ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME ÉXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE AVEC CONCENTRATION EN RÉSEAUX DE TÉLÉCOMMUNICATIONS
M. Ing.

PAR
ABDERRAHIM NADIFI

ÉTUDE ET ANALYSE DES RPVs BASÉS SUR LA COUCHE LIAISON EN COMPARAISON AVEC LES RPVs BASÉS SUR LA COUCHE RÉSEAU

MONTRÉAL, LE 16 MAI 2007

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

Mme Maria Bennani, directrice du mémoire
Département de génie logiciel et des TI à l'École de technologie supérieure

M. Michel Lavoie, président du jury
Département de génie logiciel et des TI à l'École de technologie supérieure

M. Michel Kadoch, membre du jury
Département de génie électrique à l'École de technologie supérieure

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 1er MAI 2007

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ÉTUDE ET ANALYSE DES RPVs BASÉS SUR LA COUCHE LIAISON EN COMPARAISON AVEC LES RPVs BASÉS SUR LA COUCHE RÉSEAU

Abderrahim Nadifi

SOMMAIRE

La technologie *Multi Protocol Label Switching* (MPLS) offre aux opérateurs une infrastructure unique pour converger leurs services. MPLS améliore les services de *Internet Protocol* (IP) en offrant l'ingénierie de trafic, la qualité de services (QdS) et le support des réseaux privés virtuels (RPVs).

L'utilisation de MPLS pour implémenter des RPVs est une alternative viable qui garantit la sécurité et la QdS. Les RPVs réduisent considérablement les coûts d'acquisition des liens connectant les sites distants des clients en utilisant un réseau partagé comme le réseau Internet.

Layer 2 Virtual Private Network (L2VPN) est une nouvelle approche pour les RPVs à travers un réseau MPLS. Elle est basée uniquement sur la couche liaison (niveau 2 du modèle *Open System Interconnection* (OSI)). Comparé aux technologies *Asynchronous Transfer Mode* (ATM) et *Frame Relay* (FR), L2VPN est non seulement capable d'offrir les mêmes types de services, mais il est plus simple et moins dispendieux.

Ce mémoire étudie et analyse les services L2VPN. L'étude réalise une évaluation comparative des performances de L2VPN et de l'approche *Layer 3 VPN* (L3VPN) qui est basée sur la couche réseau du modèle OSI (niveau 3). L3VPN a été développé avant L2VPN pour établir des RPVs à travers un réseau MPLS. La fragmentation des paquets IP et le transport de trafic voix ont également été étudiés analytiquement.

On s'est intéressé dans la deuxième partie du mémoire à l'optimisation de la mise en place de la qualité de service (QdS) dans l'accès au réseau IP/MPLS pour les services L2VPN. Nous nous sommes intéressés au cas où l'équipement *Customer Edge* (CE) en périphérie du réseau du client est un simple commutateur. On propose l'algorithme *Dynamic Distributed Token Bucket* (DDTB) pour contrôler les flux des RPVs connectés au même CE.

STUDY AND ANALYSIS OF LAYER-2 VPN IN COMPARISON WITH LAYER-3 VPN

Abderrahim Nadifi

ABSTRACT

The technology Multi Protocol Label Switching (MPLS) offers a unique infrastructure to converge different types of services. MPLS improves the services of IP and at the same time provides the traffic engineering (TE), the quality of services (QoS) and the provisioning of virtual private networks (VPNs).

VPN reduces the costs of acquiring leased lines to connect the customer at remote sites by using a shared network like the Internet. The use of MPLS for the provisioning of VPNs is a viable alternative that guarantee the security and the QoS for customers.

This master's thesis studies the Layer 2 VPN services (L2VPN) which is a new approach for the establishment of VPNs through MPLS. It is based on the data link layer (layer 2). Compared to other technologies such as *Asynchronous Transfer Mode* (ATM) and *Frame Relay* (FR), L2VPN is much simple and cheap.

We compare and evaluate the performance of L2VPN and the Layer 3 VPN (L3VPN). The L3VPN services are based on the network layer (layer 3) and it has been widely used before L2VPN for the provisioning of VPNs through MPLS networks. The fragmentation of the IP packets and the transportation of voice have also been analytically studied.

The second contribution of this thesis is a solution that we propose for the optimization of the implementation of QoS in access to the IP/MPLS network when the customer uses a simple switch in periphery of its network (Customer Edge (CE)). We introduce to use the Dynamic Distributed Token Bucket (DDTB) algorithm to control the traffic flows of the L2VPNs that are connected to the same switch CE.

REMERCIEMENTS

Je tiens à remercier mon directeur de recherche Madame Maria Bennani, professeure au département de génie logiciel et des TI, de me donner l'opportunité de réaliser ce projet de recherche. Elle m'a accordé sa disponibilité et son aide précieuse à chaque fois que j'ai rencontré des difficultés dans mon travail. Le projet de recherche était pour moi une expérience enrichissante sur le plan technique et relationnel.

Je tiens à exprimer ma gratitude à M. Guy Côté, responsable du projet chez Bell Canada, de me donner l'accès au laboratoire de Bell Canada. Je remercie aussi M. Nicolas Manen qui travaille chez Bell Canada pour sa disponibilité et son aide à me familiariser avec les différents équipements de tests.

Je tiens aussi à remercier les deux étudiants en maîtrises M. Marc-André Breton et M. Olivier Truong pour leurs conseils judicieux. Je souhaite également faire part de mes remerciements à M. Mohamed El Hachimi pour sa contribution dans les développements et son aide dans la rédaction du mémoire et ses corrections.

Finalement j'aimerais aussi remercier tous ceux qui ont participé de près ou de loin au bon déroulement de mon projet, à savoir tous les membres du *Laboratoire de gestion de réseaux informatiques et de télécommunications* (LAGRIT) qui m'ont accueilli et aidé à rédiger mon mémoire.

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS ET SIGLES

AAL5	ATM Adaptation Layer 5
AC	Attachment Circuit
ACL	Acces Control List
AF	Assured Forwarding
ATM	Asynchronous Transfer Mode
AToM	Any Transport over MPLS
Bc	Committed Burst
Be	Excess Burst
BGP	Border Gateway Protocol
CBQ	Class-Based Queueing
CDP	Cisco Discovery Protocol
CE	Customer Edge
CIR	Committed Information Rate
CoDec	Coder Decoder
CoS	Class of Service
DiffServ	Differentiated Services
DSCP	Differentiated Services Code Point
EF	Expedited Forwarding
EIGRP	Enhanced Interior Gateway Routing Protocol
E-LSP	EXP-Inferred- LSP
EoMPLS	Ethernet over MPLS
FCS	Frame Check Sequence
FEC	Forwarding Equivalency Class
FIB	Forwarding Information Base
FQ	Fair Queueing
FR	Frame Relay

GRE	Generic Routing Encapsulation
HDLC	High Data Link Control
HVPLS	Hierarchical VPLS
IBGP	Interior Border Gateway Protocol
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IP	Internet Protocol
IPLS	IP-Only LAN Service
IPSec	IP security
ISP	Internet Service Provider
L2	Layer 2
L2TP	Layer 2 Tunneling Protocol
L2VPN	Layer 2 Virtual Private Network
L3	Layer 3
L3VPN	Layer 3 Virtual Private Network
LAN	Local Area Network
LAGRIT	LAboratoire de Gestion de Réseaux Informatiques et de Télécommunications
LDP	Label Distribution Protocol
LER	Label Edge Router
LLQ	Low Latency Queue
L-LSP	Label-Inferred- LSP
LSP	Label Switched Path
LSR	Label Switching Router
MAC	Media Access Control
MAN	Metropolitan Area Network
MPLS	Multi Protocol Label Switching
MTU	Maximum Transfert Unit

n-PE	network-PE
OSI	Open System Interconnection
OSPF	Open Shortest Path First
P	Provider
PDU	Protocol Data Unit
PE	Provider Edge
PHB	Per Hop Behavior
PIM	Protocol Independent Multicast
PPP	Point to Point Protocol
PQ	Priority Queueing
PSTN	Public Switched Telephone Network
PVC	Permanent Virtual Circuit
PW	Pseudo Wire
PWE3	Pseudo Wire Emulation Edge-to-Edge
QdS	Qualité de Service
RD	Route Distinguisher
RFC	Request For Comment
RIP	Routing Information Protocol
RPV	Réseau Privé Virtuel
RSVP	Resource-Reservation Protocol
RSVP-TE	Resource -Reservation Protocol-Traffic Extension
RTP	Real Time Protocol
SIP	Session Initiation Protocol
SLA	Service Level Agreement
SONET	Synchronous Optical NETwork
STP	Spanning Tree Protocol
TB	Token Bucket
TCP	Transport Control Protocol

TDM	Time Division Multiplexing
TE	Traffic Engineering
TLV	Type, Length, Value
ToS	Type of Service
UDP	User Datagram Protocol
UNI	User to Network Interface
u-PE	User-PE
VC	Virtual Circuit
VCID	Virtual Circuit Identifier
VLAN	Virtual LAN
VoATM	Voice over ATM
VoFR	Voice over Frame Relay
VoIP	Voice over IP
VoIPoL2VPN	Voice over IP over L2VPN
VoIPoL3VPN	Voice over IP over L3VPN
VoIPoMPLS	Voice over IP over MPLS
VoL3VPN	Voice over L3VPN
VoMPLS	Voice over MPLS
VP	Virtual Path
VPLS	Virtual Private LAN Service
VPN	Virtual Private Network
VPWS	Virtual Private Wire Service
VR	Virtual Routing
VRF	Virtual Routing and Forwarding
WAN	Wide Area Network
WFQ	Weighted Fair Queueing
WG	Working Group

INTRODUCTION

Les fournisseurs de service ont enregistré une croissance exponentielle du trafic dans leurs réseaux de télécommunication. Cette augmentation des flux d'information a provoqué une grande consommation de la bande passante. D'autre part, il existe différents réseaux par type de service (voix, données, vidéo, etc.). L'évolution de ces différents réseaux a augmenté la complexité et le coût de l'administration d'un tel environnement. Les opérateurs ont donc besoin d'une restructuration qui rend le fonctionnement de leurs réseaux efficace et concurrentiel.

La technologie IP a permis d'établir une architecture de réseau de communication qui supporte différentes classes de trafic. Cette architecture est unique pour tous les services offerts. Elle constitue une solution moins chère et évolutive. Cependant, il existe plusieurs problèmes dans les infrastructures IP, comme par exemple, les problèmes de la qualité de service qui sont dus à la perte de paquets, le délai et la gigue.

La nouvelle technologie MPLS a apporté le remède aux lacunes de IP. MPLS allie cheminement intelligent et commutation performante. Elle permet aussi de fournir des mécanismes pour le déploiement de nouveaux services (gestion de la qualité de service, ingénierie de trafic, etc).

La recherche effectuée dans ce mémoire s'est intéressée à l'utilisation de MPLS dans l'établissement des réseaux privés virtuels (RPVs). Les RPVs constituent un service très important pour les clients. Il permet de réduire les frais d'acquisition des liens privés. Les sites distants peuvent être connectés par l'utilisation d'un réseau virtuel à travers un réseau public.

La première partie du mémoire réalise une évaluation comparative des performances des deux approches fournies par MPLS pour la mise en place des RPVs, soit L2VPN et L3VPN. Les caractéristiques de chaque approche et les différences entre les deux sont

mises en évidence. L'étude évalue aussi l'efficacité de chaque approche dans l'utilisation de la bande passante. La comparaison a été effectuée pour différentes tailles de paquets. La fragmentation des paquets IP et le transport de la voix dans les deux types de RPV ont été aussi analysés.

Dans la deuxième partie du mémoire on s'est intéressé à l'amélioration de la mise en place de la QdS dans les L2VPNs. Dans les L2VPNs, l'équipement *Customer Edge* (CE) en périphérie du réseau du client peut être un simple commutateur. Suite à une série de tests sur une plateforme réelle, on a montré que lorsque plusieurs L2VPNs sont connectés à un CE commutateur, la QdS ne peut pas être garantie. En effet, les mécanismes de la qualité de service implémenté dans un commutateur sont très limités.

Ce mémoire présente une solution pour l'optimisation de la mise en place de la QdS lorsque le CE est un simple commutateur. La proposition est un algorithme qui a été nommé *Dynamic Distributed Token Bucket* (DDTB). Il permet de contrôler le flux des L2VPNs connectés au même CE.

Le reste de ce mémoire est organisé comme suit : Le chapitre 1 présente un état de l'art qui décrit MPLS, les services L3VPN et les services L2VPN. Cette revue de littérature couvre tous les mécanismes des deux approches et leurs fonctionnalités. Le chapitre 2 compare les deux approches selon plusieurs critères et présente une évaluation comparative approfondie des performances des deux approches. Le chapitre 3 décrit le support de la qualité de service dans les servies L2VPN. L'algorithme DDTB est proposé comme solution à la problématique reliée à la QdS. Les simulations ont été réalisées sur OPNET et les résultats obtenus sont analysés.

CHAPITRE 1

ÉTAT DE L'ART DE MPLS, L3VPN ET L2VPN

Ce chapitre est un état de l'art des réseaux privés virtuels basés sur *Multiprotocol label switching* (MPLS). Premièrement, une introduction est présentée sur la technologie MPLS. Deuxièmement, une présentation est fournie sur la technologie *Layer 3 Virtual Private Network* (L3VPN). Les deux architectures utilisées pour établir les services L3VPN sont exposées. Troisièmement, l'approche *Layer 2 Virtual Private Network* (L2VPN) est discutée ainsi que toutes les architectures possibles pour l'implémenter.

1.1 L'architecture de MPLS

La technologie MPLS offre une infrastructure unique pour converger les différents services. MPLS est plus adapté au trafic IP. Une spécification complète de l'architecture MPLS est fournie dans [1]. La figure 1.1 illustre un exemple de réseau MPLS.

L'objectif initial de MPLS était d'apporter la vitesse de la commutation de la couche liaison à la couche réseau. MPLS allie acheminement intelligent et commutation performante [2]. En effet, les datagrammes sont encapsulés dans MPLS en leurs ajoutant une étiquette (*label*) de taille fixe [3] et les routeurs acheminent les datagrammes en regardant uniquement dans leurs tables d'étiquettes.

Les objectifs de MPLS se sont multipliés. MPLS améliore les services de IP en offrant :

a. L'ingénierie de trafic pour optimiser l'utilisation des ressources des réseaux.
b. La qualité de services (QdS).
c. Les réseaux privés virtuels pour isoler le trafic d'un client au sein du réseau public de l'opérateur.

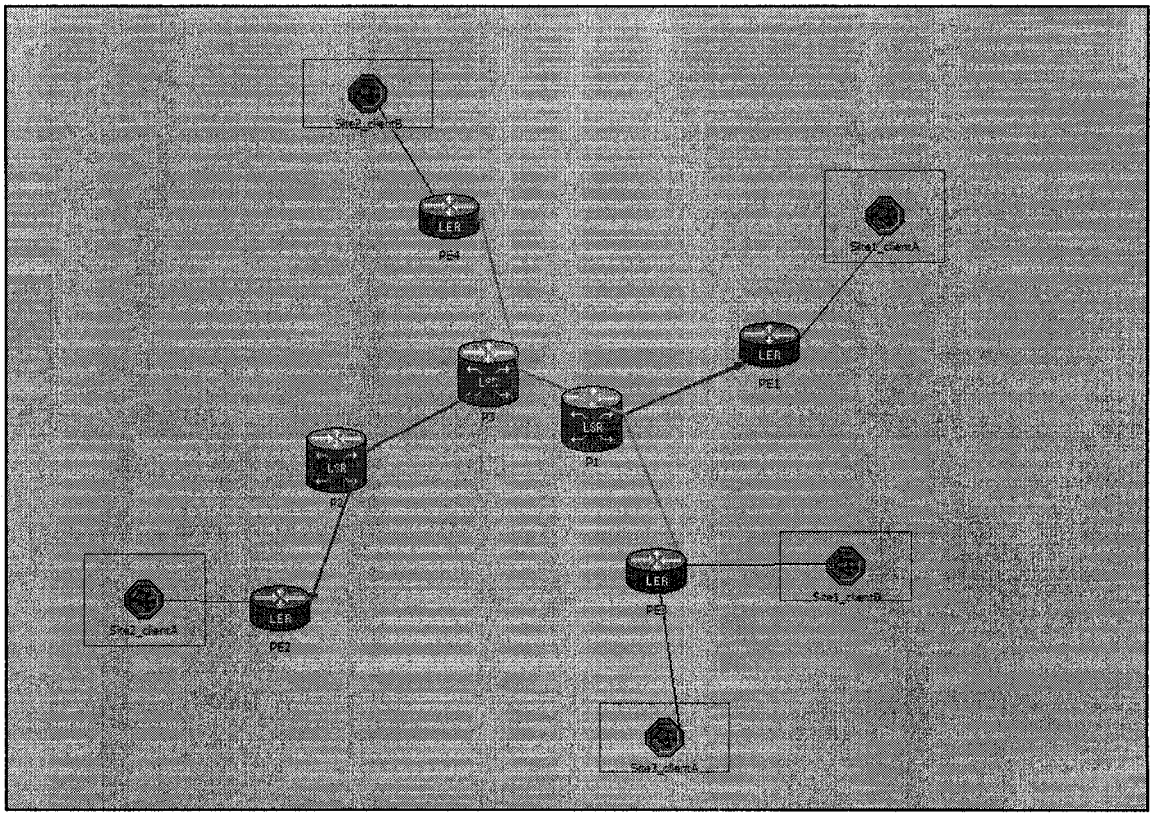


Figure 1.1 ***Exemple de réseau MPLS.***

Comme le montre la figure 1.1, le réseau IP/MPLS est constitué de routeurs de périphérie (*Provider Edge* (PE)) et de routeurs de la dorsale (*Provider* (P)). Un lien *Label Switched Path* (LSP) est définit comme étant le chemin établis entre deux routeurs PEs. Les LSPs sont établis de manière dynamique en utilisant le protocole *Label Distribution Protocol* (LDP), qui est un protocole de distribution des étiquettes MPLS, ou avec le protocole *Resource -Reservation Protocol-Traffic Extension* (RSVP-TE).

Le protocole LDP est utilisé pour établir un LSP entre deux routeurs PEs. Le protocole RSVP-TE peut également être utilisé pour attribuer une bande passante au LSP ou utiliser l'ingénierie de trafic pour sélectionner le meilleur chemin spécifié selon la qualité de service demandée ou les objectifs du TE.

