Réf:2001/ / / *Soutenu à la session de Juillet 2001*

Université Manouba

ECOLE NATIONALE DES SCIENCES DE L'INFORMATIQUE

RAPPORT

DE MÉMOIRE DE FIN D'ÉTUDES

Présenté en vue de l'obtention du titre

d'**INGÉNIEUR EN INFORMATIQUE**

Par:

Abdelbasset TRAD

SUJET:

Étude des performances de TCP dans un environnement à très haut débit

Encadré par:

Mr **Hossam AFIFI**

Organisme:

INRIA Sophia Antipolis

À la mémoire de mon oncle
À mes très chers parents
pour leurs sacrifices et leur patience
À mon frère et à mes soeurs
À toute ma famille
À tous mes amis
Je dédie ce travail.

Remerciements

J'exprime ma profonde gratitude à Monsieur Hossam AFIFI pour son encadrement, ses conseils judicieux et ses remarques pertinentes qui m'ont aidé à mener à bien mon stage.

Je tiens à exprimer mes remerciements les plus vifs à Monsieur Walid DABBOUS, Chef du projet Planète à l'unité de recherche INRIA Sophia Antipolis, pour sa disponibilité et son encouragement.

Je remercie également tous les membres du projet Planète pour leur amitié, ainsi que pour l'ambiance de travail agréable qui règne au sein du projet.

Résumé

Le contrôle de flux est une tâche primordiale permettant d'assurer l'utilisation efficace de la bande passante disponible sur les liens réseaux. Le protocole de transport fiable TCP responsable de cette tâche a prouvé une grande performance de fonctionnement sur différents types de liens réseaux.

L'introduction de la fibre optique comme support de transmission et de la technique de multiplexage en longueur d'onde WDM relative ont permis d'atteindre des vitesses de transmission très élevées.

Ce mémoire a pour objectif d'étudier les différents mécanismes de contrôle de flux TCP et d'évaluer leur performances sur une connexion à très haut débit.

Mots-clés

TCP, contrôle de flux, multiplexage en longueur d'onde (WDM), réseaux haut débit, bande passante, débit, expérimentations.

Abstract

The flow control is a primordial task to assure an efficient utilization of the bandwidth available on network links. The reliable transport protocol TCP responsible for this task has proven high performance on different types of network links.

The introducion of fiber optic and related Wavelength Division Multiplexing technology is resulting in ever-higher transmission speeds.

This work aims to study TCP flow control mecanisms and to evaluate their performances on a high speed fiber optic connection.

Keywords

TCP, flow control, Wavelength Division Multiplexing (WDM), high speed networks, bandwidth, throughput, experimentations.

Table des matières

Table des figures

Chapitre 1

Introduction générale

Ce stage de mémoire de fin d'études s'est déroulé à l'unité de recherche INRIA Sophia Antipolis au sein du projet Planète.

1.1 Présentation de l'INRIA

L'INRIA Institut National de Recherche en Informatique et en Automatique est un établissement public à caractère scientifique et technologique sous la double tutelle des ministères chargés de la recherche scientifique et de l'industrie.

Il est constitué de cinq unités de recherche qui sont:

- L'unité de recherche de Paris Rocquencourt.
- L'unité de recherche de Rennes créée en 1980.
- L'unité de recherche de Sophia-Antipolis créée en 1983.
- L'unité de recherche de Lorraine créée en 1984.
- L'unité de recherche de Rhône-Alpes 1992.

L'institut s'est fixé cinq objectifs majeurs dans son plan stratégique pour la période 1999-2003 :

- Maîtriser l'infrastructure numérique en sachant programmer, calculer et communiquer sur l'Internet et sur des réseaux hétérogènes,

- Concevoir de nouvelles applications exploitant le Web et les bases de données multimédias,
- Savoir produire des logiciels sûrs,
- Concevoir et maîtriser l'automatique des systèmes complexes,
- Combiner simulation et réalité virtuelle.

Le principe d'organisation de la recherche de l'INRIA est de créer des projets de recherche comprenant au plus 25 personnes avec des objectifs clairement définis et un chef de projet devant coordonner les travaux des membres de l'équipe.

Quelques chiffres [1]

Ressources budgétaires

Dotation de l'état : 442 MFr HT

Ressources propres : 174 MFr HT

Ressources humaines

Titulaires INRIA : 724

Post-Doctorants et Contractuels: 256

Doctorants : 550

Chercheurs et enseignants d'autres organismes : 230

Conseillers, collaborateurs divers et invités : 430

L'organisation décentralisée (5 unités de recherche), les petites équipes autonômes et évaluées régulièrement de l'INRIA lui ont permis d'amplifier ses partenariats, 47 projets de recherche sur 87 sont communs avec les universités, les grandes écoles et les organismes de recherche, et de renforcer son implication dans les travaux de valorisation des résultats de recherche et le transfert technologique : 600 contrats avec l'industrie et un peu moins d'une cinquantaine de sociétés sont issues de l'INRIA.

1. Ces chiffres datent de décembre 2000

1.2 L'unité de recherche INRIA Sophia Antipolis

L'unité de recherche de l'INRIA à Sophia Antipolis a été créée en 1983. Des recherches et des actions de développement y sont menées sur les thèmes retenus par l'INRIA, centrés autour des technologies de l'information:

- réseaux et systèmes,
- génie logiciel et calcul symbolique,
- interaction homme-machine, images, données et connaissances,
- simulation et optimisation de domaines complexes.

S'appuyant sur 25 projets de recherche qui rassemblent près de 300 scientifiques (chercheurs, ingénieurs et stagiaires), l'unité de recherche joue un rôle au plus haut niveau international, en particulier, mais pas uniquement, dans les domaines de la vision par ordinateur, des réseaux et de l'Internet ou de la programmation temps-réel. La présence de nombreux visiteurs étrangers témoigne de la richesse de ces contacts internationaux.

1.3 Présentation du projet PLANÈTE

Planète: *Protocoles et applications pour l'Internet.*

Le projet Planète fait suite au projet Rodéo de l'INRIA Sophia Antipolis et à l'action Planète de l'INRIA Rhône Alpes.

Les activités du projet Planète sont centrées sur la conception, la mise en oeuvre et l'évaluation des protocoles et des applications Internet. L'objectif principal du projet est de proposer de nouvelles architectures, services et protocoles pour un Internet dans lequel les mobiles seront supportés de façon transparente.

Les travaux de recherche du projet s'articulent autour de plusieurs axes dont:

- La conception et l'évalutation de protocoles et applications multimédia pour mobiles sur Internet,
- L'étude de l'impact des nouveaux supports de transmission sur les protocoles,
- Le passage à l'échelle du routage multipoint,
- Le support de la qualité de service dans l'Internet,

– Les applications interactives multi-utilisateurs.

Le projet est impliqué dans plusieurs activités de recherche avec des partenaires académiques et industriels dont le projet VTHD sur les performances des protocoles de communication sur une plate-forme “vraiment” haut débit.

1.4 Cadre général du stage

***Projet VTHD**: Vraiment Très Haut Débit pour applications de l'Internet nouvelle génération.*

Ce stage a été effectué dans le cadre du projet VTHD qui vise à développer les briques technologiques nécessaires pour la mise en place d'un Internet de nouvelle génération [11]. Parmis les objectifs du projet on cite les objectifs suivants:

– Déployer une plate-forme d'expérimentation à très haut débit et démontrer sa capacité à associer montée en bande passante et raffinement du modèle de service.
– Valider ou adapter les protocoles Internet de contrôle de flux et de congestion à l'environnement d'un réseau très haut débit (>1 Gigabit/s) et d'évaluer leur capacité à gérer des transferts de bout en bout à très haut débit (>100 Mbits/s).
– contribuer de manière significative à une action de fédération des efforts pour l'Internet de nouvelle génération.

La plate-forme d'expérimentation comporte un réseau dorsal à 2,5 Gbit/s interconnectant différents sites localisés à Paris, à Rennes et à Sophia Antipolis. Le site de Sophia est interconnecté au réseau VTHD par une liaison en gigabits.

Ce réseau déployé sur l'infrastructure de transport de FRANCE TELECOM, repose sur une architecture de réseau IP/WDM.

Grâce à sa capacité en débit exceptionnelle le réseau VTHD permet de:

– Atteindre les potentialités attendues du calcul informatique distribué.
– Sauvegarder en ligne de très grandes bases de données.
– Diffuser la TV haute définition sur un réseau Internet.
– Développer les techniques nécessaires à l'apprentissage du geste médical à distance.

– Accélérer les processus industriels.

Le projet est divisé en 6 sous-projets qui s'inscrivent dans trois mouvements correspondant:

(i) au déploiement de la plate forme d'expérimentation,

(ii) à la mise en place et à l'exploitation d'une plate-forme multiservice sur l'infrastructure de transport IP/WDM,

(iii) aux expérimentations applicatives.

Le sous projet relatif au contrôle de flux et de congestion dans un environnement très haut débit est placé sous la responsabilité de l'INRIA.

Chapitre 2

Présentation du sujet

2.1 Introduction

La demande de bande passante par utilisateur sur le Web a explosé ces dernières années , l'arrivée de l'audio et de la vidéo qui génèrent de très hauts débits sur celui-ci ne va pas ralentir ce besoin. Comme parallèlement le nombre d'utilisateurs augmente chaque année, les réseaux actuels vont arriver très vite à la saturation [1].

L'introducion de la fibre optique comme support de transmission et la technique de multiplexage en longueur d'onde WDM (Wavelength Division Multiplexing) qui exploite l'énorme bande passante disponible sur la fibre ont permis de répondre à cette demande.

Le protocole de transport fiable TCP responsable du contrôle de trafic et de congestion dans les réseaux à commutation de paquets qui a prouvé sa robustesse et son adaptabilité de fonctionnement sur plusieurs types de liens réseaux, rencontre plusieurs problèmes de performance et de fiabilité dans le cas des réseaux haut débit caractérisés par une vitesse de transmission très élevée. De ce fait, plusieurs améliorations ont été introduites au protocole afin de l'adapter à ce type de réseaux.

Ce chapitre présente la technique de transmission WDM ainsi que les différents mécanismes de contrôle de flux et de congestion du protocole TCP.

2.2 Techniques de transmission optique

2.2.1 Réseaux de transmission de données

On définit trois générations de réseaux de transmission de données. La première génération de réseaux, qui comprend les réseaux Ethernet, Token bus et Token ring, n'utilisait pas la technologie optique et les débits étaient inférieurs à 10 Mbits/s. La deuxième génération correspond à l'introduction de la fibre optique comme support de transmission. Elle a permis l'évolution des réseaux de première génération, en remplaçant le câble en cuivre par la fibre optique. Par ailleurs de nouvelles architectures sont apparues, telles que FDDI [1] *(Fiber Distributed Data Interface)*, les débits varient alors de 100 Mbits/s à quelques Gbits/s. Enfin, les réseaux de troisième génération, dits "tout-optique" [2], utilisent de nouvelles approches afin d'exploiter la principale caractéristique de la fibre optique, à savoir la bande passante. Les débits théoriques avoisinent alors plusieurs Tbit/s [1].

2.2.2 Multiplexage optique

La technologie générale utilisée afin d'exploiter les dizaines de térahertz de bande passante spectrale disponibles dans la fibre optique est le multiplexage dans le domaine optique, ce qui signifie que la capacité de la fibre est divisée par des moyens optiques en plusieurs canaux accessibles individuellement et indépendamment.

La division de la bande passante en canaux peut être réalisée dans la dimension temporelle, on parle de multiplexage temporel (*TDM: Time Division Multiplexing*) ou dans la dimension des fréquences (ou longueurs d'onde) on parle alors de multiplexage en longueur d'onde (*WDM: Wavelength Division Multiplexing*).

2.2.3 Multiplexage temporel TDM

Le multiplexage TDM consiste à imbriquer temporellement différents canaux de communication en trames successives. Le multiplexage temporel revient à superposer les flots d'informations des différents canaux en les décalant les uns par rapport aux autres (voir

1. Réseau en fibre optique du type anneau à jeton.
2. Aucune conversion opto-électronique n'est effectué sur le paquet, des commutateurs optiques sont utilisés.

figure 2.1). Cela nécessite une synchronisation précise. À la réception chaque canal temporel est démultiplexé puis acheminé vers sa destination. On peut réaliser les fonctions de multiplexage/démultiplexage temporels avec des circuits intégrés ultra-rapides (40 Gbits/s en laboratoire) [2].

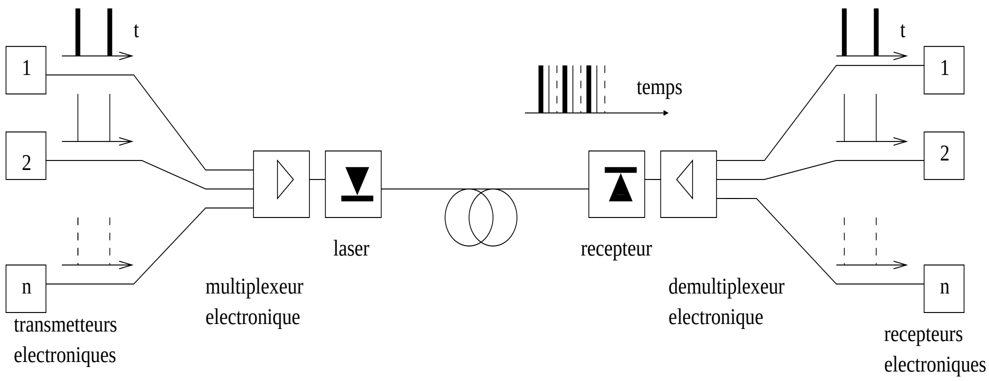


FIG. 2.1 – *Multiplexage temporel TDM*
Les suites de signaux provenant de sources différentes sont intercalés

2.2.4 Multiplexage en longueur d'onde WDM

Le multiplexage WDM consiste à découper le spectre optique d'une fibre en un nombre d'intervalles de longueurs d'onde (ou fréquences) distincts de telle sorte que chaque intervalle supporte un canal de communication transmettant un débit désiré.

Les différentes longueurs d'onde utilisées se situent dans les fenêtres centrées autour des longueurs d'onde $1.3\mu m$ ou $1.5\mu m$. Le nombre total de longueurs d'onde dépend de l'espacement entre ces dernières (l'espacement des canaux doit valoir au moins 6 fois leur bande passante pour pour éviter les interférences). Si l'espacement entre longueurs d'onde est relativement grand (supérieur à 1nm), ce multiplexage est qualifié de multiplexage en longueur d'onde WDM, sinon pour des intervalles de 0.1 à 1nm, il s'agit de multiplexage à répartiton en fréquence FDM et l'espacement entre canaux s'exprime en fréquence. Enfin pour des intervalles inférieurs à 0.1 nm, ce multiplexage est appelé DWDM (*Dense Wavelength Division Multiplexing*) [1]. Ces trois notions recouvrent le même type de multiplexage, seul l'espacement entre les longueurs d'onde diffère.

Les canaux WDM sont accessibles par des émetteurs laser réglés sur des longueurs d'onde spécifiques. Le multiplexage et le démultiplexage sont effectués par des composants optiques passifs (de façon similaire à la décomposition des couleurs par un prisme).

Ainsi, la coexistence de plusieurs canaux WDM sur une même fibre permet l'exploitation de l'énorme bande passante optique.

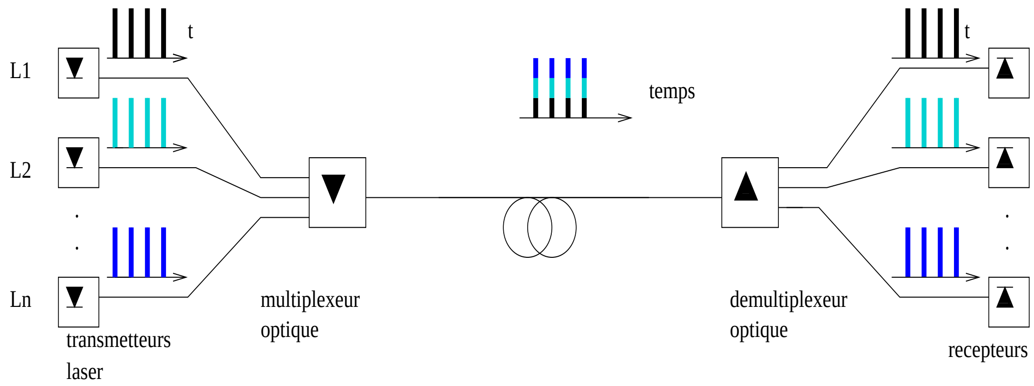


FIG. 2.2 – *Multiplexage en longueur d'onde WDM*

Les signaux sont transportés simultanément mais à une longueur d'onde différente d'une source à une autre.

2.2.5 Avantages des supports de transmission optique

La principale propriété de la fibre optique est sa formidable capacité à transporter de l'information grâce à une bande passante de 30 THz.

Parmi les autres avantages des fibres optiques figure la capacité à obtenir des taux d'erreurs bit faibles: de l'ordre de 10^{-15} dans les réseaux de données [1]. Ils sont dûs à l'insensibilité des fibres aux perturbations électromagnétiques et à l'absence d'interaction entre photons (contrairement aux électrons).

L'exploitation de ces grandes capacités de transmission nécessite l'adaptation du protocole TCP responsable du contrôle de flux et de congestion à l'environnement d'un réseau très haut débit.

