Une méthode de différenciation de pertes pour améliorer la performance des protocoles de transport sur réseaux sans-fil

Wassim Ramadan
Eugen Dedu Julien Bourgeois
Laboratoire d'Informatique de l'Université de Franche-Comté (LIFC)
1 cours Leprince-Ringuet, 25200 Montbéliard, France
{Wassim.Ramadan, Eugen.Dedu, Julien.Bourgeois}@pu-pm.univ-fcomte.fr

***Résumé*—Dans cette étude nous présentons une nouvelle approche pour améliorer les performances des protocoles de transport sur les réseaux sans-fil. Un des principaux problèmes encore non résolus des réseaux sans-fil est que les pertes qui apparaissent en raison de perturbations temporaires dans les réseaux sans-fil sont considérées comme des pertes de congestion par la source, ce qui provoque une réduction non nécessaire du débit. Ce genre de protocoles sont conçus pour fonctionner généralement sur des réseaux filaires où les pertes sont majoritairement des pertes de congestion. Leurs performances sont dégradées lorsqu'on les utilise sur un réseaux sans-fil. ECN (*Explicit Congestion Notification*) peut contrôler la congestion dans les réseaux filaires grâce à une gestion active de la file d'attente avec RED (*Random Early Detection*). Le présent document propose une méthode de différenciation de pertes (*EcnLD*) sur la base d'ECN. Cette méthode est appliquée à *TCPLike*, l'un des deux contrôles de congestion actuellement présents dans le nouveau protocole de transport DCCP (*Datagram Congestion Control Protocol*). Nos résultats indiquent que *EcnLD* est une bonne approche pour optimiser le contrôle de congestion et donc les performances de protocoles de transport sur les réseaux sans-fil.**

Mots-clés : Réseaux sans-fil, Protocole de transport, Contrôle de congestion, ECN, RED.

I. Introduction

A. La différentiation de pertes

Les réseaux sans-fil sont aujourd'hui largement déployés et sont souvent utilisés pour accéder à des services sur Internet. Pourtant, les protocoles de transport n'offrent pas les mêmes performances sur les réseaux filaires que sur les réseaux sans-fil [1][2]. Dans les réseaux filaires, les pertes sont principalement dues aux congestions des routeurs car ceux-ci rejettent les paquets reçus quand leurs files d'attente sont remplies ou bien même quand elles commencent à se remplir. C'est la raison pour laquelle les pertes dans les réseaux filiares sont considérées comme une indication de congestion. Par contre, dans les réseaux radio, les pertes se produisent souvent pour d'autres causes que la congestion, par exemple, à cause d'une interférence, d'une mauvaise qualité de la liaison radio, ou de l'éloignement de la station de base et du mobile.

Ce qui entraîne la dégradation des performances des protocoles de transport sur les réseaux sans-fil c'est le fait que le protocole TCP (*Transport Control Protocol*) [3], conçu généralement pour les réseaux filaires, et largement utilisé par les applications d'Internet, considère toute perte de données comme une perte liée à une congestion ce qui n'est pas toujours le cas. Par conséquent, il réduit le taux de transmission ce qui n'est pas nécessaire en cas de pertes sans-fil.

Il existe de nombreuses propositions sur la manière d'optimiser les performances des protocoles de transport sur les réseaux sans-fil. L'idée principale commune est que le protocole de transport ne devrait réduire son taux de transmission que dans le cas de congestion [1], [4], [5], [6].

B. Le contrôle de congestion

Le contrôle de congestion est une caractéristique majeure du protocole TCP. Il est utilisé par TCP pour atteindre de hautes performances et éviter l'effondrement du réseau (*congestion collapse*). Le mécanisme de contrôle de congestion essaie de maintenir le taux des données entrant dans le réseau en-dessous du taux qui cause une congestion. Le récepteur acquitte toutes les données envoyées par l'émetteur ce qui lui permet d'en déduire les conditions du réseau. En effet, les informations obtenues à partir de ces accusés de réception ou même leurs absences renseignent l'émetteur. Ce mécanisme de contrôle de congestion fait appel à deux algorithmes principaux : le démarrage lent (*Slow Start*) et l'évitement de congestion (*Congestion Avoidance*). En mode démarrage lent, le débit augmente de façon exponentielle, alors que dans le mode évitement de congestion il augmente de manière linéaire. A chaque fois que l'émetteur détecte une perte, il divise à moitié son débit et il utilise un mécanisme de retransmission pour les paquets perdus.

C. L'acquittement sélectif ou option TCP SACK

La version originale du protocole TCP utilise un schéma d'acquittement cumulatif pour acquitter des groupes de paquets. Dans ce schéma, si dans un groupe de paquets envoyés c'est le premier paquet qui est perdu, le récepteur ne peut pas dire qu'il a reçu les autres paquets. Ainsi, l'émetteur pense que tout le groupe est perdu et il retransmet tous les paquets de ce groupe. Pour éviter cette retransmission, il est possible d'utiliser le schéma d'acquittement sélectif *(Selective acknowledgment)*, option *(SACK)*, défini dans la RFC2018 [7], grâce auquel le récepteur est autorisé à acquitter tous les paquets correctement reçus. Ainsi, l'émetteur peut retransmettre plus rapidement les paquets perdus.

