dynamique via le chronomètre de l'ordonnanceur.
- Ordonnance les transmissions selon les contraintes des différents types de données.
- Assure la transmission à partir de tous les streams sans exception.
- Accorde des privilèges aux streams selon leur priorité.
- Réduit la perte de paquet due à l'attente de transmission sur les streams.

Toutefois, il est nécessaire de vérifier l'apport de cette contribution. C'est pourquoi, nous procédons dans la suite à la simulation de CQ-mSCTP et à l'analyse des résultats obtenus.

CHAPITRE 4

Simulation et Analyse des Résultats

Après élaboration de l'architecture de CQ-mSCTP, nous spécifions dans le présent chapitre les détailles de son implémentation sous linux-Mandriva avec le simulateur Network Simulator-NS2 dans sa version 2.31. Nous analysons ensuite les résultats des simulations, obtenues à l'aide d'outils spécifiques (NAM, XGraph). Toutefois, pour commencer, nous donnons un bref aperçu du fonctionnement du simulateur NS2.

1. Présentation de Network Simulator- NS2

1.1. Introduction

En parallèle au développement technologique, une panoplie de simulateurs de réseaux ont été développé. Nous citons, entre autres, Network Simulator 2 [43] et [19], OPNET [44] ou GloMoSim [45] ou encore Qualnet [46]. NS2 présente autant d'avantage qui en font le simulateur de réseaux le plus utilisé par la communauté Ad Hoc [20], [11]. Il offre dans son package une implémentation Open Source de SCTP selon le RFC 2960 [25]. C'est pourquoi, nous le déployons pour évaluer les performances de CQ-mSCTP.

NS a été développé, dans sa version 1.0 en 1989 au Laboratoire National de Lawrence Berkeley- LBNL. Depuis 1995, Son évolution en versions (2.xx) fait partie du projet VINT (Virtual InterNetwork Testbed) dirigé par l'université de Californie du sud, financé par le DARPA en collaboration avec Xerox PARC et LBNL [10]. NS est aujourd'hui disponible sur le site de référence, http://www.isi.edu/nsnam/ns [19].

NS fournit un environnement relativement détaillé permettant de réaliser des simulations fiables des protocoles et des liens filaires et sans fil sur l'une ou l'autre des couches du modèle TCP/IP ainsi que leurs interactions. Il permet de plus l'extension de protocoles existants et leur intégration dans d'éventuelles simulations. NS se distingue par le support des techniques de visualisation pour mieux interpréter les résultats. Néanmoins, il n'en est qu'un simulateur qui doit s'approcher d'avantage, sans cesse, des conditions d'un environnement réel pour minimiser le taux des résultats erronés.

1.2. Installation

Pour effectuer nos simulations, nous avons installé le package NS-ALLINONE-2.31 contenant NS en version 2.31 ainsi que les principaux programmes requis sur un système d'exploitation Linux dans sa distribution MANDRIVA 2008.

Les étapes de l'installation sont décrites ci-dessous :

1) Décompresser le package ns-allinone-2.31.tar.gz
2) A partir du répertoire ns-allinone-2.31 qui résulte de la décompression, lancer la console de commandes (ou la touche F4).
3) Lancer la commande *'./install'*. Cette commande spécifie les chemins des fichiers et répertoires de NS selon le *homelogin* de l'utilisateur, i.e : le chemin de décompression de NS. Cependant, NS n'est encore pas utilisable. La figure suivante montre le résultat visuel de l'installation. La console affiche les chemins des bibliothèques utilisées par NS.

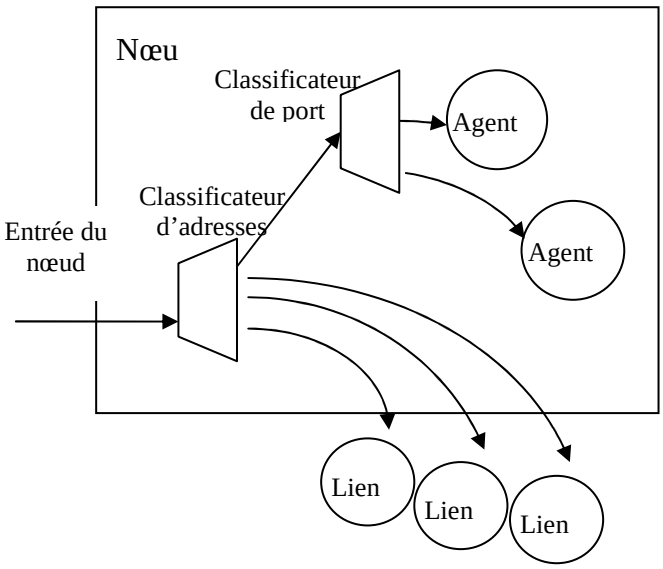


Fig 25 : *Console du résultat après installation de NS*

4) Pour pouvoir utiliser NS, il faut définir ces chemins dans les variables d'environnement de LINUX. Les instructions suivantes permettent d'ouvrir le fichier caché des variables d'environnement pour y définir les chemins des bibliothèques : *'gedit ~/.bashcr'* si on est sur GNOME

ou *'kate ~/.bashcr &'* si on est sur KATE

Après enregistrement du fichier, nous lançons la commande *$ source ~/.bashcr* indispensable pour réinitialiser ces variables.

5) Pour vérifier que NS est accessible sur Linux, la commande '*ns*' doit afficher '%'.
6) A partir du répertoire *'Home/homelogin/ns-allinone-2.31/ns-2.31'*, lancer la dernière commande *'./validate'* qui génère les fichiers *'.o'(object)* de NS et permet ainsi d'exécuter des simulations.

La figure suivante montre l'arborescence des fichiers obtenus après installation. Nous y identifions entre autres, les trois répertoires NS2, NAM, Tclcl [47] :

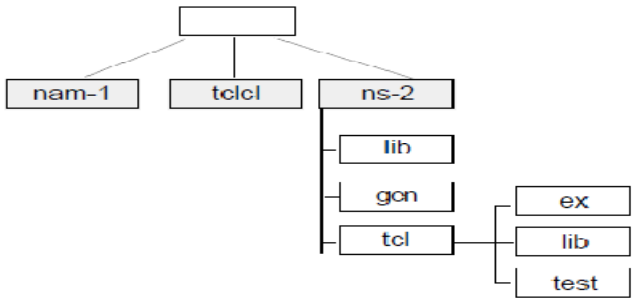


Fig 26: *Arborescence des fichiers de la distribution de NS*

- `ns-2`, ce répertoire contient l'ensemble des fichiers .h et .cc de NS.
- `nam`, l'outil de visualisation des résultats de la simulation: l'animateur réseau.
- `tclcl`, contient les sources du code assurant la liaison entre l'interpréteur et le simulateur. `tcl-object.tcl` est l'un des principaux fichiers de ce répertoire.

1.3. Principe de fonctionnement de NS

1.3.1. Dualité de NS

NS2 est conçu en une approche d'évènements discrets [23] et réalisé en orienté objet avec du code open source. NS est écrit en C++ et muni d'un interpréteur OTcl-Object Tool Control Language. Le package de NS2 inclus une hiérarchie de classes compilée dont les objets sont écrits en C++ et une hiérarchie de classes interprétée, dont les objets sont écrits en OTcl. Ces deux hiérarchies sont étroitement liées comme illustré sur la figure 27. En effet, quand l'utilisateur crée un nouvel objet via l'interpréteur OTcl, un objet correspondant appelé 'objet reflet' est aussi instancié dans la hiérarchie compilée C++.

Ces objets peuvent être accédés aussi bien en OTcl qu'en C++ grâce à la mise en place de structures de données (fichiers et variables) en commun et de procédures d'appel entre OTcl et C++.

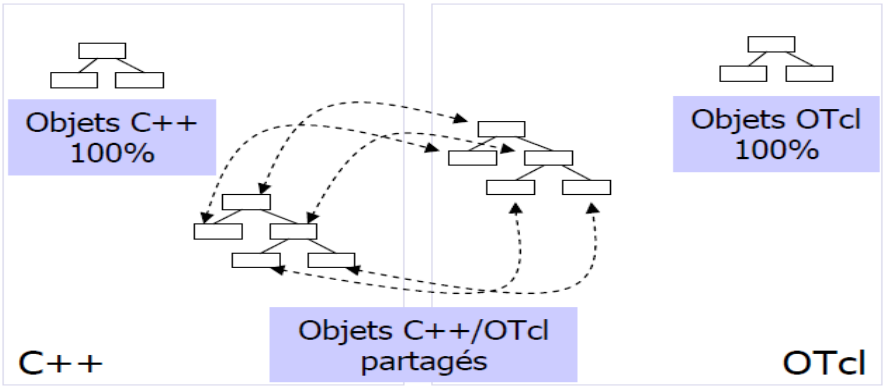


Fig 27 : *Dualité des classes OTCL et C++*

La raison de cette double hiérarchie et de ce double langage, est que d'un coté l'utilisation d'OTcl est simple et conviviale et ne nécessite, de plus, que l'interprétation du code. C'est pourquoi, il est utilisé pour modéliser les éléments de chaque simulation et en modifier les configurations en toute simplicité. De l'autre coté, C++ permet une manipulation précise et efficace des octets, des paquets, et plus généralement des divers algorithmes utilisant de larges bases de données. La programmation avec C++ est plus complexe, moins rapide et nécessite la compilation du code. C'est pourquoi, il est utilisé pour créer les classes de base et pour traiter un grand nombre de données (tel que calcul des tables de routage, mouvement des mobiles…).

L'implémentation du simulateur même, au plus bas niveau, définit six classes (Tcl, TclObject, TclClass, TclCommand, EmbeddedTcl, InstVar) instanciées par les programmes C++ pour utiliser l'interpréteur et définir les principales commandes de haut niveau pour accéder aux variables C++ et OTcl.

