Design d'une interface cross-layer et implémentation de métriques complexes pour les réseaux ad-hoc

Hervé Aïache, Vania Conan, Laure Lebrun, Jérémie Leguay, Stéphane Rousseau, Damien Thoumin

Thales Communications[1]
160 Bd de Valmy, BP 82
92704 Colombes cedex, France
{firstname.name}@fr.thalesgroup.com

RÉSUMÉ. *L'approche* cross-layer *suscite de plus en plus d'intérêts dans le domaine des réseaux mobiles. Toutefois, les outils de validation expérimentale sont encore peu nombreux. Ce papier présente une extension logicielle majeure à XIAN (Cross-layer Interface for wireless Ad hoc Networks). Il repose sur un framework générique permettant d'expérimenter l'implémentation de design* cross-layer *sous plateforme Linux avec des cartes sans fil 802.11 en utilisant le driver MadWifi. XIAN rend accessible les informations disponibles au niveau des couches MAC/PHY et offre un cadre expérimental permettant la création de métriques pouvant utiliser des informations locales et/ou des informations de voisinages. Afin d'illustrer les possibilités données par cette extension, nous prenons comme exemple l'implémentation de la métrique ETX (Expected Transmission Count) et fournissons les résultats de cette étude expérimentale.*

ABSTRACT. *In the highly dynamic and unpredictable environment of MANETs,* cross-layer *design is receiving growing interest but lacks experimental validation tools. This paper presents XIAN (Cross-layer Interface for wireless Ad hoc Networks), a generic framework for experimenting cross-layer designs in Linux testbeds with 802.11 wireless cards using the MadWifi driver. XIAN can be used as a service by other layers to access MAC/PHY configuration and performance information and to create automatically complex metrics from both local and neighbour node measurements. The defined and implemented software architecture introduces the XIAN Nano-protocol and its automated management. We exemplify their benefits through the implementation of the well-known ETX (Expected Transmission count) metric and we provide results from real experimentations.*

MOTS-CLÉS : *Réseaux Ad hoc, Cross-layer, QoS, métrique*

KEYWORDS: *Ad hoc Networks, Cross-layer, QoS, metric*

1. Ce travail a été soutenu par le projet RNRT Airnet et le projet européen IST FP6 Chorist

1. Introduction

Outre le fait que dans les Réseaux Ad hoc Mobiles (MANETs [COR 99]), les nœuds sont susceptibles de se déplacer, la qualité des liens radio établis entre deux noeuds évolue au cours du temps. En effet, la connexion radio peut passer d'un état très bon à un état très faible en très peu de temps et inversement. Certaines études montrent que le trafic du réseau lui-même est à l'origine de ces dégradations.

D'autre part, on souhaite accueillir dans les réseaux ad hoc de plus en plus d'applications nécessitant une certaine qualité de service. Il est donc nécessaire d'avoir une connaissance très précise de l'état du réseau à un instant donné afin de pouvoir satisfaire ce genre d'applications. Un des objectifs de notre étude est de trouver un compromis entre connaître l'état du réseau et ne pas surcharger inutilement les liens radio par des paquets de contrôle. Ainsi, la bande-passante disponible pour les applications n'est pas diminuée par d'innombrable paquets de contrôle.

Le modèle OSI tel qu'il est présenté permet de déceler l'évolution de la qualité des liens grâce à l'envoi fréquent de paquets de contrôle. Cet envoi est fait non seulement par la couche MAC mais aussi par les couches supérieures telles que la couche réseau. Ce mode de fonctionnement engendre ainsi des duplications d'information et de contrôle au sein d'un même noeud et par conséquent engendre une perte de bande-passante.

Notre approche consiste à réutiliser les informations disponibles dans les couches basses et de les remonter vers les couches supérieures. Cette approche, connue sous le nom cross-layer, permet de diminuer considérablement le nombre de paquets de contrôle nécessaires pour vérifier la qualité des liens (suivant différents critères). La communication peut se faire soit entre les couches adjacentes (e.g. Kawadia et al. [KAW 03]), ou éventuellement entre des couches non adjacentes (e.g. Conti et al. [CON 04]). Si cette approche existe déjà, son expérimentation reste difficile du fait d'un manque de support API approprié des cartes 802.11 utilisées dans les testbeds ad hoc.