D. Les protocoles de transport pour les applications temps réel

Aujourd'hui, les applications de streaming vidéo sont utilisées partout sur Internet et elles souffrent de la trop grande

fiabilité du protocole TCP. En effet, une vidéo pourrait accepter un certain niveau de pertes sans trop dégrader la qualité de la vidéo. D'autre part, UDP *(User Datagram Protocol)* [8], qui est largement utilisé pour le streaming multimédia, manque d'un mécanisme d'évitement de congestion et d'un mécanisme de contrôle de flux. En conséquence, UDP pourrait conduire dans de nombreux cas, à une très faible qualité des données reçues à cause de la passivité de flux UDP.

DCCP (*Datagram Congestion Control Protocol*), récemment standardisé dans la RFC4340 [9] est un protocole de transport sans fiabilité, mais avec un contrôle de congestion, ce qui le rend plus adapté pour la diffusion de données multimédia. Son point fort est de permettre le choix du protocole de contrôle de congestion entre *TCPLike* [10], qui est similaire à TCP avec l'option SACK activée, ou TFRC *(TCP-Friendly Rate Control)* [11]. Comme décrit dans [9], DCCP implémente des connexions bidirectionnelles et unicasts. Il possède, en outre, les propriétés suivantes : négociation du mécanisme de contrôle de congestion et mécanismes d'acquittement afin de communiquer des informations concernant les paquets perdus et les paquets marqués ECN (*Explicit Congestion Notification*), voir II-A. Mais 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.

A partir de toutes ces constatations et parce que les réseaux sans-fil sont aujourd'hui largement déployés, il nous semble pertinent de modifier le contrôle de congestion pour qu'il prenne en compte la cause des pertes. Ainsi, dans ce papier nous proposons une nouvelle approche, *EcnLD (Ecn Loss Differentiation*, différenciation de perte basée sur ECN) sur la base de *TCPLike* dans DCCP. D'une part, *EcnLD* utilise ECN comme principal facteur de différenciation de pertes. D'autre part, *EcnLD* a recours au *RTT Round Trip Time* afin de palier, dans plusieurs cas, à la faiblesse d'ECN pour différencier les pertes dues au sans-fil de celles dues à la congestion (voir section II-B).

Ce document est organisé comme suit : La section II montre la stratégie adoptée par *EcnLD* en tant que nouvelle méthode de classification de pertes. Dans la section III, nous présentons nos résultats de simulation pour évaluer les performances de *EcnLD*. La section IV présente un état de l'art des méthodes utilisées pour distinguer les pertes dues à la congestion de celles dues aux perturbations sur le réseau sans-fil. Enfin, la section V énonce les conclusions et quelques perspectives.

II. *EcnLD*, ECN Loss Differentiation

Pour être en mesure de faire la différence entre les pertes dues à la congestion de celles dues aux perturbation du canal sans-fil, *EcnLD* nécessite que les routeurs intermédiaires entre l'émetteur et le récepteur soient compatibles ECN. Cela impose aux routeurs d'appliquer une gestion active de la file d'attente *Active queue management* de type RED *Random Early Detection*.

A. *Le principe d'ECN,* Explicit Congestion Notification

ECN est une extension d'IP qui est définie dans la RFC3168 [12] et qui fonctionne avec RED, voir section suivante. ECN permet la notification de la congestion de bout en bout dans le réseau sans perdre de paquets. Il est un élément facultatif, et est utilisé seulement lorsque les deux points de terminaison de connexion veulent l'utiliser. Ainsi, un routeur compatible avec ECN met à jour un bit dans l'en-tête IP des paquets pour indiquer la congestion. En réponse à ce signal ECN, le destinataire du paquet acquitte le paquet en le marquant aussi pour l'expéditeur qui réagit, à son tour, comme si c'était un paquet perdu.

B. *Gestion active de la file d'attente, RED*

De nos jours, les routeurs implémentant une stratégie active de gestion des files d'attente comme RED sont nombreux. A l'instar de *Droptail* qui rejette les paquets seulement quand la file d'attente est pleine, ce qui introduit une congestion dans le réseau et une inégalité de partage des ressources, RED exerce un partage équitable entre les différents flux. Il est utilisé également pour la gestion de la congestion en fournisant des informations *feedbacks* négatives aux émetteurs par le rejet aléatoire de paquets de sa file d'attente afin de signaler l'arrivée d'une congestion. ECN propose de marquer ces paquets au lieu de les rejeter. Pour ce faire, RED maintient de nombreuses valeurs : la taille de la file d'attente q_l ; la moyenne de la file d'attente q_{ave} ; le seuil minimum de la file d'attente q_{th_min} et le seuil maximum de la file d'attente q_{th_max}. Si $q_{ave} < q_{th\_min}$, tous les paquets passent sans rejet ou marquage ECN ; entre th_min et th_max les paquets sont marqués ou rejetés avec une probabilité qui augmente avec l'augmentation de q_{ave}. Quand $q_{ave} > q_{th\_max}$ tous les paquets sont rejetés.