1.2 Les réseaux privés virtuels dans MPLS

Le terme Réseau Privé Virtuel (RPV) réfère à un nombre de sites communiquant à travers un réseau qui est utilisé par plusieurs clients. Lorsque les ressources des différents sites connectés par le RPV sont accessibles uniquement par le même client, alors il s'agit d'un intranet. Mais lorsque les ressources des sites sont accessibles par différents clients, le RPV est un extranet [4].

Le déploiement des réseaux privés virtuels dans MPLS a été développé car leurs topologies prouvent une flexibilité intéressante pour les opérateurs. Pour implémenter un RPV dans MPLS, un PE, connectant le client, doit avoir un tunnel établi vers les autres routeurs PE connectant les autres sites des clients. Le tunnel peut être de type MPLS, mais il peut aussi être de type *Generic Routing Encapsulation* (GRE), IP Security (IPSec) ou *Layer 2 Tunneling Protocol* (L2TP), ce qui permet de ne pas imposer l'usage de MPLS dans le cœur du réseau.

La connaissance des RPVs se trouve uniquement en périphérie au niveau des routeurs PEs. Les routeurs Ps du cœur de réseau n'ont aucune information sur les RPVs ou les routes IP extérieures au réseau de l'opérateur. Un PE n'a connaissance d'un RPV que s'il connecte des sites à ce RPV. La commutation dans le cœur est assurée par les routeurs Ps traversés par des tunnels reliant les routeurs PEs. Ce modèle sert de base aux RPV de niveau 2 (L2VPN) et de niveau 3 (L3VPN).

Dans [1] les auteurs expliquent la conception et le déploiement de MPLS et les réseaux privés virtuels à travers cette infrastructure.

Au départ, L3VPN, appelé aussi *BGP MPLS VPN*, a été la seule solution pour le déploiement des RPVs dans une infrastructure MPLS, mais une variété de solutions a

été suggérée afin de palier les lacunes de L3VPN. La figure 1.2 illustre l'arbre des RPVs basés sur MPLS qui montre leurs proliférations au niveau 2 et au niveau 3.

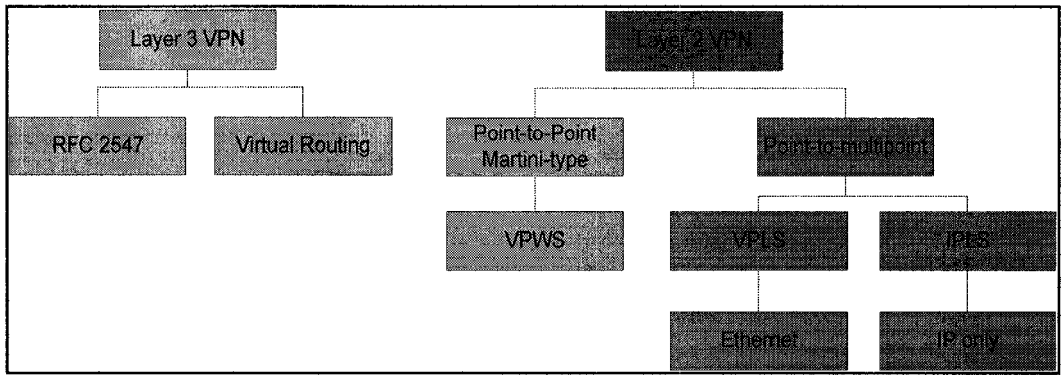


Figure 1.2 ***L'arbre des RPVs MPLS.***

Cet arbre montre différentes solutions RPV. Chaque solution a ses avantages et ses inconvénients. Il faut les identifier afin de pouvoir choisir la solution qui répond le mieux aux besoins des clients.

1.3 Layer 3 Virtual Private Network (L3VPN)

Dans un L3VPN, un routeur PE peut être connecté à plusieurs CEs. Le PE doit maintenir une table *VPN Forwarding Instance* (VFI) pour chaque RPV. Un VFI est une table qui contient les informations de routage pour chaque RPV. Lorsque le PE reçoit un paquet de la part d'un CE, il se base sur le lien transportant le paquet afin d'identifier le VFI associé à son RPV, ensuite il achemine le paquet en se basant sur les informations contenues dans le VFI [4].

1.3.1 RFC 2547

Dans ce modèle, l'entité logique VFI est un *Virtual Routing and Forwarding* (VRF). Un VRF est une table d'acheminement par site. Chaque routeur PE a besoin de maintenir un nombre de VRFs séparés. Seulement les routes qui sont échangés entre les sites ayant au moins un RPV en commun, sont stockées dans le VRF. Ceci est nécessaire pour empêcher la communication entre des sites qui ne sont pas connectés par un RPV.

Les services L3VPN nécessitent le routage pour la topologie spécifique du réseau virtuel privé du client, donc la conception des VRFs qui contiennent les routes des clients. Cela inclut aussi l'assignation des attributs de la communauté étendue (*extended communities*) du protocole *Border Gateway Protocol* (BGP). Ces attributs sont utilisés par les routeurs PE dans la distribution des routes.

Il faut aussi établir entre les routeurs PEs et les équipements CEs l'échange des routes des clients (*Peering*) qui constitue une composante essentielle dans les services L3VPNs. Le protocole BGP est utilisé pour effectuer la distribution des informations de routage qui concernent les RPVs des clients à travers le réseau MPLS de l'opérateur. Le processus d'échange des routes, entre les sites des clients dans le cœur du réseau du fournisseur d'accès, est décrit dans [5]. Un exemple d'échange des routes est illustré dans la dans la figure 1.3.

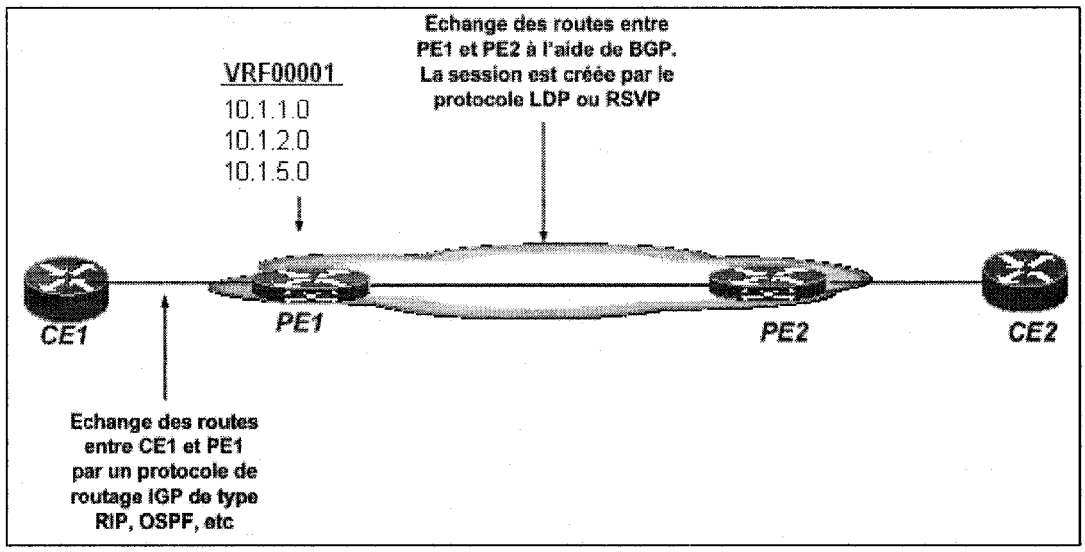


Figure 1.3 ***BGP MPLS VPN.***

Dans la figure 1.3, le CE1 et le CE2 connectent des sites qui ont un RPV en commun. Afin que ces sites puissent communiquer, un mécanisme d'échange des routes est nécessaire. VRF0001 est un exemple d'une table VRF qui stocke les routes d'un site connecté derrière CE1. Cette table de routage a été échangée entre CE1 et PE1 en utilisant un protocole de routage IGP de type RIP, OSPF, IBGP ou EIGRP. Ensuite, le PE1 doit déclarer ces routes au routeur PE2 à l'aide de BGP. La session BGP peut être créée par le protocole LDP ou RSVP. Finalement, les routeurs PEs envoient ces tables de routages aux routeurs CEs.

Dans les RPVs, l'utilisation des adresses IP privées était un problème pour la distribution des routes des clients. Une solution est spécifiée dans [5] pour permettre à un client d'utiliser dans ses RPVs les mêmes adresses IP privés qu'un autre client. Elle consiste à ajouter aux adresses IP un préfixe unique de 8 octets "*Route Distinguisher* (RD)". Donc si deux RPVs différents utilisent la même adresse IP, le PE la transforme en deux adresses IP différentes en utilisant le RD propre à chaque RPV.

1.3.2 Virtual Router (VR)

Ce modèle a été proposé par le fabriquant *Nortel* qui développe cette approche. L'architecture VR est une solution alternative à la proposition RFC 2547. Le PE maintient un routeur virtuel pour chaque RPV. L'architecture de cette solution est décrite dans [6]

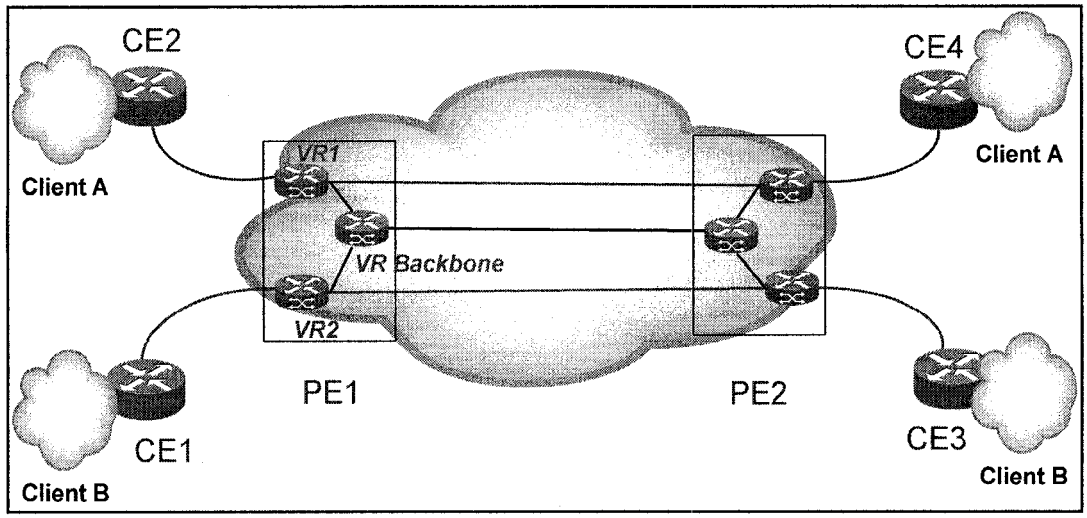


Figure 1.4 ***L'architecture VR pour L3VPN.***

La figure 1.4 illustre l'approche VR qui maintient une séparation des données de chaque site dans le cœur du réseau MPLS. Dans ce modèle, l'entité logique VFI est un *Virtual Router* (VR). Un VR agit comme un routeur virtuel résidant à l'intérieur du routeur PE. Chaque VR utilise une instance de protocole de routage pour transmettre les informations aux autres VR.

Il n'y a pas de limitation sur le type de tunnel connectant deux routeurs PE (IP, IPSec, GRE, MPLS). Le CE est raccordé à un VR à l'aide d'une technologie niveau 2, ou à l'aide d'un mécanisme de tunnel IPSec, L2TP ou GRE. Pour chaque client RPV, le PE

utilise un VR spécifique. Les différents VR dans le PE sont connectés à un *VR Backbone* (voir figure 1.4). Ce dernier est connecté au réseau de la dorsale IP/MPLS. Un LSP connectant deux PEs peut être originaire d'un VR ou du *VR Backbone*. Le LSP qui commence du *VR Backbone* est partagé entre plusieurs RPVs. Un LSP utilise une pile de deux étiquettes MPLS. La distribution des étiquettes peut se faire à l'aide de BGP, LDP ou RSVP.

L'inconvénient de cette approche est la complexité de sa configuration dans le réseau MPLS. La mise à l'échelle s'avère aussi une limitation à cette architecture.

Pour conclure, L3VPN ne répond pas à tous les besoins des clients. À titre d'exemple, si un client utilise des protocoles réseaux différents d'IP, L3VPN ne peut pas lui offrir la connectivité dont il a besoin. En effet, L3VPN supporte uniquement IP.

La prochaine partie discute la solution alternative L2VPN et les différents modèles proposés pour implémenter des L2VPNs.

1.4 Layer 2 Virtual Private Network (L2VPN)

Le but de cette section est de rendre intelligible la solution L2VPN. Les deux architectures principales de L2VPN sont présentées. Il s'agit de *Virtual Private LAN Services* (VPLS) et *Virtual Private Wire Service* (VPWS). Les deux propositions se distinguent par les caractéristiques du service offert aux clients.

Les deux architectures sont décrites dans [7]. Il n'existe encore aucun standard sous forme de RFC pour L2VPN. Les *drafts* qui ont été publiés par l'organisme *Internet Engineering Task Force* (IETF) sont accessibles via le site internet du *Working Group L2VPN* (http://www.ietf.org/html.charters/l2vpn-charter.html).

1.4.1 Virtual Private Wire Service

VPWS offre un service similaire à la technologie *Asynchronous Transfer Mode* (ATM) et *Frame Relay* (FR). Les sites sont reliés par des connexions point-à-point en utilisant des *Pseudo-Wires* (PW). Les sites connectés communiquent entre eux comme s'ils faisaient partie d'un seul *Local Area Network* (LAN). La figure 1.5 illustre un exemple de quatre sites reliés par deux circuits VPWS.

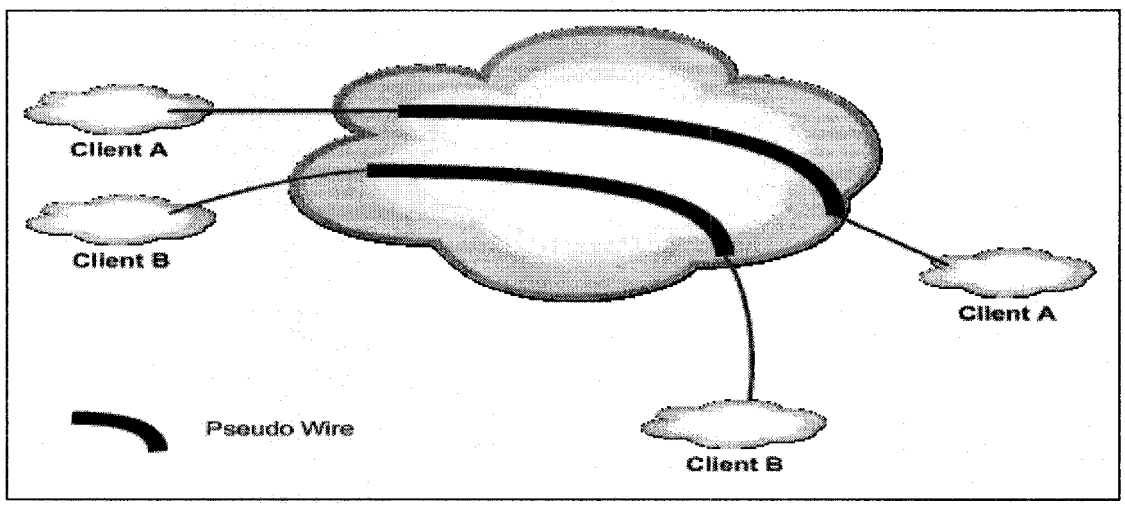


Figure 1.5 ***La topologie de VPWS.***

Le trafic du client est encapsulé en gardant l'entête de la couche 2. Ceci est possible grâce aux PWs. Si par exemple les réseaux locaux d'un client utilisent la technologie Ethernet, un PW Ethernet permettra de transporter les PDUs Ethernet/802.3 sur un réseau MPLS. La figure 1.6 illustre le modèle de référence pour VPWS.

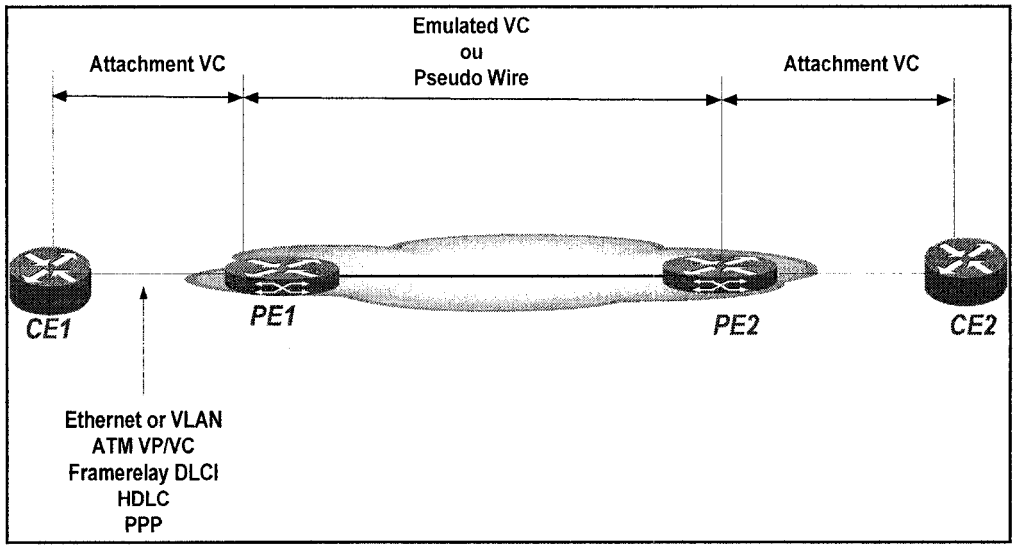


Figure 1.6 ***Modèle de référence pour VPWS.***

La figure 1.6 montre un circuit VPWS. On peut constater qu'il est composé des éléments suivants :

a. Le *Attachment circuits* (AC) est un lien physique ou virtuel qui attache un CE à un PE. Un AC peut être, parmi plusieurs technologies, un circuit virtuel (ATM, Frame Relay), un port Ethernet, un VLAN, un lien HDLC ou une connexion PPP.
b. Le *Emulated VC* est utilisé pour offrir une connexion niveau 2 entre deux routeurs PEs. Sa fonction principale est d'émuler des services de type ATM, Frame Relay, Ethernet et TDM sur un réseau MPLS. Il s'appelle aussi un *Pseudo Wire (PW)*.

VPWS est un service L2VPN qui associe un AC à un seul PW. Les trames reçues sur le AC seront transmises sur le PW correspondant. De même, les trames reçues sur le PW seront transmises sur le AC correspondant. Le contenu de l'entête de la trame ne joue aucun rôle dans le processus d'acheminement.