Ce papier propose une extension de XIAN [AIA 06] (Cross-layer Interface for wireless Ad hoc Networks) qui rend accessible des informations contenues dans les couches MAC/PHY et facilite l'intégration et l'évaluation de concepts *cross-layer*. XIAN est aujourd'hui disponible sur Linux avec le driver *MAD* [MAD] 802.11. XIAN a été conçu d'une part dans le but d'encourager les implémentations basées sur le cross-layer, et de faciliter son déploiement dans les plateformes MANET. L'extension apporte la possibilité d'ajouter de nouvelles métriques plus ou moins complexes utilisant des informations locales ou provenant des noeuds voisins. Le *nano-protocole XIAN* a pour vocation de faciliter l'ajout de ces métriques. Nous illustrons ces nouvelles fonctionnalités à la fin du papier, par l'implémentation de la métrique ETX proposée par De Couto et al. [DEC 03].

L'organisation du papier est la suivante. Dans la partie 2, nous présentons l'approche et le concept donnés dans XIAN. La partie 3 décrit le *nano-protocole XIAN*.

Les parties 4 et 5 sont consacrées à l'implémentation et l'application de la métrique ETX sur une étude expérimentale. Enfin la partie 6 conclut le papier et donne différentes directions pour les prochaines évolutions de XIAN et des approches cross-layer.

2. XIAN : une interface Cross-layer pour les réseaux sans fil ad hoc

XIAN a été développé dans le but de faciliter les études expérimentales dans les réseaux Ad-Hoc. Pour cela, XIAN simplifie considérablement les communications entre les couches adjacentes mais aussi non-adjacentes du modèle OSI. La plupart des informations échangées sont des métriques permettant de statuer sur la qualité des liens et des ressources disponibles. Ces données calculées par les couches basses sont immédiatement utilisables par les couches supérieures et peuvent être réutilisées pour calculer de nouvelles métriques. Ces échanges se font par un mécanisme de question/réponse entre les couches. Nous décrivons ici en détails l'architecture logicielle de XIAN et nous présentons le type d'informations/états/statistiques qu'il offre aux autres composants du système Linux.

2.1. *Architecture logicielle de XIAN*

XIAN contient trois composants principaux.

– Le ***Kernel Space XIAN Interface*** (KSI) est implémenté comme un module noyau. Il interagit directement avec le driver *MadWifi* pour en extraire les statistiques et les états de chaque connexion.

– Le ***User Space XIAN Interface*** (USI) duplique le KSI mais au niveau de l'espace utilisateur. Cette API est contenue dans une librairie C dans le but de faciliter son intégration au sein de programmes utilisateur. (e. g. les démons de routage et les applications).

– Le ***XIAN Information Transport Module*** (ITM) permet de récupérer des informations et des statistiques contenues dans le driver *MadWifi* en passant par les Interfaces décrites précédemment. Ce module est implémenté dans cette version de XIAN comme un *"character device"*.

En sus, un composant complémentaire, le ***XIAN User Space Extended Interface*** (USEI), fournit aux utilisateurs un moyen simple pour faire des moyennes, des combinaisons de métriques ou encore noter des changements significatifs pour une métrique particulière. Il intéragit directement avec le USI et peut être décomposé en trois ensembles de fonctions principaux :

– ***Measurement functions***, qui fournit des métriques calculées à partir de métriques élémentaires prises directement via l'USI ou à partir de moyennes.

– ***Operation functions***, où sont implémentés les opérateurs mathématiques nécessaires au calcul de certaines métriques (telles que la moyenne, le min, le max, ...).

– ***Relevance functions***, qui met en oeuvre le comparateur correspondant permettant d'indiquer si une différence significative arrive entre deux métriques calculées (typiquement entre les précédents et les nouveaux).

Ces cinq composants sont représentés dans la figure 1. De plus, nous y indiquons la manière dont ils interagissent les uns avec les autres et comment on fournit les états internes du driver MAC aux autres composants Linux.