2.3 Le protocole TCP

2.3.1 Présentation du protocole TCP

Le développement du protocole TCP a commencé dès le début du projet ARPAnet . De façon similaire à l'Internet d'aujourd'hui, le réseau ARPAnet ne garantissait pas un transfert fiable des données. Ceci signifie que le réseau ne garanti pas la livraison des paquets IP aux machines destinataires. En cas de congestion, le réseau peut jeter des paquets sans informer ni l'émetteur ni le récepteur de ce problème. A cela vient s'ajouter le fait que les routes utilisées dans l'Internet peuvent changer de façon dynamique, d'où la possibilité de dé-séquencement des paquets. Or le réseau Internet a été conçu pour des applications où la fiabilité est primordiale. D'où la nécessité d'avoir au niveau des couches supérieures à IP, les moyens nécessaires pour la détection des pertes et la retransmission des paquets perdus.

Le protocole TCP protocole de niveau 4 défini dans la RFC 793 [7], offre un service de transfert *bidirectionnel*, *fiable* avec *contrôle de flux* au dessus d'une couche réseau non fiable en commutation par paquets.

TCP assure entre autre la retransmission des paquets perdus dans le réseau, la délivrance de paquets en séquence à l'application de destination et le contrôle de la vitesse de transmission.

Le service TCP est utilisé par plusieurs applications de transfert de données (FTP, Telnet, SMTP, HTTP...). L'accès à ce service est effectué à travers l'interface *Socket* du système d'exploitation.

2.3.2 Le service de fiabilité

L'émetteur TCP bufferise tous les segments qu'il envoie et attend un acquittement du récepteur (ACK) avant de les détruire. Il démarre également un temporisateur (RTO) qui, s'il expire avant l'arrivée d'un acquittement, entraîne la retransmission des données du segment.

Chaque octet émis à un numéro de séquence, le numéro d'un segment est le numéro du premier octet qu'il contient.

Un acquittement contient un numéro de séquence strictement supérieur à tous les octets déjà reçus, ainsi on acquitte tous les segments qui ont un numéro de séquence inférieur.

Pour optimiser l'utilisation de la bande passante, TCP n'envoie pas un ACK systématiquement après la réception d'un paquet de données, il vérifie s'il a des données à envoyer, si c'est le cas TCP envoie un segment de données tout en mettant à jour le champ acquittement de sorte à acquitter le dernier segment reçu (technique de "piggybacking").

2.3.3 Le contrôle de flux

TCP se base sur le mécanisme de "*fenêtre coulissante*" (sliding window) afin d'éviter la perte des paquets au niveau du récepteur par écrasement de paquets non encore consommés par les couches supérieures [7].

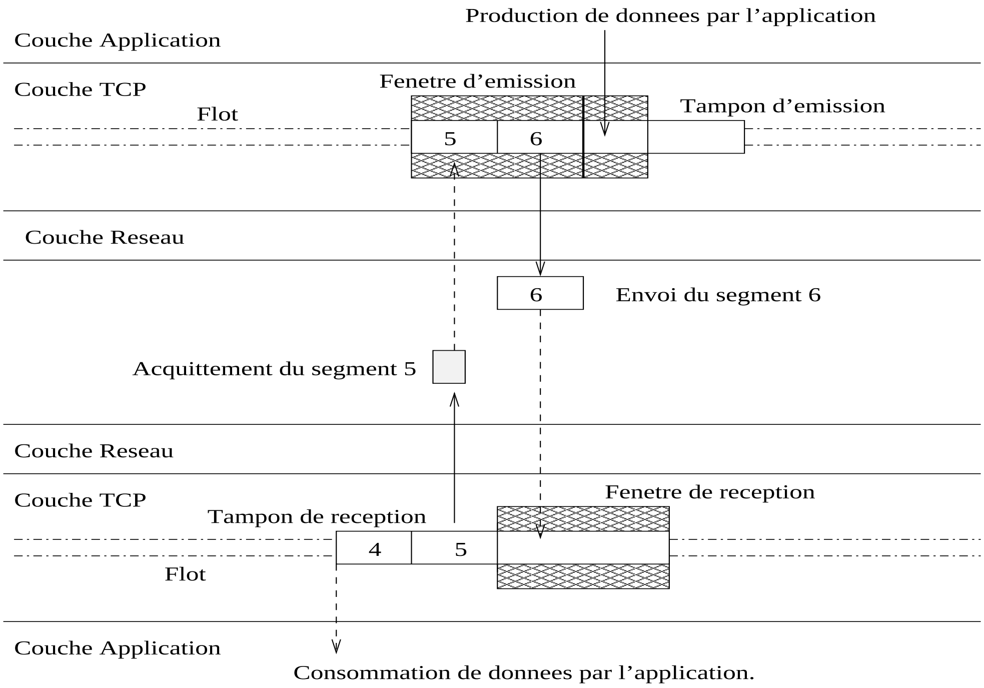


FIG. 2.3 – *Mécanisme de fenêtre coulissante*

L'émetteur doit connaître la taille de la fenêtre de réception (place encore libre dans le buffer de réception) pour déterminer la quantité de données à envoyer. L'émetteur ne peut pas émettre plus que la taille de la fenêtre signalée par le récepteur avant de recevoir un acquittement indiquant que certaines données ont été consommées par l'application (figure 2.3).

Un mécanisme de *fenêtre coulissante* est utilisé par sens de communication.

La taille du buffer d'émission n'est pas corrélée à la taille de la fenêtre d'émission, celui ci sert à bufferiser les données qui ne peuvent pas être émises.

2.4 Contrôle de flux et de congestion dans les réseaux TCP/IP

2.4.1 Principes fondamentaux

L'état de congestion dans les réseaux de paquets est défini comme un état où l'ensemble des ressources requises par les sources excèdent ceux du réseau. Dans ce cas, les performances, e.g. taux de pertes et délai des paquets, se dégradent très rapidement. Pour faire face aux problèmes de congetion, il est nécessaire de contrôler les ressources du réseau ainsi que les flux qui y circulent. Le *contrôle de trafic* [3] désigne l'ensemble des actions permettant de *prévenir la congestion.* Le *contrôle de congestion* désigne l'ensemble des actions permettant de *minimiser les effets lors d'une congestion* et de rétablir au plus vite l'état normal du réseau [3].

Il existe plusieurs types de mécanismes de contrôle de trafic qu'on peut classer en 3 classes:

2.4.1.1 Contrôle réactif

Chaque source doit, de façon dynamique, ajuster son trafic pour partager équitablement (entre les différentse sources) les ressources du réseau. Pour ce faire, le réseau doit indiquer de manière *explicite* ou *implicite,* aux sources les changements d'états. Chaque changement doit se traduire par un ajustement du trafic de la source.

2.4.1.2 Contrôle préventif

Chaque source doit décrire, *préalablement*, son trafic au réseau. Le réseau réserve en conséquent les ressources (e.g. mémoire et bande passante) correspondant aux paramètres décrivant le trafic. En cas d'insuffisance de ressources le réseau n'accepte pas le trafic en question. Ceci permet de minimiser les chances d'une congestion.

2.4.1.3 Contrôle hybride

Combine les caractéristiques du contrôle préventif et réactif. Le réseau réserve pour chaque source un minimum de ressources correspondant aux paramètres de trafic spécifiés. Les

3. Les termes contrôle de trafic et contrôle de flux sont utilisés de façon interchangeable

sources sont autorisées à excéder le trafic spécifié pendant les périodes où les ressources du réseau ne sont pas toutes réservées ou utilisées (inactivité d'autres sources). Dans ce cas le réseau n'offre pas de garantie au trafic excédant celui spécifié au départ.

2.4.2 Mécanismes de contrôle de flux de TCP

Le protocole TCP, responsable du contrôle de flux mettait en oeuvre un algorithme basé sur l'utilisation du champ fenêtre du protocole. Cependant TCP n'utilisait aucun mécanisme de contrôle de trafic permettant de prévenir les congestions dans le réseau. L'Internet souffrait d'un phénomène de congestion donnant lieu à des retransmissions très souvent inutiles, suite à l'expiration du temporisateur de retransmission.

La version TCP Tahoe qui intègre un ensemble d'algorithmes de contrôle de trafic a été proposée. De nouvelles modifications ont donné le jour à TCP Reno, New Reno et ensuite à TCP Vegas.

L'idée derrière les mécanismes de contrôle de congestion supportés par TCP est que chaque source de données contrôle sa vitesse de transmission indépendamment des autres de façon à réaliser les objectifs suivants:

- Utiliser au maximum la bande passante disponible sur les liens du réseau.
- Réduire le nombre de paquets en attente de transmission dans les routeurs.
- Distribuer la bande passante disponible de façon équitable entre les différents utilisateurs.

2.4.2.1 algorithme Slow Start et Congestion Avoidance

L'algorithme Slow Start et Congestion Avoidance [6] introduit dans TCP Tahoe est du type basé sur fenêtre ("window based") et utilise une notification implicite de l'état de congestion dans le réseau. Cet algorithme permet à la source de contrôler la quantité de données soumise au réseau. Les variables introduites par cet algorithme sont les suivantes:

- ***cwnd*** (congestion window): Taille de la fenêtre de congestion, elle représente la quantité maximale de données que la source peut transmettre sur le réseau avant de recevoir un acquittement.

– ***ssthresh*** (slow start threshold): Seuil utilisé pour déterminer la phase de l'algorithme qui doit être exécutée (Slow Start ou Congestion Avoidance), généralement *ssthresh* est initialisé à la valeur de *rwnd* [4]

Le ***min (cwnd, rwnd)*** est utilisé pour déterminer la quantité de données qui doit être transmise sur le réseau.

Slow Start

Après l'établissement d'une connexion ou après l'expiration d'un temporisateur de retransmission, l'émetteur fixe la taille de la fenêtre de congestion à 1 segment (*MSS*). A chaque réception d'un acquittement la taille de la fenêtre de congestion est incrémentée de 1 *MSS*:

$$cwnd = cwnd + MSS$$

Ceci revient à doubler la valeur de *cwnd* à chaque fois qu'une fenêtre de congestion entière est acquittée. Cet algorithme continue jusqu'à ce que la fenêtre de congestion atteigne un seuil (ssthresh). Il est à noter que le terme Slow Start peut induire une confusion puisque la taille de la fenêtre croît de façon exponentielle et permet de remplir, assez rapidement, le lien. Mais cette croissance est lente (slow) par rapport à la version de TCP où il n'existait pas de mécanisme de contrôle de congestion et où la source transmet dès le début d'une connexion une quantité de données égale à la taille de *rwnd.*

4. Receiver's advertised window: taille de la fenêtre d'émission indiquée par le récepteur.

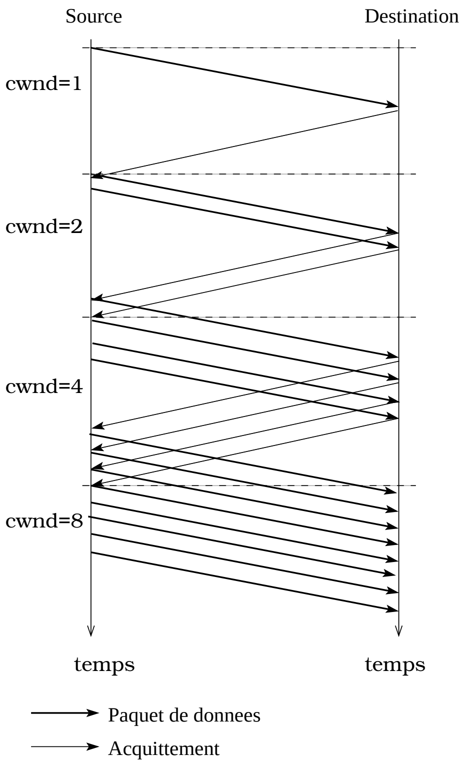


FIG. 2.4 – *La phase "Slow Start"*

Congestion Avoidance

Après la phase de Slow Start, qui prend fin au franchissement du seuil *ssthresh,* la fenêtre évolue de façon linéaire: pour chaque acquittement reçu, la taille de la fenêtre est incrémentée d'une fraction de MSS égale à *(MSS/cwnd)*:

$$cwnd = cwnd + MSS * \frac{MSS}{cwnd}$$

Ceci revient à incrémenter la fenêtre par un *MSS* à chaque fois qu'une fenêtre de congestion entière est acquittée. Dans cette phase la croissance est linéaire plutôt qu'exponentielle pour éviter de revenir trop rapidement à une phase de congestion. Cette phase continue jusqu'à détection de perte de paquet qui peut se faire de deux manières: soit par l'expiration d'un temporisateur que la source déclenche lors de la transmission du paquet soit par la réception de quatre acquittements de la part de la source portant le même numéro de séquence (acquittements dupliqués), à ce point TCP met à jour la valeur du seuil *ssthresh* à la moitiée de la fenêtre de congestion (voir figure 2.6). Quand la fenêtre de congestion atteint sa taille maximale donnée par la taille maximale de la fenêtre du récepteur, sans qu'il y ait des pertes, la fenêtre de congestion ne croît plus et reste constante jusqu'à la détection de la prochaine perte [10].

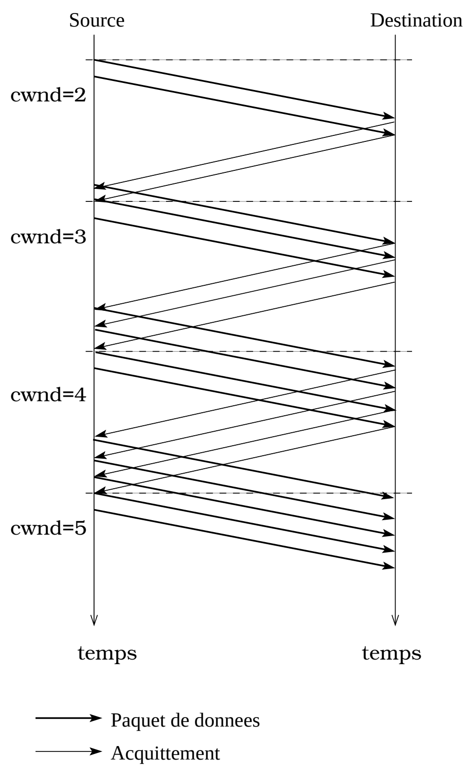
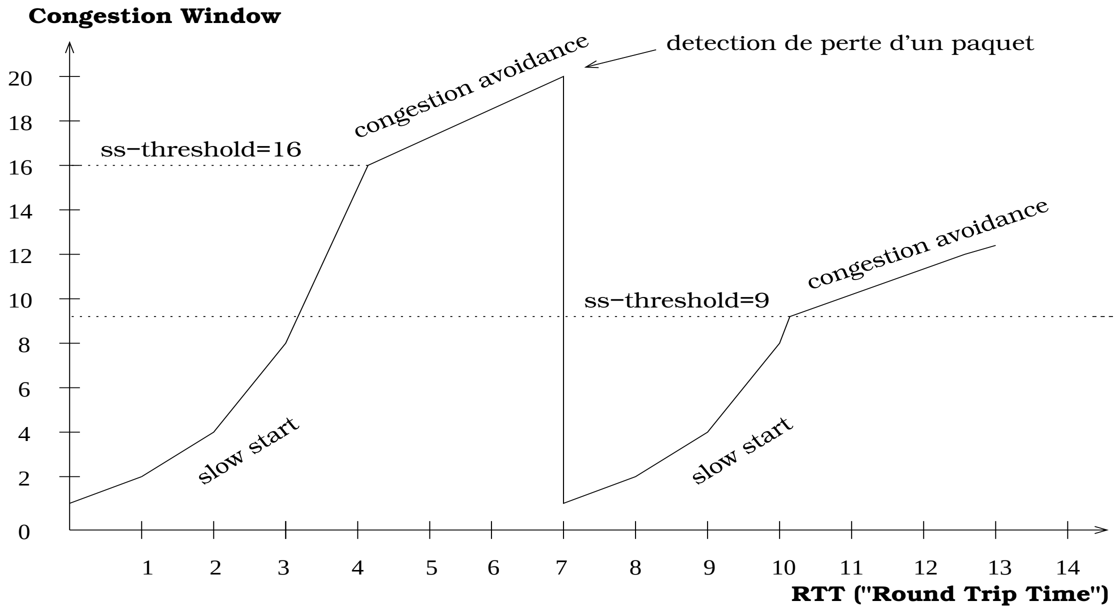


FIG. 2.5 – *La phase "Congestion Avoidance"*



[5]

FIG. 2.6 – *Évolution de la fenêtre de congestion de TCP en fonction du RTT*

Le fait de passer à chaque perte d'un paquet la taille de la fenêtre à un, gaspille de la bande passante. De plus, une détection plus rapide des pertes de paquets permet d'avoir un débit plus élevé. Ces deux constatations ont permis d'implémenter de nouveaux mécanismes TCP qui sont décrits dans le paragraphe suivant.

2.4.2.2 Algoritmes de retransmission rapide et de récupération rapide

L'algorithme décrit dans le paragraphe 2.4.2.1 constitue la brique de base pour le contrôle de trafic dans TCP. Cet algorithme réagit aux pertes de paquets, considérées comme un feedback implicite, pour ajuster la taille de la fenêtre de congestion et par conséquent le trafic de l'émetteur. Dans tout système de contrôle il est très important de reçevoir une indication au plus vite afin de refléter au maximum l'état courant du système et réagir rapidement aux changements. La détection de pertes de paquets TCP se faisait en utilisant le temporisateur de retransmission. Or, à cause de contraintes matérielles et logicielles, plusieurs systèmes d'exploitation n'offrent qu'une très grande granularité pour les retransmissions (de l'ordre de 300 à 500 ms). Si le RTT est relativement faible par rapport au temporisateur de retransmision (c'est souvent le cas pour les réseaux à haut

5. Sur la figure 2.6 le RTT est représenté comme constante pour simplifier la représentation. En pratique, le RTT est toujours variable dans le temps.

débit), toute perte de paquet engendre la dégradation de performance de TCP. Ceci est causé par le temps d'inactivité élevé dû à l'attente de l'expiration du RTO (temporisateur de retransmission).