Les procédures génériques pour un PW Ethernet sont :

a. Le préambule et le *Frame Check Sequence* (FCS) de la trame Ethernet sont supprimés.
b. Le *Control Word* est ajouté à la trame résultante pour effectuer le séquencement des trames. Cette option n'est pas obligatoire.
c. Le "PW label" est ajouté à la trame résultante pour le démultiplexage des PWs.
d. Finalement, la trame résultante est encapsulée dans un tunnel et transmise.

Une méthode d'établissement et de maintien des PWs Ethernet est fournie dans [8]. Les trames des clients peuvent être encapsulées dans des PWs Ethernet ou dans des PWs Ethernet VLAN. Dans les PWs Ethernet, les trames peuvent ne pas avoir des étiquettes VLAN, mais dans les PWs Ethernet VLAN, les trames doivent absolument avoir une étiquette VLAN.

1.4.1.1 Pseudo-Wire Ethernet

Si la trame Ethernet n'a pas de VLAN, elle sera encapsulée avec un entête comme c'est décrit dans [8]. Si la trame est déjà encapsulée (VLAN), deux cas se présentent :

a. Si la trame arrive avec un VLAN tag identifiant un RPV (service-delimiting), l'encapsulation est utilisée localement par le PE pour identifier le trafic des clients. Cette encapsulation doit être supprimée avant que la trame ne soit encapsulée dans le PW. Au PE de l'autre extrémité, la trame pourrait être marquée par un VLAN tag s'il y a besoin de distinguer le trafic du client. De même, si une trame arrive à un port de PE sur ATM ou Frame Relay VC qui identifie une instance VPLS, l'encapsulation ATM ou FR doit être supprimée.
b. Si la trame arrive avec un VLAN qui n'est pas utilisé pour distinguer le trafic des clients, comme par exemple, pour identifier un domaine VLAN dans le réseau L2 du client. Ce genre de VLAN tag doit être conservé.

1.4.1.2 Pseudo-Wire Ethernet VLAN

Si le PW est configuré en mode Ethernet VLAN (*tagged mode*), chaque trame envoyée dans le PW doit être absolument marquée par un VLAN tag. Deux cas se présentent :

a. Si la trame, à son arrivée au PE, dispose d'un VLAN tag pour identifier le trafic du client au niveau de PE ou pour identifier un domaine VLAN du client, alors, l'encapsulation doit être conservée. La trame, comme telle, sera encapsulée et envoyée dans le PW.
b. Si la trame, à son arrivée au PE, ne dispose pas d'un VLAN tag, il lui sera imposé un tag nul avant d'être encapsulée dans le PW.

Le PW Ethernet VLAN offre un moyen simple pour préserver les bits 802.1p du client utilisés pour le marquage de QoS. Si le PE ne peut pas supporter les deux types de PW (Ethernet et Ethernet VLAN), il doit envoyer l'étiquette "*Label Release*" avec le code "*Unknown FEC*" comme c'est indiqué dans [9].

1.4.2 Virtual Private LAN Service (VPLS)

Virtual Private LAN Service (VPLS) est un service qui propose une connectivité Ethernet multipoint-à-multipoint à travers une infrastructure IP/MPLS. Le VPLS étend le réseau local (LAN) du client à de multiples sites en fournissant une connectivité multipoint de niveau 2 entièrement maillée.

Le VPLS s'appuie sur l'Ethernet pour fournir un service multipoint simple. Le réseau de l'opérateur IP/MPLS est vu comme un commutateur Ethernet auquel sont connectés les différents sites. Les postes clients des sites distants sont connectés comme s'ils appartiennent au même réseau local Ethernet.

L'utilisation des protocoles de routage IP/MPLS ainsi que l'utilisation des étiquettes MPLS, apportent à l'infrastructure de l'opérateur une souplesse et une capacité de déploiement sur une grande échelle. La figure 1.7 illustre le service Ethernet multipoint-à-multipoint offert par VPLS.

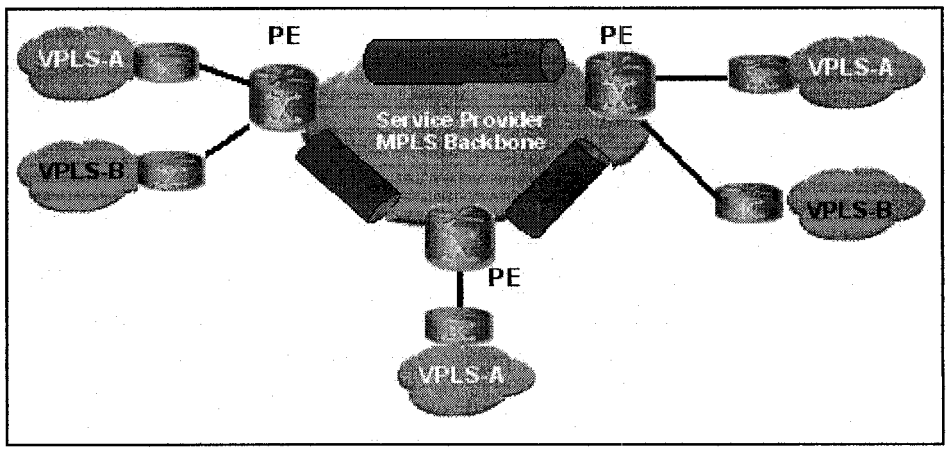


Figure 1.7 ***Le modèle de référence pour VPLS [10].***

Chaque routeur PE du réseau IP/MPLS doit intégrer les fonctionnalités VPLS. Un domaine VPLS est constitué de plusieurs routeurs PEs. Chaque PE participant à un domaine VPLS doit activer une instance VPLS participant à ce domaine. Une instance VPLS est appelée *Virtual Switch Instance* (VSI) parce qu'elle simule le fonctionnement d'un commutateur virtuel.

Les PEs doivent supporter certaines fonctionnalités. Ils ont besoin d'inonder les autres PEs, participants dans un domaine VPLS, des trames d'adresses MAC inconnues (*flooding*), enregistrer les adresses MAC (l'auto-apprentissage des adresses MAC), etc. Ainsi, pour éviter des problèmes de mise à l'échelle, VPLS doit être capable de gérer un très grand nombre d'adresses MAC.

Il existe différentes approches pour supporter un grand nombre d'adresses MAC :

a. Utiliser des routeurs pour les dispositifs CEs. Le routeur sera vu par le réseau de l'opérateur via son unique adresse MAC de niveau 2.
b. Si le CE est un commutateur Ethernet, toutes les adresses MAC (celles des serveurs et des périphériques incluses) des sites connectés sont visibles sur le PE qui connecte le CE. Les adresses MAC arrivant par un port d'entrée connectant le client devront être gérées, voire limitées, pour ne pas engorger le réseau.

Un équipement CE est relié à un routeur PE, situé dans le réseau de l'opérateur, via une interface Ethernet. Il doit apprendre les adresses MAC provenant des commutateurs et des routeurs connectés, et il doit établir ou terminer les tunnels de données dans l'infrastructure MPLS.

Le service est transparent pour le client. Il peut envoyer tout type de trafic sans avoir recours à l'opérateur. Une fois la connexion entre les PEs est établie, l'instance VPLS d'un PE est prête à recevoir des trames Ethernet d'un site client, et peut commuter ces trames sur le LSP approprié en fonction de l'adresse MAC destination. Cela est possible car VPLS permet au routeur PE d'agir comme un commutateur Ethernet avec une table d'adresses MAC par instance VPLS. En d'autres termes, l'instance VPLS sur le routeur PE dispose d'une table MAC alimentée par apprentissage des adresses MAC sources lorsque les trames Ethernet entrent par des ports physiques ou logiques spécifiques, exactement de la même façon qu'avec un commutateur Ethernet traditionnel.

Une fois que la trame Ethernet arrive par un port d'entrée connectant le client, l'adresse MAC destination est comparée dans la table MAC et la trame est transmise sans altération (si, bien sûr, l'adresse MAC correspondante se trouve dans la table MAC) à l'intérieur du LSP qui va la délivrer au PE adéquat connectant le site distant visé. Si l'adresse MAC n'est pas connue, la trame Ethernet est répliquée et transmise à tous les ports logiques associés à cette instance VPLS, excepté le port d'entrée par lequel la

trame est arrivée. Une fois que le PE reçoit en retour une trame de la machine qui détient cette adresse MAC, la table MAC est mise à jour dans le PE. Les adresses MAC n'ayant pas été utilisées après un certain temps sont automatiquement éliminées de la table, exactement comme sur un commutateur Ethernet.

IPLS est une approche qui fonctionne de la même façon que VPLS mais il supporte uniquement le trafic IP. Sa spécification est décrite dans [11]. Les dispositifs CEs sont des routeurs, mais les PEs acheminent les paquets IP en se basant sur les adresses MAC et non sur les adresses IP. Ainsi IPLS est aussi une technologie VPN niveau 2 (L2VPN).

1.4.3 VPLS hiérarchique (HVPLS)

La réplication des paquets et la quantité des adresses MAC sont deux préoccupations pour les dispositifs PEs. Dans un VPLS, si le nombre des PEs augmente, cela entraîne une augmentation du nombre des copies des paquets qui doivent être générés. En effet, les PEs ont besoin de répliquer des paquets pour effectuer une diffusion générale (*broadcast*) ou une multidiffusion (*multicast*). Dépendamment des capacités des routeurs utilisés, le traitement des paquets peut avoir un impact sur les ressources des routeurs.

Un modèle hiérarchique peut améliorer l'extensibilité de VPLS. *Hierarchical VPLS* simplifie la signalisation[1] et les exigences de la réplication des paquets. Deux types de routeurs PEs sont définis dans ce modèle : *user-facing PE* (u-PE) et *network PE* (n-PE). Le dispositif CE est connecté directement au u-PE qui fait passer le trafic VPLS global pour être acheminé en se basant sur le VSI correspondant.

---

[1] La signalisation établit les tunnels MPLS et distribue les étiquettes MPLS entre les PEs.

Les u-PEs doivent supporter les fonctionnalités *Layer 2 switching*. Cela crée une topologie *hub-and-spoke*[2] (voir [12] pour plus de détails sur cette topologie) entre le n-PE et les u-PEs qui lui sont connectés. La figure 1.8 montre un HVPLS utilisant une topologie *hub-and-spoke*.

Un u-PE peut être connecté à un PE ou deux PEs (*dual-homing*). Le maillage complet[3] des LSPs est alors établi seulement entre les n-PEs. Ainsi, on a une hiérarchie qui améliore l'extensibilité.

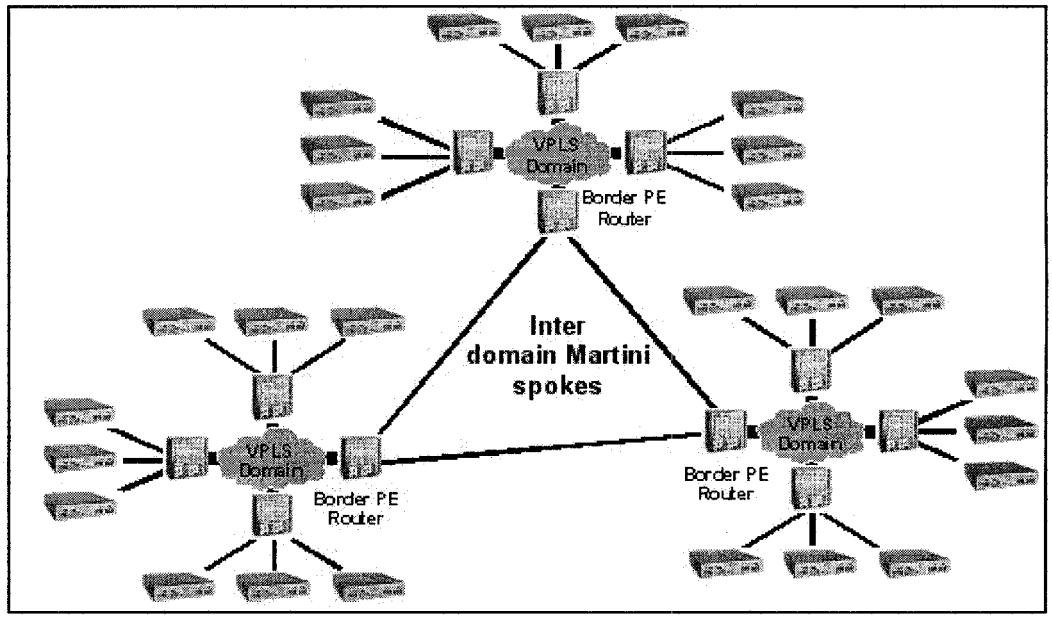


Figure 1.8 ***HVPLS utilisant Inter Domain Martini Spokes [13].***

Dans cette approche, HVPLS crée une topologie hub-and-spoke où plusieurs domaines VPLS sont connectés par des *pseudo-wires* selon le standard Martini [10] ou par un réseau L3VPN. Chaque routeur PE agit comme hub pour chaque domaine VPLS et les

---

[2] Cette topologie utilise un point de connexion central à partir duquel on peut atteindre chaque élément situé en périphérie du réseau.

[3] Un maillage complet (*full mesh*) est une topologie qui a lieu lorsque chaque nœud a un circuit le connectant vers un autre nœud du réseau.

tunnels Martini le relient aux nœuds périphériques (*spokes*). Cette topologie permet d'augmenter l'extensibilité du plan de contrôle et du plan de commutation en minimisant le nombre total de LSPs à gérer et par la distribution des réplications des paquets entre les nœuds. Avec HVPLS et l'utilisation de IGMP/PIM, il est possible d'optimiser l'arborescence de réplication (*replication tree*) pour qu'elle soit aussi efficace que l'arborescence du multicast de la couche 3.

Une autre approche consiste à construire des réseaux VPLS larges en utilisant un réseau fédérateur IP VPN basé sur RFC 2547 pour connecter les routeurs PEs, comme c'est illustré dans la figure 1.9.

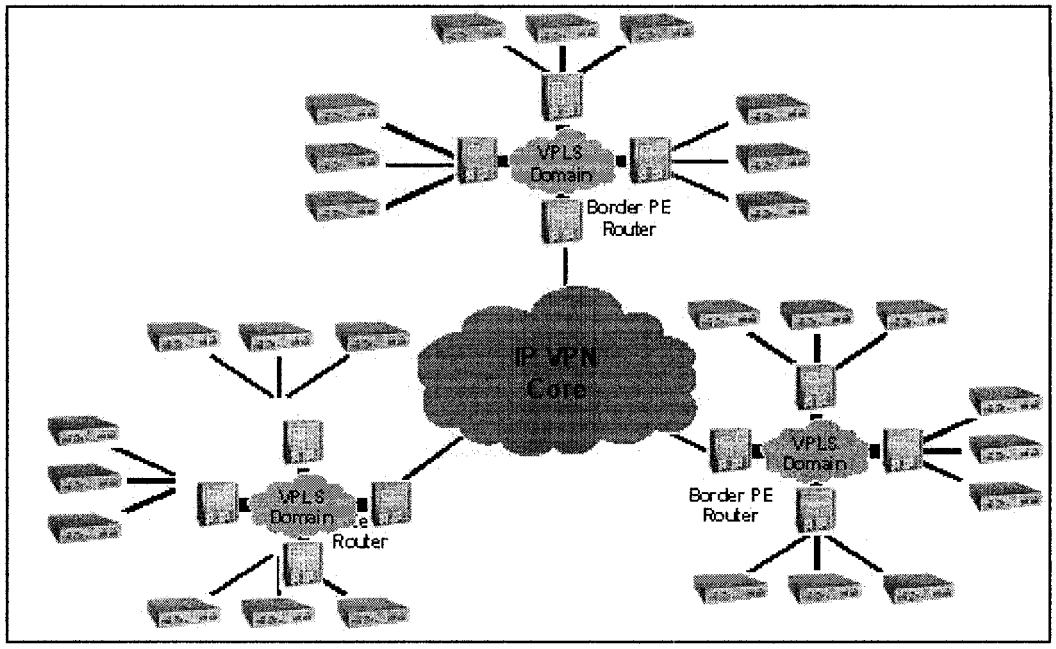


Figure 1.9 ***HVPLS utilisant une dorsale IP VPN [13].***

Cette approche utilise BGP pour relier les routeurs PEs. Donc les domaines VPLS sont interconnectés par un L3VPN à travers une infrastructure IP/MPLS.

1.4.4 Any Transport over MPLS (AToM)

Cisco a développé un produit, parmi les services L2VPN, appelé *Any Transport over MPLS* (AToM). Cette technologie vise à connecter des sites distants par un lien niveau 2 quelles que soient les technologies de la couche liaison utilisées sur chaque site (FR, ATM, Ethernet principalement). Ces connexions s'appuient sur le concept de PW. Les sites distants qui ont besoin d'être connectés peuvent ne pas utiliser la même technologie niveau 2. Il faut noter que le transport du trafic de la couche 2 est totalement transparent pour les clients. Ils n'ont aucune connaissance de l'existence du réseau MPLS.

L'établissement des L2VPNs en utilisant la technologie AToM est expliqué dans [14]. Il donne aussi des exemples de configurations de AToM avec des routeurs Cisco.

La technologie *Cisco AToM* permet, par exemple, de transporter le trafic Ethernet (unicast, broadcast et multicast) dans un réseau MPLS. Les trames Ethernet sont transportées entre les sites des clients à travers un réseau MPLS en utilisant un des trois modes :

a. *Port mode* - L'identification du trafic d'un client est basée sur le numéro de port physique du routeur PE auquel le client est connecté. La trame Ethernet est transportée sans le préambule ni le FCS. L'utilisation du *control word* est optionnelle. Dans ce mode, il est nécessaire que les routeurs PEs utilisent la même valeur du VC ID.
b. *Vlan mode* - Les interfaces connectant le CE au PE doivent supporter 802.1Q. Le trafic d'un client est identifié par son étiquette 802.1Q en associant chaque VC à un VLAN ID mais, dans le cœur du réseau, le trafic est encapsulé dans des PWs. Dans le mode VLAN, les PEs ne mémorisent pas les adresses MAC des trames Ethernet (MAC Address Learning). D'autre part, le protocole Spanning-Tree n'est pas utilisé.

c. *VLAN Rewrite mode* - La configuration est la même que le mode VLAN. La seule différence est le fait que pour le même *Emulated VC*, les PEs n'utilisent pas le même VLAN ID. Le routeur PE modifie le VLAN ID de la trame et le remplace par un autre VLAN ID. L'intérêt de ce mode est le fait que les sites du client qui sont reliés n'ont pas besoin d'utiliser les même VLAN IDs. Cela peut être utile pour une entreprise qui désire utiliser les services L2VPN en conservant la configuration des VLANs qu'elle utilise déjà. Donc elle n'aura pas besoin de modifier la configuration des VLANs de son réseau afin de relier ses sites par L2VPN.