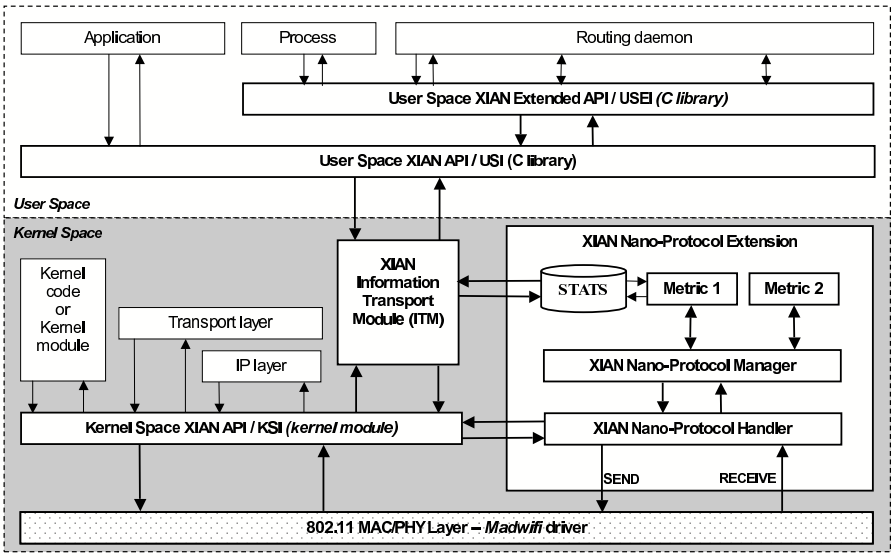


Figure 1. *Architecture logicielle de XIAN pour linux.*

L'ITM est un module orienté message implémenté comme un *character device*. Les métriques échangées via son interface, rendues accessibles à USI et KSI, sont de deux types :

– des métriques de base maintenues à jour par le driver *MadWifi*

– des métriques complexes implémentées dans l'extension *XIAN nano-protocol* présentée dans ce papier.

2.2. *Les interfaces de base de XIAN*

XIAN permet un accès facile à différentes métriques de base disponibles dans le driver *MadWifi*. Elles peuvent être organisées en trois groupes :

– ***Etats de configuration***, qui concerne la configuration courante des différents paramètres du périphérique 802.11. On peut par exemple y récupérer la taille des files

d'attente et la fréquence utilisée.

– ***Aggregated metrics***, il s'agit de compteurs qui fournissent le statut global d'utilisation de l'interface 802.11. Par exemple, les informations rapportées peuvent être le nombre de trames reçues, rejetées ou avec un mauvais BSSID, le nombre de réceptions échouées (e.g. à cause d'un mauvais CRC ou d'un problème de décryptage).

– ***Per neighbour/link metrics***, qui conserve les informations par voisin récupérées par la couche MAC, par exemple le nombre de trames de données ou bytes reçues/transmises, la force du signal (RSSI) ou le nombre de transmissions renouvelées.

Ainsi, environ 180 fonctions ont été développées et intégrées dans l'API XIAN afin de couvrir chacune des métriques disponibles (i.e. dans KSI et dans USI, la duplication de l'interface au niveau de l'utilisateur. Sur ces 180 métriques, 40 d'entre elles contiennent des informations du voisinage et le reste concerne plus particulièrement le noeud lui-même.

3. L'extension *XIAN nano-protocol*

Comme indiqué sur la figure 1, l'extension *XIAN nano-protocol*, disponible sur le site officiel [XIA], offre des mécanismes implémentant des échanges bidirectionnels de métriques entre noeuds voisins. Le format de ces messages est donné sur la figure 2. Ces paquets sont envoyés de manière périodique et contiennent les informations sur les mesures cross-layer définies. Une fois reçues, ces métriques sont collectées par le Kernel Space XIAN Interface, qui interagit directement avec le driver MadWifi.