C. *Fonctionnement de la méthode* EcnLD

A notre connaissance, il n'y a dans la littérature que *TCP-Eaglet* qui fait appel à ECN pour une classification de pertes. *TCP-Eaglet* évite d'une part de traiter les pertes arrivant en *Slow Start*. D'autre part, il ne prend pas en considération le cas où une rafale de paquets arrive tout d'un coup à un routeur où la taille des données dépasse largement la capacité de sa file d'attente. Il peut y avoir ainsi un nombre important de pertes sans marquage ECN même en mode *Congestion Avoidance*. A l'instar de *TCP-Eaglet*, *EcnLD* prend ces situations en compte. D'abord, il ne fait pas de différence entre *Slow Start* et *Congestion Avoidance*. Ensuite, il a recours au *RTT* pour palier aux insuffisances d'ECN. De cette manière, *EcnLD* couple ECN et *RTT* pour faire une méthode qui est d'un côté performant et d'un autre côté compatible avec les réseaux existants.

En résumé, on considère qu'une perte est due à une congestion si et seulement si :

1) $ecn > 0$
2) OU $n > 0$ ET $RTT_{cur} > RTT_{ave} + (RTT_{var}/2)$
   où ecn represente le nombre de paquets marqués EC, n le nombre de pertes indiqué par l'accusé de réception, RTT_{cur} le *RTT* actuel, RTT_{ave} la moyenne de *RTT* et (RTT_{var}) la déviation de *RTT*.

III. Mesures de performance

A. *Topologie de simulation*

La figure 1 représente la topologie du réseau employé pour la réalisation de simulations. Dans cette topologie, nous utilisons un scénario mixte filaire/sans-fil. Elle se compose de deux émetteurs établissant des connexions avec deux récepteurs. la liaison entre les routeurs R1 et R2 est de capacité C 11MBit/s. Les liens entre le routeur R1, les émetteurs/récepteurs et les

liens entre le routeur R2 et les émetteurs/récepteurs sont d'une capacité de 50Mbits/s plus grande que C, afin que le lien R1-R2 soit le goulot d'étranglement du réseau filaire. Le réseau sans-fil utilise la norme 802.11bg pour les communications sans-fil et a donc une bande passante de 11Mb/s et 54Mb/s. Chaque nœud fait appel à RED pour la gestion de la file d'attente qui utilise les valeurs par défaut. ECN est activé sur tous les routeurs. La taille du paquet est de 500 octets et le temps de simulation est de 50 secondes.

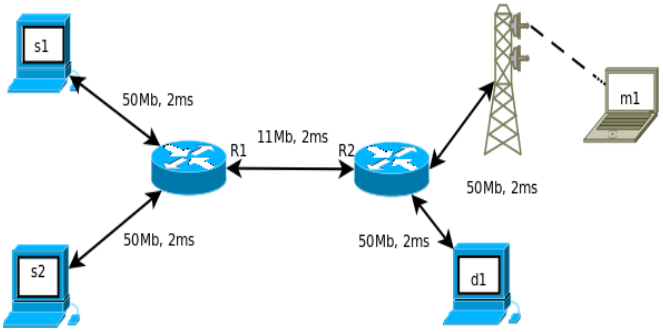

FIG. 1. La topologie du réseau de la simulation

Les tests sont faits avec le simulateur ns2 version 2.31 [13], qui est utilisé pour évaluer la performance d'*EcnLD* par rapport au *TCPLike* d'origine et ensuite, pour mesurer la pertinence d'*EcnLD* et pour comparer les résultats de classification à ceux obtenus par *TCP-Eaglet*. Pour ce faire, nous avons ajouté un modèle d'erreur sur le réseau sans-fil qui varie de 0% à 10% ce qui représente la plupart des cas dans la réalité à notre opinion. Nous avons également implémenté la méthode *TCP-Eaglet*. Tout cela sur une version de DCCP écrite par [14], maintenue par nos soins [1] et implémentée dans ns2. La section suivante présente nos résultats de simulation.