Retransmission rapide (Fast Retransmit)

Pour avoir une indication plus rapide de la perte de paquets, l'idée *d'acquittements dupliqués* a été introduite [12], elle peut être décrite de la manière suivante:

À la réception d'un segment dont le numéro de séquence n'est pas celui attendu, le récepteur acquitte le dernier segment reçu dans l'orde. À la réception d'un certain nombre d'acquittements dupliqués, l'émetteur TCP devine alors qu'il y a eu perte de segments. Au lieu d'attendre l'expiration du RTO, l'émetteur retransmet le paquet perdu.

Comme il y a un risque de déséquensement dans l'Internet et pour éviter de retransmettre inutilement des paquets, la retransmission du paquet estimé perdu ne se fait qu'après la réception de trois acquittements dupliqués.

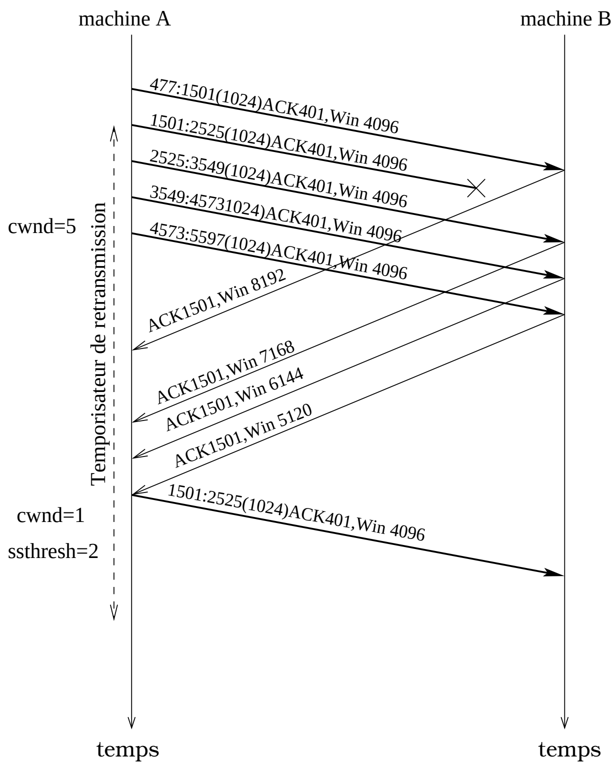


FIG. 2.7 – *Exemple de retransmission rapide d'un paquet perdu*

Recouvrement rapide (Fast Recovery)

C'est une optimisation de l'algorithme de retransmission rapide, ce dernier, suite à une détection de perte de paquets initialise la fenêtre de congestion à 1 et rentre dans la phase Slow Start souvent susceptible de baisser les performances de TCP. Le recouvrement rapide est un algorithme qui s'exécute juste après la retransmission rapide et peut être résumé comme suit:

- à la réception du 3 ème acquittement dupliqué réduire la fenêtre de congestion à la moitié (mais pas moins de 2 segments) *:*

$$ssthresh = \frac{cwnd}{2}$$

$$cwnd = ssthresh$$

- incrémenter *cwnd* de 3 MSS (nombre de segments qui ont été transmis sur le réseau et bufferisés par le récepteur):

$$cwnd = ssthresh + 3 * MSS$$

- pour chaque acquittement dupliqué reçu incrémenter *cwnd* de 1 MSS et envoyer un nouveau segment (car étant donné que les acquittements dupliqués ne permettent pas d'acquitter les paquets envoyés et qui se trouvent dans le buffer réception, il faut augmenter *cwnd* pour éviter que l'émetteur arrive à une situation où il ne peut plus transmettre de paquets et donc il y aurait une chute de débit).
- à la réception d'un acquittement du segment retransmis, réduire *cwnd* à la valeur de *ssthresh* (précédemment calculée). Quitter la phase de recouvrement rapide et rentrer dans la phase "congestion avoidance".

L'algorithme de retransmission rapide est intégré dans la version Tahoe et Reno de TCP. Le second, l'algorithme de recouvrement rapide, est intégré dans la version Reno uniquement et constitue l'unique changement par rapport à version Tahoe.

2.4.3 Améliorations de TCP

2.4.3.1 TCP Reno

La seule amélioration apportée dans TCP Reno par rapport à TCP Tahoe est l'ajout de l'algorithme de *recouvrement rapide.* Cet algorithme évite de rentrer dans la phase *"Slow Start"* dans laquelle TCP perd considérablement ses performances suite à une perte de paquet, TCP transmet alors de nouveaux paquets juste après la phase de *recouvrement rapide* ensuite passe directement à la phase *"Congestion Avoidance"* après avoir reçu un acquittement du paquet perdu.

Plusieurs études de performance ont montré que TCP Reno donne des performances plus meilleurs que celles de TCP Tahoe en cas de perte d'un seul paquet par fenêtre de congestion . En cas de pertes multiples, les performances de TCP Reno se dégradent.

Pour remédier à ce problème un nouvel algorithme permettant d'éviter les time-outs pour TCP Reno (en cas de pertes multiples) a été introduit [9], c'est la version TCP New Reno.

2.4.3.2 TCP New Reno

Pour TCP Reno, chaque perte de paquet dans une fenêtre cause la réduction de *ssthresh* à la moitié, dans le cas de perte de plus que deux paquets par fenêtre le RTO expire et la phase Slow Start ne peut plus être évitée.

Afin d'éviter cette dégradation de performances une modification à été apporté à la phase "Fast Recovery" dans TCP New Reno. Cette modification définit une procédure appelée "Fast Recovery" qui commence quand le 3ème ACK dupliqué est reçu et qui se termine à l'expiration d'un RTO ou à la réception d'un ACK qui acquitte la totalité des données transmises y compris celles envoyées au début de la procédure "Fast Recovery" [9]. L'algorithme peut être décrit de la façon suivante:

1. Quand le 3 ème ACK dupliqué est reçu et l'émetteur n'est pas déjà entrain d'exécuter la procédure "Fast Recovery" faire:

   $$ssthresh = \frac{cwnd}{2}$$

   enregistrer le numéro de séquence le plus élevé qui a été transmis dans une variable *"recover"*.

2. Retransmettre le segment perdu et affecter à cwnd la valeur de ssthresh plus 3*MSS (nombre de segments qui ont été transmis sur le réseau et bufferisés par le récepteur):

$$cwnd = ssthresh + 3 * MSS$$

3. Pour chaque acquittement dupliqué reçu incrémenter *cwnd* de 1 MSS et envoyer un nouveau segment.
4. À la réception d'un acquittement de nouvelles données (cet ACK peut être une réponse à la retransmission de l'étape 2, ou à une retransmission plus récente), si cet ACK acquitte le segment de numéro de séquence "*recover*" alors il est considéré comme l'acquittement de tout les segments envoyés depuis la transmission du paquet perdu et jusqu'à la réception du 3 ème ACK dupliqué. Si l'ACK reçu est de valeur inférieur à *"recover"* (il est appelé acquittement partiel), c'est une indication de perte de paquet, alors on retransmet le premier segment non acquitté sans attendre trois acquittements dupliqués et on ne réduit pas *ssthresh.*
5. La phase de recouvrement rapide continue jusqu'à ce que tout les paquets transmis avant de rentrer dans cette phase soient acquittés

2.4.3.3 TCP Vegas

À l'opposé de TCP Tahoe et TCP Reno où on continue à augmenter la taille de la fenêtre jusqu'à détection de perte de paquet considérée comme indication de congestion du réseau (systèmes réactifs), TCP Vegas modifie (augmente ou diminue) la taille de la fenêtre à partir d'observations effectuées sur les RTT (Round Trip Times) des paquets émis (système préventif). Si les RTT observés sont larges TCP Vegas déduit que le réseau commence à être congestionné et diminue par conséquent la taille de la fenêtre de congestion. Si les RTT se réduisent, TCP Vegas reconnait que le réseau n'est plus congestionné et commence à augmenter la taille de la fenêtre.

Pendant la phase "*Congestion Avoidance*" la taille de la fenêtre est modifiée comme suit:

$$cwnd = cwnd + MSS, \quad si \;\; diff < \frac{\alpha}{baseRTT}$$

$$cwnd = cwnd, \quad si \;\; \frac{\alpha}{baseRTT} \leq diff \leq \frac{\beta}{baseRTT}$$

$$cwnd = cwnd - MSS, \quad si \quad \frac{\beta}{baseRTT} < diff$$

où:

- $diff = \frac{cwnd}{baseRTT} - \frac{cwnd}{RTT}$
- RTT est la valeur observée du RTT.
- $baseRTT$ est la plus petite valeur des RTT observés.
- $\alpha\, et\, \beta$ sont des constantes.

Pendant la phase Slow Start la taille de la fenêtre est incrémentée d'une valeur égale à la moitiée de celle utilisée pour TCP Tahoe et TCP Reno.

TCP Vegas permet d'anticiper les états de congestion en ajustant le taux de transmission et modifie le mécanisme de Slow Start en évitant de créer des pertes de paquets pour trouver la bande passante.

2.5 Objectifs de l'étude

Le travail consiste en une première partie à identifier les problèmes relatifs au fonctionnement de TCP sur des réseaux haut débit et à décrire les améliorations introduites au protocole pour l'adapter à de tels environnements.

Une seconde partie est consacrée à la présentation des expérimentations qui ont été effectuées dans le but d'étudier les performances de TCP dans un environnement haut débit ainsi qu'à l'analyse des résultats obtenus.

Les tests ont été réalisés sur deux types de connexions utilisant la fibre optique comme support de transmission:

1. Une connexion entre deux machines en point à point.
2. Une connexion de bout en bout sur le réseau expérimental VTHD.

Chapitre 3

TCP et les réseaux haut débit

3.1 Introduction

Plusieurs études ont montré que TCP peut fonctionner de manière fiable et assez performante sur différents types de liens Internet dont la capacité de transmission de données varie de quelques centaines de bits/s (connexion Modem à 300 bits/s) jusqu'à l'ordre de centaines de Mbits/s, cependant plusieurs problèmes liés à la performance de TCP apparaissent dans le cas des réseaux à grand produit délai-bande passante. Ce paramètre significatif est le produit de la bande passante[1] (bits/s) et du délai entre l'émetteur et le récepteur (RTT), il présente la quantité de données émises et non encore acquittées permettant d'avoir une utilisation efficace de la bande passante disponible [5].

Si ce produit est supérieur à 10^5 bits le réseau est qualifié de réseau à grand produit délai-bande passante. Par exemple pour un réseau de fibre optique ayant un délai de 10ms et une bande passante de 300 Mb/s ce produit dépasse 10^6 bits.

En effet l'introduction de la fibre optique a permis d'atteindre des vitesses de transmission très élevées sur les liens, et vu que l'implémentation de TCP ne prenait pas en considération de tels aspects des problèmes apparaissent dans le cas des réseaux haut débit.

Une analyse de ces problèmes ainsi qu'un ensemble de solutions relatives seront présentés dans ce chapitre.

1. La vitesse de transmission la plus élevée pouvant être atteinte, déterminée par la vitesse de l'élement le plus lent sur un chemin du réseau.

3.2 Limitation de la taille de la fenêtre TCP

Dans un réseau caractérisé par une bande passante réduite la transmission d'une fenêtre de données sur le lien peut être schématisée par la figure 3.1 où une fenêtre de taille 80 Kbits contenant 10 segments de taille 8Kbits est transmise sur un lien de capacité 64 Kbits/s.

Dans ce cas le récepteur commence à reçevoir des segments de données avant que l'émetteur n'achève la transmission de la moitié de la fenêtre d'émission, la bande passante du lien est bien utilisée.

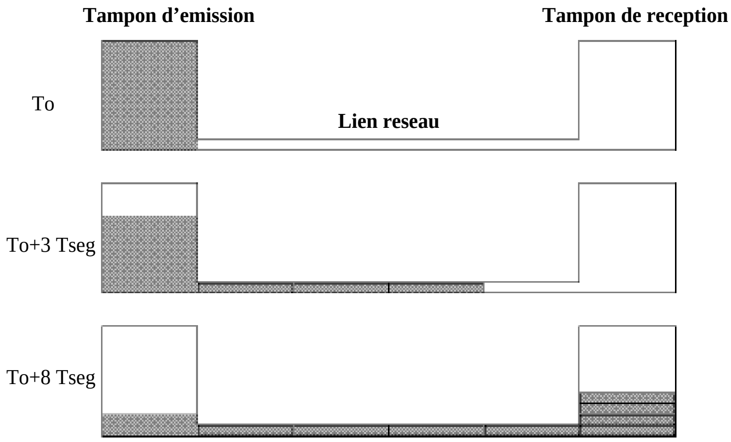


FIG. 3.1 – *Transmission de données sur un réseau de capacité réduite.*

La figure 3.2 représente le transfert d'une fenêtre de même taille que l'exemple précédent sur un lien à grande vitesse de transmission. Dans ce cas la totalité de la fenêtre TCP est émise avant que le récepteur ne reçoive le premier octet transmis. Ceci résulte en une sous-utilisation de la capacité du lien vu que l'émetteur ne peut émettre des données d'une nouvelle fenêtre qu'après la réception du premier acquittement de la fenêtre précédente.

La vitesse de transmission élevée pour les réseaux haut débit est obtenue grâce à la quantité de données énorme pouvant transiter sur le lien (grande capacité du lien), toutefois l'entête TCP utilise un champ de 16 bits pour indiquer la taille de la fenêtre de réception (rwnd) ce qui limite la taille maximale de la fenêtre à 65 Ko (2^{16}).

Une nouvelle option de l'entête TCP (*"window scale option"*) a été définie [5] afin de permettre d'avoir des fenêtres d'émission de taille plus grande, cette option définie un *facteur multiplicatif* pouvant avoir une valeur maximale de 14.

Cette option est indiquée à l'établissement d'une connexion au niveau des segments <SYN> et <SYN,ACK>, la taille de la fenêtre est obtenu en décalant à gauche l'entier indiqué par le champ "window size" de celui indiqué par le champ "window scale option".

La taille maximale pouvant être atteinte est alors de 2^{30} c'est à dire 1Go.

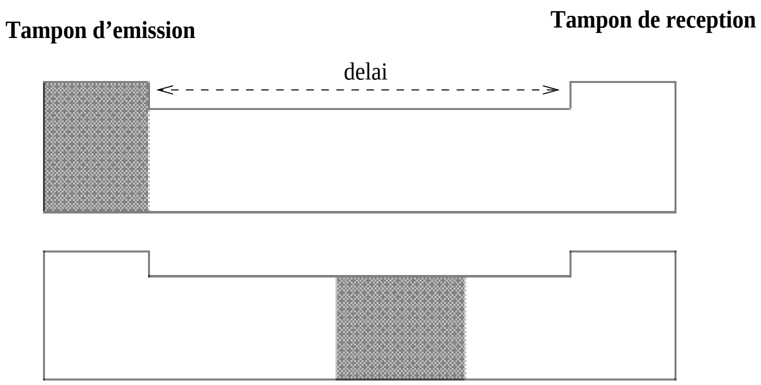


FIG. 3.2 – *Transmission de données sur un réseau haut débit*

3.3 Rétablissement après une perte de plusieurs paquets

La perte de paquets dans un réseau à grand produit délai-bande passante résulte en une dégradation remarquable du débit d'une connexion TCP vu que cette perte entraîne la réduction de la fenêtre d'émission à la moitié et l'exécution de la phase *"Slow Start"*. L'introduction des algorithmes "Fast Retransmit" et "Fast Recovery" [6] a permis le rétablissement de TCP après la perte d'un seul paquet par fenêtre d'émission (voir figure 3.3), cependant l'expansion de la taille de la fenêtre d'émission augmente la probabilité de perte de plusieurs paquets par fenêtre.

Pour remédier à ce problème un mécanisme d'acquittements séléctifs (connu sous l'appelation "Selective ACKnowledgment" *ou* SACK) a été proposé [8].