La classe OTcl Simulator, par exemple, fournit les interfaces pour créer et gérer la topologie, initialiser le format des paquets et choisir le planificateur d'évènements. Elle stocke intérieurement des références à chaque élément de la topologie. Un script devra toujours commencer par l'instanciation d'une variable de cette classe avant de configurer les éléments de la topologie à simuler. Ces éléments consistent en les nœuds, les liens et les agents ainsi que leurs caractéristiques.

1.3.2. Objets de l'architecture des réseaux NS

- Les nœuds sont des instances de la classe 'Node'. Ils ont pour fonction de recevoir des paquets, de les examiner et de les mapper à leurs interfaces sortantes appropriées. Les classificateurs sont des parties intégrantes aux nœuds où chacun traite l'un des champs des paquets reçus par les nœuds.
- Les liens entre les nœuds sont définis dans la classe 'Link' et 'SimpleLink'. Plusieurs types de liaisons sont supportés, comme le point à point, le broadcast ou les liaisons sans fil pour la mobilité.
- Les agents sont instanciés de la classe 'Agent' et font partie intégrante des nœuds. Leur rôle, comme pour les protocoles de transports, est de générer et de réceptionner des paquets.
- Néanmoins, NS permet de générer des paquets en utilisant des simulateurs d'application existants ou des générateurs de trafics. Ces deux types de générateurs sont décrits dans la classe Application. Ils peuvent être associés aux agents de transport.

Les figures 28 et 29 ci-après montrent les interactions entre ces différents objets de NS.

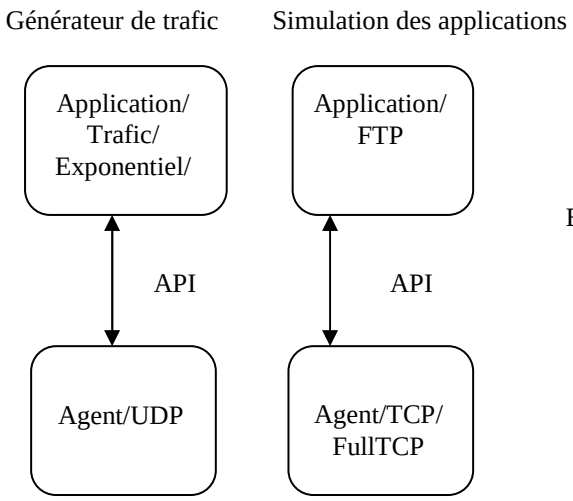


Fig 28: *Exemple d'interaction entre Agents et Application*

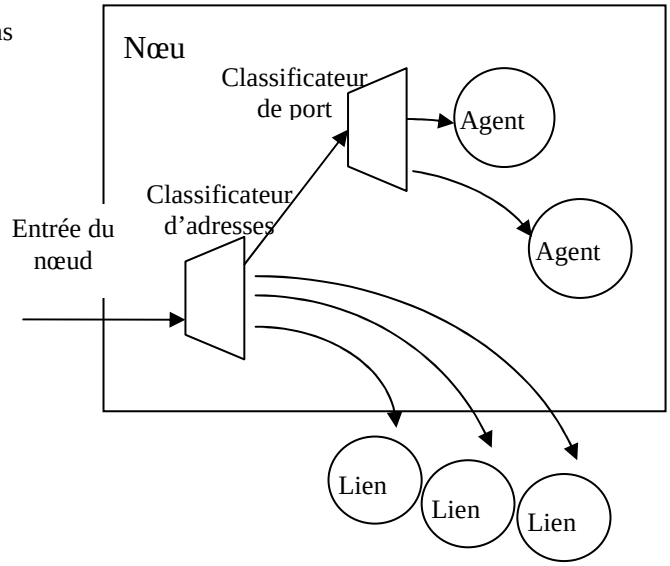


Fig 29: *Eléments d'un nœud filaire de NS*

Pour simuler la mobilité des nœuds, NS définit de nouveaux types de nœuds. En effet, les points d'accès selon la terminologie de NS fournissent l'accès à Internet aux nœuds mobiles et référencent les agents mère et visité. De plus, la classe '*MobileNode*' permet d'instancier des nœuds mobiles et de définir leurs attributs. Les caractéristiques de la mobilité telle que le mouvement des nœuds, les mises à jour de localisation ou les limites de la topologie sont implémentées en C++. Par contre, les composants réseau comme le nœud lui-même (classificateur, caractéristiques des liens,…) sont implémentés en OTcl.

NS permet de définir et d'utiliser d'autres variantes des architectures des réseaux telles que les protocoles de routage, les protocoles de la couche MAC, les modèles de propagation radio, …etc.

1.3.3. Outil de traitement des résultats de simulation

NS présente les résultats des simulations sous forme de fichiers traces d'extension '*.tr*'. Ce type de fichiers rapporte les évènements réseau indexés par le temps sous forme de lignes. Chaque ligne regroupe un nombre important de détails tels que les arrivées/départs/suppression de paquets, rupture de lien et le suivi des états d'un grand nombre de nœuds. La prise en compte de tous ces détails au niveau des utilisateurs complique la compréhension et l'analyse des résultats. Les outils de visualisation et d'analyse répondent à ce problème en permettant à l'utilisateur de prendre en considération les résultats des simulations très rapidement.

NAM Network AniMator est un outil d'animation qui supporte l'affichage de la topologie, l'animation des échanges de paquets et des outils d'observation de données divers. NS permet de faire appel à NAM pour interpréter les fichiers de trace, d'identifier visuellement les modèles de communication et de mieux comprendre les interactions et les causalités [10]. Xgraph, PERL, AWK, Gnuplot sont également des outils complémentaires à NS mais d'un autre type. Ils permettent une interprétation statique des fichiers de trace sous forme de graphes et de diagrammes.

Notons que la visualisation de la mobilité n'a été intégrée au NAM que récemment. On peut distinguer les nouds mobiles qui apparaissent sans liens entre eux, les paquets échangés entre nœuds mobiles sont représentés par des petits points lorsque les mobiles sont à portée l'un de l'autre. Les points d'accès représentant la frontière entre le réseau filaire et le réseau sans fil sont rattachés au réseau filaire par un lien et communiquent avec les nœuds mobiles. L'émission des 'beacons' sur l'interface radio est simulée par l'apparition périodique de cercle autour du point d'accès, indiquant en même temps leur portée. Cependant, NAM présente des lacunes et doit être amélioré pour supporter la mobilité de manière cohérente. En effet, le dernier nœud mobile affiché est systématiquement au même endroit dans la fenêtre, quelque soit la position qu'on indique. Il ne part du bon endroit que lorsqu'il commence à se déplacer. De plus, lorsque les simulations spécifient des nœuds mobiles et fixes, l'affichage ne peut pas être contrôlé totalement : on ne peut pas donner explicitement la position des nœuds filaires dans la fenêtre d'animation. Même les points d'accès, pour lesquels la position joue un rôle important puisqu'elle définit une zone d'accessibilité, ne peuvent pas être affichés au bon endroit [10].

1.4. L'agent SCTP

La version 2.31 de NS inclut les spécifications d'un agent SCTP selon le RFC 2960 [25]. Le *SCTPAgent* traite, entre autres, les opérations suivantes :

L'établissement et fermeture des associations : l'ouverture d'une association passe par les quatre étapes mais *SCTPAgent* n'implémente pas toutes les spécifications relatifs à l'échange de cookie. De même, l'association est fermée en fin du temps de simulation puisque *SCTPAgent* n'implémente pas une fermeture négociée par les deux extrémités.

La transmission des données s'effectue en respect au séquencement des TSNs et en utilisant les mécanismes d'acquittement sélective. *SCTPAgent* détecte l'échec de transmissions sur les chemins et gère les retransmissions des paquets. Il permet également la livraison des chunks dans l'ordre des SSNs ou sans respect de cet ordre. *SCTPAgent* implémente le mécanisme de mulistreaming mais requière que l'application déployée active son utilisation. L'ordonnancement entre streams se fait selon l'algorithme de Round Robinson, comme spécifié dans [26].

Cependant, la plateforme NS ne supporte pas plus d'une interface réseau pour un nœud. C'est pourquoi, le multihoming de SCTP est implémenté logiquement via l'utilisation d'un nœud principale et de plusieurs nœuds interface. La figure 30 montre que le nœud principal est lié aux autres nœuds par des liens unidirectionnels utilisés uniquement pour récolter des informations de routage. La transmission des paquets s'effectue du correspondant vers l'une de ces interfaces directement.

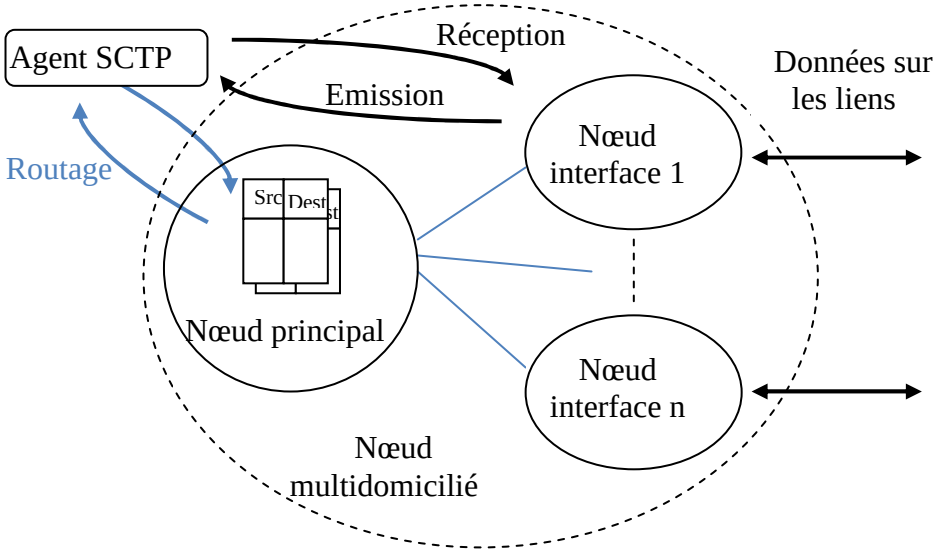


Fig 30 : *Nœud multidomicilié de NS*

2. Implémentation et résultats

La mise en œuvre de notre conception de CQ-mSCTP consiste en premier à intégrer nos modifications au code C++ de NS. Ces extensions s'agissent en la définition d'une application capable de traiter le multistreaming ainsi que des modifications sur l'agent SCTP de NS. Pour assurer un fonctionnement cohérent de ces extensions et pour permettre à l'application générée d'utiliser les services de l'agent CQ-mSCTP, nous effectuons également des modifications sur différents fichiers de control de NS. Nous soumettons, ensuite, les fichiers C++ à une topologie écrite en OTcl pour récupérer des résultats de simulation et les analyser.