B. Les résultats de simulation

1) EcnLD *vs* TCPLike*:* Pour montrer le gain de débit grâce à la classification des pertes d'*EcnLD*, nous comparons les performances d'*EcnLD* avec celles de *TCPLike*. Pour cela, une simulation est répétée onze fois, une fois sans taux d'erreur et dix fois avec un taux d'erreur qui varie de 1% à 10% sur la partie radio. Ce même test est réalisé deux fois en changeant le nombre de retransmissions MAC (une et deux retransmissions au niveau de la couche MAC en plus de la transmission initiale) et ce, pour chaque type de contrôle de congestion, *EcnLD* et *TCPLike*. Le but de ce changement est de pouvoir obtenir un nombre significatif de pertes radio ce qui arrive dans la réalité mais pas dans ns2. L'émetteur de cette simulation est le nœud s1 qui crée une connexion avec le mobile m1 sur la partie sans-fil. Nous présentons les résultats dans deux cas : d'abord, sans avoir une concurrence sur le réseau. Ensuite, en présence d'un flux TCP, entre l'émetteur s2 et le récepteur d1, qui apparaît deux fois : de 1 à 20 secondes et de 25 à 45 secondes dont le but est de créer un trafic en mode *Slow Start* plusieurs fois quand les files d'attente risquent d'être pleines d'un seul coup.

[1]http://lifc.univ-fcomte.fr/~dedu/ns2

1) **Scénario sans concurrence :** Dans ce scénario, le flux testé profite seul de toutes les ressources disponibles. Les figures 2 et 3 montrent les résultats de comparaison en fonction de la bande passante occupée pendant la communication entre *EcnLD* et *TCPLike*, et le tableau I présente en détail et par nombre de paquets de données envoyés et reçus par l'émetteur/récepteur *EcnLD* et *TCPLike*. On peut noter que dans la figure 2, *EcnLD* prend le dessus et que cette supériorité augmente avec l'augmentation du taux de pertes. En effet, le récepteur d'*EcnLD* reçoit entre environ 1000 et 5000 paquets de données supplémentaires par rapport au récepteur *TCPLike* et c'est pour un taux d'erreur supérieur à 3%. Par contre, la différence diminue entre les deux avec la hausse du nombre de retransmissions ce qui s'explique par le nombre très petit de pertes radio avec la retransmission des paquets erronés par le point d'accès. A remarquer également que *EcnLD* a pu avoir une meilleure performance dans le cas de deux transmissions "Une seule retransmission" MAC que de trois. Cela vient du fait que le nombre de retransmissions MAC a une influence non-négligeable sur le *RTT*.

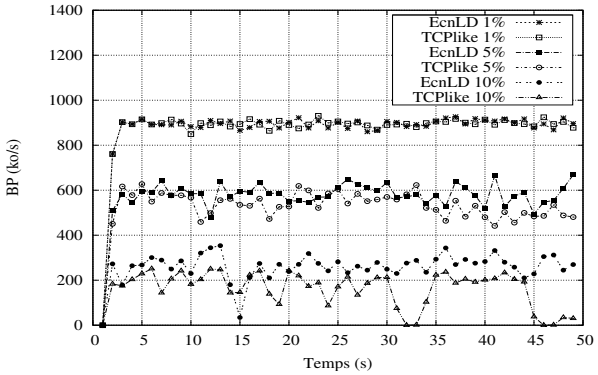

FIG. 2. *EcnLD* vs *TCPLike* pour une retransmission Mac.

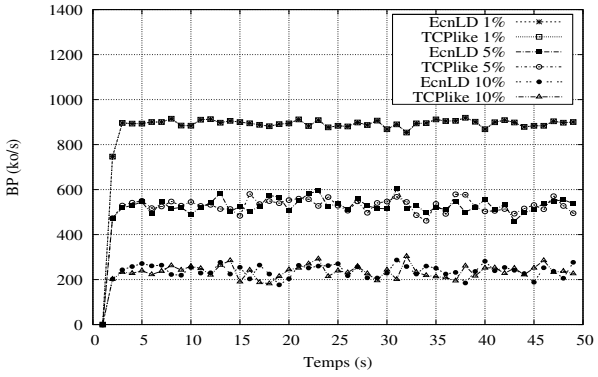

FIG. 3. *EcnLD* vs *TCPLike* pour deux retransmissions Mac.

2) **Scénario en concurrence avec un flux TCP :** Les figures 4 et 5 et le tableau II montrent les résultats de cette simulation. Il est évident que *EcnLD* a l'avantage par rapport à *TCPLike* grâce à sa capacité à distinguer les pertes dues à la congestion de celles dues au sans-fil. Cet avantage apparaît clairement sur le tableau II, pour une

Une retransmission	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
EcnLD/ Envoi	46705	42238	38012	34699	31259	27736	24539	21608	18558	16077	12865
TCPLike/ Envoi	46705	42235	37918	33809	29425	25555	21453	17815	13831	10906	8121
EcnLD/ Réception	46627	42124	37849	34496	31040	27396	24184	21153	18103	15631	12374
TCPLike/ Réception	46627	42125	37774	33597	29230	25300	21167	17475	13498	10558	7791
Deux retransmissions	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
EcnLD/ Envoi	46705	42124	37527	33187	29126	25071	21751	18975	16426	13454	11408
TCPLike/ Envoi	46705	42124	37527	33187	29133	25081	21774	18337	16020	13131	11347
EcnLD/ Réception	46627	42040	37453	33086	29006	24957	21628	18764	16096	13240	11235
TCPLike/ Réception	46627	42040	37453	33086	29034	24961	21625	18215	15867	12961	11168