A chaque métrique est associé un nombre de paramètres de configuration, par exemple la fréquence à laquelle sont envoyés les paquets sur le réseau, la fonction de traitement invoquée à la réception de ces paquets pour mettre à jour la métrique. Une configuration type pour une métrique se caractérise par : un identifiant de configuration, un nombre associé à la métrique, le périphérique à utiliser, l'adresse MAC destination pour le rapport, la fréquence d'envoi (en millisecondes) et un pointeur de fonction qui se charge du traitement de l'information. Chaque métrique est implémentée sous la forme d'un module noyau Linux utilisant l'API fournie par XIAN.

Dans le protocole que nous avons implémenté, l'élément principal du message XIAN est la métrique. Les caractéristiques d'une métrique XIAN sont son type (i.e. un nombre identifiant la métrique), un identifiant de la configuration de la métrique, un type d'encodage, l'adresse MAC associée à la métrique et la valeur de la métrique. Les autres champs composent l'entête du message XIAN :

– ***Version*** : la version du nano-protocole XIAN.

– ***Id*** : l'identifiant du message.

– ***Length*** : le nombre de métriques composant le message.

– ***Payload*** : un ensemble d'informations sur les métriques.

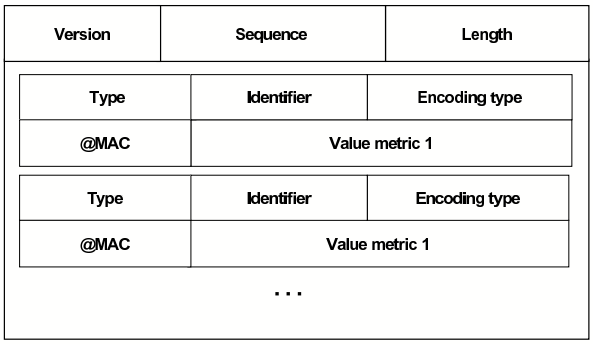


Figure 2. *Format des messages du nano-protocole XIAN*

Comme montré sur la figure 1, quatre composants logiciels, tous implémentés en tant que module noyau, supporte l'intégration de métriques complexes au sein du framework XIAN.

3.1. *Le XIAN Nano-Protocol Handler*

Ce module se charge des messages XIAN échangés entre les noeuds. Sa première fonctionnalité est d'envoyer les messages XIAN reçus vers le module XIAN Nano-Protocol Manager, qui se trouve juste au-dessus dans l'architecture.

De plus, ce module se charge de construire et envoyer les messages XIAN requis par les métriques complexes ayant été implementées dans le framework. Diverses métriques sont alors échangées entre les noeuds voisins. Il faut noter que un mécanisme a été mis en place afin de ne pas utiliser la bande-passante de manière excessive lorsque plusieurs métriques complexes requièrent l'échange de même métrique de base.

3.2. *XIAN Nano-Protocol Manager*

Ce module permet la gestion et l'utilisation de plusieurs métriques. Il est au coeur de l'architecture XIAN. Il permet en outre un mécanisme d'abonnement et désabonnement pour la configuration de plusieurs métriques.

A la réception de messages XIAN, le XIAN Nano-Protocol Manager analyse et isole les différentes métriques présentes dans le message. Pour chaque métrique, il exécute la fonction appropriée qui se charge de la mise à jour des valeurs correspondantes. Il a aussi la possibilité de connaître l'adresse MAC du noeud origine du message et si nécessaire lui communiquer des informations. Dans le cas où la métrique n'est pas définie elle est automatiquement ignorée.

3.3. *Le module de statistiques*

Le module précédent est nécessaire pour la mise en place de nouvelles métriques utilisant des informations bi-directionnelles. Une fois traitées, ces métriques doivent être accessibles à d'autres modules ou processus. Cette fonctionnalité est assurée par le module XIAN Statistics. Ce module permet la sauvegarde des différentes métriques dans une base de données. Pour chaque métrique, il est possible de retenir la valeur, l'identifiant de configuration et l'adresse MAC du noeud origine (le noeud qui a envoyé l'information).