2.1. Générateur de trafic

Notre application est une *'SCTP aware application'*, capable de générer des données à transmettre en tenant compte du dispositif de multistreaming de la couche inférieure. Son implémentation identifie la sous-classe '*SCTPApp1*' de la classe '*Application*'. Dans le fichier d'entête '*SCTP_App1.h*', nous avons utilisé des variables '*bindable*' à l'implémentation TCL pour identifier les paramètres requis d'une association SCTP. Il s'agit entre autres du nombre de streams à déployer et de la fiabilité des transmissions. Quant au fichier '*SCTP_App1.cc*', il contient les procédures qui génèrent les chunks à transmettre à des intervalles aléatoires. La structure d'un chunk spécifie sa taille, l'identification du stream correspondant ainsi que sa fiabilité. Notons que cette implémentation ne distingue pas entre les types de données. En effet, nous nous contentons de spécifier le numéro de stream de chaque chunk et de tester les performances par rapport à ces numéros. La couche application transmet ces chunks aux agents SCTP via l'appel de la méthode '*sendmsg()*' avec la structure de données à transmettre comme paramètre :
agent_->sendmsg(agent_->size(), (char*)data_to_transport);

2.2. L'agent CQ-mSCTP

Notre implémentation de l'agent CQ-mSCTP se base sur le code de l'agent SCTP de NS. Les modifications que nous apportons à l'agent SCTP touchent aux structures des fichiers .h et .cc et leurs procédures qu'elles soient implémentées pour émission ou réception des chunks (données de la couche application vers SCTP ou dans le sens contraire). Notons que la procédure '*sendmsg()*' est le déclencheur de toute instance SCTP à partir de la couche Application. De même, c'est la procédure '*recv()*' évoquée au niveau des protocoles réseaux qui déclenche la réception de paquets sur SCTP.

2.2.1. Les structures de données

Parmi les modifications apportées à '*SCTP.h*', nous citons ce qui suit :

i. La structure d'un stream au niveau de la couche SCTP est définie par les champs suivants :

```
STRUCT SCTPInStream_S
{ Le mode du stream (reliable ou non)
  Le prochain SSN sur le stream
  La liste des chunks sur le stream}
```

Nous rajoutons un 4ème champ de type double '*sChunkListWeight*', utilisé pour déterminer le seuil de transmission correspondant au stream.

ii. Nous définissons deux nouvelles sous classes de la classe '*TimerHandler*': '*ChunkListBufferTimer*' et '*StreamBufferTimer*' utilisées selon le sens de transmission pour contrôler le seuil accordé à chaque stream. Ces timers sont utilisés respectivement dans les procédures '*updateallstreams()*' et '*sendmsg()*'. Ces classes sont définies comme suit :

```
class ChunkListBufferTimer : public TimerHandler
{
public:
 ChunkListBufferTimer (SctpAgent *a) : TimerHandler(), opAgent(a)
{ }
protected:
   virtual void expire(Event *);
  SctpAgent *opAgent;
};
class StreamBufferTimer : public TimerHandler
{
public:
 StreamBufferTimer (SctpAgent *a) : TimerHandler(), opAgent(a) { }
protected:
  virtual  void expire(Event *);
  SctpAgent *opAgent;
};
```

iii. Pour stocker les données provenant de la couche application et prêts à être transmis vers les couches basses, nous utilisons la structure : '*sAppLayerBuffe*r' de NS. Ce buffer empile les chunks selon leur ordre d'arrivée, quelques soit leur numéro de stream. Nous définissons de plus, une nouvelle structure de buffer associée à chaque stream : '*sStreamBuffer*'.

iv. Les seuils associés à chaque stream sont des variables de type double '*bindable*', c'est-à-dire que leur valeur initiale est définie dans le fichier '*default.tcl*' contenant les valeurs par défaut des différents agents de NS.

2.2.2. Les méthodes

Nous présentons dans ce qui suit quelques procédures ajoutées ou modifiées du fichier '*SCTP.cc*' :

i. Passtostream()

Cette procédure est définie sur NS pour attribuer un chunk provenant de la couche inférieure à un stream de SCTP. Nous en modifions le code pour initialiser la valeur du champ '*sChunkListWeight*', qui correspond à la durée de transmission accordée selon le type de stream. Cette modification indique de plus, la présence de chunk sur les streams.

ii. *ChunkListBufferTransmission()*

NS définit la procédure '*UpdateAllstreams*' qui transmet les chunks en attente sur tous les streams de SCTP vers la UPL. Elle utilise une boucle qui passe par chaque stream du 1er au dernier et transmet tous les chunks selon l'algorithme de Round Robinson. Notre extension n'utilise plus cette procédure mais nous définissons la procédure '*ChunkListBufferTransmission()*' qui effectue également un parcourt en boucle de tous les streams mais en chronométrant le temps de transmission afin de ne pas dépasser le seuil défini pour chaque stream. '*ChunkListBufferTransmission()*' utilise la méthode '*Resched(t)*' du timer correspondant qui chronomètre le temps de transmissions. Dès que ce temps atteint la valeur 't', elle appelle la fonction '*expire ()*' du timer.

iii. New insert node()/ New delete node ()

Nous définissons la procédure '*NewInsertNode()*' et '*NewDeleteNode()*' qui, respectivement, insert et supprime les paquets en queue/tête d'une liste et établissent un chaînage double. Ces procédures sont utilisées dans '*void StreamBufferTransmission()*'. La première ajoute un paquet dans un buffer de transmission commun aux streams. Quant à la seconde, elle supprime un nœud du buffer de son stream après sa transmission au buffer commun (de SCTP→couche inférieure).

iv. Process init chunk / Process initAck chunk()

Ces procédures sont utilisées lors de l'initialisation de l'association. Elles définissent un tableau dont le nombre de cases correspond au nombre de streams déterminé par l'utilisateur. Chaque case contient les paramètres correspondants au stream tel que le mode du stream, le NSS suivant, la valeur du seuil correspondant à chaque stream.

v. Recv()

Cette procédure est utilisée pour faire passer un paquet de la couche réseau à l'agent SCTP. NS y défini différent comportement selon l'état de l'association lors de l'appel de '*recv()*'. Notre modification est implémentée lorsque l'association est en état établie ce qui signifie que l'instance CQ-mSCTP peut recevoir des chunk. Le temps de réception des chunks est soumis au timer '*ChunkListBufferTimer()*'. '*Recv()*' fait également appel à la procédure '*ProcessDataChunk()*'.

vi. ProcessDataChunk()

'*ProcessDataChunk()*' est une procédure de NS qui définit d'abord les paramètres de transmission tels que Congestion window, rwnd, TSN,... Ensuite, elle fait appel à la procédure '*PassToStream()*' pour mettre un chunk provenant des couches basses sur un stream de SCTP. selon NS, elle fait également appel à '*UpdateAllStreams()*' qui transmet tous les chunks en attente sur les streams de SCTP, vers la UPL selon Round Robinson. Nos modifications remplacent cet appel par l'appel de la procédure '*ChunkListBufferTransmission()*'.

vii. StreamBufferTransmission()

Cette procédure permet de transférer les données de leur stream à un buffer commun en respectant le seuil de chaque stream. '*StreamBufferTransmission()*' comme pour '*ChunkListBufferTransmission()*' utilise la méthode 'Resched(t)' du timer correspondant qui chronomètre le temps de transmissions.

viii. Send msg()

'*Sendmsg()*' implémente deux processus : dépose d'abord le chunk sur le stream correspondant, ensuite elle envois les chunks empilés à la couche inférieure.

Pour effectuer la première procédure, nous utilisons un tableau de tampons '*AppLayerBuffer*' tel que chaque case correspond à un stream. Le code insère chaque chunk provenant de la UPL dans le tampon correspondant à son N° de stream. La deuxième procédure de '*sendmsg()*' n'est déclenchée que si l'association est en état établi. Notre implémentation utilise la procédure '*StreamBufferTransmission()*'. Ensuite, l'appel à la procédure '*sendmuch()*' de NS permet l'envoi des chunks, un par un, à partir du buffer commun vers la couche basse.

Les diagrammes suivants sur les figures 31 et 32 illustrent l'interaction entre les couches application, transport et réseau selon le déploiement de CQ-mSCTP. Le premier Diagramme identifie les séquences d'appel de procédures pour le passage d'un chunk de la couche application vers la couche réseau. Quant au second diagramme, il montre les séquences du passage d'un chunk de la couche réseau vers la couche application.