Dans ce chapitre la technologie MPLS a été introduite. Ensuite un état de l'art des RPVs basés sur MPLS a été présenté.

1.5 Problématique

Les services L3VPN ont été développés pour répondre aux besoins des opérateurs qui veulent établir des réseaux privés virtuels à travers une infrastructure MPLS. La référence, pour le déploiement des services L3VPN dans les noyaux des réseaux des fournisseurs de services, est représentée dans [5]. Cette architecture est utilisée pour relier des sites distants à travers un réseau de type *Metropolitan Area Network* (MAN) ou *Wide Area Network* (WAN). Son extensibilité et son utilisation de BGP et MPLS favorisent le choix de cette solution pour étendre les réseaux locaux des clients à travers le cœur du réseau IP/MPLS de l'opérateur.

Au début de son apparition, L3VPN était supposé une solution satisfaisante pour les demandes du marché, mais, avec le temps, plusieurs problèmes ont vu le jour. En effet, *BGP MPLS VPN* est très complexe à dimensionner et à maintenir. En plus, cette solution supporte uniquement le trafic IP et elle est limitée dans ses tables de routage BGP [15] [16].

Pour résoudre ces problèmes, l'organisme *Internet Engineering Task Force* (IETF) a développé les services L2VPN. Il est difficile de trouver dans la littérature de l'information les concernant. L'architecture des services L2VPN n'a pas encore atteint une certaine maturité. Les *drafts* publiés par le *Working Group L2VPN,* mis en place par l'IETF, ne constituent pas encore une norme finale de production. Ils n'ont pas encore progressé pour devenir des *Request For Comment* (RFC). Pour cette raison, les opérateurs se contentent pour le moment d'offrir aux clients uniquement les services L3VPN et ne s'aventurent pas encore dans le déploiement des services L2VPN.

L2VPN est une solution de commutation pour le transport des trames niveau 2 à travers un réseau IP/MPLS. Le transport du trafic de la couche 2 est totalement transparent pour les clients. Ces derniers n'ont aucune connaissance de l'existence du réseau MPLS. Le but est d'avoir des réseaux privés virtuels totalement indépendants de la couche réseau.

Les services L2VPN peuvent se résumer à :

a. Grouper plusieurs réseaux niveau 2 d'une entreprise ou d'un opérateur en un seul réseau IP/MPLS offrant les services de la couche 2 totalement transparents aux clients.
b. Étendre les réseaux locaux (LAN) à travers le cœur du réseau de l'opérateur.
c. Offrir des services Ethernet multipoint-à-multipoint et point-à-point.

Les deux approches répondent de manières différentes aux besoins de la connectivité des clients. La solution L2VPN a de nombreuses caractéristiques qui font d'elle un concurrent à la solution L3VPN. Afin d'aider à identifier la solution qui répond le mieux aux besoins des clients souhaitant connecter leurs réseaux locaux distants à travers une infrastructure MPLS, nous avons effectué une comparaison des deux approches en se basant sur plusieurs critères.

Dans le but de mesurer la différence entre les deux technologies en terme de performance, une étude théorique une étude analytique a été réalisée. L'étude analyse l'impact de l'encapsulation des trames dans un réseau IP/MPLS sur les performances de L2VPN et L3VPN. Nous avons choisi d'examiner le débit des flux de trafic transportés et l'efficacité dans l'utilisation de la bande passante dans un réseau IP/MPLS.

D'autre part, un des avantages des services L2VPN est la possibilité d'utiliser des commutateurs comme CEs pour connecter les réseaux des clients au réseau MPLS de l'opérateur. Nous avons pu voir a travers notre étude que pour utiliser un commutateur comme équipement CE, la mise en place de la qualité de service n'est pas garantie si plusieurs client sont connectés au même CE.

On s'est intéressé dans la deuxième partie du mémoire à la mise en place de la QdS dans les RPVs basés sur MPLS. Le but ultime des opérateurs est de garantir aux clients un service qui respecte les paramètres du contrat *Service Level Agreement* (SLA) entre le client et l'opérateur, comme par exemple : le délai, la gigue, la réservation de la bande passante, la perte des paquets, etc. La QdS est nécessaire pour gérer la congestion et utiliser les liens de manière efficace.

D'autre part, les flux de trafic dans un L2VPN passe uniquement par le niveau 2. En effet, les L2VPNs sont basés sur la couche liaison et aucun routage n'est effectué entre les sites des clients. Pour se connecter au réseau de l'opérateur, les clients peuvent utiliser simplement des commutateurs.

Cependant, nous avons pu voir a travers notre étude que pour utiliser un commutateur comme équipement CE, la mise en place de la QdS n'est pas garantie si plusieurs client sont connectés au même CE. En effet, un commutateur est très limité dans l'implémentation de la qualité de service. Un commutateur dispose uniquement dans ses

interfaces de sortie de 4 à 8 files d'attentes. Donc si plusieurs RPVs partagent la même file d'attente, il y aura concurrence dans l'utilisation de la bande passante.

Dans L3VPN, ce problème ne se présente pas car les équipements CE sont des routeurs. En effet, un routeur permet d'utiliser des files d'attentes virtuelles et on peut associer une file d'attente logique par RPV.

Le présent mémoire propose comme solution un algorithme que nous avons appelé "*Dynamic Distributed Token Bucket*". L'algorithme a été développé afin de contrôler le flux de chaque L2VPN connecté au CE. La solution garantit pour chaque RPV le débit définit dans son contrat. Ce débit peut varier s'il y a de la bande passante non utilisée.

CHAPITRE 2

ÉTUDE COMPARATIVE DES SERVICES L2VPN ET L3VPN

Dans ce chapitre on présente une étude analytique des performances des services L2VPN et des services L3VPN. Les deux technologies sont implémentées dans un réseau MPLS de manières similaires. La différence réside dans le type d'encapsulation du trafic du client afin de l'acheminer à sa destination à travers MPLS. Après une observation poussée du fonctionnement des services L2VPN et des services L3VPN, les différentes étapes d'encapsulation des trames ont été analysées.

Le but de cette étude est d'évaluer l'impact de l'encapsulation sur les performances des services L2VPN et les services L3VPN. Pour effectuer cette étude, nous avons choisi d'examiner le débit des flux de trafic transportés et l'efficacité dans l'utilisation de la bande passante pour les deux technologies. Pour ces paramètres, la différence entre les valeurs a été quantifiée et analysée.

L'étude a montré une différence de débit lorsque les paquets sont de petites tailles. En effet, dans les L2VPNs, le trafic causé par les entêtes ajoutés devient immense si les paquets sont de petites tailles. Par contre, dans les L3VPNs, les entêtes ajoutés utilisent moins de bande passante.

En raison de l'utilisation de petits paquets dans les communications de voix, une étude du transport de la voix dans les deux types de RPV a été réalisée. L'étude comparative utilise différents *Coder Decoder* (CoDec) de voix. Le multiplexage des connexions voix a été aussi examiné afin d'évaluer son impact sur les performances.

2.1 Choisir L2VPN ou L3VPN

Cette partie présente les principales différences techniques entre le déploiement des services L2VPN et les services L3VPN. La comparaison entre les deux approches se base sur les critères suivants :

a. Dimensionnement
b. Déploiement
c. Gestion et maintenance
d. Type de trafic
e. Mise à l'échelle

2.1.1 Dimensionnement

Le dimensionnement des services L3VPN nécessite la conception du routage pour la topologie spécifique du réseau virtuel privé du client. Il faut aussi établir entre les routeurs PEs et les équipements CEs l'échange des routes des clients (*Peering*) qui constitue une composante essentielle dans les services L3VPN

Le dimensionnement des services L2VPN est beaucoup plus simple. Chaque routeur PE a seulement besoin de connaître les autres routeurs PEs afin d'établir avec eux les VCs pour construire le réseau privé virtuel désiré.

2.1.2 Déploiement

Le déploiement des services L3VPN nécessite des routeurs PEs sophistiqués capables de gérer plusieurs tables de routage. Les fournisseurs d'accès à Internet et les opérateurs des larges réseaux IP utilisent déjà le protocole BGP dans leurs infrastructures. Ils opteront pour le déploiement des services L3VPN pour bénéficier des sessions BGP déjà établies. Ensuite, il faudra mettre en place des tunnels pour connecter les routeurs PEs entre eux.

Les services L2VPN utilisent des équipements PEs simples. Les opérateurs qui n'ont pas implémenté BGP ou qui n'ont pas l'intention de déployer BGP pour offrir les services des réseaux virtuels privés à leurs clients opteront pour la solution L2VPN.

2.1.3 Gestion et maintenance

Pour effectuer la gestion des L3VPNs, les ingénieurs auront besoin de manipuler plusieurs sessions BGP et les routes BGP des clients avec leurs différents attributs. Le stockage de ces informations requiert une intense activité opérationnelle qui est sans doute soumise à d'éventuelles erreurs humaines. Par conséquence, la détection d'une mauvaise configuration sera une tâche difficile. D'autre part, comme dans la plupart des réseaux IP à grande échelle, l'existence de plusieurs systèmes autonomes (SA) augmente la complexité de la gestion du réseau.

La gestion des L2VPNs est beaucoup plus simple que celle des L3VPNs. L'opérateur ne gère pas les routes des clients. Ces derniers doivent gérer la distribution de leurs routes et échanger le trafic avec les autres équipements CEs. Étant donné que l'utilisation du protocole BGP n'est pas nécessaire, la gestion et le dépannage (*troubleshooting*) ne seront pas des opérations compliquées. Même si les opérateurs utilisent le protocole BGP pour effectuer la signalisation, la gestion des L2VPNs requiert simplement la manipulation des VCs. De plus, pour chaque routeur PE, les ingénieurs s'occupent seulement d'une seule table de routage.

2.1.4 Type de trafic transporté

Les services L3VPN offre le transport du trafic IP uniquement, tandis que les services L2VPN offre le transport de tout trafic niveau 3 : IPv4, IPv6, IPX, DECNet, etc. En utilisant les services L2VPN, les compagnies, qui utilisent des protocoles différents de IP, auront moins de restrictions.

D'autre part, plusieurs organisations ont déjà commencé l'expérimentation d'IPv6 dans le but d'y migrer. Pour continuer à offrir la connectivité à ces organisations, l'utilisation des services L3VPN nécessitera d'apporter des modifications au standard actuel. Par contre les services L2VPN continueront à leurs offrir la connexion même si le réseau de l'opérateur n'as pas été mis à jour pour supporter IPv6.

2.1.5 Mise à l'échelle

Les limitations des deux technologies sont à peu près similaires. Les routeurs LSR supportent un nombre limité de LSPs et de VCs. Un fichier de configuration d'un équipement LSR doit contenir toutes les informations reliées à la maintenance des réseaux virtuels privés. Cependant ce fichier a une taille maximale pour stocker toutes ces informations. Dans le cas d'un L3VPN, le fichier de configuration contient la définition des VRFs, des attributs de la communauté étendue de BGP et les politiques de filtrages des routes échangées. Pour un L2VPN, le fichier de configuration contient les adresses des autres équipements PEs et les numéros des ports associés aux réseaux virtuels privés.

Un autre facteur limitant pour les services L3VPN est le nombre maximum des routes qui peuvent être enregistrés dans un routeur PE. Cette contrainte est due au fait que les équipements PEs enregistrent les routes qui proviennent de tous les RPVs qui leurs sont connectés.

La contrainte imposée par la taille des fichiers de configuration dans les L2VPNs peut être réduite en implémentant le mécanisme de Découverte Automatique[4] [17] qui permet la création automatique d'un maillage complet de LSP entre les PEs. Ainsi la mise à

---

[4] La découverte automatique (Auto-Discovery) est un mécanisme qui peut être utilisé par chacun des routeurs PEs pour participer à un domaine RPV et découvrir tous les autres PEs partenaires.

l'échelle des services L2VPN est moins affectée par la contrainte de la taille maximale des fichiers de configuration des routeurs PEs.

Pour conclure, on a vu qu'il existe deux solutions pour l'implémentation des réseaux virtuels privés à travers une infrastructure IP/MPLS, L2VPN ou L3VPN. L'approche L3VPN est basée sur le standard *BGP MPLS VPN*. Cette solution transporte uniquement le trafic IP et elle supporte plusieurs RPVs en utilisant le mécanisme de distribution des routes proposé avec BGP. L'approche L2VPN utilise les normes de facto définis dans les drafts de *martini* [7]. Cette solution est plus récente que L3VPN. Elle permet de transporter des trames niveau 2 telle quelle à travers un réseau MPLS. L'intérêt de cette approche est son indépendance de la couche 3.

La comparaison des deux solutions montre qu'il n'est pas possible de déterminer si une approche est meilleure que l'autre. Chaque approche offre une solution qui répond de manière différente aux besoins de connectivité des clients. Afin de répondre le mieux aux besoins des clients souhaitant connecter leurs réseaux locaux distants, il est indispensable de bien analyser les forces et les faiblesses de chaque approche. Le reste de ce chapitre réalise une évaluation comparative des deux approches.

Dans l'étude qui suit, on suppose qu'entre le client et les routeurs PEs, le trafic est identifié par son VLAN ID, mais, dans le cœur du réseau, le trafic est encapsulé dans des *pseudo-wires*. Les interfaces connectant les CEs aux PEs doivent supporter 802.1Q. Donc les liens CE-PE sont configurés sur des sous-interfaces et chaque VC est associé à un VLAN ID.

2.2 Étude analytique des performances de L2VPN

L'étude se concentre sur le transport de la technologie Ethernet. Actuellement la technologie Ethernet est la technologie LAN la plus déployée dans les réseaux locaux des clients. Le standard Ethernet a évolué de manière très significative. Le débit géré par cette technologie a connu une croissance exponentielle en passant de 10Mbps à 10 Gbps [18].

Les services L2VPN offrent des connexions Ethernet longues distances. La figure 2.1 illustre l'encapsulation des trames Ethernet dans un L2VPN.

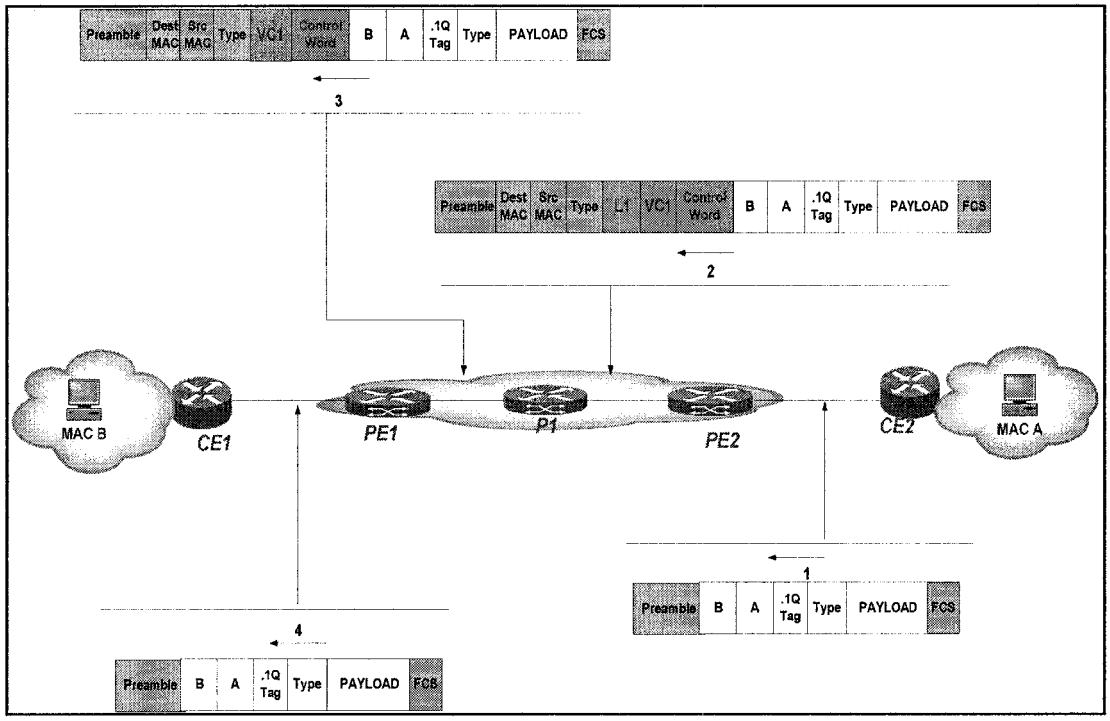


Figure 2.1 ***Encapsulation des trames Ethernet dans L2VPN.***

La figure 2.1 illustre les étapes d'encapsulation de la trame Ethernet dans L2VPN. Les équipements CE1 et CE2 connectent les deux réseaux locaux aux routeurs périphériques

PE1 et PE2 respectivement. Dans les sites du client, le trafic est constitué de paquets IP encapsulés dans 802.1Q (VLAN).

Le champ VC1 représente la première étiquette dans la pile des étiquettes MPLS (*Demultiplexer field*) qui est une étiquette ajoutée pour le démultiplexage des L2VPNs. Le champ L1 est la deuxième étiquette dans la pile (*Tunnel header*) qui sert à acheminer la trame dans le réseau MPLS [3]. La trame Ethernet est transportée sans le préambule ni le FCS. L'utilisation du champ *control word* est optionnelle.

Dans la figure 2.1, on voit les différentes étapes dans l'acheminement des trames Ethernet de la source vers la destination :

a. Le CE2 est relié au PE2 par un *Attachment Circuit* (AC) de type Ethernet. Ce lien sert à transiter le trafic des clients vers le réseau MPLS. Le trafic de chaque client est identifié par le tag VLAN dans l'entête Ethernet.
b. Le PE2 reçoit la trame, supprime le préambule et le champ FCS. Il l'encapsule dans un *pseudo wire* en lui ajoutant deux étiquettes MPLS (VC1 et L1), un entête Ethernet, un préambule et un FCS.
c. Le P2 effectue l'opération "*swapping*" de la trame. Cette opération consiste à regarder dans la table des étiquettes MPLS afin d'envoyer la trame sur le bon port de sortie de P2 avec la bonne étiquette. Étant donné que P2 est le dernier nœud dans le réseau MPLS avant d'atteindre PE1, alors il supprime l'étiquette L1 et achemine la trame vers PE1. Cette opération s'appelle "*Penultimate Hop Popping*".
d. Finalement le PE1 reçoit la trame, supprime le préambule, le champ FCS, les étiquettes MPLS, et l'entête Ethernet. Ensuite il reconstitue le préambule et le FCS et il l'envoie sur le bon AC qui le relie avec CE1.