3.4. *Le module de configuration de métriques* cross-layer

La fonction principale de ce module est de configurer le comportement pour une métrique donnée. Il est possible de charger plusieurs modules et ainsi définir plusieurs métriques. Plusieurs paramètres peuvent être précisés tels que l'identifiant, le type de métrique, le périphérique utilisé, l'adresse MAC destination, la fréquence (en millisecondes) des rapports et plus particulièrement le traitement souhaité pour cette métrique. Le traitement des métriques peut être soit local soit nécessiter une communication avec les autres noeuds et sera fournie par l'intermédiaire d'un pointeur vers la fonction de traitement.

4. Implémentation de la métrique ETX dans XIAN

Nous allons à présent illustrer tout le fonctionnement de XIAN à travers un exemple simple d'implémentation de métrique. Pour se faire, nous avons choisi la métrique ETX proposée par De Couto et al. [DEC 03]. Cette métrique est bi-directionnelle et permet de connaître la qualité d'un lien et avertir d'une congestion ou d'interférences qui pourraient compromettre la communication. Il est évident que cette métrique peut s'avérer être très utile pour la mise en place de qualité de service dans un réseau. D'autres métriques auraient pu faire l'objet de cette même étude telles que la *médium time metric* (MTM) proposée par Awerbuch et al. [AWE 04] qui sélectionne le meilleur chemin en termes de bande-passante ou bien la *available bandwidth* introduite par Déziel et al. [DEZ 05], ou encore la métrique présentée par Iannone et al. [IAN 04] qui combine le taux de succès de transmission, le niveau d'interférence et le débit. La partie suivante se focalise sur la métrique ETX et son implémentation avec l'extension XIAN nano-protocol.

4.1. *ETX*

La métrique *expected transmission count* (ETX) mesure le nombre moyen de transmissions nécessaires pour la bonne réception d'un paquet. Cette mesure peut être faite sur chaque lien du réseau. Ainsi, pour connaître le nombre moyen de transmis-

sions nécessaires pour atteindre une destination, il suffit de faire la somme d'ETX sur chacun des liens traversés.

ETX est une mesure bi-directionnelle. Elle combine le *forward delivery ratio*, noté Df, et le *reverse delivery ratio*, Dr. Df est la probabilité qu'un paquet arrive avec succès au noeud suivant et Dr est la probabilité que le paquet ACK arrive lui-aussi au noeud source. Une bonne transmission est l'association de ces deux étapes. ETX est donc égale à l'inverse du produit $Df * Dr$.

$$ETX(link) = \frac{1}{Df * Dr} \quad [1]$$

Cette métrique se calcule par l'envoi périodique de paquets aux noeuds voisins. Ces paquets sont comptabilisés et numérotés ce qui permet d'en déduire aisément le nombre de transmissions réussies. Chacun des paquets envoyés contient ce nombre (calculé localement) ce qui permet a un noeud voisin de calculer la valeur d'ETX.

4.2. *Détails de l'implémentation*

La métrique ETX a été implémentée dans le framework XIAN comme un module noyau spécifique. Au moment de charger le module ETX, nous configurons le XIAN Nano-Protocol Manager afin qu'il envoie la valeur Dr dans un message de broadcast niveau MAC toutes les T millisecondes. Un nouveau type ainsi qu'un nouvel identifiant de configuration de la métrique ETX sont alors créés.

Le paramètre pf fourni au moment de l'enregistrement de la métrique via la fonction $register\_id$ correspond à la fonction de traitement appelée à chaque réception de la métrique ETX. Le message de broadcast contient la valeur Dr pour la liste d'adresses MAC fournie par la fonction $insert\_mac\_to\_broadcast$ qui est invoquée pour chaque message de broadcast reçu quand aucune adresse MAC n'est présente. A la réception de ces messages, le module noyau correspondant à ETX stocke la valeur de la métrique D_r dans le module de statistiques via leur interface commune et la fonction $update\_xian\_stat$. Alors le module métrique incrémente un compteur qu'il stocke dans le module de statistiques. A partir des D_r et D_f obtenus, un calcul de ETX est réalisé puis stocké dans le module statistiques mettant ainsi à disposition une estimation de la qualité du lien.

En parallèle, un thread est réveillé toutes les W millisecondes. Il calcule alors D_f avec la valeur du compteur qu'il récupère et le nombre de messages estimés durant ces W millisecondes. Par la suite, D_f est stocké dans le module STATS et le compteur est remis à zéro. ETX est alors recalculé et mis à jour dans le module STATS.