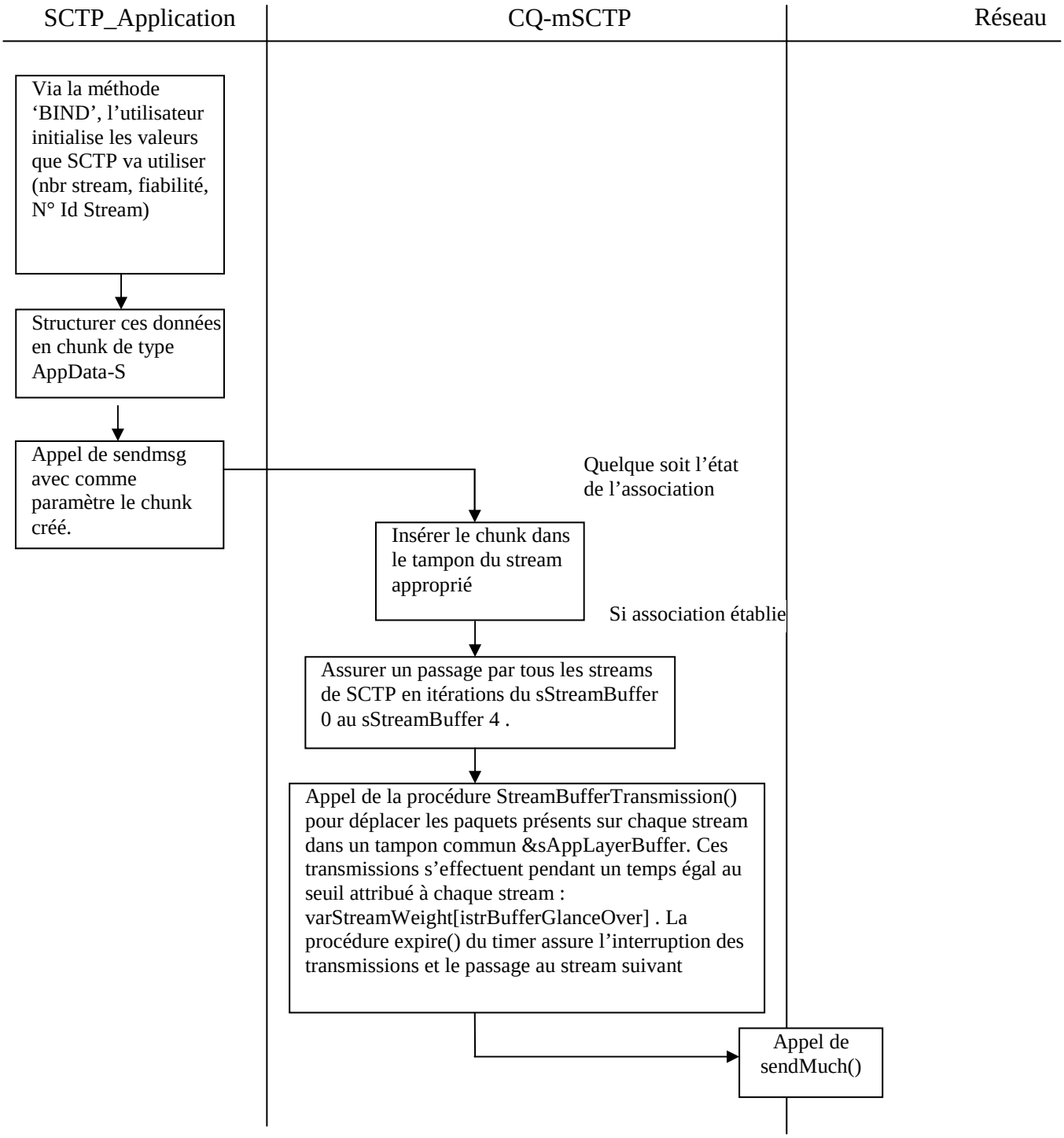


Fig 31 : Diagramme de transmission d'un chunk de la couche application à la couche réseau

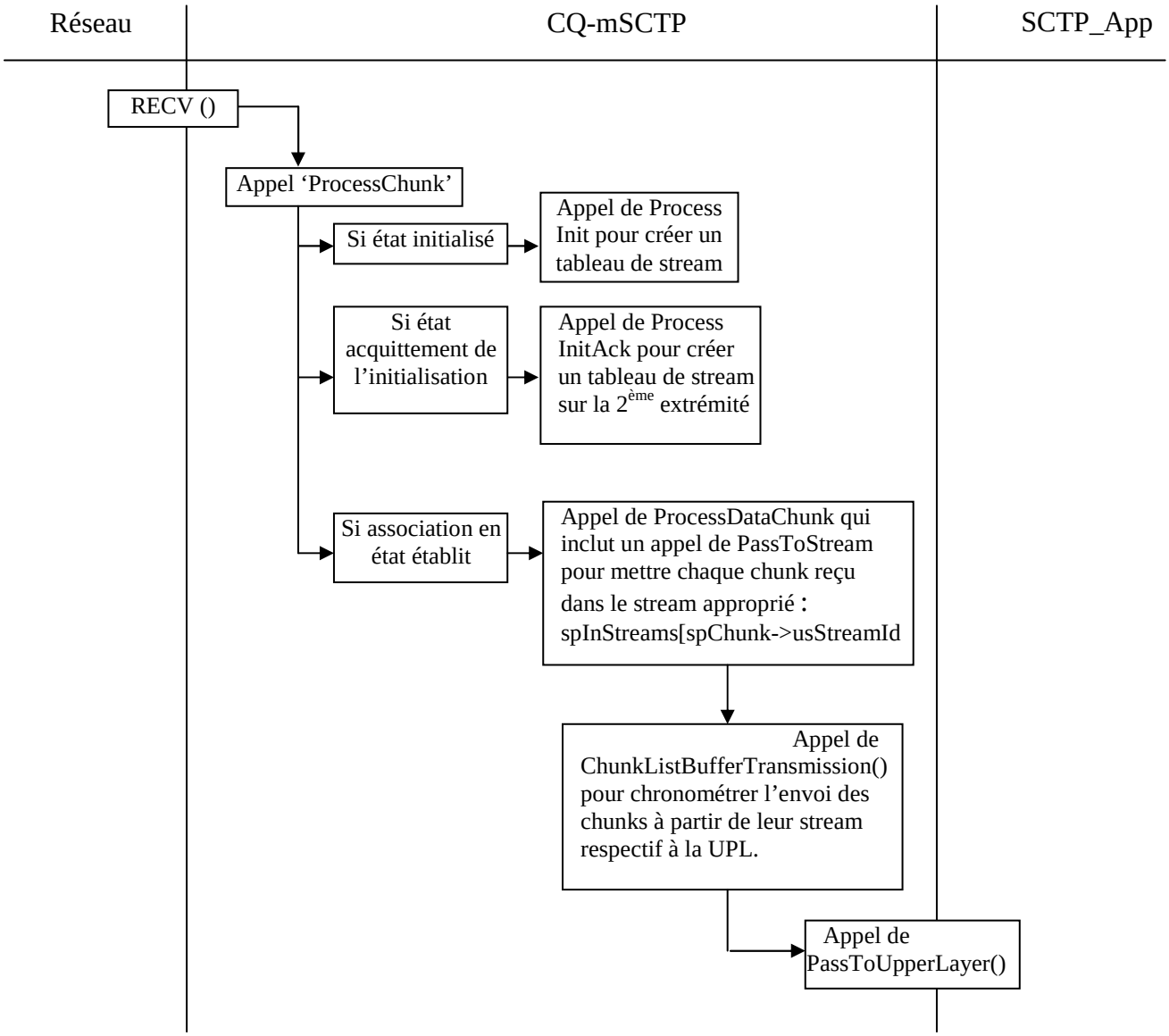


Fig 32 : Diagramme de transmission d'un chunk de la couche réseau à la couche application

2.3. Script TCL

La seconde étape de la mise en œuvre de CQ-mSCTP sous NS consiste à produire un scénario en TCL. Il s'agit de simuler une topologie de réseaux sans fils pour tester les fonctionnalités implémentées en C++. Cette étape offre un moyen simple pour varier les éléments des scénarios et déterminer les meilleurs paramètres de configuration de notre conception.

2.3.1. Topologie de simulation

La figure suivante montre la topologie du réseau, développée pour simuler l'environnement de transmission. Nous y repérons un environnement mixte composé de nœud filaires et sans fils. Il s'agit d'un routeur mobile multidomicilié ayant deux interfaces utilisées alternativement, trois nœuds fixes et deux nœuds mobiles derrière le RM ainsi que deux stations de base. Les mouvements des RMs correspondent un à « ping-pong » entre deux cellules. Notons que cette topologie est largement utilisée pour simuler SCTP dans le cadre de la mobilité des nœuds [32], [33], [30].

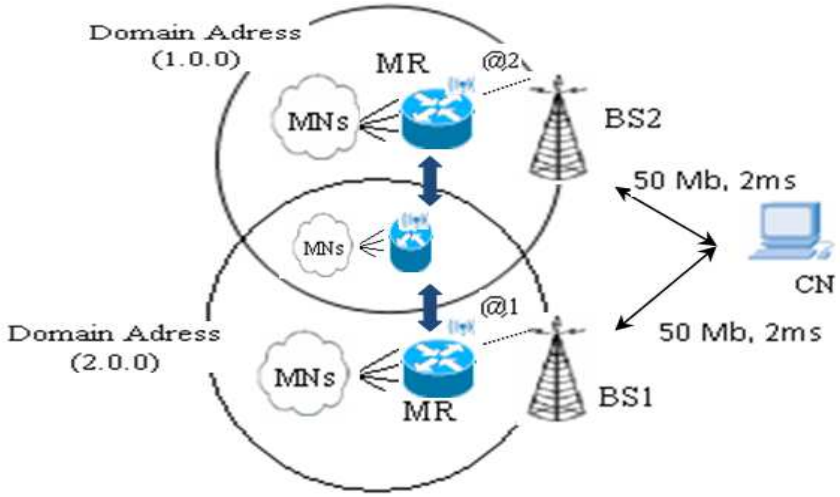


Fig 33 : *Topologie de simulation*

Comme indiquée sur la figure 33, la bande passante des liens filaires entre le correspondant et les stations de base est de 50Mbs, avec un délai de 2 ms. Le réseau sans-fil utilise la norme 802.11b pour les communications sans-fil et a donc une bande passante de 11Mb/s. La taille des paquets est de 500 octets (spécifiée en C++ lors de la génération des paquets sur l'application), le temps de simulation est de 450 secondes quant à la surface de simulation elle est de 800 ×800 m^2.