Le trafic circulant entre le CE et le PE est identifié par son VLAN ID. Dans le cœur du réseau, le trafic est encapsulé dans des *pseudo-wires*. Les interfaces connectant

l'équipement CE à l'équipement PE doivent supporter le standard 802.1Q afin de pouvoir utiliser plusieurs liens VC pour connecter le PE au CE. Ainsi, sur la même interface globale on peut configurer plusieurs liens CE-PE sur des sous-interfaces. Chaque lien VC correspond à un VLAN ID. Le tableau suivant présente la taille en octets des différents champs ajoutés pour l'encapsulation de la trame Ethernet (voir figure 2.1).

Tableau 2.1

Calcul de la taille totale des entêtes dans L2VPN

Champ ajouté	**Taille (octets)**
VLAN	4
Deux étiquettes MPLS	8
Control Word	4
Deux entêtes Ethernet	28
Préambule	8
FCS	4
Entête final	56

Le trafic Ethernet est encapsulé en lui ajoutant 56 octets d'entêtes. La taille de l'entête finale n'est pas négligeable. On verra par la suite comment cet ajout peut affecter les performances des services L2VPN.

On s'intéresse à mesurer le débit physique dans L2VPN pour les différentes tailles des paquets IP. Pour le calculer, il suffit de multiplier la taille de la trame par le nombre des trames commutées par seconde. Notons Y le débit physique, X la taille initiale en octets du paquet IP, N le nombre des paquets commutés par seconde et O la taille finale des entêtes ajoutés au paquet IP en octets.

Pour obtenir le débit physique dans L2VPN, il suffit de multiplier le nombre des paquets commutés par seconde par la taille totale de la trame. La formule (2.1) permet de calculer le débit théorique en fonction de la taille et le nombre des paquets commutés par seconde. Chaque trame est composée d'octets.

$$Y\,(\text{bps}) = N \cdot (X + O) \cdot 8 \tag{2.1}$$

D'autre part, entre les trames Ethernet, il doit y avoir un intervalle de temps (*inter frame gap)* d'au moins 96 bits time. C'est le temps nécessaire pour transmettre 96 bits. Il est de 9.6 µs pour 10Mbps, 960 ns pour 100 Mbps et 96 ns pour 1Gbps.

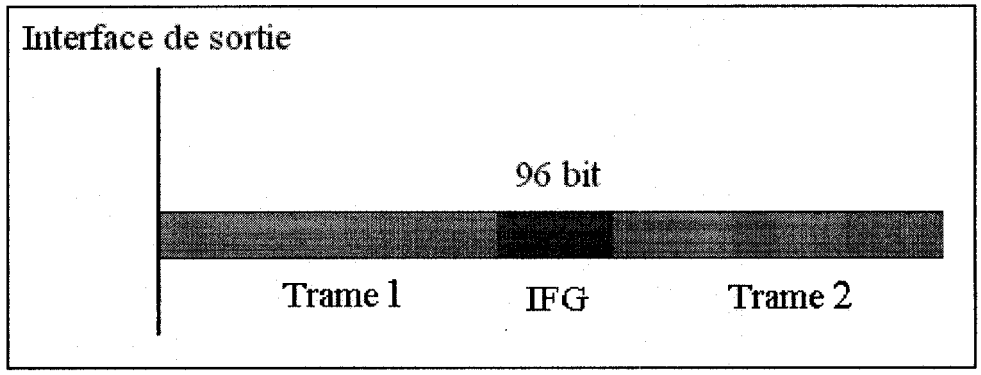


Figure 2.2 ***Inter Frame Gap (IFG).***

Notons V la vitesse de propagation du lien (en Mbps). Le nombre de paquets commutés par seconde est donné par la formule (2.2)

$$N = \frac{V}{X + O + IFG} \tag{2.2}$$

Supposant que les liens physiques dans le réseau sont configurés à un débit de 100Mbps. Le tableau suivant illustre le nombre de paquets commutés par seconde N1 et le débit Y1 lorsque l'on varie la taille des paquets IP.

Tableau 2.2

Débit physique des trames Ethernet dans L2VPN

X (octets)	N1	Y1 (Mbps)
64	94696	90,1
128	63775	93,9
256	38580	96,3
512	21551	98
1024	11446	99
1400	8514	99,2

Les résultats affichés dans le tableau ci-dessus montrent que plus la taille d'une trame est grande, plus son débit est élevé. Par contre, plus la taille des paquets IP est grande plus le nombre des paquets commutés par seconde est petit. Ces résultats sont conformes aux équations (3.1) et (3.2) qui montrent que le débit Y1 est proportionnel à la taille X des paquets IP et N1 est inversement proportionnel à X.

Par la suite, les débits théoriques offerts par L3VPN seront aussi calculés afin de les comparer avec les résultats obtenus pour L2VPN.

2.3 Étude analytique des performances de L3VPN

Les services L3VPN offrent des connexions longues distances à travers un réseau IP/MPLS. La figure 2.3 illustre l'encapsulation des paquets IP lorsque les deux sites du client sont connectés via un L3VPN.

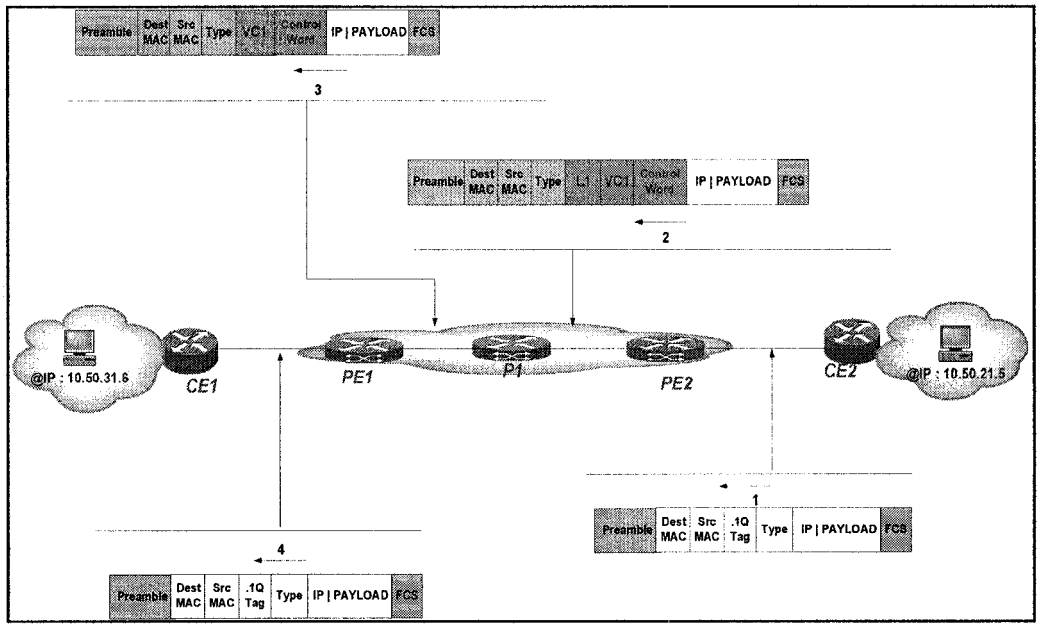


Figure 2.3 ***Encapsulation des paquets IP dans L3VPN.***

Comme il a été déjà mentionné L2VPN et L3VPN utilise le même principe. Les deux approches encapsulent le trafic dans MPLS avec une pile de deux étiquettes MPLS. (VC1 et L1). Le champ VC1 représente la première étiquette dans la pile des étiquettes MPLS qui est ajoutée pour le démultiplexage des L2VPNs. Le champ L1 est la deuxième étiquette dans la pile qui sert à acheminer la trame dans le réseau MPLS.

Lorsque CE2 reçoit une trame, il lui rajoute une étiquette VLAN pour identifier le VRF associé au client afin qu'elle passe par le bon VC reliant CE2 au PE2. Ensuite le PE2 encapsule la trame dans MPLS en lui ajoutant les deux étiquettes L1 et VC1 qui correspondent au L3VPN du client. Il peut aussi utiliser le champ *Control Word* mais il reste optionnel.

Dans un L2VPN on parle de commutation des trames Ethernet et dans un L3VPN on parle plutôt de routage des paquets IP. Contrairement à L2VPN, les trames du client qui sont encapsulées dans MPLS ne gardent pas les entêtes de la couche liaison. Elles sont

supprimées par le PE2 avant d'encapsuler les paquets IP dans MPLS. À la sortie du réseau MPLS, le PE3 restitue l'entête de la couche liaison avant d'envoyer les trames vers CE3. Le tableau suivant présente la taille en octets des différents champs ajoutés dans un L3VPN.

Tableau 2.3

Calcul de l'entête total

Champ ajouté	**Taille (octets)**
Deux étiquettes MPLS	8
Entête Ethernet	14
Préambule	8
FCS	4
Entête total	34

La taille totale des entêtes qui ont été additionnés au trafic est de 34 octets. De la même façon que pour L2VPN, le nombre de paquets commutés par seconde et le débit théorique, lorsqu'on varie la taille des trames Ethernet, ont été calculés. Comme dans le cas de L2VPN, les liens physiques dans le réseau sont supposés être configurés à un débit de 100Mbps. Les résultats sont présentés dans le tableau ci-dessous.

Tableau 2.4

Débit physique des trames Ethernet en utilisant L3VPN

X (octet)	N2	Y2 (Mbps)
64	113636	89,1
128	68681	93,1
256	40322	96,02
512	22084	97,84
1024	11595	98,87
1400	8596	99,15

Si on compare les résultats obtenus dans le cas de L2VPN et L3VPN, on constate qu'il n'y a pas une grosse différence sauf pour les petits paquets. Le tableau suivant illustre cette différence pour les différentes tailles des trames Ethernet qui ont été générées.

Tableau 2.5

Différence entre les débits de L2VPN et L3VPN

X (octet)	L2VPN Y1 (Mbps)	L3VPN Y2 (Mbps)	Y1-Y2 (Mbps)
64	90,1	89,1	1
128	93,9	93,1	0,8
256	96,3	96,02	0,28
512	98	97,84	0,16
1024	99	98,87	0,87
1400	99,2	99,15	0,05

2.4 Synthèse des résultats analytiques

Le débit est plus élevé dans les L2VPNs. Ceci est dû à la différence dans la taille totale des entêtes ajoutés aux paquets pour les encapsuler dans le réseau MPLS. La taille finale des trames Ethernet dans un L2VPN est plus élevée que dans un L3VPN.

D'après ces résultats, l'architecture L2VPN s'avère plus performante que l'architecture L3VPN. Mais l'analyse, qui suit, montre que ces résultats sont trompeurs. En effet, les résultats obtenus concernent les débits physiques. Un client s'intéresse plus au débit applicatif de son trafic. Une comparaison correcte doit prendre en compte le pourcentage des données dans le trafic transporté par les deux types de réseaux privés virtuels. Notons P le pourcentage des données, alors nous obtenons l'équation (2.3).

$$P = \frac{X}{X + O} \tag{2.3}$$

Notons P1 et P2 les pourcentages des données dans L2VPN et L3VPN respectivement. Notons aussi O1 et O2 les *Overheads* ajoutés par L2VPN et L3VPN respectivement. Le tableau suivant présente les résultats du calcul de P1 et P2 pour différentes tailles initiales de paquets.

Tableau 2.6

Pourcentage des données dans une trame Ethernet

	L2VPN		L3VPN	
X (octets)	**X + O1 (octets)**	**P1 (%)**	**X+O2 (octets)**	**P2 (%)**
64	120	53,333	106	60,377
128	184	69,565	170	75,294
256	312	82,051	278	85,906
512	568	90,140	554	92,418
1024	1080	94,814	1066	96,060
1400	1456	96,153	1442	97,087

La figure 2.4 illustre graphiquement la variation du pourcentage des données lorsqu'on varie la taille des paquets IP entre 1 et 1400 octets.

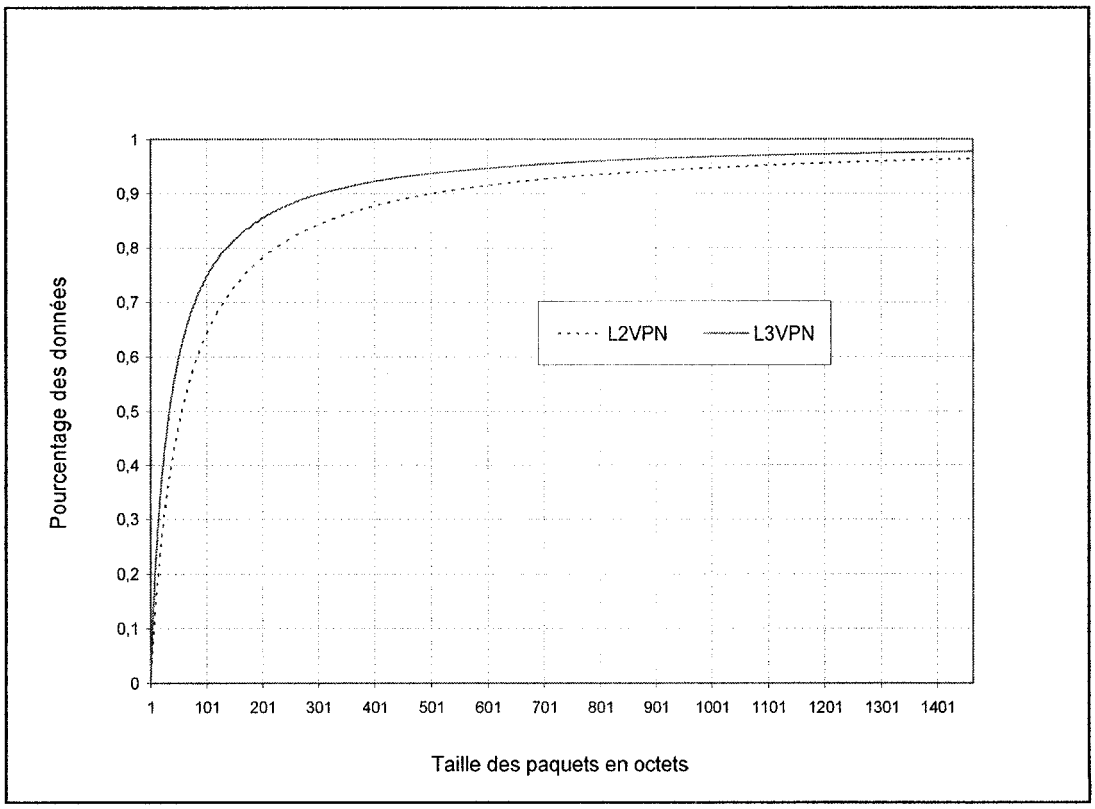


Figure 2.4 ***Pourcentage des données dans L2VPN et L3VPN.***

En connaissant le pourcentage des données dans les trames, on est capable de calculer les débits applicatifs en utilisant la formule (2.4).

$$Débit\_applicatif = Débit\_physique \times Pourcentage\_des\_données \qquad (2.4)$$

Le tableau suivant présente l'application de l'équation (2.4) dans le calcul du débit applicatif. Les valeurs de D1 et D2 représentent les débits applicatifs dans L2VPN et L3VPN respectivement.

Tableau 2.7

Calcul du débit applicatif

	L2VPN		L3VPN		
X (octet)	**P1 (%)**	**D1 (Mbps)**	**P2 (%)**	**D2 (Mbps)**	**D2-D1 (Mbps)**
64	53,333	48,484	60,377	54,237	5,752
128	69,565	65,30	75,294	70,329	5,023
256	82,051	79,012	85,906	82,580	3,568
512	90,140	88,275	92,418	90,459	2,183
1024	94,814	93,772	96,060	94,990	1,217
1400	96,153	95,367	97,087	96,286	0,918

On constate que les valeurs du débit applicatif dans un L3VPN sont bien plus élevées que dans un L2VPN. Les performances des services L2VPN sont limitées à cause de la taille des entêtes ajoutés dans le processus d'encapsulation des paquets IP. En effet, lorsque les paquets sont petits, leur nombre est considérablement élevé. Ainsi, il y a aura plus de bits IFG et d'entêtes dans le trafic. Pour cette raison, la quantité des données transmises est moins élevée lorsque les paquets sont petits.

La figure 2.5 illustre graphiquement la variation du débit applicatif pour les deux technologies lorsqu'on varie la taille des paquets entre 1 et 1400 octets, ainsi que la différence entre les débits applicatifs dans L2VPN et L3VPN.

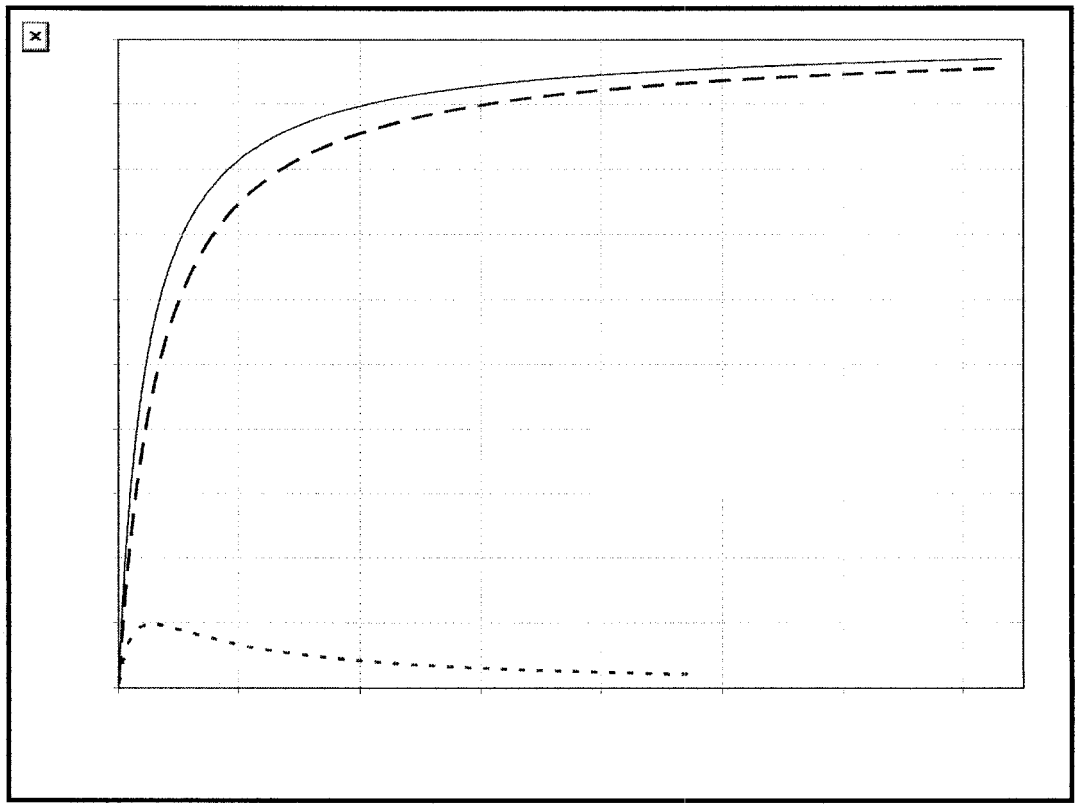


Figure 2.5 ***Débit applicatif dans L2VPN vs L3VPN.***

En analysant les résultats, on peut conclure que dans les L2VPNs, le trafic causé par les entêtes ajoutés devient immense si les paquets sont de petites tailles. Les ressources du réseau MPLS sont consommées inutilement par les entêtes. Par contre, dans les L3VPNs, les entêtes ajoutés utilisent moins de bande passante. Le trafic du client dans un L2VPN atteindra sa limite plus rapidement que dans un L3VPN.