TAB. I
EcnLD VS *TCPLike* SANS CONCURRENCE EN FONCTION DU NOMBRE DES PAQUETS ENVOYÉS/REÇUS.

retransmission où la différence est assez significative, environ 1000 et 4000 paquets de plus reçus par m1. Par contre, cette différence est beaucoup moins importante pour deux retransmissions dues au nombre très petit des pertes sans-fil après trois transmissions MAC ce qui n'est pas le cas dans la réalité. Dans le cas ou il n'y a pas d'erreur sur le sans-fil, les performances dans les deux cas sont identiques ce qui prouve qu'*EcnLD* exerce le même comportement que *TCPLike* dans des situation habituelles. On peut également remarquer dans les deux figures que *EcnLD*, à l'instar de *TCPLike*, a une tendance d'avoir une variation de *RTT* assez petite et il n'y a dans aucun cas une chute importante du débit.

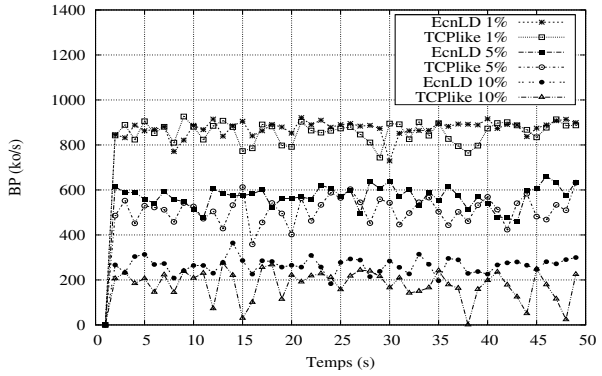

FIG. 4. *EcnLD* vs *TCPLike* pour une retransmission Mac et un flux TCP.

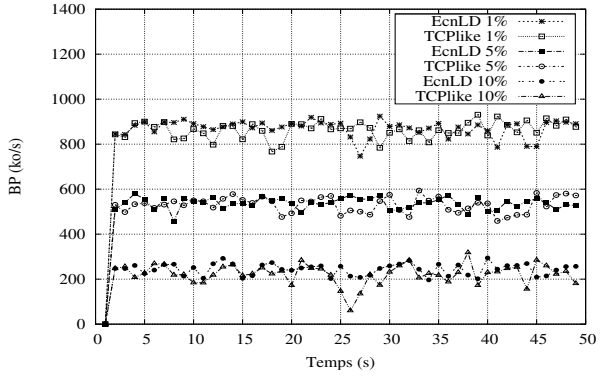

FIG. 5. *EcnLD* vs *TCPLike* pour deux retransmissions Mac et un flux TCP.

2) EcnLD *vs Eaglet:* Sachant que *TCP-Eaglet* n'est pas proposé pour être utilisé avec DCCP, nous nous sommes intéresses à valider notre méthode par rapport à son hypothèse.

1) Scénario avec concurrence sur un réseau sans-fil de 11Mb/s :
   Nous comparons dans cette simulation *EcnLD* et *TCP-Eaglet* dans les mêmes conditions utilisées pour comparer *EcnLD* et *TCPLike*. Le taux de classification de pertes pour *EcnLD* et *TCP-Eaglet* est un pourcentage représentant le taux des pertes correctement distinguées en utilisant chacun sa méthode sur le nombre total de pertes pendant la simulation. Le calcul de ce taux nécessite la connaissance parfaite de la cause de perte. Les résultats de classification sont montrés dans le tableau III. Premièrement, dans le cas d'une seule retransmission Eaglet a pu se montrer meilleur qu'*EcnLD*, que ça soit d'un point de vue classification avec un taux bien plus élevé ou que d'un point de vue performance surtout à partir de 6% de taux de perte. La raison de cette mauvaise note d'*EcnLD* est qu'il a pris assez de précautions pour ne pas provoquer une congestion dans le réseau. Deuxièmement, dans le cas deux retransmissions, *EcnLD* s'est stabilisé en ayant un taux de classification moyenne de 79% contre environ 63.6% pour Eaglet.
2) Scénario avec concurrence sur un réseau sans-fil de 54Mb/s
   Ce test a pour but de montrer que ECN ne suffit pas tout seul pour différencier les pertes et faire de sorte que l'émetteur n'abuse pas en transmettant des paquets plus que le réseaux ne peut accepter. On met dans cette simulation la capacité du réseaux sans-fil à 54Mb/s dans le but d'avoir un goulot d'étranglement sur le réseaux filaire ou ce goulot est partagé entre plusieurs flux. Les résultats de classification entre les pertes de congestion et les pertes radio, voir tableau IV montrent que le taux de fiabilité de classification d'Eaglet varie entre 2.5% et environ 41% contre entre 52.2% et 74.6% pour *EcnLD* dans le cas d'une transmission MAC et il est de 2% à 16.2% au lieu d'entre 41.4% et 74.6% dans le cas de deux retransmission. Ce taux est nettement supérieur pour *EcnLD* que pour Eaglet la raison pour laquelle Eaglet exerce un taux de réception plus élevé mais en envoyant beaucoup plus de paquets sur le réseaux ce qui le rend pas adapté au transfert à large échelle. Dans le cas d'*EcnLD* presque la plupart les paquets envoyés sont reçus grâce à sa capacité d'adapter aux conditions du réseau sans pour autant perdre beaucoup de performance.