Les principaux paramètres utilisés pour définir les caractéristiques des nœuds mobiles du modèle de simulation sont listés dans le tableau suivant :

Paramètre	Valeur
Type de propagation radio	Propagation/TwoRayGround
Protocole utilisé par la couche MAC	Mac/802.11
Type de l'antenne	Antenna/OmniAntenna
Type du routage Ad Hoc	DSDV
Type de support physique	Phy/WirelessPhy
Interface de la file d'attente	Queue/DropTail/PriQueue
Type d'adressage	hiérarchique
Activer ou non un agent de trace sur le nœud	On
Portée de transmission	250m

Tab 5 : *Paramètres utilisés pour les nœuds mobiles*

2.3.2. Instances de simulation

Dans le script de simulation, nous avons varié différents paramètres afin de tester notre implémentation sous diverses conditions. Nous citons ci-après ces éléments :

Le schéma de mobilité : les routeurs mobiles changent de position ainsi que leur attachement aux points d'accès. De même, les nœuds mobiles intérieurs aux RMs changent leur point d'ancrage. Nous varions également les vitesses de déplacement des équipements afin de voir l'impact sur la qualité de service offerte par l'agent CQ-mSCTP.

Adressage des nœuds : contrairement à une mobilité effective, le changement de position des nœuds sous NS ne modifie pas leur adressage. C'est pourquoi, nous prenons en charge l'attribution d'adresses aux nœuds pour tester l'implémentation lors du déploiement du mécanisme de multihoming.

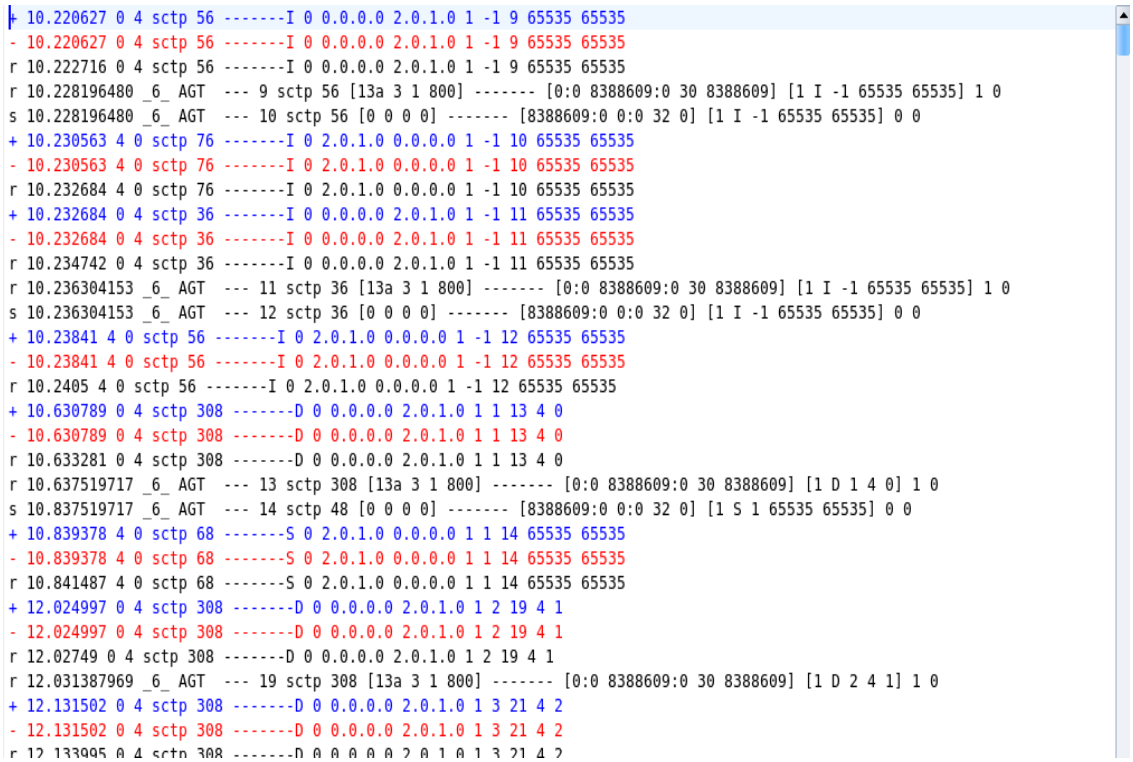
Génération du flux : nous avons implémenté cinq instances d'application '*SCTPApp1*' pour générer les données transmises à CQ-mSCTP. Chaque instance doit déposer les données sur un stream spécifique ce qui permet de distinguer sous NS les types de données (vidéo, audio, …). De plus, l'intervalle de temps entre deux productions consécutives de paquet obéit à une génération aléatoire. Nous pouvons également varier les temps de commencement et arrêt de génération de paquets.

Timers de chaque stream : la variation du seuil de transmission accordé à chaque stream fait partie intégrante de notre conception. Nous avons utilisé différentes valeurs de ce poids en respectant le type de données associé à chaque stream. Les meilleurs résultats sont réalisés dans notre environnement de simulation en utilisant les valeurs (60, 50, 20, 10, 5) ms pour les streams (0, 1, 2, 3, 4). Cependant, même si ces valeurs reflètent notre conception d'un ordonnancement adapté aux types de données, elles n'échappent pas au caractère expérimental et doivent être soumises à des validations dans d'autres environnements de simulation.

2.4. Résultats

Après simulation de CQ_mSCTP, nous récupérons les résultats sur différentes formes. Il s'agit des fichiers traces relatifs à SCTP dont les informations sont synthétisées grâce aux outils Xgraph et awk. Notons que l'utilisation de l'outil NAM ne permet pas de visualiser la mobilité selon notre topologie mais elle permet tout de même de générer des fichiers traces du NAM.

Les figures 34 et 35 correspondent respectivement aux fichiers d'extension '.tr' et'.nam' générés suite à une simulation selon les seuils (60, 50, 20, 10, 5) ms pour les streams (0, 1, 2, 3, 4).



```
+ 10.220627 0 4 sctp 56 -------I 0 0.0.0.0 2.0.1.0 1 -1 9 65535 65535
- 10.220627 0 4 sctp 56 -------I 0 0.0.0.0 2.0.1.0 1 -1 9 65535 65535
r 10.222716 0 4 sctp 56 -------I 0 0.0.0.0 2.0.1.0 1 -1 9 65535 65535
r 10.228196480 _6_ AGT  --- 9 sctp 56 [13a 3 1 800] ------- [0:0 8388609:0 30 8388609] [1 I -1 65535 65535] 1 0
s 10.228196480 _6_ AGT  --- 10 sctp 56 [0 0 0 0] ------- [8388609:0 0:0 32 0] [1 I -1 65535 65535] 0 0
+ 10.230563 4 0 sctp 76 -------I 0 2.0.1.0 0.0.0.0 1 -1 10 65535 65535
- 10.230563 4 0 sctp 76 -------I 0 2.0.1.0 0.0.0.0 1 -1 10 65535 65535
r 10.232684 4 0 sctp 76 -------I 0 2.0.1.0 0.0.0.0 1 -1 10 65535 65535
+ 10.232684 0 4 sctp 36 -------I 0 0.0.0.0 2.0.1.0 1 -1 11 65535 65535
- 10.232684 0 4 sctp 36 -------I 0 0.0.0.0 2.0.1.0 1 -1 11 65535 65535
r 10.234742 0 4 sctp 36 -------I 0 0.0.0.0 2.0.1.0 1 -1 11 65535 65535
r 10.236304153 _6_ AGT  --- 11 sctp 36 [13a 3 1 800] ------- [0:0 8388609:0 30 8388609] [1 I -1 65535 65535] 1 0
s 10.236304153 _6_ AGT  --- 12 sctp 36 [0 0 0 0] ------- [8388609:0 0:0 32 0] [1 I -1 65535 65535] 0 0
+ 10.23841 4 0 sctp 56 -------I 0 2.0.1.0 0.0.0.0 1 -1 12 65535 65535
- 10.23841 4 0 sctp 56 -------I 0 2.0.1.0 0.0.0.0 1 -1 12 65535 65535
r 10.2405 4 0 sctp 56 -------I 0 2.0.1.0 0.0.0.0 1 -1 12 65535 65535
+ 10.630789 0 4 sctp 308 -------D 0 0.0.0.0 2.0.1.0 1 1 13 4 0
- 10.630789 0 4 sctp 308 -------D 0 0.0.0.0 2.0.1.0 1 1 13 4 0
r 10.633281 0 4 sctp 308 -------D 0 0.0.0.0 2.0.1.0 1 1 13 4 0
r 10.637519717 _6_ AGT  --- 13 sctp 308 [13a 3 1 800] ------- [0:0 8388609:0 30 8388609] [1 D 1 4 0] 1 0
s 10.837519717 _6_ AGT  --- 14 sctp 48 [0 0 0 0] ------- [8388609:0 0:0 32 0] [1 S 1 65535 65535] 0 0
+ 10.839378 4 0 sctp 68 -------S 0 2.0.1.0 0.0.0.0 1 1 14 65535 65535
- 10.839378 4 0 sctp 68 -------S 0 2.0.1.0 0.0.0.0 1 1 14 65535 65535
r 10.841487 4 0 sctp 68 -------S 0 2.0.1.0 0.0.0.0 1 1 14 65535 65535
+ 12.024997 0 4 sctp 308 -------D 0 0.0.0.0 2.0.1.0 1 2 19 4 1
- 12.024997 0 4 sctp 308 -------D 0 0.0.0.0 2.0.1.0 1 2 19 4 1
r 12.02749 0 4 sctp 308 -------D 0 0.0.0.0 2.0.1.0 1 2 19 4 1
r 12.031387969 _6_ AGT  --- 19 sctp 308 [13a 3 1 800] ------- [0:0 8388609:0 30 8388609] [1 D 2 4 1] 1 0
+ 12.131502 0 4 sctp 308 -------D 0 0.0.0.0 2.0.1.0 1 3 21 4 2
- 12.131502 0 4 sctp 308 -------D 0 0.0.0.0 2.0.1.0 1 3 21 4 2
r 12.133995 0 4 sctp 308 -------D 0 0.0.0.0 2.0.1.0 1 3 21 4 2
```