La différence entre les débits applicatifs dans L2VPN et L3VPN est inversement proportionnelle à la taille des trames. Plus les paquets sont petits, plus le pourcentage des données est petit. La différence entre D1 et D2 est plus évidente pour les petits paquets. Sa plus grande valeur est `9,74 Mbps` et elle correspond aux paquets de taille `56 octets`.

Les performances des services L2VPN peuvent être améliorées en utilisant des algorithmes de compression d'entêtes. Le *MPLS Working Group* au sein de l'IETF s'est intéressé à la compression d'entête dans les LSPs. la compression des entêtes des flux transitant dans MPLS est décrite dans [19].

Étant donné que les communications voix utilisent des petits paquets, on s'est intéressé à analyser et comparer les différentes possibilités pour transporter la voix en utilisant les deux approches. La section suivante présente une étude détaillée de la communication voix dans L2VPN et L3VPN.

2.5 Étude de cas : Communication de la voix dans L2VPN vs L3VPN

Cette section fournit une étude du transport de la voix dans les deux types de RPV. L'étude comparative utilise différents CoDecs de voix. Le multiplexage des communications voix a été aussi examiné afin d'évaluer son impact sur les performances.

L'utilisation du transport de la voix sous forme de paquets a été développée pour les réseaux IP, ATM, FR et MPLS. Plusieurs opérateurs ont migré les applications de voix des réseaux TDM vers des réseaux à commutations de paquets. Cela permet d'avoir un seul groupe d'ingénieurs et de gestionnaires pour maintenir un seul réseau dédié à la voix et les données au lieu d'avoir deux réseaux séparés [20].

Le déploiement de la téléphonie IP a été possible grâce au déploiement de la QdS, qui garantit une latence et un taux de perte de paquets acceptable, et l'interopérabilité du réseau IP avec le réseau traditionnel de la téléphonie PSTN [21].

2.5.1 Avantages de l'utilisation de MPLS pour la communication voix

Nous avons vu l'intérêt de l'utilisation d'un réseau à commutation de paquets dans le but d'offrir une architecture unique pour les services de voix et de données. Toutefois les deux types de trafic ne se comportent pas de la même manière vis-à-vis de la congestion. Les mécanismes de contrôle de congestion qu'utilise le protocole TCP adaptent le débit du trafic de données à l'état de congestion du réseau [22]. Si des paquets sont rejetés, TCP les retransmettra. Cependant le trafic voix, qui utilise le protocole UDP, est très sensible à la congestion. Un délai excessif n'est pas toléré. Les paquets voix perdus ne peuvent pas être retransmis car l'application voix est interactive. Le trafic voix doit avoir un très bas taux de perte et un très bas délai.

L'ingénierie de trafic et le support du mécanisme *Differentiated Services* (DiffServ) dans un réseau MPLS représentent une nouvelle façon pour offrir la qualité de service dans les réseaux. Le DiffServ assure le marquage du trafic afin de le traiter dans les points de congestion selon sa priorité [23]. Les bits EXP dans l'étiquette MPLS sont utilisés pour attribuer la priorité adéquate au trafic. Le trafic voix est marqué avec une haute priorité et les ressources sont réservées au flux à chaque nœud du réseau MPLS.

Dans [24], les auteurs identifient les fonctionnalités nécessaires pour implémenter l'ingénierie de trafic dans un réseau MPLS. Cela permettra de garantir au trafic voix les ressources du réseau nécessaire. La réservation des ressources pour la voix peut être effectuée en utilisant le protocole RSVP [25]. Si par exemple le plus court chemin (le chemin IP normal) n'a pas assez de ressources pour supporter un appel, l'ingénierie de trafic trouvera un chemin moins congestionné pour envoyer le flux.

Les tunnels établis à l'aide du protocole RSVP peuvent être vus comme des lignes de communication privées. La taille du tunnel peut être choisie de manière à supporter un

nombre prédéfini d'appels. Pour qu'un appel puisse utiliser un tunnel donné, il faut lui appliquer l'étiquette MPLS appropriée.

Si le nombre d'appels reçus dépasse la capacité du tunnel, il existe différentes options. Lorsque le tunnel approche l'épuisement de sa capacité, un ajustement dynamique de la bande passante peut être utilisé ou un nouveau tunnel peut être signalé. Si les ressources réservées du réseau sont excédées alors tout appel reçu sera automatiquement perdu. Dans [26] les auteurs suggèrent une architecture de réseau MPLS où un serveur SIP, en plus d'aider à l'établissement des sessions, sert également à la gestion dynamique des ressources.

2.5.2 Les deux modèles proposés pour le transport de la voix dans MPLS

Deux approches ont été proposées. La première proposition *Voice over MPLS* (VoMPLS). Elle encapsule la voix directement dans MPLS en éliminant IP de la pile des protocoles. La deuxième proposition est *Voice over IP over MPLS* (VoIPoMPLS). Elle utilise MPLS pour transporter la voix sur IP.

Le standard VoIPoMPLS utilise la même pile de protocoles utilisée par VoIP (RTP/UDP/IP). Le trafic est ensuite encapsulé dans MPLS. Le standard VoIPoMPLS a l'avantage de bénéficier des protocoles utilisés par VoIP pour contrôler les appels et les équipements (MGCP, H.323, SIP, MEGACO, etc). Le *draft,* publié par l'IETF, [27] offre un premier modèle de référence pour la proposition VoIPoMPLS.

Le standard VoMPLS encapsule les échantillons de voix directement dans MPLS. Il offre un moyen de transport efficace pour le transport de la voix en éliminant les entêtes RTP/UDP/IP. Le standard VoMPLS est spécifié dans [28]. Elle définit le format de l'entête de VoMPLS et comment la voix est encapsulée directement dans MPLS. Mais elle ne donne pas les détails de son fonctionnement dans les réseaux MPLS, comme par

exemple la signalisation pour la voix, l'architecture et les fonctions des passerelles et des routeurs. De plus, il existe différentes façons pour l'implémentation de VoMPLS et le but de cette spécification est de supporter toutes les possibilités de déploiement de VoMPLS. La figure 2.6 illustre l'architecture de référence pour VoMPLS.

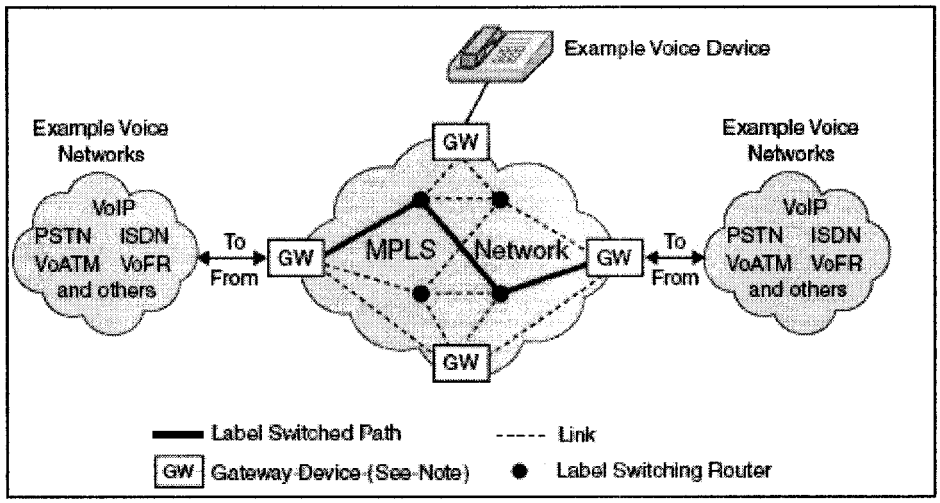


Figure 2.6 ***Architecture de référence pour VoMPLS [21].***

Le réseau MPLS est composé de passerelles, des routeurs LSR et des chemins LSP. Les passerelles peuvent être connectées entre elles directement ou indirectement via les routeurs LSR. La passerelle supporte entre autres les fonctionnalités d'un routeur PE.

La proposition VoMPLS a été refusée par l'IETF en faveur de la proposition VoIPoMPLS [21]. Le standard VoMPLS a été ensuite adopté par l'organisme MPLS Forum. Les arguments derrière le refus de cette proposition sont :

a. L'élimination de IP limite la connectivité au niveau des périphériques du réseau MPLS. Ainsi le standard VoMPLS ne sera pas approprié pour amener la voix jusqu'aux postes des clients.
b. L'utilisation des algorithmes de compression d'entête donnera à VoIPoMPLS une efficacité similaire à celle de VoMPLS.

2.5.3 Évaluation du transport de la voix dans L2VPN vs. L3VPN

Dans ce mémoire *Voice over IP over L2VPN* (VoIPoL2VPN) réfère au transport de VoIP dans L2VPN, *Voice over IP over L3VPN* (VoIPoL3VPN) réfère au transport de VoIP dans L3VPN et *Voice over L3VPN* réfère au transport de la voix directement dans L3VPN (VoL3VPN).

Le reste de cette section est une évaluation comparative de VoL2VPN vs. VoL3VPN en termes d'efficacité de l'utilisation de la bande passante pour la voix. Différents CoDecs de voix ont été utilisés.

2.5.3.1 Application du modèle VoMPLS

D'après la description de VoMPLS, ce modèle ne peut pas s'appliquer à L2VPN. Dans les services L2VPN, le trafic dans un réseau local est encapsulé tel quel dans MPLS. Cependant VoMPLS transporte la voix dans MPLS directement sans la couche IP ni liaison. Le déploiement de VoMPLS est restreint au réseau MPLS et il n'est pas possible de le ramener jusqu'aux postes clients.

Dans VoMPLS, la voix est transportée dans des "*primary subframes*" qui contiennent un entête VoMPLS suivie par un échantillon de la voix. Dans l'entête on trouve le champ "*channel identifier*" qui permet le multiplexage des connexions voix et le champ "*length indicator*" qui permet le transport de plusieurs échantillons de la voix provenant d'une même conversation [28]. La contrainte est que le nombre de bits par échantillon fois le nombre d'échantillons se conforme à la limite du nombre d'octets. Chaque échantillon audio est représenté par un nombre fixe d'octets. Ce nombre change d'un CoDec de voix à un autre. La figure 2.7 illustre la structure des données pour VoMPLS transporté sur Ethernet.

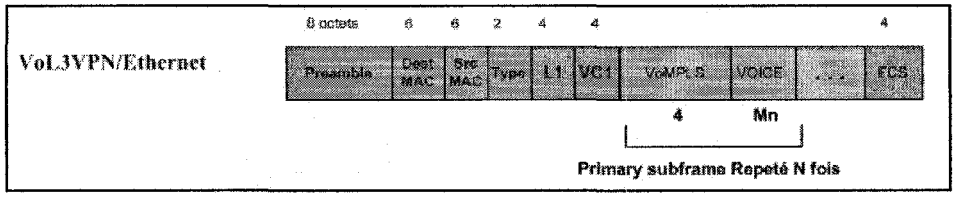


Figure 2.7 ***Structure des trames pour VoMPLS.***

Dans cette section les notations suivantes sont utilisées comme paramètres :

a. n : Nombre d'octets dans un échantillon audio.

b. M : Nombre d'échantillons audio provenant d'une seule connexion et multiplexé dans un seul paquet de transport.

c. N : Nombre de connexions actives générant des paquets voix multiplexés dans un même paquet de transport.

Il faut noter que N n'est pas le nombre total des connexions voix. Il est seulement le nombre des connexions actives qui génèrent les paquets voix à transporter. Si la détection des silences est activée, une fraction f sera supprimée de l'unité de données à transporter. Dans ce cas N sera égal au nombre total des connexions de voix multiplié par f. une valeur typique de f est 0.6 qui indique que en moyenne 60% de la conversation est un silence [29].

Les données sont encapsulées en utilisant une pile d'étiquettes MPLS. Les commutateurs ATM peuvent être utilisés comme des LSRs. Le type d'encapsulation MPLS pour envoyer des étiquettes ou des trames MPLS vers ou à partir des ATM-LSRs est spécifié dans [30]. Le document spécifie l'utilisation de l'encapsulation *ATM adaptation layer 5* (AAL5) pour MPLS. Il faut noter que les bits de bourrage (*padding*) sont ajoutés lorsque la taille de l'unité de données n'est pas un multiple de 48 octets.

Notons F la taille totales des entêtes qui sont indépendants du nombre des échantillons de voix et V la taille de l'entête VoMPLS ajouté pour chaque échantillon audio. F et V peuvent être calculés à partir de la figure 2.7 qui indique le nombre d'octets dans chaque entête ajouté pour l'encapsulation des échantillons de voix dans une trame Ethernet transportant VoMPLS. Le tableau suivant illustre les valeurs de F et V.

Tableau 2.8

Calcul du débit applicatif [29]

	F (octets)	**V (octets)**
VoL3VPN/Ethernet	34	4

Le nombre des octets de voix est donné par la formule (2.5). La taille de l'*Overhead* est donnée par la formule (2.6).

$$N \cdot M \cdot n \tag{2.5}$$

$$F + N \cdot V \tag{2.6}$$

L'*Overhead* introduit à la voix a un effet sur le pourcentage de la bande passante utilisée par la voix. Ce pourcentage indique l'efficacité dans l'utilisation de la bande passante. Il est obtenu en divisant le nombre d'octets de voix par la taille totale de la trame. Notons E l'efficacité dans l'utilisation de la bande passante par la voix. La formule (2.7) calcule la valeur de E.

$$E = \frac{N \cdot M \cdot n}{N \cdot M \cdot n + N \cdot V + F} \tag{2.7}$$

Dans le but d'évaluer l'impact de la variation du paramètre M sur E, on a choisi d'utiliser deux valeurs différentes de M, soit M1 et M2. Les valeurs de M1 sont les valeurs par défaut de M et sont spécifiées dans [28]. Notons aussi M2 les valeurs de M qui correspondent à un délai total du groupage des échantillons audio égaux à 30 ms. Les caractéristiques des CoDecs utilisés sont données dans le tableau ci-dessous.

Tableau 2.9

Caractéristiques des CoDecs [29]

CoDec	Débit (Kb/s)	n	Délai (ms)	M1	M2
G711	64	40	5	2	6
G723.1 (SB-ADPCM)	64	40	5	4	6
G723.1(5,3kb/s)	5,3	20	30	1	1
G723,1(6,4kb/s)	6,4	24	30	1	1
G726(ADPCM)	32	20	5	2	6
G727(EADPCM)	32	20	5	4	6
G728(LD-CELP)	16	5	2.5	4	12
G729	8	10	10	2	3

Les deux tableaux suivants illustrent les résultats du calcul de l'efficacité E pour M=M1 et M=M2 respectivement. La valeur N des connexions actives a été fixée à 1.

Tableau 2.10

Efficacité E pour M=M1

CoDec	VoL3VPN/Ethernet
G711	67,79%
G723.1 (SB-ADPCM)	80,80%
G723.1(5,3kb/s)	34,48%
G723,1(6,4kb/s)	38,70%
G726(ADPCM)	51,28%
G727(EADPCM)	67,79%
G728(LD-CELP)	34,48%
G729	34,48%

Tableau 2.11

Efficacité E pour M=M2

CoDec	VoL3VPN/Ethernet
G711	86,33%
G723.1 (SB-ADPCM)	86,33%
G723.1(5,3kb/s)	34,48%
G723,1(6,4kb/s)	38,70%
G726(ADPCM)	75,94%
G727(EADPCM)	75,94%
G728(LD-CELP)	61,22%
G729	44,11%

En comparant les deux tableaux, on peut conclure que l'efficacité E peut être améliorée en augmentant la valeur du paramètre M qui indique le nombre d'échantillons de voix transportés par le même paquet de transport. Cependant, plus le nombre d'octets de voix transportés par un seul paquet augmente, plus la perte de trafic voix, qui peut être occasionnée, augmente. La Recommandation G.711 indique que la perte de trafic durant 32 à 64 ms est disruptive parce qu'elle affecte les phonèmes de la parole. La perte de cellule de 4 à 16 ms est acceptable parce qu'elle n'affecte pas la parole. Pour cette raison et pour faciliter les calculs, le paramètre M est fixé à M1 dans le reste de cette section.

On peut aussi conclure que l'utilisation de VoMPLS avec une seule connexion voix n'est pas efficace. L'utilisation d'une seule connexion de voix est nécessaire dans le réseau d'accès dont les destinations sont différentes. Mais dans le réseau VoMPLS, il est important de multiplexer, entre les passerelles, les échantillons audio de plusieurs connexions voix dans le même paquet de transport afin d'augmenter l'efficacité dans l'utilisation de la bande passante.

Dans le but d'évaluer l'impact de la variation du paramètre N sur les valeurs de E, le calcul précédant a été rétabli en variant N de 1 à 10. Les résultats sont illustrés graphiquement dans les figures suivantes selon le CoDec utilisé.