IV. TRAVAUX ANTÉRIEURS

Dans la littérature, de nombreux types de méthodes ont été proposées pour rendre possible la différenciation de pertes :

Une Retransmission	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
EcnLD/ Envoi	44890	41291	37528	34074	30656	27224	23769	20577	18235	15406	12991
TCPLike/ Envoi	44890	40449	37555	32672	28572	24365	19983	16480	14069	11304	8857
EcnLD/ Réception	44842	41167	37377	33879	30396	26911	23416	20201	17842	14976	12496
TCPLike/ Réception	44842	40352	37418	32475	28334	24132	19713	16206	13733	10966	8524
Deux Retransmissions	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
EcnLD/ Envoi	44890	41148	36992	32483	28924	25386	21702	18831	15571	13528	11701
TCPLike/ Envoi	44890	40811	36875	33204	28869	25131	21360	18532	15833	13415	10759
EcnLD/ Réception	44842	41062	36902	32334	28805	25319	21566	18703	15431	13370	11526
TCPLike/ Réception	44842	40727	36783	33072	28780	25055	21238	18399	15669	13281	10567

TAB. II
EcnLD VS *TCPLike* EN PRÉSENCE D'UN FLUX TCP.

Une retransmission	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
EcnLD/Envoi	44890	41291	37528	34074	30656	27224	23769	20577	18235	15406	12991
Eaglet/Envoi	44890	41369	37782	34568	31247	28065	24823	22575	19519	17348	15149
EcnLD/Réception	44842	41167	37377	33879	30396	26911	23416	20201	17842	14976	12496
Eaglet/Réception	44842	41237	37624	34346	30957	27770	24475	22171	19084	16840	14521
EcnLD/ Correct	100%	72%	70%	69%	61%	62%	53%	53%	55%	58%	55%
Eaglet/ Correct	100%	59%	84%	84%	86%	92%	90%	92%	89%	87%	88%
Deux retransmissions	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
EcnLD/ Envoi	44890	41148	36992	32483	28924	25386	21702	18831	15571	13528	11701
Eaglet/ Envoi	44890	40626	37071	33256	28753	25387	21603	18611	16089	13510	11605
EcnLD/ Réception	44842	41062	36902	32334	28805	25319	21566	18703	15431	13370	11526
Eaglet/ Réception	44842	40485	36895	32961	28584	25307	21432	18457	15865	13246	11297
EcnLD/ Correct	100%	83%	85%	82%	76%	76%	73%	69%	81%	65%	79%
Eaglet/ Correct	100%	60%	50%	47%	50%	81%	79%	77%	51%	51%	54%

TAB. III
EcnLD VS EAGLET AVEC CONCURRENCE SUR UN RÉSEAU SANS-FIL DE 11MB/S.

Une retransmission	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
EcnLD/ Envoi	68422	64469	57823	51423	44831	37883	32717	27612	22894	19622	15462
Eaglet/ Envoi	71681	67805	63667	58773	52276	48371	41069	34946	28619	27285	22626
EcnLD/ Réception	68270	64249	57609	51223	44491	37465	32251	27100	22365	18842	14865
Eaglet/ Réception	68044	63103	59707	54360	48754	44541	38828	32745	26501	24413	20404
EcnLD/ Correct	74.6%	53.0%	61.3%	52.2%	58.0%	59.1%	60.7%	60.7%	55.8%	64.6%	55.7%
Eaglet/ Correct	3.8%	2.5%	4.1%	4.5%	10.1%	11.9%	26.3%	29.2%	32.8%	29.5%	40.8%
Deux retransmissions	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
EcnLD/ Envoi	68422	64497	58629	51673	44749	39028	32736	26685	22042	17797	14584
Eaglet/ Envoi	71681	68649	61015	55672	50548	41841	34525	30121	24492	20775	17062
EcnLD/ Réception	68270	64279	58454	51455	44535	38691	32424	26536	21933	17675	14305
Eaglet/ Réception	68044	63626	57013	52145	46117	39379	32503	27445	22773	18795	15299
EcnLD/ Correct	74.6%	55.2%	60.7%	52.5%	51.1%	41.4%	51.9%	50.9%	48.9%	54.9%	67.6%
Eaglet/ Correct	3.8%	2.0%	2.2%	2.8%	2.5%	5.9%	9.5%	5.6%	10.5%	8.5%	16.2%

TAB. IV
EcnLD VS EAGLET AVEC CONCURRENCE SUR UN RÉSEAU SANS-FIL DE 54MB/S.