Fig 34 : *Résultat de simulation selon le format du fichier trace de SCTP*

Les détails de transmission des chunks CQ-mSCTP sur le fichier trace correspondent, comme le montre la figure 34 aux champs suivant : le type d'évènement (r/reçu , +/arrivée , -/départ , s/émission, d/suppression du chunk) et son temps d'occurrence sont réprésentés sur les 1er et 2ème champs. Le type de paquets (SCTP), taille des paquets, type de chunk (D, I, S/Sack, H) correspondent respectivement aux champs (5, 6, 7). Quant aux chapms 12, 14 et 15, ils représentent respectivement le numéro du TSN ou du SACK, l'identificateur du stream et le SSN. Comme les chunks de contrôle ne possèdent pas de TSN ni de SSN, ces champs sont identifiés par des (-1) ou (65535).

```
h -t * -a 1.0.1  -s 5 -x 50 -y 50 -Z 0  -z 20 -v circle -c black
n -t * -a 2.0.1  -s 6 -x 50 -y 50 -Z 0  -z 20 -v circle -c black
n -t * -a 1.0.2  -s 7 -x 50 -y 50 -Z 0  -z 20 -v circle -c black
V -t * -v 1.0a5 -a 0
W -t * -x 800 -y 800
A -t * -n 3 -p 0 -o 0xffffffff -c 31 -a 1
A -t * -h 1 -m 1023 -s 22
A -t * -h 2 -m 2047 -s 11
A -t * -h 3 -m 2047 -s 0
n -t * -a 2.0.0 -s 4 -S UP -v circle -c RED -i RED -x 150 -y 250 -Z 0
n -t * -a 0.0.0 -s 0 -S UP -v circle -c GREEN -i GREEN -x 300 -y 450 -Z 0
n -t * -a 0.1.0 -s 1 -S UP -v circle -c BROWN -i BROWN -x 400 -y 550 -Z 0
n -t * -a 0.2.0 -s 2 -S UP -v circle -c yellow -i yellow -x 200 -y 450 -Z 0
n -t * -a 1.0.0 -s 3 -S UP -v circle -c RED -i RED -x 100 -y 250 -Z 0
l -t * -s 0 -d 3 -S UP -r 5000000 -D 0.002 -c black
l -t * -s 0 -d 1 -S UP -r 5000000 -D 0.002 -c black
l -t * -s 0 -d 4 -S UP -r 5000000 -D 0.002 -c black
l -t * -s 2 -d 0 -S UP -r 5000000 -D 0.002 -c black
+ -t 10.220626663 -s 0 -d 4 -p sctp -e 56 -c 0 -i 9 -a 0 -x {0.0.0.0 2.0.1.0 -1 -------I null}
- -t 10.220626663 -s 0 -d 4 -p sctp -e 56 -c 0 -i 9 -a 0 -x {0.0.0.0 2.0.1.0 -1 -------I null}
h -t 10.220626663 -s 0 -d 4 -p sctp -e 56 -c 0 -i 9 -a 0 -x {0.0.0.0 2.0.1.0 -1 ------- null}
r -t 10.222716263 -s 0 -d 4 -p sctp -e 56 -c 0 -i 9 -a 0 -x {0.0.0.0 2.0.1.0 -1 -------I null}
r -t 10.228196480 -s 6 -d 6 -p sctp -e 56 -c 2 -a 0 -i 9 -k AGT
+ -t 10.228196480 -s 6 -d -1 -p sctp -e 56 -c 2 -a 0 -i 10 -k AGT
- -t 10.228196480 -s 6 -d -1 -p sctp -e 56 -c 2 -a 0 -i 10 -k AGT
```

Fig 35 : *Résultat de simulation selon le fichier trace de NAM*

Nous évaluons notre implémentation à travers la variation des seuils accordés aux streams selon la topologie décrite ci-dessus. Les figures suivantes, figure 36 et figure 37 montrent le taux de transmission à partir de chaque stream durant les seuils suivant respectivement (50, 50, 50, 50, 50) et (60, 50, 20, 10, 5). Néanmoins tous ces résultats dépendent des différentes instances de simulations citées dans la section 2.3.2 de ce chapitre, particulièrement l'intervalle de temps ainsi que le taux de génération des chunks sur chaque streams.

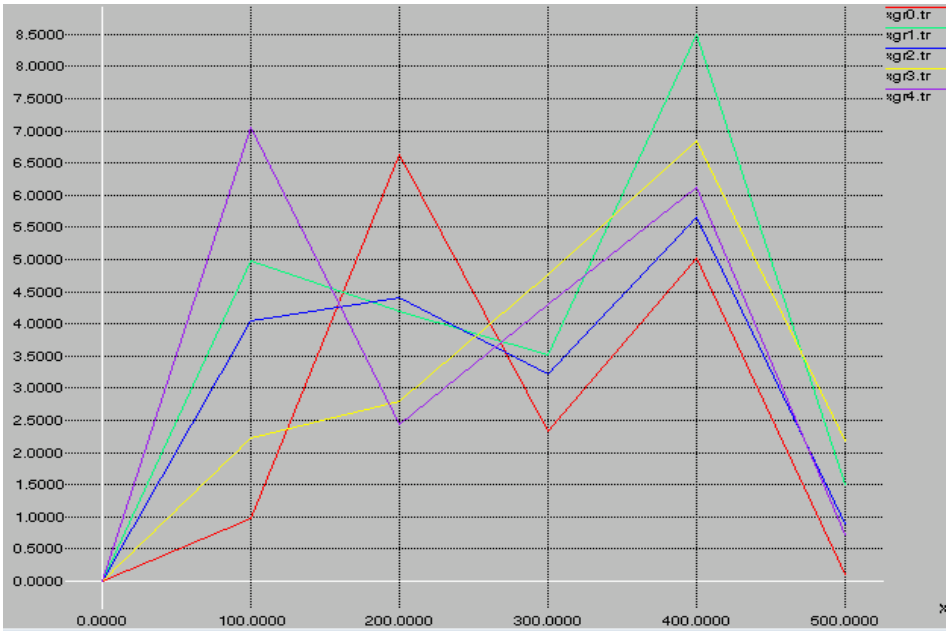


Fig 36 : *Taux de transmission à partir des streams CQ-mSCTP en utilisant les seuils* (50, 50, 50, 50, 50)

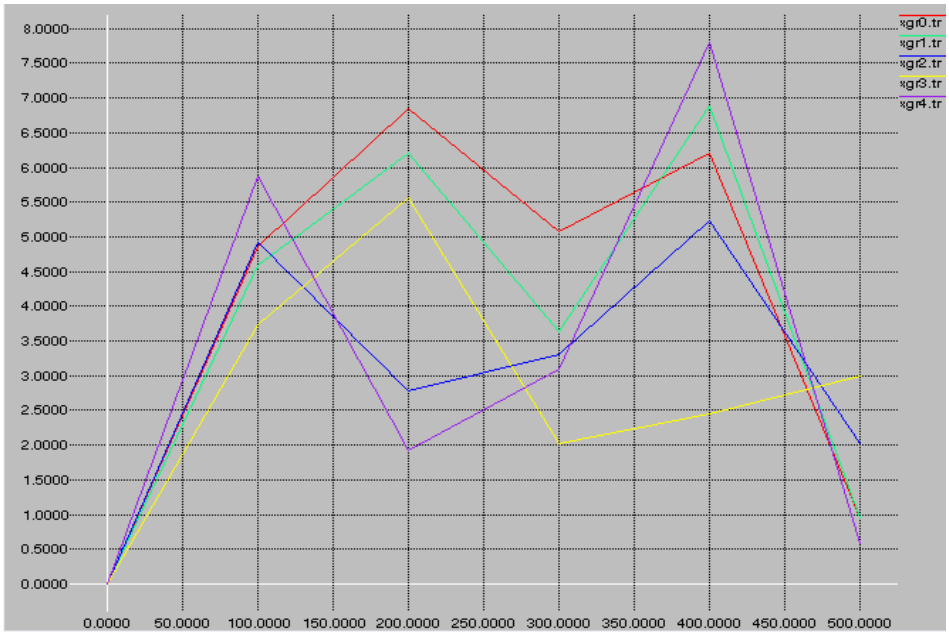


Fig 37 : *Taux de transmission à partir des streams CQ-mSCTP en utilisant les seuils* (60, 50, 20, 10, 5)

3. Analyse et validation de l'implémentation

Dans ce qui suit, nous évaluons les performances de notre implémentation. Nous procédons par une approche de comparaison des résultats de transmission entre CQ-mSCTP, le SCTP originel de NS ainsi que l'implémentation basée priorités P-SCTP spécifiée par [30]. L'analyse effectuée a pour objectif de valider le support de la QoS nécessaire aux transmissions multimédia via notre extension CQ-mSCTP. Nous vérifions également que CQ-mSCTP n'induit pas l'affaméité des streams moins prioritaires.