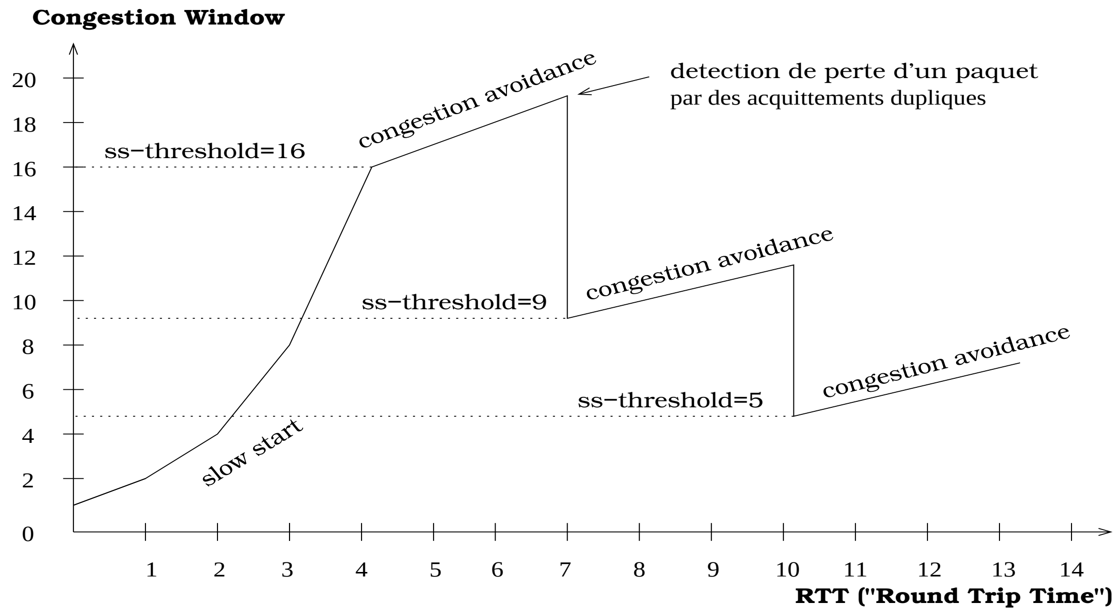


FIG. 3.3 – *Recouvrement rapide: rétablissement de la perte d'un seul paquet par fenêtre.*

3.3.1 TCP SACK

L'extention de SACK utilise deux nouvelles options de l'entête TCP :

- "SACK-permitted": pour indiquer l'utilisation de SACK, envoyée à l'établissement d'une connexion.
- L'option SACK : envoyée après l'établissement de la connexion pour indiquer les blocs reçus.

Le principe de *SACK* consiste à reporter dans les acquittements les blocs contigus de segments reçus et non encore acquittés, l'émetteur retransmet alors les segments n'ayant pas été reçus en se référant aux intervalles de numéros de séquences vides entre les blocs indiqués.

Chaque bloc contigu de données reçues et non encore acquittées est identifié par deux entiers de 16 bits:

- Origine relative: le premier numéro de séquence du bloc relativement au numéro indiqué par le champ ACK du premier segment de la fenêtre.

– Taille du bloc: taille du bloc en octets.

L'option SACK permet de spécifier au maximum une liste de 10 blocs reçus et non encore acquittés.

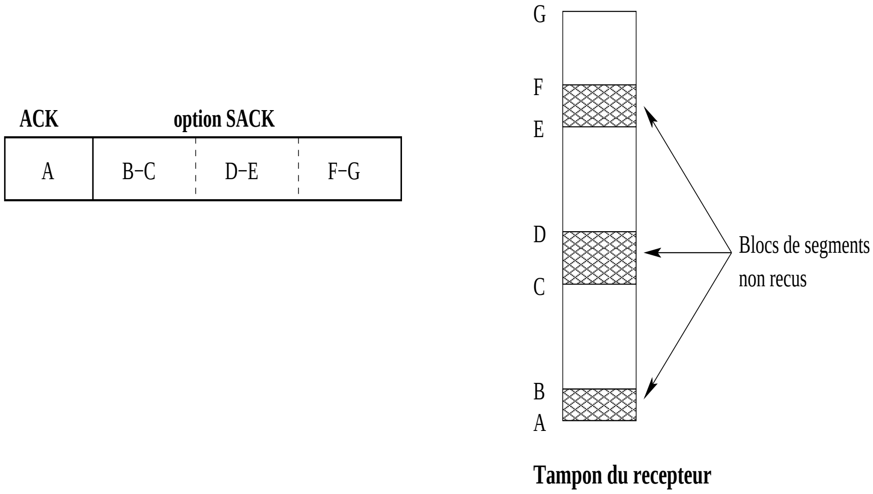


FIG. 3.4 – *Acquittements séléctifs*

L'implémentation SACK de TCP préserve l'algorithme de base de TCP Reno, la différence est seulement au niveau de la réaction à la perte de plusieurs paquets par fenêtre de données.

3.3.2 SACK avec l'option "window scaling"

Si l'option "window scaling" est utlisée, les 16 bits du champ de l'option SACK deviennent insuffisants pour indiquer l'origine et la taille des blocs reçus. La solution qui a été retenue est de multiplier les champs de l'option SACK par le même facteur de l'option "window scale".

Cette solution implique une perte de précision puisque l'option SACK va indiquer seulement des entiers multiples du facteur "window scale", mais cette perte de précision ne peut induire une erreur que si le facteur "window scale" est supérieur à la valeur MSS[2].

2. Maximum Segment Size: la plus grande quantité d'information pouvant être transmise dans un segment, négociée à l'établissement d'une connexion.

3.4 Mesures du RTT

L'implémentation de TCP garantie un transfert fiable de données en retransmettant les segments n'ayant pas été acquittés pendant un intervalle de temps (RTO). Une estimation dynamique et précise du temporisateur de retransmission est déterminante pour la performance de TCP puisqu'elle permet d'éviter des retransmissions inutiles causées par une sous estimation du RTO ou un retard de la détection de perte de paquets causé par une sur estimation du RTO.

La valeur du RTO est déterminée en fonction de la moyenne et de la variance du RTT (intervalle de temps entre l'émission d'un segment et la réception d'un acquittement de ce segment).

Plusieurs implémentations de TCP effectuent les mesures du RTT en utilisant comme échantillon un seul paquet par fenêtre, ceci permet d'avoir une approximation adéquate du RTT pour des fenêtres de taille réduite. Pour une vitesse de transmission très élevée une telle mesure engendre une erreur d'estimation du RTT vu la fréquence d'échantillonnage faible par rapport à la fréquence des données.

Afin d'avoir une estimation plus précise du RTT pour les réseaux haut débit, un mécanisme de mesure du RTT(Round Trip Time Measurement) qui utilise une nouvelle option de TCP: *estampille de temps* a été introduit [4, 5].

3.4.1 Estampille de temps TCP ("TCP timestamps option")

L'option d'estampille de temps TCP contient deux champs de 4 octets chacun:

- TSval (Time Stamp value): valeur d'estampille placée par l'émetteur.
- TSecr (Time Stamp echo reply): valeur d'estampille envoyée par le récepteur, elle doit être égale à la valeur de TSval contenue dans le segment acquitté afin d'indiquer à l'émetteur le segment par rapport auquel il doit faire la mesure du RTT.

Le champ TSecr contient la valeur la plus récente de TSval.

La valeur du RTT est obtenue en effectuant la différence entre l'instant d'émission d'un segment et l'instant de réception d'un acquittement qui contient une valeur de TSecr égale à TSval du segment envoyé.

La figure 3.5 illustre ce mécanisme sur un ensemble de blocs de données de numéros de séquences: A, B, C et les acquittements correspondants.

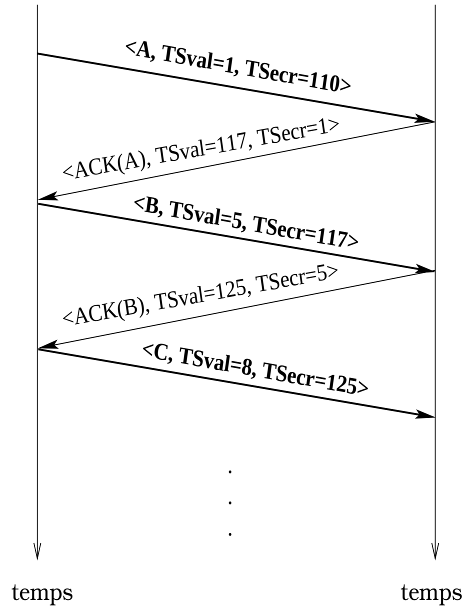


FIG. 3.5 – *Estampille de temps pour la mesure du RTT*

3.5 Conclusion

On a présenté dans ce chapitre les principaux problèmes liés à la performance de TCP qui surgissent dans le cas d'un environnement réseau haut débit à savoir:

- La limitation de la taille de la fenêtre TCP.
- Rétablissement après la perte de plusieurs paquets par fenêtre d'émission.
- Nécessité d'une estimation assez précise du RTO.

Ces problèmes constituent les axes majeurs autours desquels les expérimentations présentées dans le chapitre suivant ont été effectuées.

Chapitre 4

Tests effectués et résultats obtenus

4.1 Tests en point à point

Afin d'évaluer les performances de TCP sur une connexion à très haut débit et d'identifier les paramètres limitatifs du fonctionnement du protocole dans un tel environnement un ensemble de tests a été effectué entre deux machines liées en point à point avec une fibre optique.

Sachant que les réseaux de stations de travail sont généralement des réseaux de stations Unix, les mesures ont été réalisées en utilisant les couches de protocoles natives de la plupart des Unix: l'implémentation des sockets sur TCP/IP.

L'ensemble de ces tests ainsi que l'analyse des résultats empiriques obtenus sont présentés dans cette partie.

4.1.1 Environnement des tests

4.1.1.1 Architecture de la connexion

Les deux machines pour lesquelles les tests ont été réalisés sont connectées en point à point sur des cartes réseau Gigabit Ethernet d'une capacité de transmission de 1Gbit/s liées par une fibre optique multimode dont la bande passante théorique est de 1 Ghz.

L'architecture correspondante est représentée par la figure suivante:

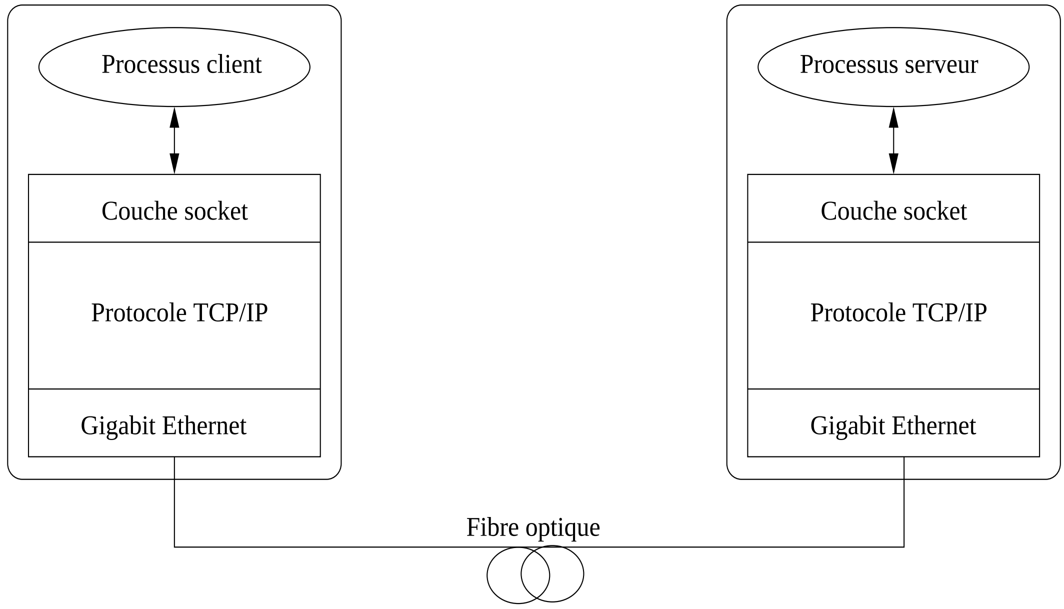


FIG. 4.1 – *Architecture de la connexion considérée*

4.1.1.2 Outils utilisés

L'outil de tests de TCP qui a été utilisé est NTTCP (New Test TCP), celui ci est lancé en mode réception sur une machine et en mode émission sur une autre transmet un nombre déterminé de buffers de taille définie en mesurant le débit de la transmission. L'outil ETHEREAL a été utilisé pour analyser les paquets transmis entre les deux machines (une présentation de ces outils ainsi que des exemples de leur exécution est donnée en annexe).

XGRAPH a été utilisé pour la visualisation des courbes de débit.

4.1.2 Mesures du débit sur la connexion point à point

4.1.2.1 Mesure du débit en TCP

Cette expérience a pour but de mesurer le débit de transmission de données en TCP entre les deux machines considérées selon l'architecture représentée par la figure 4.1, et ceci en fonction de la taille des tampons (buffers) de la socket TCP, la variation de la taille des tampons est réalisée par l'option *-w* de l'outil NTTCP.

La variation de taille de la fenêtre TCP entraînée par la variation de la taille des buffers de la socket est utilisée pour mettre en évidence les limites de performance de TCP.

Afin d'accélérer l'exécution des tests un script c-shell a été développé, ce script pilote l'ensemble de la procédure d'incrémentation de la taille des buffers, de l'execution de NTTCP et de l'affichage des courbes de débit.

- Dans cette expérience la quantité de données transmise entre les deux machines est égale à 100000 tampons de 4096 octets chacun.
- La MTU qui représente la taille maximale de l'unité de transmission qui peut être transportée sans fragmentation sur le lien est fixée à 1500 octets.
- Le délai moyen de transmission mesuré entre les deux machines est de 250μs.

La courbe représentant le débit de la connexion qui a été obtenue est la suivante:

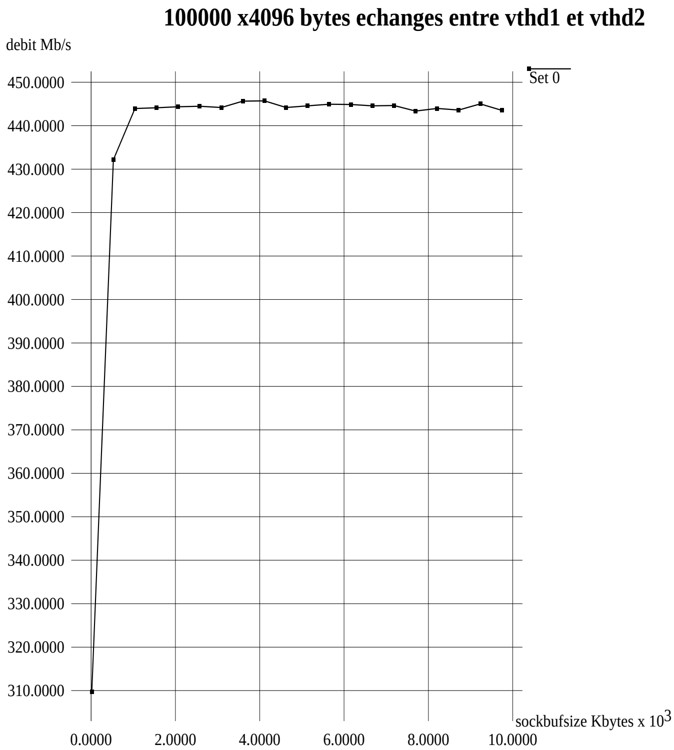


FIG. 4.2 – *Débit de la connexion TCP*

À partir de cette courbe on constate que pour une taille initiale du buffer de la socket TCP de 64 Ko qui représente la valeur prise par défault on a un débit de 310 Mbits/s, alors qu'à partir d'une valeur du buffer égale à 512Ko le débit réalise une augmentation remarquable pour atteindre une valeur de 432 Mbits/s. Pour les tailles des buffers supérieures à 1 Mo le débit se stabilise à une valeur de 445 Mbits/s.

4.1.2.2 Mesure du débit en UDP

Cette expérience à pour but d'identifier la valeur la plus élevée de débit pouvant être atteinte en UDP. Cette valeur présente le seuil maximal de débit qu'on peut avoir sur la connexion.

La courbe obtenue en se basant sur les mesures effectuées par l'outil NTTCP en utilisant UDP au lieu de TCP est la suivante:

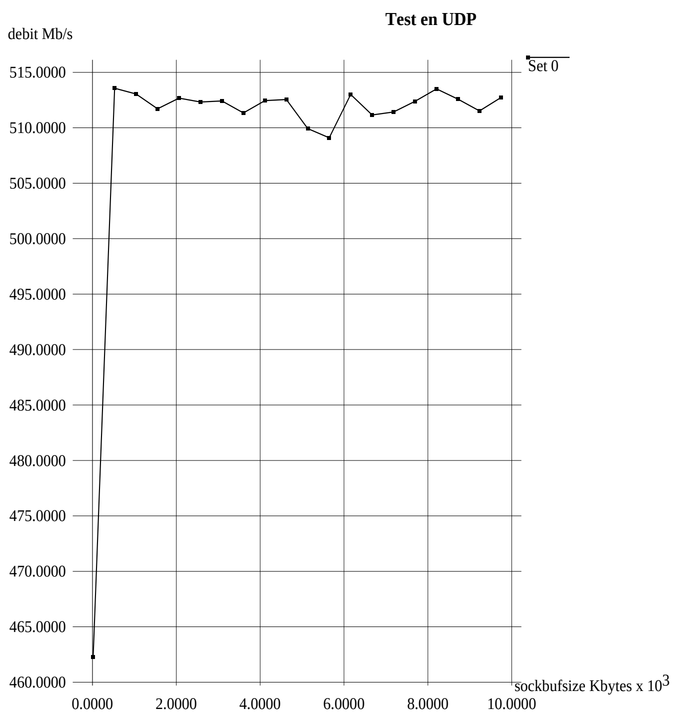


FIG. 4.3 – *Débit mesuré en UDP*

En UDP le débit obtenu dépasse celui obtenu en TCP d'environ 70 Mbits/s et atteint la valeur de 512Mbits/s. Les valeurs du débit obtenues présentent un aspect stable pour des valeurs de la taille du buffer supérieures à 512 Ko, cependant pour une taille de 64 Ko on obtient une valeur moins importante du débit: 462 Mbits/s.

4.1.2.3 Effet de l'augmentation de la taille du buffer de la socket

On s'aperçoit bien que les débits obtenus à partir des expériences précédentes ne sont pas pleinement satisfaisants vu la sous utilisation de la grande bande passante disponible sur la fibre optique. On a essayé de maximiser le débit de transmission de données en augmentant la taille du buffer de la socket TCP jusqu'à une valeur de 1Go, ceci en modifiant les paramètres *rmem_max* et *wmem_max* se trouvant respectivement dans les fichiers:

/proc/sys/net/core/rmem_max et */proc/sys/net/core/wmem_max* sous Linux.