Notons que, comme le noyau Linux ne prend pas en compte des flottants, toutes les métriques dans l'espace noyau sont sous le format spécifique $xian_float$ et accessible par la fonction $struct\ xian\_float\ get\_xian\_stat$ dans le KSI. Dans l'espace utilisateur, les résultats sont disponibles sous le format flottant avec une fonction similaire $float\ get\_xian\_float()$ dans le USI. Par ailleurs, pour terminer le processus,

la configuration de la métrique doit être effacée via la fonction $unregister\_id$ et le module déchargé.

5. Résultats expérimentaux

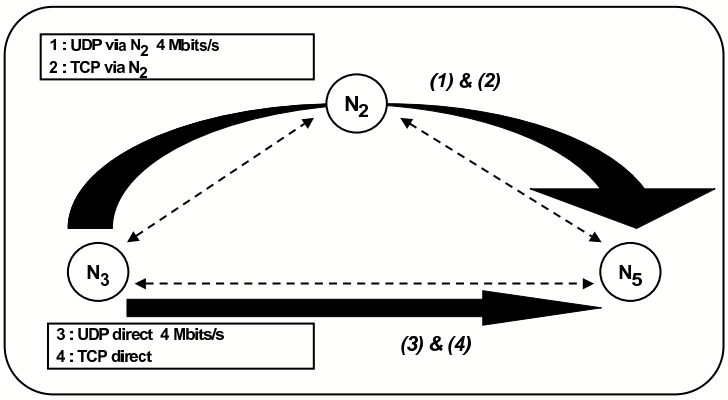
Nous présentons ici les résultats concernant ETX obtenus en plateforme. Ces résultats ont pour but de mettre en évidence les possibilités de XIAN mais aussi de mettre en évidence l'intérêt d'ETX dans le cadre du routage. La plateforme est composée de trois machines équipées de cartes Cisco Aironet Wi-Fi utilisant le chipset Atheros. On utilise le standard 802.11b à 11 Mbits/s en mode ad-hoc (sans RTS/CTS). Sur chaque machine nous utilisons *iperf* [IPE] pour générer du trafic TCP et UDP. De plus, le framework STAF/STAX [STA] permet l'automatisation des expérimentations.

Le mesures de performances suivantes ont été réalisées toutes les $delta$ secondes :

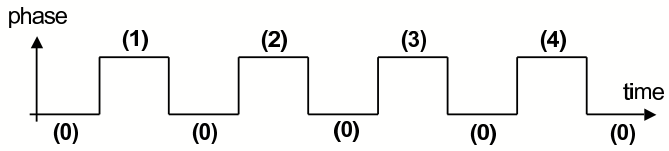
– Le ***RSSI*** (Relative Signal Strength Indicator) : la puissance reçue par les noeuds.

– Le ***Throughput*** : le nombre total d'octets reçus et envoyés par la couche MAC durant les $delta$ dernières secondes.

– L' ***ETX*** calculé sur 10 messages ayant été envoyés toutes les 200 ms.



(a) Topologie du réseau



(b) Chronogramme pour le génération de trafic

Figure 3. *Paramètres de l'expérimentation.*

Les figures 3(a) et 3(b) présentent la topologie du réseau et précise les instants où les flots transitent. Le lien entre les machines N_3 et N_5 est plus grand que les deux autres et ce de manière significative. Le trafic est généré en 5 phases durant chacune 24 secondes et séparées par une seconde.

– ***Phase (0)*** : aucun trafic.

– ***Phase (1)*** : Trafic UDP entre N_3 et N_5 en passant par N_2 avec un débit de 4 Mbits/s.

– ***Phase (2)*** : Trafic TCP entre N_3 et N_5 en passant par N_2.

– ***Phase (3)*** : Trafic UDP entre N_3 et N_5 avec un débit de 4 Mbits/s.

– ***Phase (4)*** : Trafic TCP entre N_3 et N_5.