Pour établir le premier objectif, nous examinons le taux de transmission des paquets de plus haute priorité (i.e les streams 0, 1, 2). Notons que les cinq instances d'applications utilisées commencent la génération des paquets en même temps. Ce qui implique que les cinq streams de l'association contiennent des données en attente pour transmission.

Les résultats de la simulation de la figure 38 montrent une amélioration dans le traitement des chunks sur les streams plus prioritaires via CQ-mSCTP comparés aux résultats du SCTP originel. Ces résultats démontrent que CQ-mSCTP favorise la transmission des données de haute priorité contrairement à SCTP qui ordonnance ces transmissions selon leur ordre d'arrivée sur les streams sans tenir compte de leur classe de données.

Cependant, nous notons qu'au premier intervalle de temps de la figure 38, CQ-mSCTP fournit moins de gain que P-SCTP. En effet, CQ-mSCTP offre un taux de transmission des paquets prioritaires légèrement inférieur à celui du P-SCTP. Ces résultats sont une conséquence directe à l'ordonnancement du P-SCTP qui transmet d'abord tous les chunks présents sur les streams de haute priorité avant de traiter les autres streams. Contrairement à CQ-mSCTP qui interrompe les transmissions des chunks prioritaires pour l'accorder aux chunks des streams '3' et '4'.

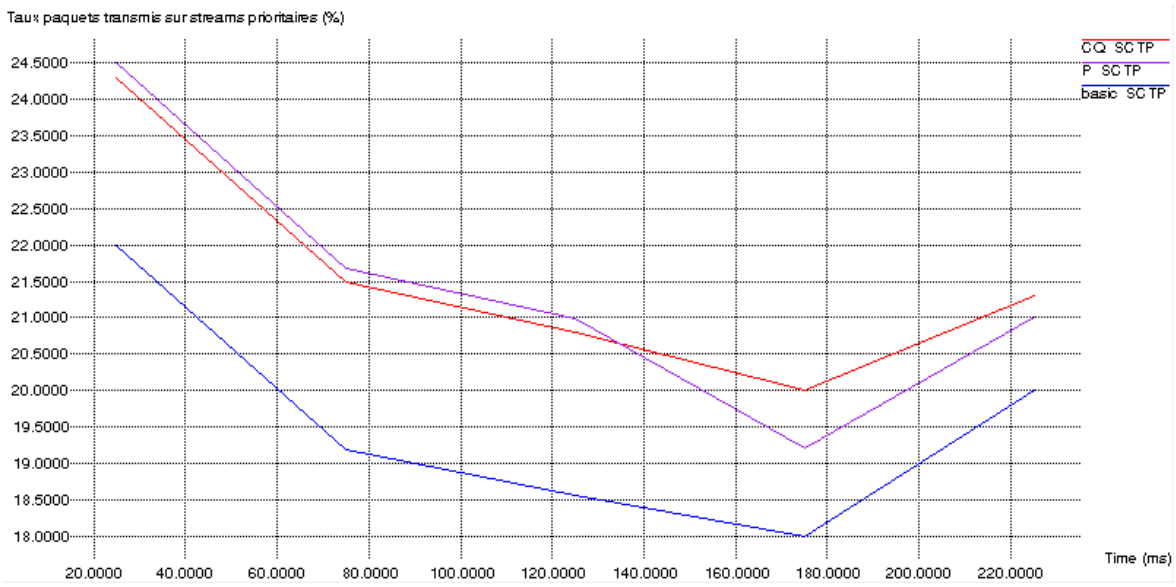


Fig 38 : *Comparaison du taux de transmission sur les streams plus contraignants*

Néanmoins, si CQ-mSCTP améliore certes la QoS de la transmission des paquets multimédia, il faudra vérifier qu'il évite les inconvénients de [30]. C'est pourquoi, nous mesurons le taux de perte de paquets sur les streams moins prioritaires.

Les résultats des tests présentés sur la figure 39 montrent la dégradation du taux de transmission des chunks sur les streams moins prioritaires en utilisant l'ordonnanceur P-SCTP comparés aux résultats de l'ordonnanceur de SCTP. Cette dégradation est due à l'expiration de la durée de vie des chunks durant leur attente sur les streams. Ces résultats montrent également que CQ-mSCTP réduit les pertes de paquets par rapport à SCTP originel sur les streams moins contraignants. La différence entre ces taux de transmission est supérieure à 2.5% des chunks transmis. En effet, P-SCTP fige l'ordonnanceur sur les streams prioritaires et néglige les autres streams. Contrairement à CQ-mSCTP et P-SCTP qui permettent un ordonnancement cyclique qui évite l'affaméité des streams.

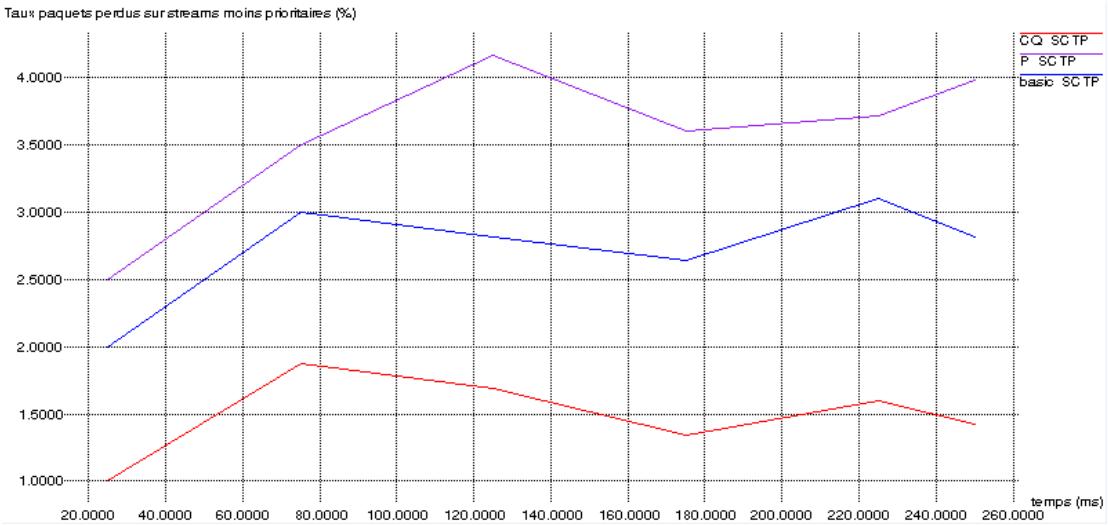


Fig 39 : *Comparaison du taux de transmission sur les streams moins contraignants*

Ces résultats démontrent que CQ-mSCTP utilisant l'algorithme de Round Robinson préemptif basé sur les VST selon le type de données respecte le compromis entre les contraintes des données associées aux streams de hautes priorités et le droit des streams moins contraignants de transmettre leur chunks.

4. Conclusion

Nous avons montré dans ce chapitre les principales étapes de notre implémentation sous NS 2.31. Nous avons décortiqué et introduit des modifications au code C++ en Open Source. Nous avons également utilisé d'autres outils et langages de programmation (TCL, NAM, XGRAPH). Les difficultés que nous avons rencontrées relèvent des spécificités de ces éléments de programmation ainsi que de leur interaction sous la plateforme de NS.

Néanmoins, nous avons pu implémenter un prototype de notre conception, simuler son comportement et obtenir des résultats via des diagrammes sous XGraph. L'analyse de ces résultats ainsi que leur comparaison avec les versions existantes de SCTP démontrent que notre conception améliore certes la qualité de service selon le type de données au niveau transport. Même si les résultats présentent une légère dégradation des transmissions de l'une des classes de données, nous estimons qu'elle répond aux exigences attendus et qu'elle peut constituer un point de départ pour d'autres travaux.

Des expériences dans des environnements plus grands et plus complexes pourraient apporter une évaluation plus exhaustive et permettre de déterminer les valeurs des seuils.

Conclusion & perspectives

Notre travail propose une modification au protocole de Transport SCTP en vu d'améliorer la qualité de service de ses transmissions dans le contexte d'une mobilité groupée.

En effet, nous avons pris connaissance dans le premier chapitre des éléments s'attachants à la mobilité des équipements sur Internet.

Dans le deuxième chapitre, nous avons examiné une panoplie de protocoles et leurs mécanismes, dont SCTP. Cette étude nous a permis de comparer les fonctionnalités de SCTP à celles offertes par d'autres protocoles qu'ils soient déployés sur le même niveau (Transport) ou sur la couche Réseau.

Nous avons conclu que SCTP par son extension mSCTP, permettant une configuration dynamique des adresses basée sur son mécanisme de multihoming, est un protocole qui favorise la gestion de la mobilité des réseaux. Cependant le déploiement de mSCTP pour assurer un Internet omniprésent impose qu'il puisse supporter les transmissions des paquets multimédia et qu'il doit répondre à leurs contraintes.

Dans le troisième chapitre, nous avons proposé un nouveau schéma d'ordonnancement de transmission entre les streams de SCTP en vu d'améliorer la qualité de service de tous type de données provenant à SCTP de la couche application ou de la couche réseau.

L'analyse des résultats obtenus des simulations de CQ_mSCTP dans le quatrième chapitre montre qu'un traitement des files de chunks sur les streams, adapté au type de données permet d'améliorer le temps de transmission des flux de haute priorité aussi bien que ceux de moindre priorité.