Ces paramètres représentent respectivement la taille mémoire réservée pour l'émission et la réception des données par la socket TCP.

Les résultats obtenus sont récapitulés dans le tableau suivant:

Taille des tampons de la socket TCP	64 KB	512 KB	1 MB	1 GB
Débit maximal atteint	310 Mb/s	433 Mb/s	445 Mb/s	447 Mb/s

TAB. 4.1 – *Débits mesurés en fonction de la taille des tampons de la socket TCP*

D'après cette expérience on conclut que le débit maximal pouvant être atteint en augmentant la taille du buffer jusqu'à une valeur de 1 Go est de 447 Mb/s.

Dans les deux expériences qui suivent on va essayer d'identifier l'effet de l'utilisation des estampilles de temps et des acquittements séléctifs de TCP sur le débit de la connexion.

4.1.2.4 Effet de l'estampille de temps TCP

Le but de cette expérience est de mettre en évidence l'impact de l'utilisation de l'estampille de temps de TCP introduite par l'option "TCP timestamps" (décrite au paragraphe 3.4.1) sur le débit de la connexion considérée. Pour cela on a désactivé cette option en affectant la valeur 0 à la variable *tcp_timestamps* se trouvant dans le fichier */proc/sys/net/ipv4/tcp_timestamps* sous Linux (cette variable étant activée par défault). Pour les mesures effectuées dans les expériences précédentes l'option "TCP timestamps" était activée.

La courbe de débit obtenue est la suivante:

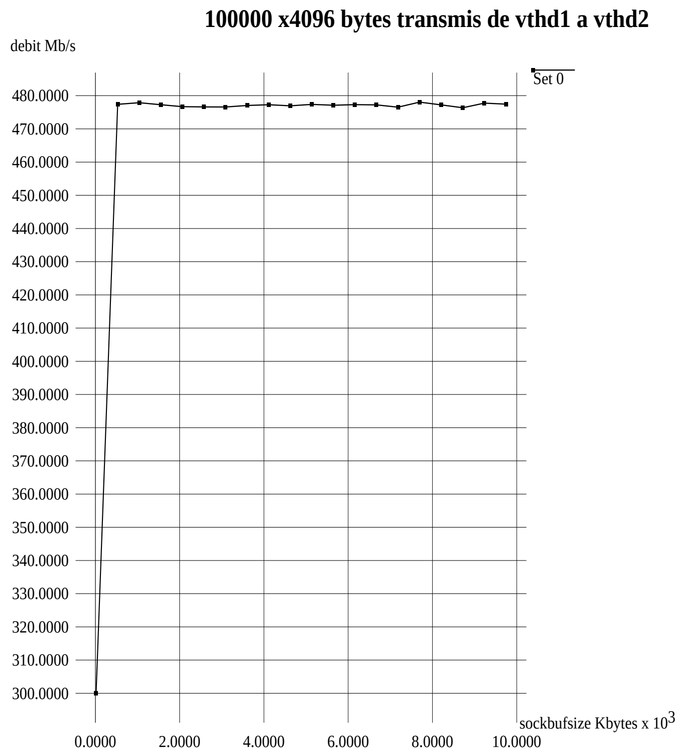


FIG. 4.4 – *Débit obtenu avec l'option d'estampille de temps désactivée*

On constate d'après cette expérience qu'un gain de 40 Mbits/s apparaît sur le débit par rapport aux tests effectués avec l'option *tcp_timestamps* activée.

4.1.2.5 Effet des acquittements séléctifs

De la même manière que l'expérience précédente on a désactivé l'option TCP SACK en mettant la variable *tcp_sack* se trouvant dans le fichier */proc/sys/net/ipv4/tcp_sack* sous Linux à la valeur 0 (cette variable est activée par défault).

La courbe obtenue est la suivante:

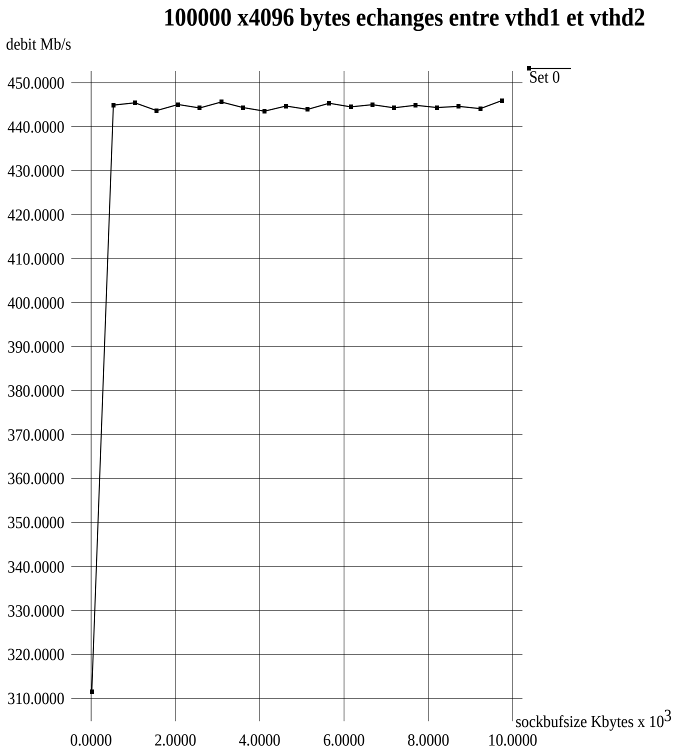


FIG. 4.5 – *Débit obtenu avec l'option TCP SACK désactivée*

On remarque que la désactivation de l'option TPC SACK n'a pas eu d'effet sur le débit de la connexion considérée puisqu'on obtient une courbe similaire à la courbe de la figure 4.2 ayant comme débit maximal 445 Mb/s.

4.1.3 Analyse des résultats obtenus

4.1.3.1 Impact de la taille des buffers de la socket

La connexion qui a été testée dans cette partie est caractérisée par un délai de transmission très réduit ($250\mu s$) vu que les deux machines considérées sont directement liées sur les cartes Gigabit Ethernet par une fibre optique multimode.

La valeur réduite du RTT implique une valeur du produit délai-bande passante peu élevée, c'est pour cette raison que le débit maximal a été atteint pour une valeur du buffer de

512Ko, et que la valeur du débit reste stable et inchangée suite à l'augmentation de la taille du buffer qui a été augmentée jusqu'à une valeur de 1Go.

Cependant le débit réduit obtenu pour une taille du buffer de 64Ko peut être expliqué par la sous-utilisation de la bande passante de la fibre optique due à l'envoi d'une quantité de données inférieure au produit délai-bande passante (voir chapitre 3).

La transmission de données en TCP pour le cas d'une taille du buffer de la socket supérieure au produit délai-bande passante peut être schématisée par la figure suivante:

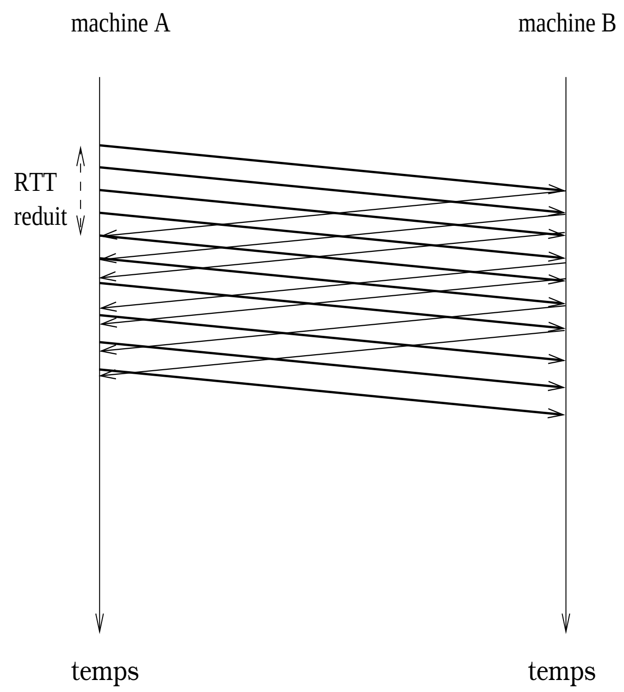


Fig. 4.6 – *Transmission de données pour un délai réduit*

4.1.3.2 Limitation de la valeur du débit

On constate que les débits engendrés pour la configuration considérée se limitent à 512Mbits/s en UDP et à 445Mbits/s en TCP, ceci est principalement dû aux limitations du "hardware":

- Le bus PCI 32 bits fonctionnant à une vitesse d'horloge de 33 Mhz déjà limité physiquement à un débit de 132 Mo/s devient vite submergé pour les hauts débits.

– La taille de la mémoire RAM importe également, une taille de 512Mo ne permet pas d'atteindre pleinement la bande passante de la fibre optique.

4.1.3.3 Effet de l'estampille de temps TCP

L'utilisation du mécanisme d'estampille de temps cause une réduction du débit de la connexion vu la charge induite pour l'émetteur et le récepteur (consommation de ressources système). L'emetteur maintient la valeur de l'estampille envoyée dans un segment et calcule la valeur estimée du RTT en fonction de l'estampille de temps envoyée par le récepteur dans un ACK.

Cependant la désactivation de l'option TCP Timestamps peut engendrer de grandes dégradations de la performance de TCP qui ne peuvent pas être mises en évidence dans le cas de la connexion considérée vu le taux de perte très réduit qui la caractérise.

4.1.3.4 Effet des acquittements sélectifs

L'utilisation du mécanisme des acquittements sélectifs SACK n'a pas prouvé son intérêt dans le cas de la connexion considérée vu que le débit mesuré en désactivant l'utilisation de l'option TCP SACK ne diffère pas de celui obtenu en permettant l'utilisation de cette option.

Ceci peut être considéré comme étant un résultat logique vu que le mécanisme TCP SACK permet de remédier à la dégradation de l'utilisation de la bande passante suite à la perte de plusieurs paquets par fenêtre de transmission alors que dans le cas de l'environnement de la connexion testée le taux de perte de paquets est assez réduit.

4.2 Tests effectuées sur le réseau VTHD

Dans le but d'évaluer les performances de TCP sur une connexion de bout en bout à très haut débit, deux machines ont été connectées sur des cartes réseau Gigabit au réseau expérimental VTHD:

– Une machine locale se trouvant à l'INRIA Sophia Antipolis.
– Une machine distante placée à l'INRIA Paris.

À la différence de la connexion en point à point testée dans la partie précédente cette connexion est caractérisée par un délai de transmission plus grand et par la présence de quatre routeurs dorsaux (principaux), exploitant les capacités du multiplexage en longueur d'onde WDM, sur la liaison connectant les deux machines.

De la même manière que pour les expériences effectuées en point à point l'outil NTTCP a été utilisé pour mesurer le débit de transmission de données sur le réseau.

L'ensemble des tests réalisés pour cette connexion ainsi que l'interprétation des résultats obtenus sont présentés dans cette section.

4.2.1 Description du réseau VTHD

4.2.1.1 Architecture physique

Le réseau dorsal (réseau principal) VTHD comporte huit routeurs interconnectés par des canaux optiques à 2,5 Gbit/s. Sur ce réseau dorsal sont raccordés 18 routeurs d'accès agrégeant le trafic généré par les réseaux locaux d'entreprise des organisations exploitant la plate-forme VTHD.

Le réseau VTHD est pour le présent un réseau fermé: il offre uniquement la connectivité entre les sites VTHD.

4.2.1.2 Routage dans le réseau IP VTHD

Le routage interne au réseau dorsal VTHD permet la distribution de l'information d'accessibilité des routes internes au réseau dorsal.

La connaissance de l'accessibilité des routes propres aux sites connectés au réseau dorsal VTHD est réalisée de manière externe par le protocole BGP-4 ce qui permet d'obtenir dynamiquement les informations d'accessibilité des sites connectés au réseau VTHD et de détecter leur inaccessibilité en cas de panne du réseau VTHD. Le réseau n'offre pas de garantie de disponibilité du service en raison de son caractère expérimental.

4.2.1.3 Systèmes de transmission du réseau

Une architecture IP/WDM où les routeurs sont directement interconnectés par des systèmes de transmission optiques WDM est adoptée pour la partie dorsale du réseau VTHD.

En associant sur une même fibre optique, plusieurs porteuses optiques ou longueurs d'onde, la capacité de transmission des fibres déployées est de l'ordre du térabit/s (10^{12}bit/s). Les longueurs d'onde interconnectant les routeurs du réseau dorsal VTHD portent chacune un débit de 2,5 Gbit/s.

Les liens d'accès interconnectant les routeurs dorsaux et les routeurs agrégeant le trafic des réseaux locaux utilisent le même principe consistant à transporter directement les flux de paquets IP sur les porteuses optiques. La technologie Ethernet dans sa version haut débit: Gigabit Ethernet, est utilisée sur les réseaux locaux. Les fibres d'accès n'exploitent pas cependant les capacités du multiplexage en longueur d'onde. Chaque site peut ainsi générer un débit de l'ordre du gigabit/s vers le réseau dorsal.

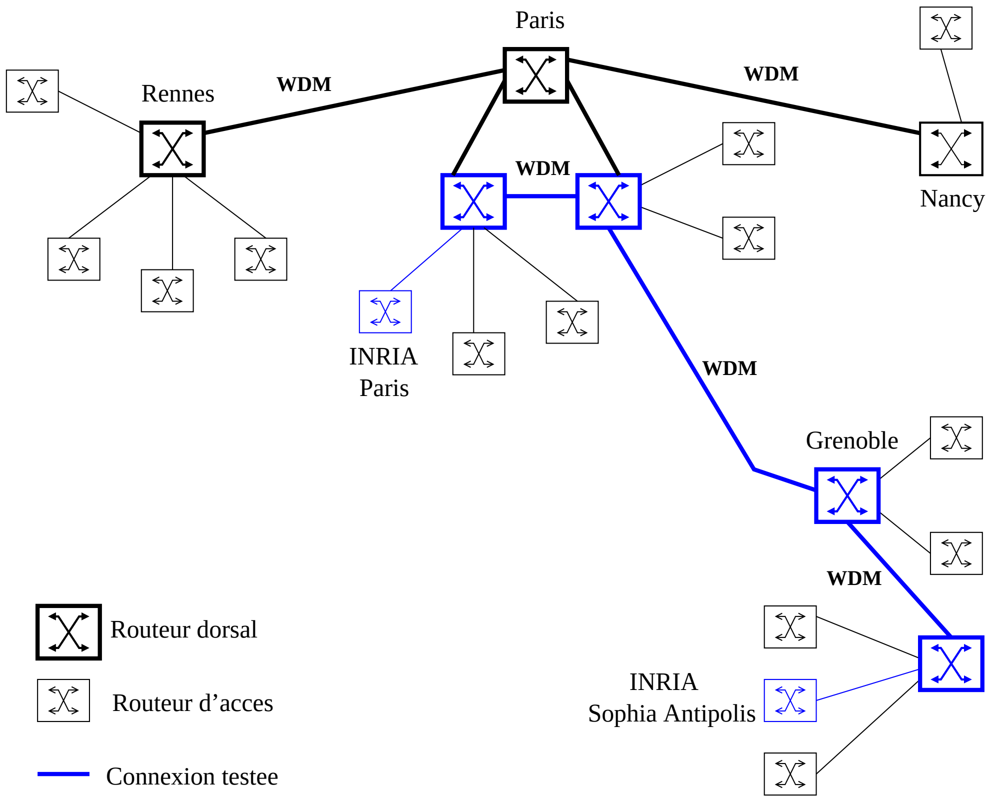


FIG. 4.7 – *Connexion du réseau expérimental VTHD testée*

4.2.2 Mesures du débit de transmission de bout en bout

4.2.2.1 Mesure du débit de TCP

Dans cette expérience on mesure le débit d'une connexion TCP de bout en bout en faisant varier la taille des tampons de la socket TCP à l'aide de l'option *-w* de l'outil NTTCP. La valeur de la MTU considérée est de 1500 octets.

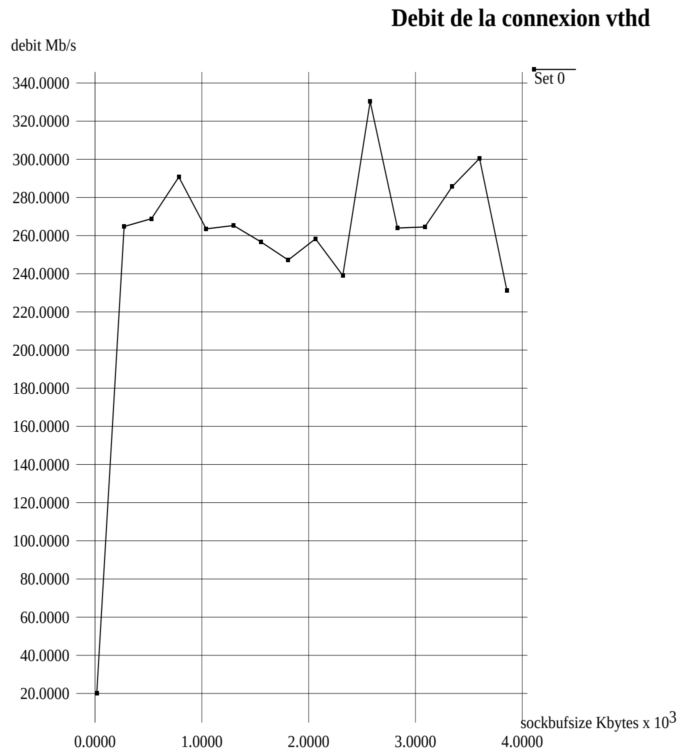


FIG. 4.8 – *Débit de TCP mesuré sur la connexion VTHD*

D'après cette courbe le débit maximal réalisé est de 330 Mbits/s, cependant on remarque que pour la valeur initiale du buffer de la socket TCP qui est de 64KB un débit beaucoup plus faible est obtenu: 20 Mbits/s.