1) Le premier type nécessite la mise en place d'un agent intermédiaire entre la source et la destination qui est normalement localisé sur la station de base. Snoop [2], [1], par exemple, est une approche TCP-aware qui fait une retransmission locale. Il s'installe sur un routeur ou sur une station de base. Puis, il enregistre une copie de chaque paquet transmis. Ensuite, il examine chaque paquet ACK et réalise des retransmissions locales quand un paquet est corrompu sur le canal sans-fil. D'autres approches, comme ELN (*Explicit Loss Notification*) [6] peuvent également être utilisées pour informer l'émetteur que la perte s'est produite sur le réseau sans-fil ou bien sur le réseau filaire. En dépit du fait que ce type d'approche est très intéressant, il est nécessaire d'apporter des modifications sur les équipements du réseau, en particulier la station de base. En outre, cette approche nécessit plus de traitement sur la station de base.
2) Le deuxième type regroupe les mécanismes de bout en bout qui ne nécessitent aucune modification des équipements du réseau. Ces méthodes peuvent être classées en deux grandes catégories : les méthodes basées sur *IAT* (*Inter Arrival Time*) et celles basées sur le temps d'aller *ROTT* (*Relative One-way Trip Time*) :
   a) Biaz [5] et mBiaz [15] utilisent *IAT* des paquets à la réception pour classifier les pertes. Biaz considère que si un paquet arrive avant son temps prévu et qu'il y a un paquet perdu, cette perte est due à une congestion antérieure. Pour les pertes de sans-fil, le premier paquet reçu après la perte devrait arriver au moment où il aurait dû être arrivé, c'est-à-dire pour n paquets perdus, si $(n+1)\,T_{min} \leq T_i < (n+2)\,T_{min}$ donc les n paquets sont considérés comme congestion. Ils sont des pertes sans-fil sinon.
   mBiaz est une version modifiée de Biaz qui améliore la classification de congestion de Biaz comme suit :
   $(n+1)\,T_{min} \leq T_i < (n+1.25)\,T_{min}$.
   b) SPLD (*Statistical Packet Loss Discrimination*) [16] repose aussi sur *IAT*. Cette méthode comporte trois modules. Un module de surveillance pour recueillir

des informations sur les paquets reçus. Ainsi, si pendant un certain temps il n'y a pas de pertes, le deuxième module, celui des statistiques, met à jour le minimum et la moyenne de *IAT*. Par contre, si une perte se produit, le troisième module, le discriminant, utilise IAT_{avg} pour classifier les pertes. SPLD considère la perte comme congestion si *IAT* est supérieur ou égal à *IAT* stable (IAT_{avg}) ; c'est une perte de sans-fil sinon.

c) Spike [17] est aussi une méthode basée sur ROTT. Dans Spike, la connexion a deux états : *spike* ou non. Une perte est considére comme congestion si la conexion est en état *spike*, sinon c'est une perte sans-fil. On entre en état *spike* si ($ROTT > B_{spikestart}$; (le seuil indiquant le maximum de *ROTT*) et on quitte cet état si ($ROTT > B_{spikeend}$ (le seuil indiquant le minimum de *ROTT*).

d) Outre la déviation et la moyenne de *ROTT*, ZigZag [15] est basé sur le nombre de pertes n. Ainsi, si :
   i) ($n = 1$ ET $rott_i < rott_{mean} - rott_{dev}/2$)
   ii) OU ($n = 2$ ET $rott_i < rott_{mean}$)
   iii) OU ($n = 3$ ET $rott_i < rott_{mean} - rott_{dev}$)
   iv) OU ($n > 3$ ET $rott_i < rott_{mean} - rott_{dev}/2$)
   Les n pertes sont dues à la liaison sans-fil, sinon à la congestion.

e) ZBS, décrit dans [15], est un algorithme hybride entre ZigZag, mBiaz et Spike. Il choisit l'un d'entre eux en fonction des conditions suivantes du réseau :
   Si ($rott < (rott_{min} + 0.05 * T_{min})$) utiliser Spike ;
   Sinon
   Si ($Tnarr < 0.875$) utiliser ZigZag ;
   Sinon si ($Tnarr < 1.5$) utiliser mBiaz ;
   Sinon si ($Tnarr < 2.0$) utiliser ZigZag ;
   Sinon : utiliser Spike.
   Où $Tnarr = T_{avg}/T_{min}$ (la moyenne et le minimum de *IAT*).

L'évaluation de performance faite dans [18] montre que les méthodes basées sur *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*.