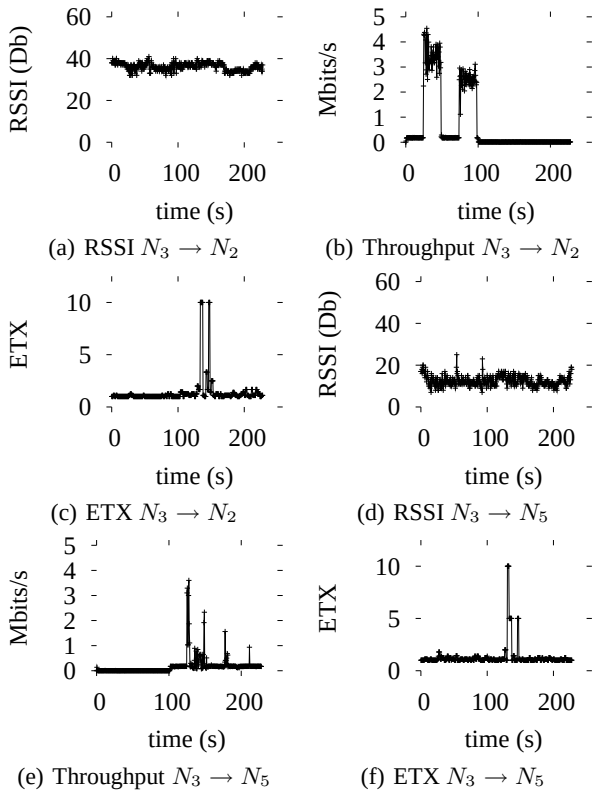
Nous présentons les résultats obtenus expérimentalement. Ces résultats sont regroupés sur la figure 4. Plus précisément, les figures 4(b) et 4(e) indiquent les débits respectifs des liens $N_3 \rightarrow N_2$ et $N_3 \rightarrow N_5$.

Une première analyse nous permet d'estimer la valeur de la bande passante disponible entre N_3 et N_2 et entre N_2 et N_5. En effet, lorsque les flux UDP et TCP issus du noeud N_3 passent par N_2 pour atteindre N_5 nous parvenons à saturer le lien. Nous en déduisons une bande-passante de 3.4 Mbits/s pour le lien $N_3 \rightarrow N_2$ et de 2.4 Mbits/s pour le lien $N_2 \rightarrow N_5$. Au contraire, quand les trafics UDP et TCP issus du noeud N_3 passent directement par le lien $N_3 \rightarrow N_5$, la bande-passante disponible sur les mêmes liens est alors de 0.1 Mbits/s. Une première conclusion est qu'une décision de routage ne tenant compte que du nombre de sauts serait inappropriée dans ce cas. Pour améliorer la qualité des décisions de routage nous introduisons donc le concept cross-layer.

Sur les figures 4(a) et 4(d) nous nous intéressons à la valeur du RSSI pour les liens $N_3 \rightarrow N_2$ et $N_3 \rightarrow N_5$. La distance significative entre les noeuds N_3 et N_5 implique une valeur du RSSI plus faible sur le lien $N_3 \rightarrow N_5$, que celui observé sur le lien $N_3 \rightarrow N_2$. Les valeurs retenues sont respectivement 12.5 dB et 36.1 dB. La valeur du RSSI pour le lien $N_2 \rightarrow N_5$ n'est pas représenté ici mais la valeur observée est de 33.4 dB. Nous aurions pu utiliser cette métrique pour prendre des décisions de routage. Toutefois, cette métrique ne tient pas compte de la contention au niveau du MAC. Du coup, cette métrique nous ne parait pas suffisament intéressante si on se place dans des conditions réelles.

Les figures 4(c) et 4(f) indiquent les valeurs ETX observées pour les liens $N_3 \rightarrow N_2$ et $N_3 \rightarrow N_5$. La valeur d'ETX sur le lien $N_3 \rightarrow N_5$ est supérieure à celle du lien $N_3 \rightarrow N_2$. Ceci indique la mauvaise qualité du lien concerné. Durant la phase (3), le trafic UDP ne peut être transféré convenablement, le noeud N_3 voit son buffer saturé ce qui a pour conséquence une dégradation de l'ETX sur le lien $N_3 \rightarrow N_2$. Cette dégradation n'est pas seulement due à ce phénomène mais aussi aux collisions au niveau du lien radio.