4.2.2.2 Augmention de la taille des tampons de la socket TCP

La connexion haut débit du réseau VTHD considérée présente les caractéristiques suivantes:

- une bande passante théorique de 1Gbits/s disponible sur la fibre optique
- un délai de transmission moyen de 15ms.

ce qui implique un produit délai-bande passante assez élevé (15 Mbits).

Pour mettre en évidence l'influence de la quantité de données transmise sur le débit de la connexion ayant un grand produit délai-bande passante, on a procédé à la variation de la taille des buffers d'émission et de réception de la socket TCP en modifiant à chaque fois les valeurs se trouvant respetivement dans les fichiers:

/proc/sys/net/core/wmem_max et */proc/sys/net/core/rmem_max* sous le système Linux.

Les valeurs du débit obtenues sont récapitulées dans le tableau suivant:

Taille du buffer de la socket TCP	64Ko	128Ko	512Ko	1Mo	512Mo	1Go
Débit (Mbits/s)	21,5	43,1	182	267	279	320

TAB. 4.2 – *Débit en fonction de la taille des tampons de la socket TCP*

D'après ce tableau on remarque bien que pour des buffers de taille inférieure à 1MB les débits engendrés sont relativement faibles par rapport à ceux obtenus pour 512MB ou 1GB et qui dépassent les 270Mbits/s.

En passant de la taille du buffer prise par défault par le système Linux qui est de 64Ko à une taille de 1Go un gain très important (300Mbits/s) a été obtenu pour le débit de la connexion.

4.2.2.3 Mesure du débit en fonction de la variation de la MTU

La MTU qui représente la taille de l'unité de transfert d'information maximale pouvant être transportée sans fragmentation sur le lien peut s'avérer déterminante pour les performances de TCP étant donné que cette valeur représente la taille des trames qui acheminent les datagrammes entre la machine source et destination en transitant par les éléments intermédiaires sur le réseau. La MTU peut avoir de grands effets sur l'utilisation de la capacité du réseau dans le cas des réseaux hauts débits, c'est ce qu'on va essaye de mettre en évidence à partir de cette expérience.

La MTU ayant une valeur de 1500 octets par défaut a été modifiée en utilisant la ligne de commande:

ifconfig *[interface] [adresse]* ***mtu*** *[valeur_mtu]*

La courbe du débit obtenue en diminuant la MTU à 1000 octets est la suivante:

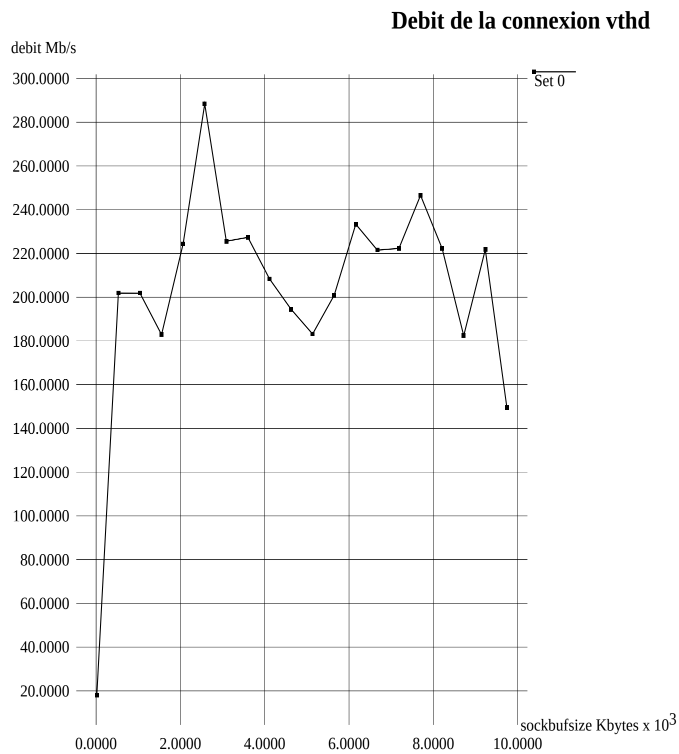


FIG. 4.9 – *Courbe du débit TCP pour une MTU de 1000 octets*

La figure 4.10 représente le débit mesurée en fonction de l'augmentation de la taille des buffers de la socket pour une MTU de valeur 800 octets.

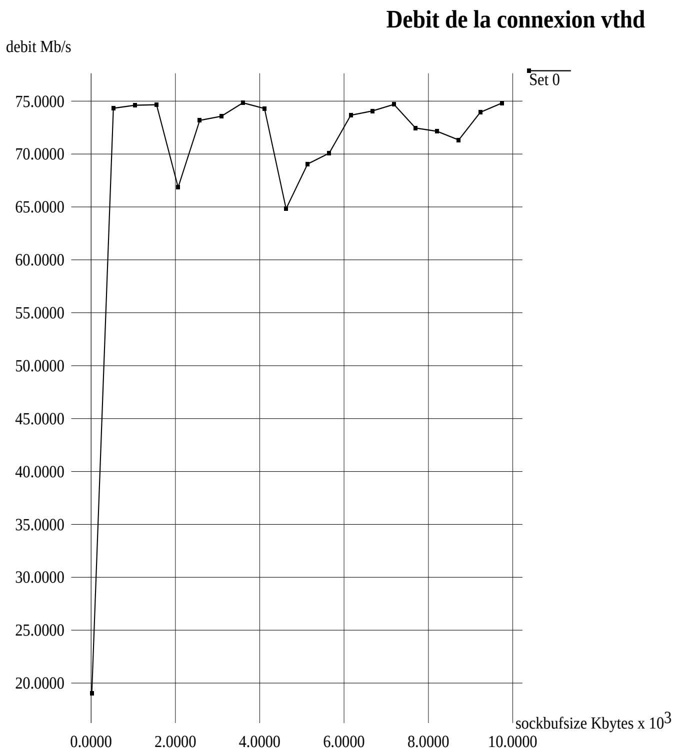


FIG. 4.10 – *Courbe du débit TCP pour une MTU de 800 octets*

Pour une valeur de la MTU égale à 800 octets on constate qu'il y a eu une chutte du débit qui s'est limité à 75Mbits/s, pour une MTU de 1000 octets le débit diminue aussi mais reste à un niveau relativement proche du débit mesuré avec une MTU de 1500 octets.

Les courbes de débit obtenues pour des valeurs de la MTU supérieures à 1500 octets sont représentées sur la figure 4.11 et la figure 4.12.

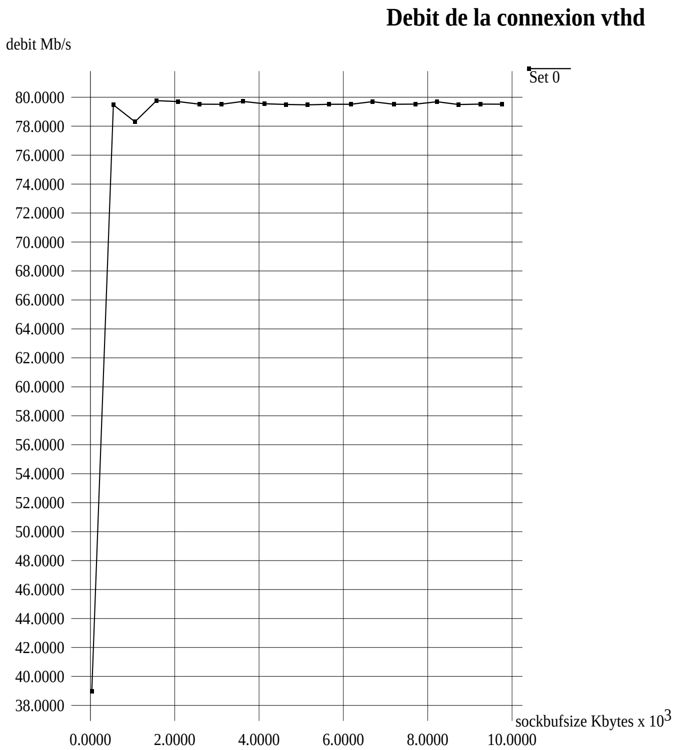


FIG. 4.11 – *Courbe du débit TCP pour une MTU de 2000 octets*

En opposition au résultat attendu on a remarqué que pour des valeurs de la MTU supérieures à 1500 octets le débit stagne à une valeur de 80 Mbits comme le montre la figure 4.11 pour une MTU de 2000 octets et la figure 4.12 pour une MTU de 9000 octets.

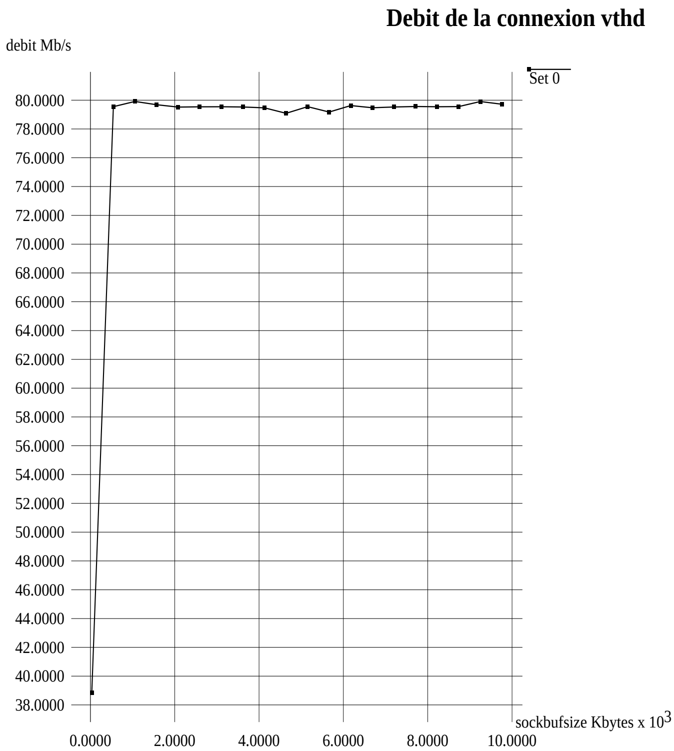


FIG. 4.12 – *Courbe du débit TCP pour une MTU de 9000 octets*

4.2.2.4 Effet de l'utilisation des estampilles de temps

Le mécanisme d'estampilles de temps de TCP (voir 3.4.1) a été introduit dans le but d'assurer la fiabilité et la performance du fonctionnement de TCP dans le cas des réseaux haut débit caractérisés par une vitesse de transmission des données très élevée et ceci en permettant une estimation précise et dynamique du RTO à partir du calcul des RTT ce qui permet d'éviter le gaspillage de la bande passante causé par la retransmission inutile de paquets.

Afin de mettre en évidence l'effet de l'utilisation de cette option on a procédé à sa désactivation (en exécutant la commande *echo 0> /proc/sys/net/ipv4/tcp_timestamps* sous Linux).

On a obtenu la courbe de débit suivante:

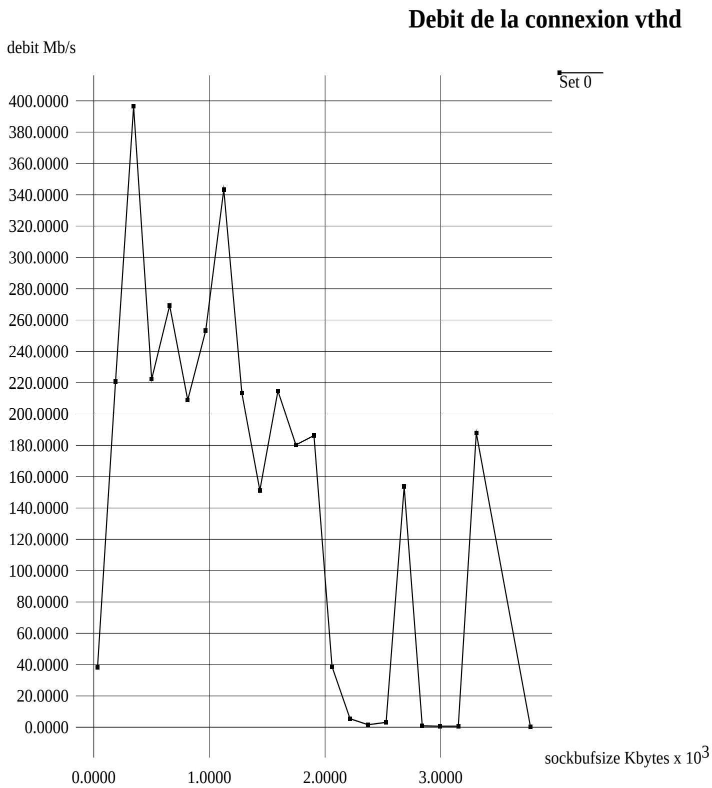


FIG. 4.13 – *Courbe du débit de TCP avec désactivation de l'option TCP Timestamps*

Il est bien clair à partir de cette courbe que la performance de TCP se dégrade de manière remarquable lorsque l'option d'estampilles de temps de TCP n'est pas utilisée. Suite à l'augmentation de la taille des buffers de la socket TCP et en conséquent l'augmentation de la quantitée de données transmises, le débit chutte de plus en plus vers des valeurs plus faibles pour atteindre des valeurs inférieures à 10 Mbits/s.

Cependant une valeur du débit qui dépasse les valeurs déjà obtenues lorsque l'option TCP Timestamps était activée a été obtenue (397Mbits/s) mais ceci étant pour une taille du buffer de 300Ko, c'est à dire pour un flux de données qui n'est pas très important.

4.2.2.5 Effet de l'utilisation du mécanisme TCP SACK

Dans le but d'identifier l'effet de l'utilisation des acquittements sélectifs sur le débit de la connexion TCP haut débit considérée une mesure du débit a été réalisée après désactivation de l'option TCP SACK par la commande: *echo 0 > /proc/sys/net/ipv4/tcp_sack*.

La courbe obtenue est la suivante:

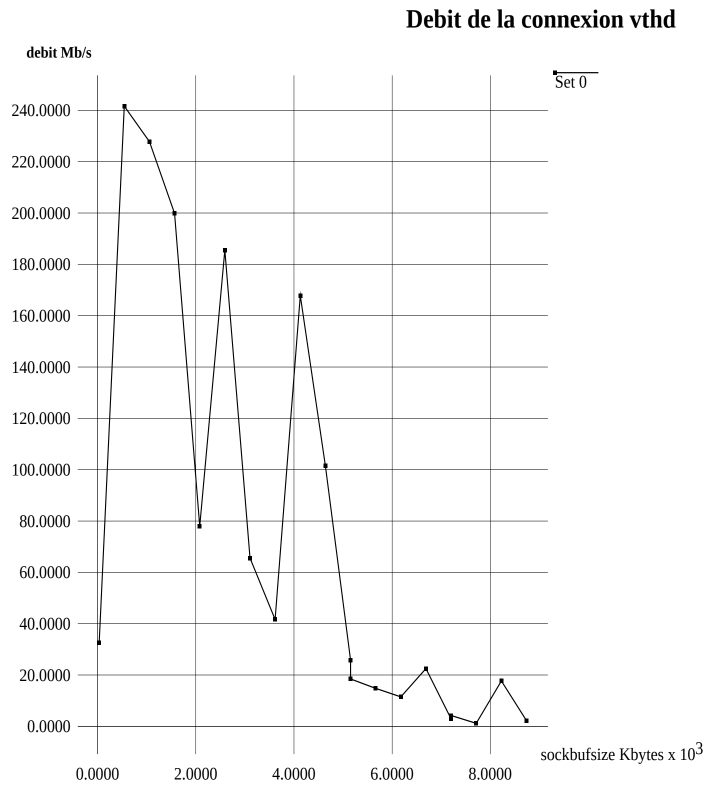


FIG. 4.14 – *Courbe du débit de TCP avec désactivation de l'option TCP SACK*

D'après cette courbe la valeur maximale du débit est de 240 Mbits/s ce qui présente une diminution de 90 Mbits par rapport au débit maximal engendré losque l'option TCP

SACK était activée.

On constate que le débit de la connexion diminue progressivement lorsqu'on augmente la taille du buffer de la socket pour atteindre des valeurs très faibles et voir inacceptables du débit ($<$ 10 Mbits/s).

4.2.3 Analyse des résultats

4.2.3.1 Effet de la bufferisation

La bande passante disponible sur le lien réseau qui est de 1Gbits/s et le délai de transmission moyen qui est de 15ms font que le produit délai-bande passante soit élevé (15Mbits), ce qui nécessite une taille du buffer de la socket supérieure à 1,87Mo afin de remplir le canal de transmission et de maintenir une utilisation efficace de la bande passante.

C'est pour cette raison que pour une taille du buffer de 64Ko, qui est la taille prise par défaut par le système Linux, le débit est assez faible (21,5Mbits/s). Ce cas peut être schématisé par la figure 4.15 où la source n'arrive pas à utiliser pleinement la bande passante disponible sur le lien vu la limitaion de la taille du buffer. Ce problème a pu être résolu en augmentant la taille du buffer de la socket TCP sous Linux, le débit atteint alors les 300Mbits/s pour un buffer de 1Go.