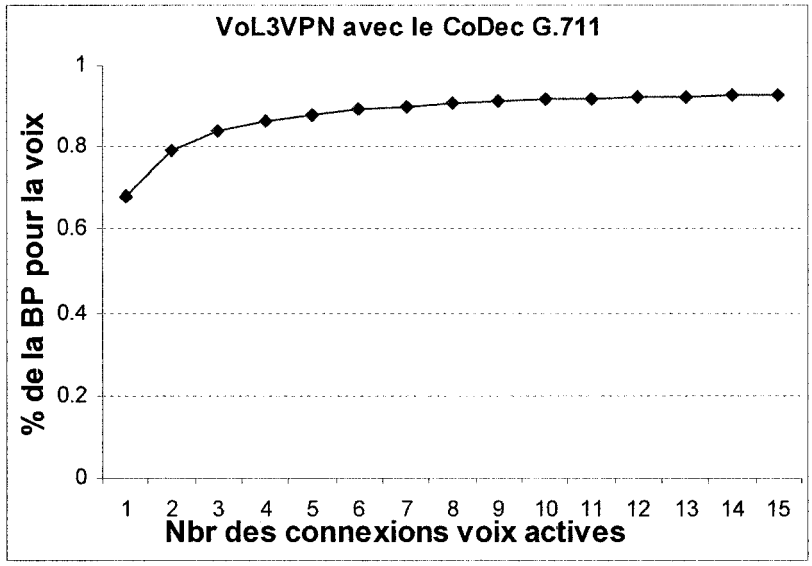


Figure 2.8 *VoL3VPN avec le CoDec G.711.*

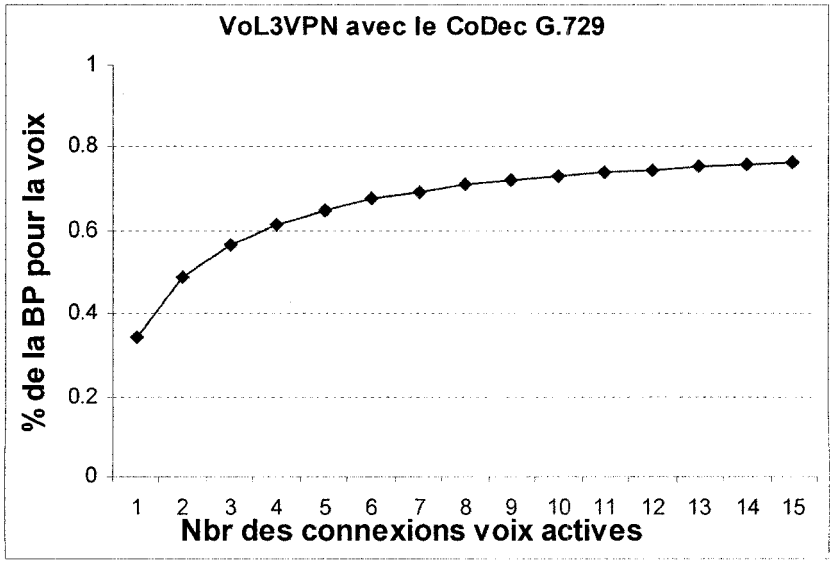


Figure 2.9 *VoL3VPN avec le CoDec G.729.*

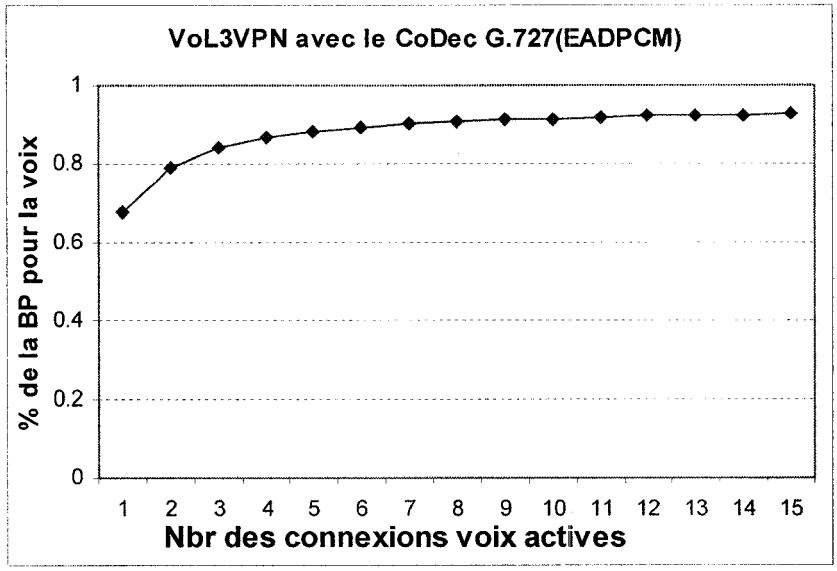


Figure 2.10 *VoL3VPN avec le CoDec G.727.*

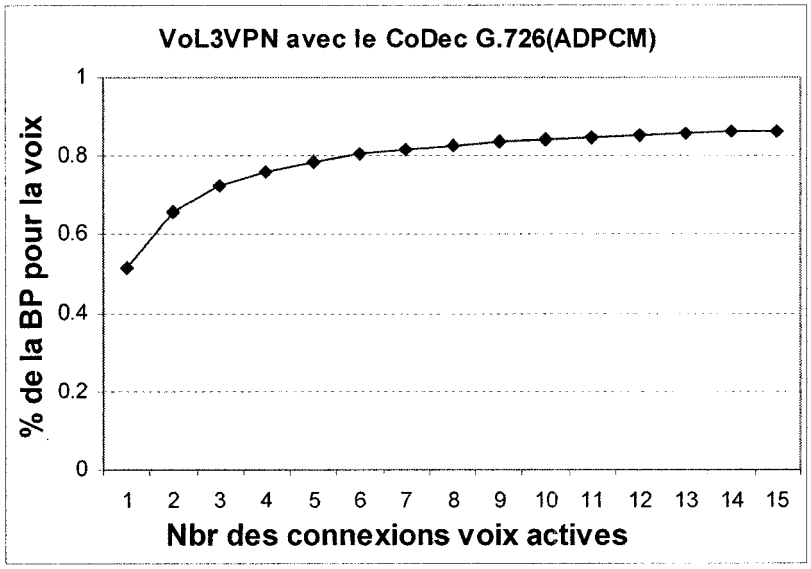


Figure 2.11 *VoL3VPN avec le CoDec G.726.*

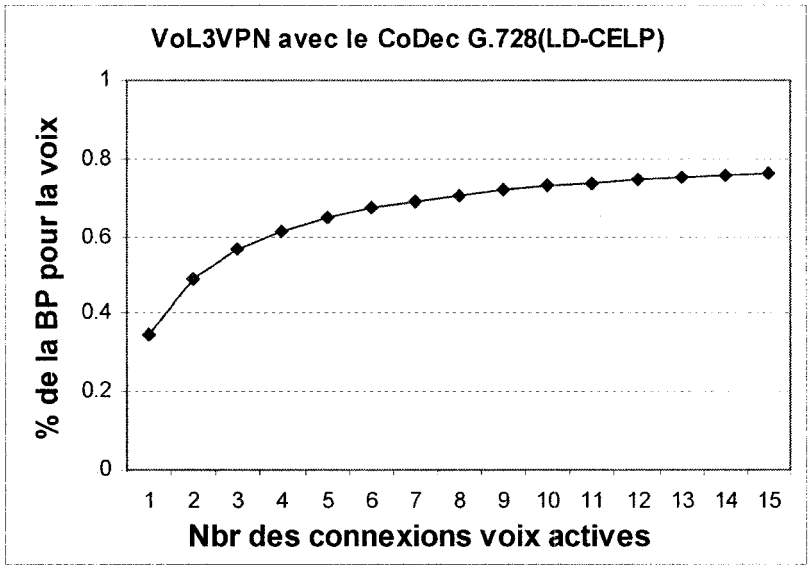


Figure 2.12 ***VoL3VPN avec le CoDec G.728.***

Les résultats changent d'un CoDec à un autre puisque les paramètres sont différents. Il est clair que plus N augmente, l'efficacité E devient plus élevée. Lorsque N dépasse 10, la différence entre les trois options de transport est inférieure à 20%.

2.5.3.2 Application du modèle VoIPoMPLS

VoIP est basé sur IP. Le protocole de transport utilisé est *User Datagram Protocol* (UDP). UDP est adopté pour le transport de VoIP malgré que TCP soit considéré comme mécanisme de transport idéal pour sa fiabilité. Le trafic voix est supporté par le protocole *Real Time Protocole* (RTP). RTP permet d'identifier la charge utile, garde le séquencement, inclut l'horodatage et surveille la livraison des données [31]. La figure 2.13 illustre la structure des données pour VoIPoL2VPN et VoIPoL3VPN.

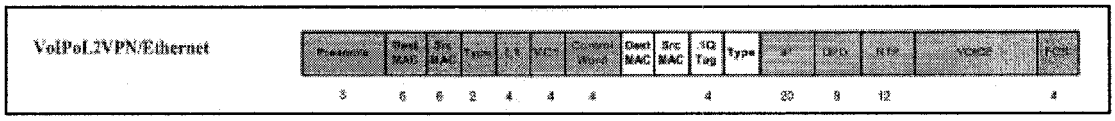


Figure 2.13 ***Structure des trames pour VoIPoL2VPN et VoIPoL3VPN.***

Pour un paquet voix, les entêtes RTP, UDP, IP (40 octets en total), sont ajoutés. Cet *Overhead* diminue énormément l'efficacité de VoIP. Cette lacune peut être réglée en utilisant des standards de compression. L'application de la compression des entêtes sur RTP/UDP/IP est spécifiée dans [32]. La taille des entêtes de 40 octets est réduite à 2-4 octets.

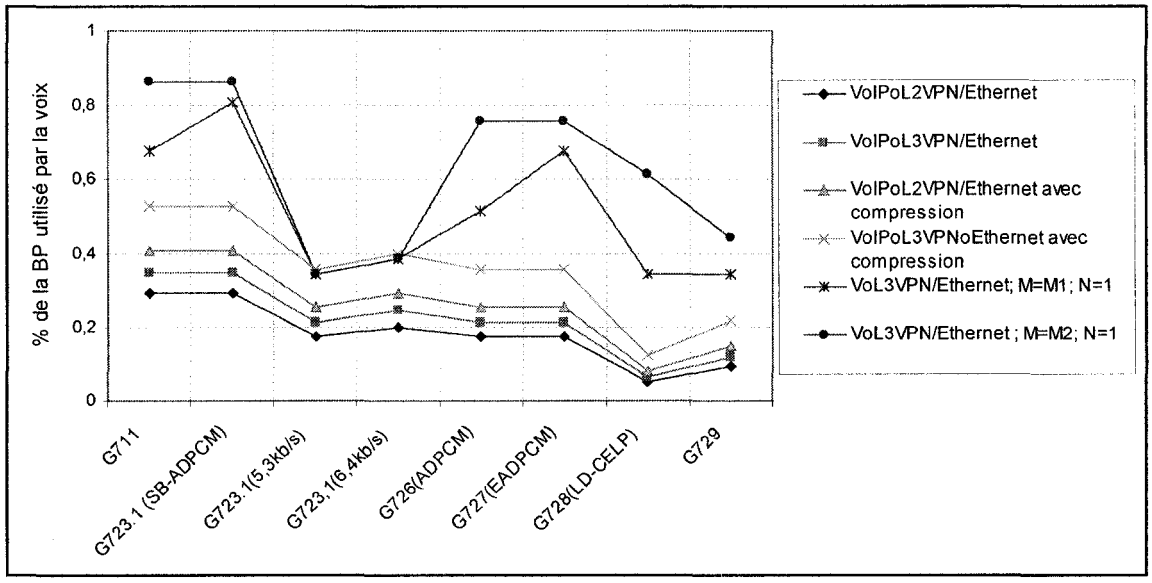
En se basant sur la figure 2.13, il est facile de calculer l'*Overhead* ajouté à la voix pour les différentes options. Le tableau suivant illustre la taille de chaque *Overhead* ajouté.

Tableau 2.12

Taille de l'*Overhead* ajouté à la voix

	Overhead (octets)
VoIPoL2VPN/Ethernet	96
VoIPoL2VPN/Ethernet avec compression	58
VoIPoL3VPN/Ethernet	74
VoIPoL3VPN/Ethernet avec compression	36

Les résultats du calcul de l'efficacité E sont illustrés graphiquement dans la figure 2.14. Dans le cas d'utilisation des entêtes RTP/UDP/IP compressés, on a supposé que la taille totale de ces entêtes a été réduite à 2 octets. Les résultats de VoL3VPN ont été ajoutés afin de les comparer avec VoIPoL2VPN et VoIPoL3VPN.



$$E = \frac{N \cdot M \cdot n}{N \cdot M \cdot n + N \cdot V + F}$$

Figure 2.14 ***Les résultats de calcul de l'efficacité E.***

Les résultats montrent une efficacité plus élevée pour VoL3VPN par rapport à VoIPoL3VPN et VoIPoL2VPN. Dans VoL3VPN la voix est transportée directement dans MPLS, donc les entêtes RTP/UDP/IP ne sont pas utilisés et cela réduit énormément l'*Overhead* ajouté à la voix. En plus, VoL3VPN a l'avantage de pouvoir transporter plusieurs paquets voix et de multiplexer plusieurs connexions de voix dans un seul paquet de transport. Les résultats de l'efficacité pour VoL3VPN seront meilleurs si on avait utilisé des valeurs plus élevées pour le paramètre N.

L'efficacité de VoIPoL3VPN augmente lorsqu'on utilise la compression d'entêtes. Elle s'est même approchée de la valeur de l'efficacité réalisée par VoL3VPN lorsqu'on utilise le CoDec G.723. En effet les valeurs de M1 et M2 pour ce CoDec sont fixés à 1. Réduire la valeur de M à 1 exclue la possibilité de multiplexer plusieurs paquets de voix

et ainsi diminue l'efficacité de VoL3VPN. De plus, l'utilisation des algorithmes de compression donne à VoIPoL3VPN une efficacité similaire à VoL3VPN.

VoIPoL2VPN a une faible efficacité à cause de la conservation de l'entête Ethernet. On peut noter que l'efficacité peut être améliorée si on utilise la compression d'entête. L'encapsulation de la voix IP/UDP/RTP et dans MPLS génère un *Overhead* de 56 octets pour L2VPN. Elle dépasse la taille du paquet voix dont la taille varie entre 5 et 40 octets selon le CoDec voix utilisé. L'utilisation de la compression s'avère nécessaire pour offrir une efficacité meilleure.

Le multiplexage RTP peut aussi améliorer l'efficacité de VoIPoL2VPN et VoIPoL3VPN, mais il n'est pas encore standardisé.

2.6 La fragmentation des paquets IP dans L2VPN vs. L3VPN

Les paquets IP peuvent transiter à travers des réseaux aux technologies différentes. Chaque réseau définit une taille maximale pour les datagrammes IP que l'on appelle MTU (1000 pour Arpanet, 1500 octets dans Ethernet et 4470 octets dans FDDI). Il est donc impossible de fixer une taille maximale des paquets IP pour éviter la fragmentation [33].

La fragmentation prend lieu lors de la transition d'un réseau à un autre dont le MTU est plus petit. Si le datagramme est trop grand pour passer sur le réseau, alors il faudra le fragmenter en le découpant en fragments de tailles inférieures au MTU du réseau et de telle façon que la taille du fragment soit un multiple de 8 octets. Le routeur transmet les fragments de façon indépendante et il n'est pas garanti que les fragments soient arrivés dans le bon ordre. Les datagrammes contiennent les informations nécessaires pour les réassembler [33].

Les clients veulent souvent savoir si les datagrammes de leurs trafics seront fragmentés ou pas. Il est important de vérifier si la fragmentation est possible dans les réseaux virtuels L2VPN et L3VPN. Si non, quelles sont les solutions possibles pour éviter de dépasser le MTU d'un réseau intermédiaire.

2.6.1 La fragmentation dans L2VPN

Dans un tunnel L2VPN, il n'est pas possible de faire la fragmentation des paquets IP [34]. Le MTU sur le lien d'accès doit être bien choisi afin que la taille totale de la trame Ethernet ne dépasse pas le MTU du réseau. Les trames layer-2 qui dépassent le MTU du réseau après l'encapsulation sont détruites automatiquement et ne traverseront pas le réseau pour atteindre leurs destinations.

Le MTU dans le lien d'accès du L2VPN (AC) doit être choisi de manière à ce que la taille d'une trame plus l'entête final ne dépassent pas le MTU définit dans le cœur du réseau MPLS. Toute trame dépassant le MTU sera automatiquement détruite par le routeur PE [34].

Pour résoudre ce problème, le client a besoin d'une configuration propre et d'une gestion de la taille du MTU à prendre en considération. Il est possible d'utiliser le *Path MTU discovery* défini dans [35]. Il permet d'identifier tout le long du chemin entre la source et la destination le MTU standard pour ne pas rencontrer dans le réseau un routeur qui a un MTU plus petit. Cette solution informe les sources de la taille maximale des paquets IP à envoyer. L'application du mécanisme *Path MTU discovery* dans MPLS est discutée dans [35].

Si par exemple le MTU du réseau est limité à 1500 octet, la taille limite de la trame Ethernet qui pourrait être transportée dans un tunnel L2VPN est 1514 octet. La trame Ethernet est encapsulée dans MPLS en lui ajoutant deux étiquettes MPLS et le champ *control word*. Le tout est encapsulé ensuite dans Ethernet. Sachant qu'une étiquette

MPLS est de taille 4 octets, le champ *controle word* dans l'entête MPLS est de taille 8 octets et l'entête Ethernet est de taille 14 octets (voir figure 2.15). La taille des trames Ethernet qu'un client peut générer ne doit pas dépasser la taille maximale qui est calculée par la formule (2.8).

$$\text{Taille\_max} = 1514 - (4+4+8+14) = 1488 octets \qquad (2.8)$$

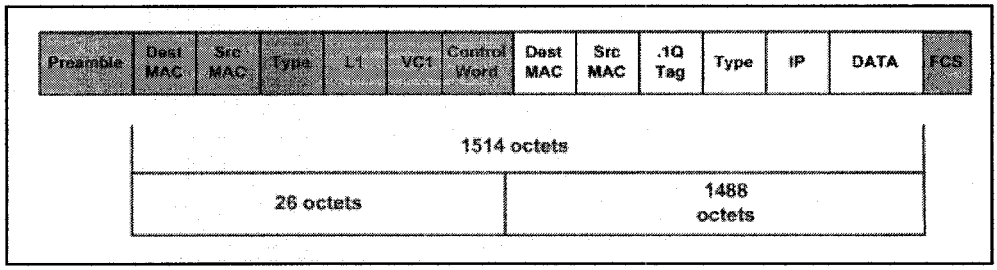


Figure 2.15 ***Taille maximale lorsque le MTU =1500 octets.***

2.6.2 La fragmentation dans L3VPN

La solution L3VPN est basée sur la couche réseau et le trafic IP. Elle devrait permettre la fragmentation des paquets IP. Dans les normes de L3VPN, aucun document ne traite encore la fragmentation.