Toutefois, nous identifions un inconvénient à notre proposition. Il s'agit de la dégradation du temps de transmission des données de haute priorité comparé à l'implémentation de [30]. Néanmoins, cette dégradation est acceptable en termes des contraintes de QoS pour le trafic multimédia.

Comme perspectives à notre étude, nous proposons d'étendre SCTP pour mieux tenir compte des différences entre données applicatives. De plus, nous pourrons porter plus d'intérêt à la définition exacte de routeur mobile dans notre environnement de simulation (NS2), ainsi qu'un meilleur contrôle du flux de données applicatives.

Références

[1]. Laurent Sorin PAUN, ‘‘Gestion de la mobilité dans les réseaux ambiants’’, Institut National Polytechnique de Grenoble, Thèse de Doctorat, Nov 2005.

[2]. Julien MONTAVONT, ‘‘Gestion des déplacements de terminaux IPv6 mobiles assistée par géolocalisation’’, Université Louis Pasteur de Strasbourg, Thèse de Doctorat, 2006.

[3]. Thierry Ernest, ‘‘Le support des réseaux mobiles dans IPv6’’, RSTI – TSI, p 573 à 597, Volume 25 – n°5/2006.

[4]. Guillaume AURIOL, ‘‘Spécification et implémentation d’une architecture de signalisation à gestion automatique de la QdS dans un environnement IP Multidomaines’’, Institut National des Sciences Appliquées de Toulouse, Thèse de Doctorat, 2004.

[5]. Ismail Adel Djama, ‘‘Adaptations inter-couches pour la diffusion des services vidéo sans fil’’, Université de Bordeaux I, Thèse de Doctorat, N°d'ordre 3665, 2008.

[6]. Wassim Ramadan, Eugen Dedu Julien Bourgeois, ‘‘Une méthode de différenciation de pertes pour améliorer la performance des protocoles de transport sur réseaux sans fils’’, Université de Franche-Comté (LIFC), JDIR'09 : 10èmes Journées Doctorales en Informatique et Réseaux, Février 2009.

[7]. Thierry Ernest, “Les Réseaux Mobiles dans IPv6, support nécessaire au Multimédia”, Première Conférence Nationale sur le Multimédia Mobile (MCube), Mars 2004.

[8]. Olga Antonova, “Introduction and Comparison of SCTP, TCP-MH, DCCP protocols’’, Helsinki University of Technology-HUT T-110.551 Seminar on Internetworking, Avril 2004.

[9]. Naveen Gundu, “Mobility vs Multihoming’’, Helsinki University of Technology-HUT T-110.551 Seminar on Internetworking, Avril 2004.

[10]. Nicolas Montavont, “La Mobilité dans les Réseaux IP’’, Université Louis Pasteur de Strasbourg, Rapport de Stage, 2000/2001.

[11]. Rami Langar, “Mécanismes de Gestion de la Mobilité et Evaluation de Performance dans les Réseaux Cellulaires tout-IP’’, Ecole Nationale Supérieure des Télécommunications, Thèse de Doctorat, Juillet 2006.

[12]. M. Guillaume Jourjon, “Toward a Versatile Transport Protocol’’, École Nationale Supérieur d’Ingénieurs de Constructions Aéronautique, Thèse de Doctorat, 2008.

[13]. Sarwar Golam Babil, “Congestion control in DCCP for real-time applications over satellites and long delay wireless links’’, University of New South Wales, Thèse de Master, 2008.

[14]. Cui Lin, Seok J. Koh, “DCCP Overview’’, http://protocol.knu.ac.kr/tech/CPL-TR-05-07.pdf.

[15]. E. Kohler, M. Handley, S. Floyd, “Datagram Congestion Control Protocol (DCCP)’’, RFC 4340, Mars 2006.

[16]. F.Buntschu, R. Scheurer, A. Delley, “SCTP une alternative à TCP et UDP’’, Flash Informatique 9/03, Ecole Polytechnique Fédérale de Lausanne, p13-20, Novembre2003.

[17]. Aparna Kanung, Janani.T, P.Narayanasamy, “Dynamic IP reconfiguration in Stream Control Transmission Protocol’’, http://www.cs.wisc.edu/~janani/sctp_paper.PDF.

[18]. M. Ratola, “Which Layer for Mobility Comparing Mobile IPv6, HIP and SCTP”, Helsinki University of Technology, HUT T-10.551 Seminar on Internetworking, 2004.

[19]. “TheNetwork Simulator – ns 2, http://www.isi.edu/nsnam/ns”

[20]. Dominique Dhoutaut, “Etude du standard IEEE 802.11 dans le cadre des réseaux ad hoc de la simulation à l’expérimentation’’, Institut National des Sciences Appliquées de Lion, Thèse de Doctorat, 2003.

[21]. Rabah Meraihi, “Gestion de la qualité de service et contrôle de topologie dans les réseaux ad hoc’’, l’Ecole nationale supérieure des télécommunications. Thèse de Doctorat, Janvier 2005.

[22]. Van Steyvoort Thomas, “Détection distribuée des paramètres de propagation indoor dans les réseaux sans-fils’’, Institut National Polytechnique de Toulouse, Thèse de Master, 2005 – 2006.

[23]. Emmanuel Conchon, “Définition et mise en œuvre d’une solution d’émulation de réseaux sans fil’’, Institut National Polytechnique de Toulouse. Thèse de Doctorat. N°d'ordre 2385, 2006.

[24]. C. Moonjeong, L. Meejeong, S. Koh, “A Transport Layer Mobility Support Mechanism”, p 287-296, Information Networking, Networking Technologies for Broadband and Mobile Networks, International Conference ICOIN 2004, Février 2004.

[25]. R. Stewart, Q. Xie, et al., “Stream Control Transmission Protocol”, RFC 2960, Octobre 2000.

[26]. R. Stewart, L. Ong, I. R. Arias, K. Poon, P. Conrad, A. Caro, M. Tuexen, “SCTP Implementers Guide’’, draft-ietf-tsvwg-sctpimp guide -10.txt, 2003.

[27]. R. Stewart, Q. Xie, M. Tuexen, M. Kozuka “SCTP Dynamic Address Reconfiguration’’, RFC 5061, Septembre 2007.

[28]. V. K. Madisetti, D. A. Agyriou, “Transport Layer QoS Management for Wireless Multimedia Services’’, Soft.Networks Technical Report, September 2002.

[29]. J. Gerard, I. I. Heinz, P. D. Amer, “Priorities in SCTP Multistreaming’’, 8th World Multiconference on Systemics, Cybernetics and Informatics SCI , Juillet 2004.

[30]. N. Maslekar, M. Boussedjra, J. Mouzna, M. Pai, “QoS in Mobile Networks by Assigning Priorities to SCTP Streams’’, ITST 2008, p 246-252. Octobre 2008.

[31]. H. Gundersen, F. Trydal, “QoS for real-time IP traffic’’, Graduate Thesis Siv.ing. Degree in Information and Communication Technology AGDER University College, Mai 2001.

[32]. Shaojian Fu, Liran Ma, Mohammed Atiquzzaman, Yong-Jin Lee, “Architecture and Performance of SIGMA: A Seamless Mobility Architecture for Data Networks’’, p3249-3253, IEEE International Conference on Telecommunications ICC, Mai 2005.

[33]. Pulak K Chowdhury, William Ivancic, “SINEMO: An IP-diversity based Approach for Network Mobility in Space’’, Second IEEE International Conference on Space Mission Challenges for Information Technology, SMC-IT'06, Juillet 2006.

[34]. R. Stewart, Q. Xie, M. Tuexen, P. Conrad, “Stream Control Transmission Protocol (SCTP) Partial Reliability Extension”, RFC 3758, Mai 2004.

[35]. M. Handley, H. Schulzrinne, E. Schooler, J. Rosenberg, “SIP : Session Initiation Protocol” , RFC 2543, Mars 1999.

[36]. A. Matsumoto, M. Kozuka, K. Fujikawa, Y. Okabe, “TCP Multi-Home Options.Work in Progress”, IETF Internet-Draft draft-arifumi-tcp-mh-00.txt, Octobre 2003.

[37]. C. Perkins, Editor. IBM, “IP Mobility Support”, RFC 2002, Octobre 1996.

[38]. D. Johnson, C. Perkins , J. Arkko, “IP Mobility Support for IPv6’’, RFC 3775, Juin 2004.

[39]. Thierry Ernest, “Network mobility Support Goals and Requirements’’, RFC 4886, Juillet, 2007.

[40]. Y. Wang, L Fan, N. Akthar, K. Chew, R. Tafazoli, “An Aggregation-based QoS Architecture for Network Mobility”, 14th IST Mobile & Wireless Communications Summit, 2005

[41]. V. Basto, V. Freitas, “SCTP Extensions for Time Sensitive Traffic’’, Fifth International Network Conference, INC2005, Juillet 2005.

[42]. ITU Rec. G.114, “One-Way Transmission Time,” International Telecommunications Union, Février 1996.

[43]. L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu. “Advances in Network Simulation’’, IEEE Computer, volume 33 n°5, p 59-67, May 2000.

[44]. “OPNET. OPNET Technologies’’, http://www.opnet.com.

[45]. X. Zeng, R. Bagrodia, and M. Gerla. “GloMoSim: A Library for Parallel Simulation of Large-scale Wireless Networks’’, PADS '98, May 1998.

[46]. “QualNet. Scalable Network Technologies’’. http://www.scalable-networks.com

[47]. K. Fall and K. Varadhan. “The ns manual (formerly ns notes and documentation)’’, 2002, http ://www.isi.edu/nsnam/ns/doc/index.html.

[48]. E. Crawley, R. Nair, B. Rajagopalan, H. Sandick, “A Framework for QoS-based Routing in the Internet’’, RFC 2386, Aout 1998.