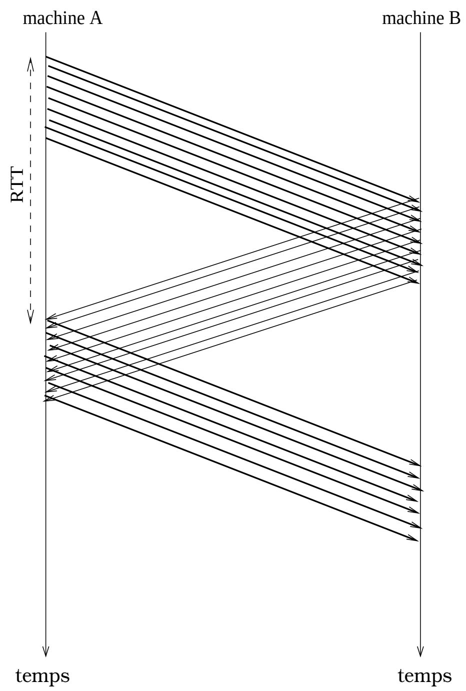


FIG. 4.15 – *Transmission de données sur un réseau haut débit*

4.2.3.2 Impact de la MTU sur la performance de TCP

Les segments TCP sont acheminés dans des datagrammes IP eux mêmes encapsulés dans des trames physiques. Chaque segment comporte en plus des données au minimum 40 octets d'entêtes IP et TCP, ce qui explique l'utilisation médiocre de la bande passante pour des valeurs de la MTU inférieures à 1500 octets (débit maximal de 75Mbits/s pour une MTU de 800 octets). Une petite MTU fait augmenter le nombre de paquets transitant sur le réseau ce qui cause la dégradation du débit liée à la croissance des traitements effectués par le protocole TCP.

D'autre part on constate qu'une grande MTU conduit également à de faibles performances (débit maximal de 80Mbits/s pour une MTU de 2000 octets), ceci est dû à la fragmentation IP effectuée par les routeurs du réseau à la réception de datagrammes qui excèdent leur capacité (MTU). À la différence des datagrammes, les fragments ne constituent pas des messages indépendants. Tous les fragments d'un datagramme doivent parvenir à destination, faute de quoi la totalité du datagramme doit être retransmise. Augmenter la taille du segment au-delà du seuil de fragmentation diminue la probabilité de transmission correcte et donc diminue le débit.

4.2.3.3 Effet des estampilles de temps

L'expérience réalisée en désactivant l'option d'estampille de temps de TCP prouve l'importance de l'utilisation de ce mécanisme dans le cas d'un transfert de données en haut débit, à l'opposé de ce qui a été constaté pour le test réalisé en point à point où l'utilisation de l'estampille de temps a montré un effet négatif sur la performance de TCP.

Ceci est dû au fait que le taux de perte de paquets croît en fonction de l'augmentation de la quantité de données envoyée par la source, les routeurs ayant des capacités de bufferisation limitées deviennent surchargés et par conséquent rejettent des paquets.

Le temporisateur de retransmission est un paramètre qui influe énormément sur les performances de TCP: si le RTO est sous-estimé TCP peut retransmettre inutilement les des segments déjà reçus et s'il est sur-estimé une perte sera détectée tardivement résultant en un temps d'inactivité de TCP assez important. Le mécanisme d'estampilles de temps s'avère primordial pour que TCP puisse se rendre compte rapidement de la perte de paquet et pour qu'il ait des informations précises et dynamiques reflétant l'état du réseau.

4.2.3.4 Avantage des acquittements sélectifs

L'expérience effectuée en désactivant l'option TCP SACK a permis de mettre en évidence le grand avantage de l'utilisation des acquittements sélectifs de TCP dans le cas d'une connexion haut débit. Lorsque les acquittements sélectifs ne sont pas utilisés le débit de la connexion se dégrade de plus en plus qu'on augmente la taille de la fenêtre d'émission. Cet effet peut être expliqué par l'augmentation de la probabilité de perte de plusieurs paquets par fenêtre d'émission losque la taille de celle ci est importante, la perte de paquets entraîne la réduction de la taille de la fenêtre d'émission à la moitié et l'exécution de la phase slow start. Les acquittements sélectifs permettent à TCP de se remettre de la perte de plusieurs paquets en évitant la grande réduction de la fenêtre au cas de l'exécution de la phase “slow start” et par conséquent permet de maintenir une utilisation stable et efficace de la bande passante du réseau.

4.3 Conclusion

D'après les expérimentations effectuées sur le réseau haut débit VTHD on peut souligner l'importance du mécanisme d'estampilles de temps de TCP dans le cas d'un réseau à très haut débit qui nécessite une estimation précise du temporisateur de retransmission afin d'éviter le gaspillage de bande passante suite à des retransmissions inutiles causées par une mauvaise estimation du RTO.

Les acquittements sélectifs de TCP prouvent aussi leur efficacité pour le cas d'un environnement réseau haut débit, ils permettent de maintenir une bonne utilisation de la bande passante en cas de perte de plusieurs paquets par fenêtre d'émission.

La taille du buffer de la socket TCP importe également grandement sur l'utilisation de la bande passante. Pour une bande passante de 10 Mbits/s ou 100 Mbits/s un buffer de 64 ko suffit, alors qu'en Gbits/s on s'aperçoit qu'un buffer d'au moins 1 Mo est indispensable voir même 512 Mo et 1 Go.

Les limitations du “hardware” font que le débit engendré (débit maximal de 320 Mbits/s en TCP) n'utilise pas pleinement la capacité en bande passante disponible sur le support de transmission optique.

Enfin la capacité de bufferisation des routeurs a une grande influence sur les performances de TCP vu la grande quantité de paquets transitant sur le réseau haut débit. Une surcharge des routeurs peut causer une grande détérioration de performances suite à un rejet de paquets.

Chapitre 5

Réalisation

5.1 Environnement

5.1.1 Environnement matériel

Au cours de ce stage, la configuration matérielle suivante a été utilisée:

- Deux machines surlesquelles les tests ont été réalisés et qui présentent la configuration suivante:

Processeur	Pentium III, 933 Mhz.
Bus	Bus PCI 32 bits, 33 Mhz
Cartes réseau	Carte 3Com 3C985 SX Gigabit Ethernet, Carte 10/100 Mbits
RAM	512 Mo
Carte mère	Asus CUV 4X-DLS

TAB. 5.1 – *Configuration matérielle des machines testées*

5.1.2 Environnement logiciel

Les tests ont été effectués sur un noyau Linux 2.2.16.

Les outils qui ont été utilisés sont:

- NTTCP (New Test TCP program) version 1.47 utilisé pour effectuer les mesures de débit (voir l'annexe A pour la description de cet outil).

- L'analyseur de réseaux ETHEREAL (voir l'annexe B)version 0.8.18.
- XGRAPH version 12.1 pour le traçage des courbes.

Pour le rapport l'éditeur de texte LyX (éditeur WYSIWYM) a été utilisé.

5.2 Travail réalisé

5.2.1 Etude théorique

La première partie du travail à été consacrée à la documentation. Au cours de cette partie, une étude bibliographique poussée a été menée afin d'assimiler les mécanismes de base du contrôle de flux et de congestion de TCP et les améliorations qui ont été proposées pour adapter le protocole aux environnements haut débit.

Cette étude s'est basée en particulier sur les RFC (Request For Comments) relatives au protocole TCP et aussi sur plusieurs articles et thèses s'intéressant à ce sujet.

5.2.2 Expérimentations

La première étape de cette partie a été consacrée à l'installation et la configuration du système Linux sur deux machines ayant chacune une interface Gigabit Ethernet ainsi qu'à l'installation et la familiarisation avec l'outil de test de performance de TCP (NTTCP).

En une seconde partie on a effectué un ensemble de tests et de mesures de débits de TCP sur une connexion point à point entre les deux machines liées par une fibre optique sur les cartes réseau gigabit. Après avoir identifier les limites de performance du protocole sur cette connexion, on s'est interessé à la réalisation de tests sur une connexion de bout en bout entre deux machines distantes du réseau VTHD.

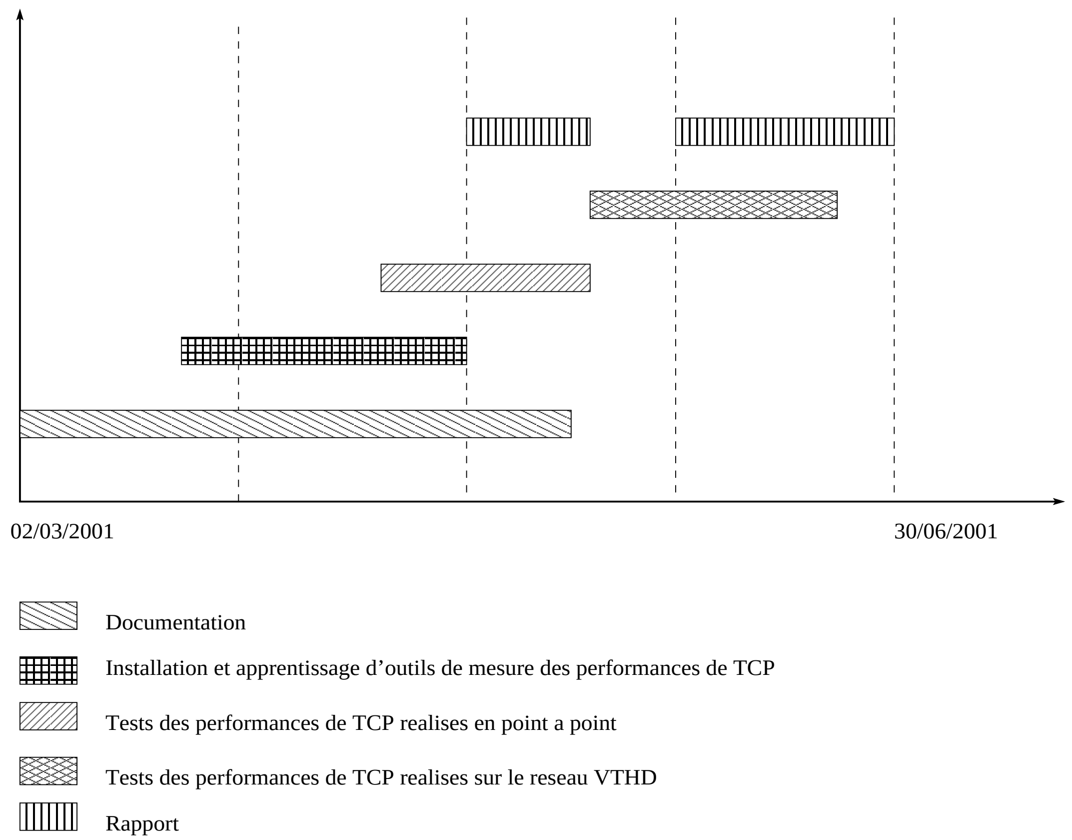
5.2.3 Perspectives

Les modèles analytiques des performances de TCP existant dans la littérature n'étant pas valables pour le cas d'un environnement réseau à très haut débit, il serait intéressant de développer une modélisation analytique permattant d'estimer les performances (débit moyen...) de TCP dans un tel environnement.

Cette modélisation permettra de mettre en évidence et d'améliorer les différents paramètres relatifs au contrôle de flux et de congestion de TCP ainsi qu'aux propriétés et comportements des éléments du réseau influençant la performance de TCP.

Chapitre 6

Chronogramme



Chapitre 7

Conclusion générale

Ce travail entre dans le cadre du développement des technologies nécessaires pour la mise en place d'un Internet de nouvelle génération et ceci à travers l'étude des performances des protocoles Internet de contrôle de flux et de congestion dans le but de les valider ou les adapter à l'environnement d'un réseau très haut débit.

On a commencé par une présentation détaillée des mécanismes de TCP, puis on a identifié les problèmes que le protocole rencontre avec les nouveaux supports de transmission et on a présenté les différentes solutions qui ont été proposées pour résoudre ces problèmes.

D'àprès les expérimentations qu'on a effectué sur une connexion à très haut débit utilisant la fibre optique comme support de transmission on peut retirer les conclusions suivantes:

- Les mécanismes de contrôle du flux de bout en bout du protocole TCP révèlent leur robustesse et leur adaptabilité à l'environnement d'un réseau à très haut débit vu l'implémentation de TCP qui ne considère pas d'hypothèses sur le type d'infrastructure ou d'équipements réseaux utilisés.
- L'utilisation du mécanisme d'estampilles de temps de TCP est très important dans le cas d'un réseau haut débit vu qu'il offre une estimation dynamique et assez précise du temporisateur de retransmission, ce paramètre influe énormément sur les performances de TCP, ce dernier n'ayant aucune connaissance des caractéristiques du réseau physique et de la charge relative se base sur l'estimation du RTO pour détecter l'état de perte de paquet et ajuster par conséquent son débit.
- Les performances de TCP sont nettement meilleures lorsque le mécanisme d'acquittements sélectifs est utilisé, vu la probabilité élevée de perte de plusieurs paquets

par fenêtre dans le cas d'un réseau haut débit caractérisé par une fenêtre d'émission de taille importante. Les acquittements sélectifs permettent d'éviter la dégradation de performance de TCP causée par l'expiration du temporisateur de retransmission et l'exécution de la phase "slow start" suite à la perte de plusieurs paquets.

– L'augmentation de la taille du buffer de la socket TCP permet une utilisation plus efficace de la bande passante disponible sur la fibre optique cependant des limitations liées à l'architecture matérielle (bus, mémoire) surviennent et font que la bande passante ne soit pas pleinement utilisée.
– La capacité de bufferisation des routeurs a aussi une grande influence sur les performances de TCP dans le cas d'un réseau haut débit vu que l'augmentation de la taille de la fenêtre d'émission résulte en une transmission d'un grand nombre de paquets ce qui peut causer une surcharge des routeurs du réseau et par suite une détérioration des performances si les routeurs ne sont pas bien dimensionnés.

Malgré la diversité des modèles et des solutions développées, le contrôle de flux et de congestion dans l'Internet reste un sujet ouvert. Des problèmes ne cessent pas de surgir avec l'expansion rapide du réseau et l'ajout de nouveaux supports de transmission. L'environnement d'un réseau à très haut débit nécessite une étude analytique poussée pour bien comprendre son impact sur TCP.

A titre personnel, ce travail m'a permis de découvrir un domaine de recherche très étendu où beaucoup de travaux sont entrain d'être réalisés.

Le stage n'a pas manqué l'aspect pratique qui a consisté à la configuration des systèmes testés à l'environnement haut débit et à la réalisation de plusieurs expérimentations sur des connexions réseau à l'aide d'outils de mesure de performance et d'analyse de réseaux.

Annexe A

NTTCP - New Test TCP program

A.1 Synopsis

nttcp [options locales] hôte distant [options distantes].

A.2 Description

Le programme *nttcp* [1] mesure le taux de transfert sur une connexion TCP, UDP ou sur une connexion UDP multicast. Pour utiliser *nttcp* il faut le lancer en réception sur une machine et en mode émission sur une autre, il transmet un nombre déterminé de buffers de taille définie en mesurant le temps réel que cela prend, le débit de transmission, le nombre d'appels système, les ressources système consommées...

Sur la machine distante on lance *nttcp* avec l'option -i, *nttcp* se met alors en attente de connexions avec d'autres machines.

Sur la machine locale *nttcp,* appelé avec le nom de la machine distante initialise la connexion avec celle ci et commence le transfert.

Par défaut, le programme transfère 2048 buffers de 4Ko (8Mo de données en total) à la machine distante. La mesure de performance est effectuée des deux côtés et les résultats sont affichés sur la machine locale. Les paramètres de la transmission peuvent être modifiés par les options de la ligne de commande [13].

1. http://home.leo.org/~bartel/nttcp

A.2.1 Options

-r

définit la direction de transfert en réception, les données sont transmises de la machine distante vers l'hôte local.

-t

définit la direction de transfert en émission, les données sont transmises de la machine locale vers la machine distante, c'est la direction prise par défaut.

-T

affiche une ligne de titre.

-u

utilise le protocole UDP au lieu de TCP (qui est pris par défaut).

-v

donne plus de détails au niveau des paramètres résultats de l'execution de nttcp.

-f format

spécifie un format défini pour l'impression des résultats.

-n

définit le nombre de buffers à écrire sur la socket de transmission, par défaut c'est 2048.

-l longueur du buffer

la longueur donnée définie la taille d'un seul buffer écrit transmis sur la socket de transmission, par défaut c'est 4096 .

-w nombre de Kilo octets

définit la taille du buffer de la socket de transmission et de réception. Cette valeur dépend du système, généralement elle est de 16Ko.

-p numéro de port

par défaut l'hôte distant lance l'écoute sur le port 5037, ceci peut être modifié par cette option.

-i

cette option active nttcp comme étant un processus démon en attente de connexions avec d'autres machines.

-m multicast IP : port

cette option est utilisée pour émettre vers l'adresse multicast et le port spécifiés.