Les expérimentations confirment qu'ETX peut être une métrique intéressante pour prendre des décisions de routage. Elle est aujourd'hui disponible sur XIAN grâce



(a) RSSI $N_3 \rightarrow N_2$ (b) Throughput $N_3 \rightarrow N_2$

(c) ETX $N_3 \rightarrow N_2$ (d) RSSI $N_3 \rightarrow N_5$

(e) Throughput $N_3 \rightarrow N_5$ (f) ETX $N_3 \rightarrow N_5$

Figure 4. *Mesures réalisées avec XIAN.*

à l'extension récemment apportée : le nano-protocol. Cependant, il est très probable que la métrique ETX ne suffise pas, c'est pourquoi nous avons implémenté XIAN en fournissant un moyen simple d'ajouter de nouvelles métriques.

6. Conclusions et travaux futurs

Dans ce papier, nous présentons l'extension de XIAN, qui permet l'intégration du concept cross-layer pour des expérimentations en Wi-Fi. L'extension proposée dans ce papier permet la mise en place rapide de nouvelles métriques ayant besoin d'informations locales ou provenant du voisinage grâce au *XIAN nano-protocol*.

Afin de valider d'une part le bon fonctionnement de l'extension de XIAN, mais aussi son utilité, nous avons implémenté la métrique ETX (Expected Transmission Count). Le code complet de XIAN et de son extension est disponible en ligne [XIA].

Nous envisageons de continuer le développement de XIAN. Cette extension est très encourageante et peut être complétée par une interface permettant l'ajout de mé-

triques et de les intégrer au niveau MAC. De plus, l'introduction d'algorithmes de routage utilisant ces métriques pourrait être développée et étudiée. Cela pourrait apporter des solutions acceptables pour des routages avec qualité de service et des routages réactifs aux changements de topologies caractéristiques des réseaux ad hoc.

7. Bibliographie

[COR 99] CORSON, S. RFC 2501 : Mobile ad hoc networking (MANET) : Routing protocol performance issues and evaluation considerations. IETF (January 1999)

[CON 04] CONTI, M., MASELLI, G., TURI, G., GIORDANO, S. Cross-layering in mobile ad hoc network design. IEEE Computer (february 2004) 48–51

[KAW 03] KAWADIA, V., KUMAR, P.R. A cautionary perspective on cross layer design. IEEE Wireless Communication Magazine (july 2003)

[AIA 06] AIACHE, H., CONAN, V., LEGUAY, J., LEVY, M. Xian : Cross-layer interface for wireless ad hoc networks. In : Proc. Med-Hoc-Net. (2006)

[MAD] MADWIFI *http ://www.madwifi.org*

[DEC 03] DE COUTO, D.S.J., AGUAYO, D., BICKET, J., MORRIS, R. A high-throughput path metric for multi-hop wireless routing. In : Proc. MobiCom. (2003)

[XIA] XIAN *http ://sourceforge.net/projects/xian/*

[DEC 02] DE COUTO, D.S.J., AGUAYO, D., CHAMBERS, B.A., MORRIS, R. Performance of multihop wireless networks : Shortest path is not enough.In : Proc. HotNets, ACM SIGCOMM (2002)

[AWE 04] AWERBUCH, B., HOLMER, D., RUBENS, H. High throughput route selection in multi-rate ad hoc wireless networks. In : Proc. WONS. (2004)

[DEZ 05] M. DÉZIEL, L.L. Implementation of an IEEE 802.11 link available bandwidth algorithm to allow cross-layering. In : Proc. WiMob. (2005)

[IAN 04] IANNONE, L., KHALILI, R., SALAMATIAN, K., FDIDA, S. Cross-layer routing in wireless mesh networks. In : Proc. ISWCS. (2004)

[IPE] IPERF. *http ://dast.nlanr.net/Projects/Iperf/*

[STA] STAF Software Testing Automation Framework. *http ://staf.sourceforge.net*