3) Dans le troisième type de méthodes, l'émetteur fait appel à ECN (*Explicit congestion notification*), proposé dans [12] comme un simple mécanisme permettant de différencier les pertes causées par la congestion des pertes dues aux erreurs de la liaison sans-fil. L'utilisation normale d'ECN pour différencier les pertes est de regarder le dernier intervalle de temps dans lequel une perte est arrivée. Si la source a déjà reçu un ECN, cela indique la présence d'une congestion, sinon et s'il existe des paquets perdus, on conclut que c'est dû au sans-fil. *TCP-Eaglet* [4] fait l'hypothèse que de cette manière le marquage ECN ne peut pas être toujours utilisé pour distinguer la congestion des pertes sans-fil. Il propose de réduire le débit seulement lorsque TCP est en mode de démarrage lent, ou bien s'il y a marquage ECN dans le mode d'évitement de la congestion.

V. Conclusion

Nous avons proposé dans cet article une méthode générale qui résoud certains problèmes liés aux faibles performances des protocoles de transport dans les réseaux sans-fil où les pertes sont temporaires, aléatoires et n'ont aucun impact sur la bande passante disponible. Cette étude a montré que le contrôle de congestion peut être plus performant sur les réseaux radio en faisant de sorte que le traitement des paquets perdus sur le sans-fil ne soit pas le même que des pertes dues à la congestion. Nous avons montré que ECN peut être utilisé avec le *RTT* pour palier à son occasionnelle faiblesse, dans le but de faire une classification acceptable entre les pertes de la congestion et les pertes radio. Nous recommandons *EcnLD* pour le transfert de la vidéo sur réseaux sans-fil car son taux de réception est très élévé (la majorité des paquets envoyés sont reçus). Nous avons encore des pistes à suivre dans cette étude en particulier pour savoir quel est l'impact du taux d'erreur sur le *RTT*, *IAT* et *ROTT* et faire le point sur les méthodes basées sur ces paramètres. D'autre part, il nous reste à faire des études pour évaluer la performance d'*EcnLD* pour une transmission réelle de la vidéo.

Références

[1] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 756–769, 1997.

[2] H. Balakrishnan, S. Seshan, and R. H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," *Wirel. Netw.*, vol. 1, no. 4, pp. 469–481, 1995.

[3] J. Postel, "RFC 793 : Transmission control protocol," Sep. 1981.

[4] S. Biaz and X. Wang, "Red for improving TCP over wireless networks," in *International Conference on Wireless Networks*, 2004, pp. 628–636.

[5] S. Biaz and N. H. Vaidya, "Discriminating congestion losses from wireless losses using inter-arrival times at the receiver," *IEEE Symposium ASSET'99*, pp. 10–17, 1999.

[6] H. Balakrishnan and R. Katz, "Explicit Loss Notification and Wireless Web Performance," in *IEEE GLOBECOM Global Interne*, Sydney, Australia, November 1998.

[7] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgment options," RFC 2018, Oct. 1996.

[8] J. Postel, "RFC 768 : User datagram protocol," Aug. 1980.

[9] E. Kohler, M. Handley, and S. Floyd, "Datagram congestion control protocol (DCCP)," RFC 4340 (Proposed Standard), Mar. 2006.

[10] S. Floyd and E. Kohler, "Profile for datagram congestion control protocol (DCCP) congestion control ID 2 : TCP-like congestion control," RFC 4341 (Proposed Standard), Mar. 2006.

[11] S. Floyd, E. Kohler, and J. Padhye, "Profile for datagram congestion control protocol (DCCP) congestion control ID 3 : TCP-friendly rate control (TFRC)," RFC 4342 (Proposed Standard), Mar. 2006.

[12] K. Ramakrishnan, S. Floyd, and D. Black, "The addition of explicit congestion notification (ECN) to IP," RFC 3168 (Proposed Standard), Sep. 2001.

[13] "The Network Simulator NS-2 (v2.31)," http://www.isi.edu/nsnam/ns/.

[14] N.-E. Mattsson, "A DCCP module for ns-2," Master's thesis, Luleå University of Technology, Sweden, May 2004.

[15] S. Cen, P. C. Cosman, and G. M. Voelker, "End-to-end differentiation of congestion and wireless losses," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 703–717, 2003.

[16] K.-H. S. Min Kyu Park and J. H. Jeong, "A statstical method of packet loss type discriminartion in wired-wireless network," in *IEEE CCNC*, Las Vegas, USA, January 2006.

[17] Y. Tobe, Y. Tamura, A. Molano, S. Ghosh, and H. Tokuda, "Achieving moderate fairness for UDP flows by path-status classification," in *LCN '00 : Proceedings of the 25th Annual IEEE Conference on Local Computer Networks*. Washington, DC, USA : IEEE Computer Society, 2000, p. 252.

[18] A. Boukerche, G. Jia, and R. W. N. Pazzi, "Performance evaluation of packet loss differentiation algorithms for wireless networks," in *Proceedings of the 2nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks, PM2HW2N*. New York, NY, USA : ACM, 2007, pp. 50–52.