A.2.2 Paramètres résultats

Le résultat du programme consiste en deux lignes de nombres, ou plusieurs lignes si la transmission multicast est utilisée. La première ligne indique les résultats des mesures pour la machine locale, la seconde affiche les mesures effectuées pour la machine distante. Le format par défaut est le suivant:

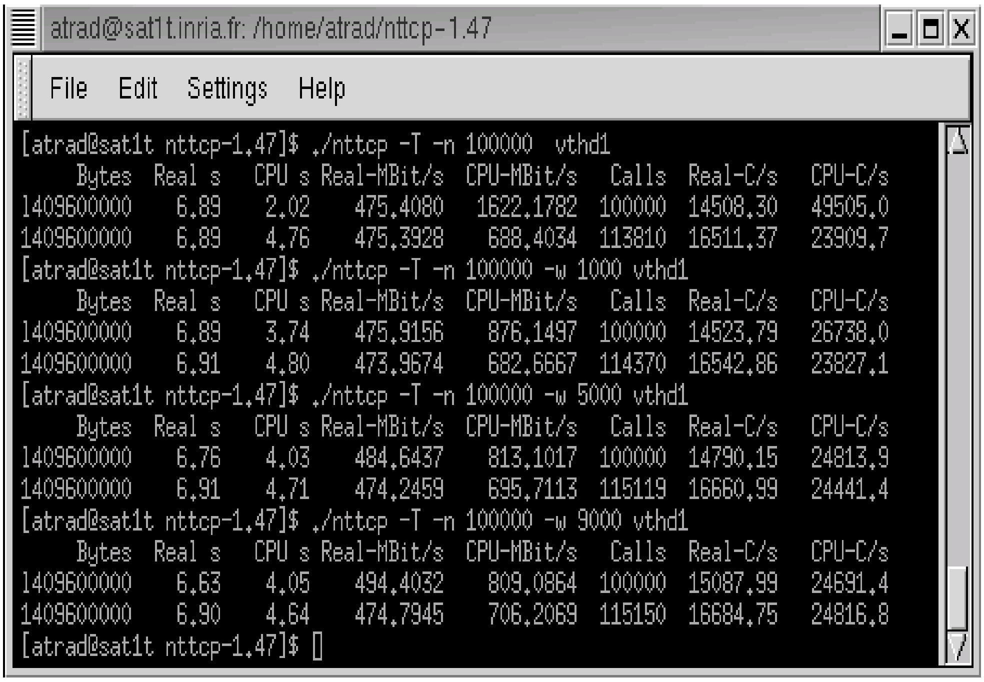
Bytes Real s CPU s Real-Mbit/s CPU-Mbit/s Calls Real-C/s CPU-C/s

Les mesures marquées avec "CPU" indiquent seulement le temps système utilisé, celles qui sont marquées avec "Real" sont des valeurs calculées en fonction du temps du début jusqu'à la fin de la transmission.

A.2.3 Exemples d'exécution de l'outil NTTCP

Exemple 1

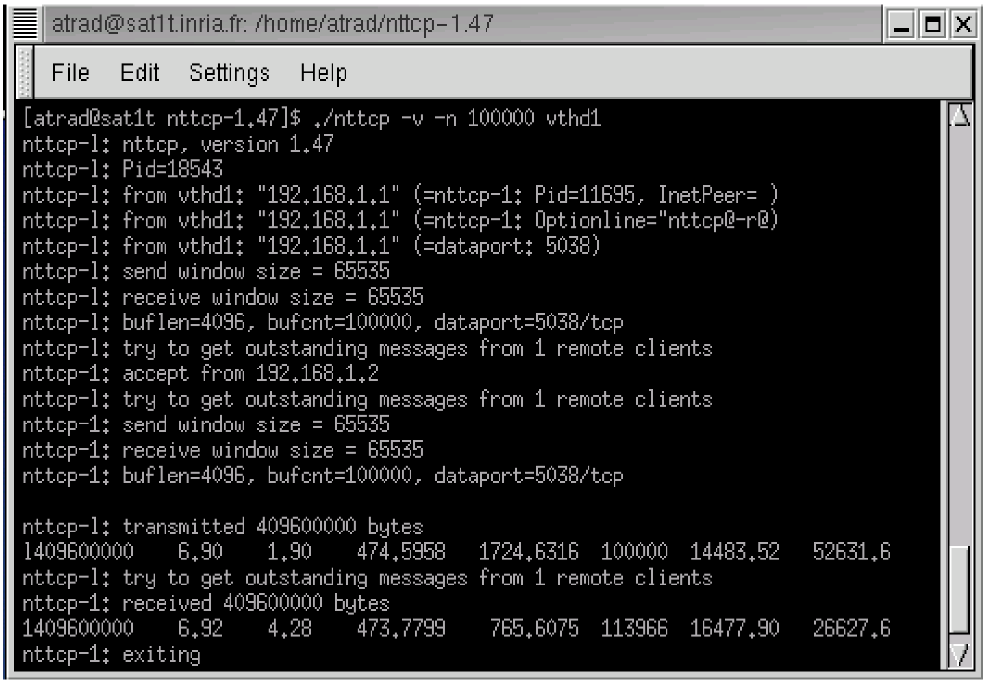
Ce premier exemple correspond à l'exécution d'un test effectué sur une connexion en point à point entre deux machines sur des interfaces réseau Gigabits avec désactivation de l'option TCP Timestamps. Le débit maximal obtenu sur la connexion TCP pour ce test est de 475 Mbits/s.



```
atrad@sat1t.inria.fr: /home/atrad/nttcp-1.47
File   Edit   Settings   Help
[atrad@sat1t nttcp-1.47]$ ./nttcp -T -n 100000  vthd1
     Bytes  Real s   CPU s Real-MBit/s  CPU-MBit/s   Calls  Real-C/s   CPU-C/s
1409600000    6.89    2.02    475.4080   1622.1782  100000  14508.30   49505.0
1409600000    6.89    4.76    475.3928    688.4034  113810  16511.37   23909.7
[atrad@sat1t nttcp-1.47]$ ./nttcp -T -n 100000 -w 1000 vthd1
     Bytes  Real s   CPU s Real-MBit/s  CPU-MBit/s   Calls  Real-C/s   CPU-C/s
1409600000    6.89    3.74    475.9156    876.1497  100000  14523.79   26738.0
1409600000    6.91    4.80    473.9674    682.6667  114370  16542.86   23827.1
[atrad@sat1t nttcp-1.47]$ ./nttcp -T -n 100000 -w 5000 vthd1
     Bytes  Real s   CPU s Real-MBit/s  CPU-MBit/s   Calls  Real-C/s   CPU-C/s
1409600000    6.76    4.03    484.6437    813.1017  100000  14790.15   24813.9
1409600000    6.91    4.71    474.2459    695.7113  115119  16660.99   24441.4
[atrad@sat1t nttcp-1.47]$ ./nttcp -T -n 100000 -w 9000 vthd1
     Bytes  Real s   CPU s Real-MBit/s  CPU-MBit/s   Calls  Real-C/s   CPU-C/s
1409600000    6.63    4.05    494.4032    809.0864  100000  15087.99   24691.4
1409600000    6.90    4.64    474.7945    706.2069  115150  16684.75   24816.8
[atrad@sat1t nttcp-1.47]$ []
```

Exemple 2

Ce second exemple est relatif au même test, l'option -v de NTTCP a été utilisée au lieu de -T, cette option permet d'avoir en résultat plus de détails concernant les paramètres de la connexion TCP.



Annexe B

L'analyseur réseau Ethereal

B.1 Description

Ethereal[1] est un analyseur de réseau pouvant être utilisé sous le système Linux ou Windows.

Il permet d'examiner les données transitant sur un réseau à partir d'une capture effectuée sur les paquets transitant par une interface réseau déterminée.

Ethereal visualise les différents champs relatifs à un paquet (protocole, entête, options, données) permettant ainsi de suivre l'évolution des différents paramètres relatifs à une connexion du réseau [14].

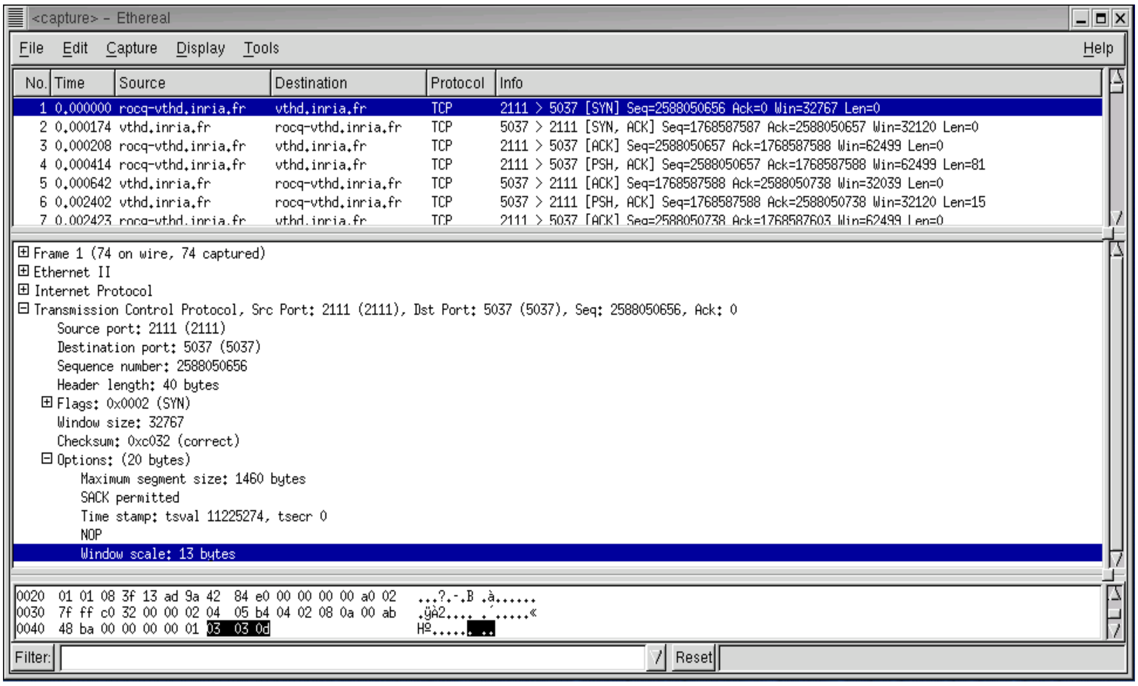
1. http://www.ethereal.com

B.1.1 Exemples d'exécution de Ethereal

Exemple 1

D'après cette capture effectuée par l'outil Ethereal on peut identifier les paramètres relatifs aux segments transitant sur la connexion TCP:

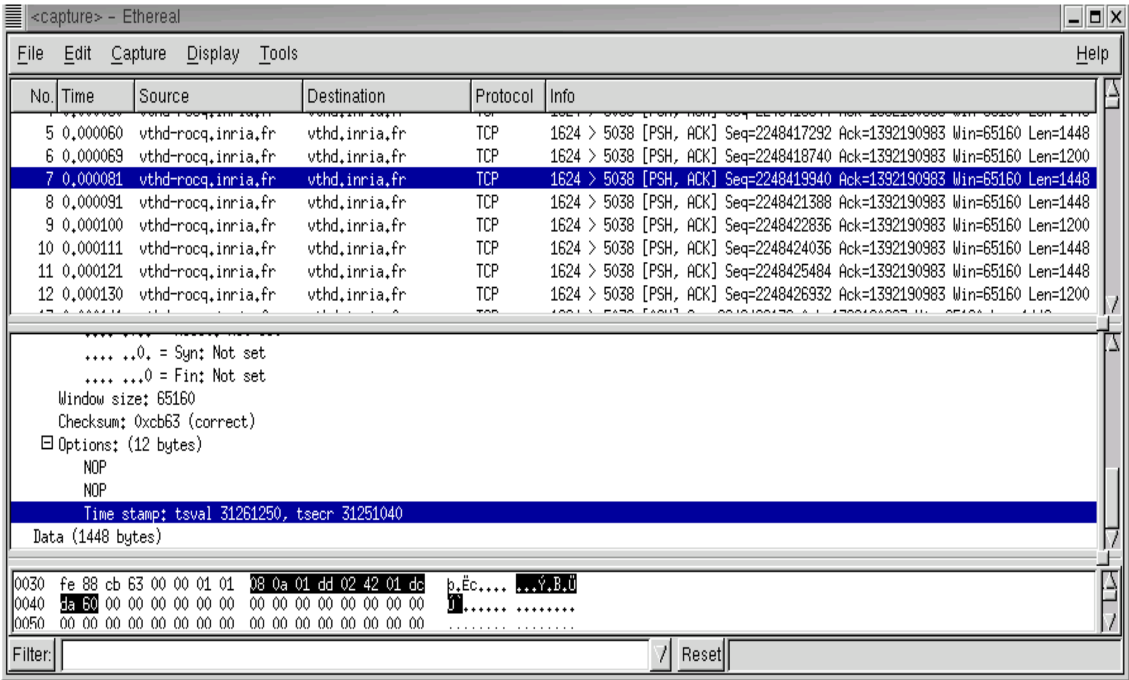
- taille de la fenêtre d'émision.
- taille maximale d'un segment MSS.
- l'option SACK.
- l'option Window scale.



Exemple 2

Cet exemple montre les estampilles de temps envoyés dans un segment TCP:

- La valeur de l'estampille transmise par la source (tsval)
- La valeur de l'estampille renvoyée par la destination (tsecr)



Annexe C

Glossaire

- **ACK** abréviation de acknowledgment, accusé de réception.
- **Bande Passante** (bandwidth) débit d'informations supporté par une connexion réseau.
- **BGP** (Border Gateway Protocol) protocole de passerelle frontière.
- **Capacité d'un réseau** le nombre maximum de paquets pouvant être transmis entre la source et la destination.
- **cwnd** (congestion window) taille de la fenêtre de congestion
- **rwnd** (receiver's advertised window) taille de la fenêtre d'émission indiquée par le récepteur.
- **Débit** (Throughput) moyenne du trafic sur une période de temps donnée, exprimée en bits par seconde.
- **FDDI** (Fiber Distributed Data Interface) réseau en fibre optique du type anneau à jeton, il permet d'avoir un débit de 100 Mbits/s.
- **Fenêtre coulissante** (sliding window) mécanisme utilisé par le protocole TCP pour contrôler le flux à émettre.
- **Mbps** Méga Bits par seconde.
- **Mo** Méga octets.
- **MSS** (Maximum Segment Size) la plus grande quantité d'information pouvant être transmise dans un segment, négociée à l'établissement d'une connexion.
- **MTU** (Maximum Transfer Unit) Taille maximale de l'unité de transmission (un paquet) qui peut être transportée sans fragmentation sur le lien.

- **RFC** (Request For Comments) série de notes qui contiennent des résultats de mesures, des idées, des observations ou des standards TCP/IP.
- **RTO** (Round Trip time-Out) délai avant retransmission calculé par TCP en fonction du délai d'aller retour.
- **RTT** (Round Trip Time) délai d'aller retour entre deux hôtes.
- **SACK** (Selective ACKnowledgment) mécanisme qui consiste à reporter dans les acquittements les blocs contigus de segments reçus et non encore acquittés.
- **Segment** unité de données du protocole TCP.
- **Socket** abstraction proposée par le système Unix pour permettre aux programmes d'application d'accéder aux protocoles TCP/IP.
- **TCP** (Transmission Control Protocol) protocole standard du niveau transport.
- **TDM** (Time Division Multiplexing) multiplexage temporel.
- **Trafic** ensemble des données qui transitent sur un réseau.
- **UDP** (User Datagram Protocol) Protocole de transport qui assure un service de remise au mieux.
- **VTHD** réseau expérimental Vraiment Très Haut Débit.
- **WDM** (Wavelength Division Multiplexing) multiplexage en longueur d'onde.

Bibliographie

[1] BEAUQUIER Bruno, Communication dans les réseaux optiques par multiplexage en longueur d'onde. Thèse de doctorat de l'université de Nice, 17 Janvier 2000, p23-26.

[2] CADRO Philippe, Evaluation des performances d'un commutateur de paquets optiques. Thèse de doctorat de l'université de Rennes 1, 16 Janvier 1998, p11-13.

[3] Elloumi Omar, TCP au dessus d'ATM: Architecture et performance. Thèse de doctorat de l'université de Rennes 1, 25 Mai 1999, p51-53.

[4] RFC 1185, TCP Extention for High-Speed Paths. V. Jacobson, R. Braden, L. Zhang. Network Working Group, Octobre 1990.

[5] RFC 1323, TCP Extentions for High Performance. V. Jacobson, R. Braden, D. Borman. Network Working Group, Mai 1992.

[6] RFC 2001, TCP Slow Start, Congestion avoidance, Fast Retransmit, and Fast Recovery Algorithms. W. Stevens, Network Working Group, Janvier 1997.

[7] RFC 793, Transmission Control Protocol, spécifications du protocole. Defense Advanced Research Projects Agency, Septembre 1981.

[8] RFC 2018, TCP Selective Acknowledgment Options. M. Mathis, J. Mahdavi, S. Floyd, A. Romanov. Network Working Group, Octobre 1996.

[9] RFC 2582, The New Reno Modification to TCP's Fast Recovery Algoritm. S. Floyd, T. Henderson. Network Working Group, Avril 1999.

[10] RFC 2581, TCP Congestion Control. M. Allman, V. Paxon, W. Stevens. Network Working Group, Avril 1999.

[11] Projet VTHD: Vraiment Très Haut Débit. France Telecom, INRIA, groupe des écoles de télécommunications. http://www.vthd.org

[12] Van Jacobson, Congestion Avoidance and Control. Actes de ACM SIGCOMM'88, Standford, 1988.

[13] Elamar Bartel, NTTCP. Université de Munchen, Octobre 1998. http://home.leo.org/~bartel/nttcp/nttcp.1.html

[14] Ethereal, http://www.ethereal.com