Layer 3 VPN devrait être capable d'accepter des trames qui dépassent le MTU du réseau en les fragmentant au niveau d'IP. En cas de fragmentation, si une trame dépasse le MTU, le paquet IP devrait être fragmenté. Le dernier fragment devrait contenir les bits restants plus les bits de bourrages (*padding*). Afin de vérifier ces hypothèses, nous avons effectué des tests dans la plateforme de Bell Canada. Le scénario des tests et les résultats sont présentés dans l'0. Les résultats ont montré que la fragmentation est bel et bien possible dans Layer-3 VPN. Si la taille d'un paquet IP dépasse le MTU du réseau, les routeurs PEs s'occuperont de la fragmentation et le réassemblage du paquet.

Cependant, il est vivement conseillé d'éviter la fragmentation et le réassemblage [36]. Ces opérations requièrent beaucoup plus de ressources que d'envoyer un seul paquet. La solution idéale pour éviter la fragmentation des paquets est la configuration et la gestion des MTUs dans les liens entre les routeurs CEs et les routeurs PEs et à travers le réseau de l'opérateur en utilisant le mécanisme *Path MTU discovery* [35].

2.7 Synthèse et conclusion

Pour conclure, le présent chapitre a d'abord proposé une étude analytique des performances de L2VPN et L3VPN. Les différentes étapes d'encapsulation du trafic dans MPLS ont été examinées dans le but de définir la quantité d'entêtes qui est ajoutée au trafic. Ces informations ont permis de calculer le débit physique et applicatif dans L2VPN et L3VPN pour différentes tailles de paquets.

On a vu qu'il y a une différence entre le débit applicatif pour L2VPN et L3VPN. Cette différence est plus évidente pour les paquets de petites tailles. Étant donné que les communications voix utilisent des petits paquets, on s'est intéressé à analyser et comparer les différentes possibilités pour transporter la voix en utilisant les deux approches.

Deux modèles ont été utilisés pour le transport de la voix à travers MPLS : VoMPLS et VoIPoMPLS. Le premier modèle est le plus efficace. Il élimine les entêtes RTP/UPD/IP en transportant la voix directement sur MPLS. En plus, il permet le multiplexage des paquets voix de plusieurs sources. Les CoDecs de voix ont des caractéristiques distinctes, donc les résultats peuvent changer si on utilise différent CoDecs de voix.

Dans le chapitre suivant, la deuxième partie de la problématique sera traitée. Elle concerne le contrôle des flux de plusieurs L2VPNs connectés à un même commutateur CE. La problématique sera illustrée, ensuite un algorithme sera proposé comme solution.

CHAPITRE 3

LA QdS DANS L2VPN, PROBLÉMATIQUE ET SOLUTION

Ce chapitre présente une solution à la deuxième partie de la problématique. Elle concerne l'implémentation de la QdS pour les services L2VPN lorsque les équipements CEs sont des commutateurs.

On commence par présenter une introduction aux mécanismes de la QdS. Ensuite la problématique est illustrée par des scénarios de test. Les solutions possibles sont exposées. Finalement, la solution retenue, qui répond le mieux à la problématique, est présentée.

3.1 La qualité de service dans L2VPN

Cette section présente un survol des mécanismes de la QdS qui permettent de garantir les exigences des clients, comme par exemple la perte de paquets, le délai, la variation du délai, etc.

Les services L2VPN sont capables de fournir une qualité de service pour les différents types de trafic. En effet, L2VPN utilise l'infrastructure IP/MPLS qui offre plusieurs mécanismes pour le support de la QdS.

L'implémentation de la QdS dans les services L2VPN peut être divisée en trois parties : le cœur et l'accès du réseau de l'opérateur et les réseaux locaux des clients. La figure 3.1 illustre les frontières de ces trois sections.

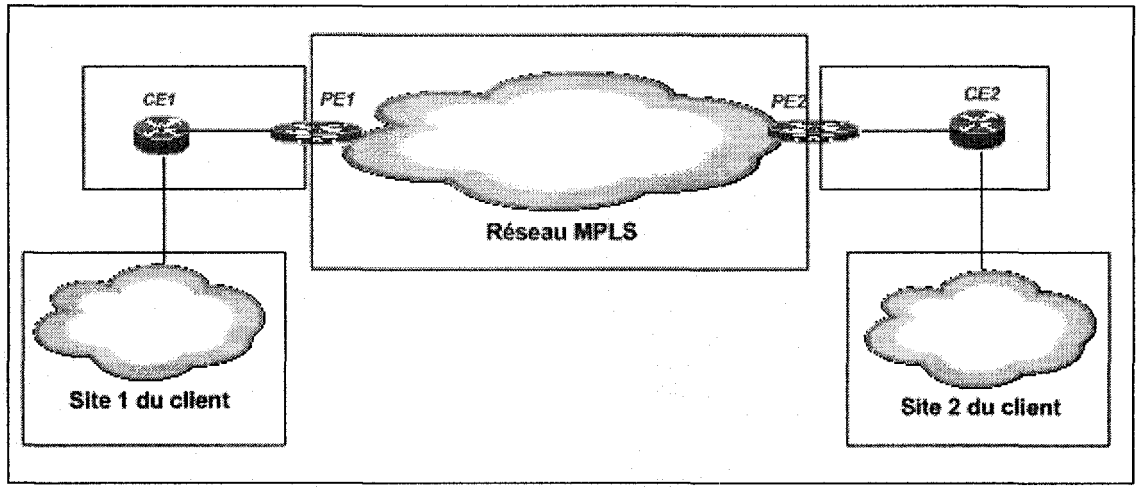


Figure 3.1 ***Les trois parties d'un L2VPN qui sont concernées par la QdS.***

La QdS dans le réseau MPLS est gérée par l'opérateur. Dans l'accès du réseau MPLS (CE-PE) la QdS peut être administrée par l'opérateur s'il héberge les équipements CE ou par l'opérateur et les clients si les équipements CEs font parties des réseaux des clients. Dans les réseaux locaux des clients, la QdS est gérée par les clients.

La QdS ne doit pas être considérée comme un mécanisme qui concerne uniquement les fournisseurs de services. Les réseaux locaux qui n'ont pas assez de bande passante ou qui transportent le trafic des applications temps réel sont aussi concernés par les problématiques de la qualité de service. Les bits *Class of Service* (CoS) de l'entête VLAN peuvent être utilisés pour prioriser le trafic dans les réseaux locaux.

Dans le reste de cette section, on présente les outils et les techniques qui sont utilisés pour mettre en place la QdS.

3.1.1 La classification et le marquage

Les clients utilisent des applications qui n'ont pas les mêmes exigences en termes de QdS. La voix sur IP par exemple a des contraintes sur le délai, la perte de paquet et la variation du délai tandis que d'autres applications peuvent supporter la perte de paquet et une valeur élevée du délai. Les trafics générés par ces applications ne doivent pas être traités de la même manière.

La classification est une opération que l'on utilise pour différentier les types de trafic. Elle consiste à identifier le trafic afin de créer des groupes de flots qui partage les mêmes exigences en termes de la qualité de service. Elle peut être réalisée n'importe où dans le réseau.

Il existe plusieurs critères pour classifier un trafic, on peut se baser sur l'adresse IP, les numéros de ports UDP ou TCP, les valeurs du champ *Type of Service* (ToS) de l'entête IP, etc.

Une fois la classification est effectuée, le trafic est marqué. Le marquage est nécessaire pour que les nœuds du réseau qui reçoivent le trafic puissent identifier la classe de trafic et ainsi le traiter convenablement.

Il existe trois types de marquages :

a. IP/DSCP – L'entête IP contient le champ *Differentiated Services Code Point* (DSCP) de 6 bit qui peut être utilisé à marquer le paquet.
b. MPLS EXP – L'étiquette MPLS contient dans son entête le champ EXP de 3 bit qui peut être utilisé à marquer le trafic dans MPLS.
c. 802.1p ou CoS – L'entête VLAN contient des bits de priorité 802.1p pour différencier le trafic dans un LAN.

Supposons que le trafic du client comprend la voix, la vidéo, les messages de routage IP, le trafic de la gestion du réseau et d'autres applications de données. Une classification typique de ce genre de trafic est proposée dans [37]. Elle comprend les classes suivantes :

a. *Voice media* – cette classe consiste en le trafic voix. Les paquets sont marqués en utilisant le DSCP *Expedited Forwarding* (EF) et le CoS 5.
b. *Interactive vidéo* – cette classe comprend seulement le trafic Interactive vidéo. Le marquage utilise le DSCP *Assured Forwarding* 41 (AF41) et le CoS 4.
c. *Streaming video* – cette classe contient uniquement du trafic vidéo. Les paquets sont marqués par le DSCP CS4 et le CoS 4.
d. *Voice/Video signaling* – cette classe concerne le trafic de signalisation. Le marquage utilise le DSCP AF31 et le CoS 3.
e. *Routing protocols* – cette classe est dédiée au trafic de mise à jour pour le routage IP. Les paquets sont marqués par le DSCP CS6 et le CoS 6.
f. *Network management* – cette classe est dédiée au trafic de la gestion du réseau. Le DSCP utilisé est CS2 et le CoS utilisé est 2.

Pour le trafic des applications de données, il est recommandé dans [37] de na pas utiliser plus que 5 classes différentes. Chaque classe est composée du trafic des applications de données qui ont les mêmes exigences en terme de la QdS. Dans [37] on trouve la classification typique du trafic des applications de données :

a. *Mission critical* – Cette classe est dédiée au trafic des applications de données indispensable pour le fonctionnement de l'entreprise. Le marquage utilise le DSPC AF3x et le CoS 3.
b. *Transactional* – Ce sont les applications client/serveur. Les paquets sont marqués par le DSCP AF2x et le CoS 2.
c. *Bulk transfers* – Ce sont les applications de transfert des gros fichiers comme FTP et les opérations de sauvegarde. Cette classe utilise DSCP AF1x et CoS 1.

d. Best effort – Cette classe regroupe le reste du trafic. Les paquets sont marqués par le DSCP 0 et le CoS 0.
e. *Scavenger* – Cette classe permet d'identifier et limiter la bande passante pour le trafic non désirable comme par exemple le programme *Kazaa peer-to-peer*. Le marquage utilise le DSCP CS1 et le CoS 1.

Il faut noter que certaines classes peuvent ne pas être utiles pour un client. Dans ce cas, il n'est pas nécessaire de les définir.

Il faut noter que l'opérateur doit vérifier si la classification du trafic du client est conforme en utilisant *Acces Control List* (ACL). Les ACLs sont des listes de contrôle d'accès qui peuvent se baser sur les numéros de ports, les adresses IP, les adresses MAC, etc. Cette vérification est indispensable lorsque le fournisseur n'a pas confiance au client. En effet un client peut marquer tout son trafic avec une haute priorité.

3.1.2 Mapping

Étant donné que dans les L2VPNs, les paquets sont commutés au niveau 2, Il est mieux de baser la classification sur les entêtes de la couche 2 tel CoS de l'entête VLAN.

En utilisant le champ CoS on est limité à 7 classes de trafic. Le trafic est regroupé dans sept classes au total laissant la classe CS7 libre pour un usage futur. Le tableau suivant illustre un exemple de sept classes de service ainsi que les valeurs du champ CoS correspondantes.

Tableau 3.1

Les classes de service définies en utilisant CoS

Classes de service	**CoS**
Contrôle	6
Voix	5
Vidéo	4
Signalisation	3
Gold	2
Silver	1
Default	0

Afin de fournir la QdS exigée par le client dans le cœur du réseau MPLS, les routeurs PEs effectuent l'opération *mapping* CoS-EXP qui établit une correspondance entre les valeurs des champs de priorité CoS dans Ethernet et les valeurs des champs de priorité EXP dans MPLS.

Si le client utilise le champ DSCP pour marquer son trafic, il est nécessaire d'effectuer l'opération *mapping* DSCP-CoS afin de remarquer le trafic en utilisant le champ CoS de l'entête Ethernet.

La figure 3.2 illustre une association entre les classes DSCP et les classes CoS ainsi que les mécanismes d'ordonnancement qui leurs sont associés [37].

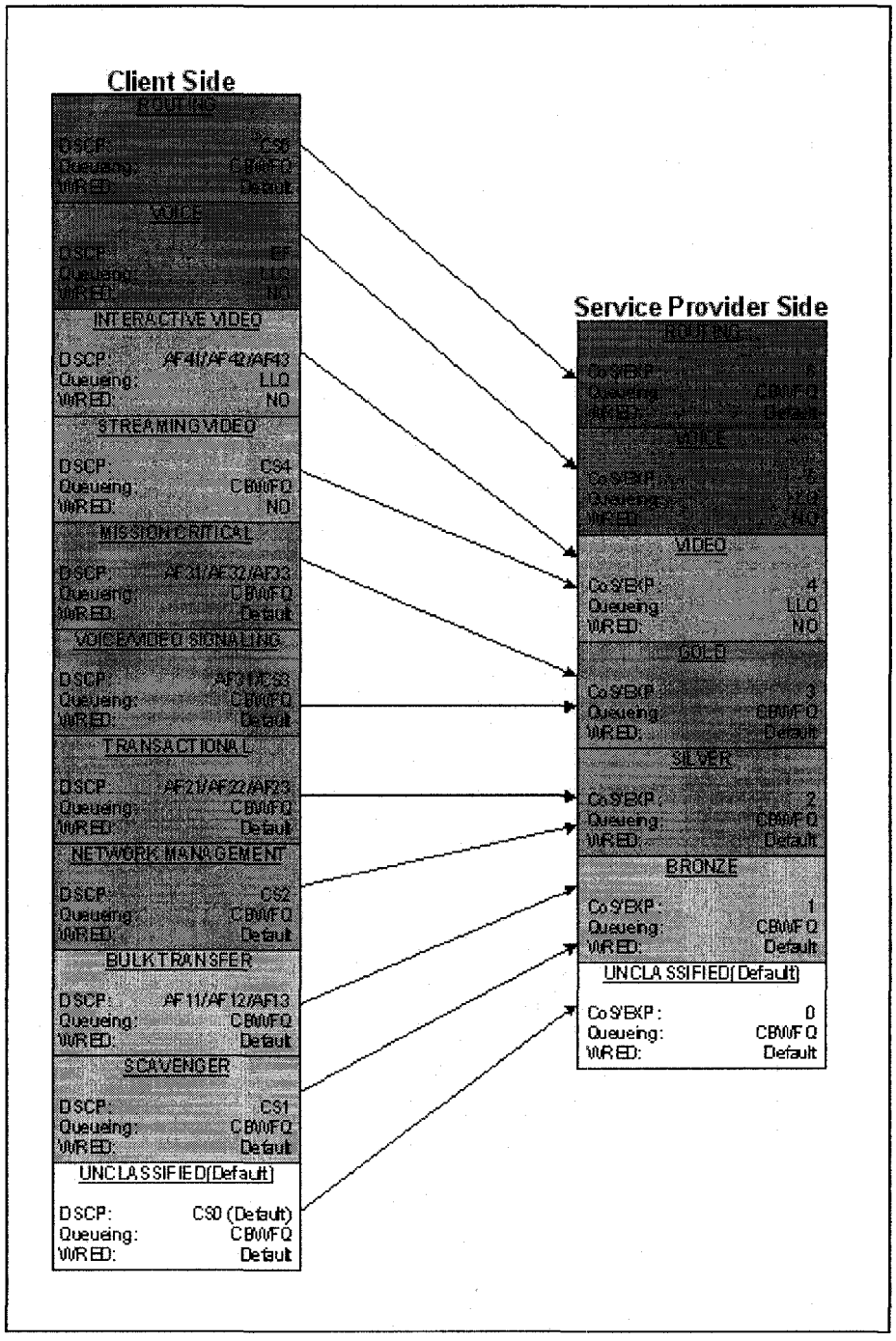


Figure 3.2 ***Correspondance des onze classes DSCP et les classes CoS [38].***

Une fois la classification et le marquage du trafic sont réalisés avec succès, l'étape suivante est l'ordonnancement. Cette opération est un mécanisme indispensable dans la mise en place de la QdS. Elle sert à placer chaque classe de trafic dans une file d'attente en lui réservant une partie de la bande passante.

3.1.3 Ordonnancement

On a vu comment le trafic est classifié et comment il est marqué. Suite à ces étapes, le classificateur doit mettre le trafic de chaque classe dans la file d'attente correspondante. Ensuite l'ordonnanceur gère les files d'attentes en cas de congestion ainsi que l'ordre de sortie des paquets. Le mécanisme d'ordonnancement traite le niveau de priorité de chaque classe et s'assure que la bande passante est bien partagée entre les différentes classes.

Il existe différent mécanismes d'ordonnancement. Dans [39] l'auteur présente les principaux algorithmes utilisés pour effectuer l'ordonnancement ainsi que d'autres plus spécifiques au constructeur Cisco Systems. Ci-dessous, une présentation brève de ces algorithmes :

a. *Priority Queueing* (PQ) – La file prioritaire est traitée en premier tant qu'elle contienne des paquets. Lorsque cette file devient vide, l'ordonnanceur passe à la file d'attente suivante. Si un paquet arrive à la file prioritaire, l'ordonnanceur revient le traiter et ensuite il continuera a traiter les files suivantes.
b. *Fair Queueing* (FQ) – Les paquets sont d'abord classifiés. Ensuite, ils sont mis dans les files d'attente qui correspondantes à leurs flots. L'ordonnanceur parcourt les files d'attente octets par octets. Un paquet n'est transmis que s'il a fini d'être parcouru.
c. *Weighted Fair Queueing* (WFQ) – Cet algorithme utilise le même principe que FQ avec la différence de pouvoir déterminer la priorité des paquets. Il distribue la bande passante entre les priorités selon leurs poids.

d. *Class-Based Queueing* (CBQ) – Cet algorithme permet de définir les types de trafic qui vont dans chacune des classes et les poids des files d'attentes.
e. *Low Latency Queue* (LLQ) – Il s'agit d'un type de file d'attente utilisé pour le trafic exigeant un délai minimal. La file LLQ constitue la file prioritaire et elle est servie en premier. Cependant, cette file dispose uniquement d'une partie de la bande passante prédéfinie. Une fois cette limite est atteinte, l'ordonnanceur passe à la file suivante.

3.1.4 Les services DiffServ

Dans le réseau MPLS, le champ EXP dans l'étiquette MPLS est utilisé pour identifier la classe de trafic des paquets afin qu'ils reçoivent le traitement adéquat.

La qualité de services peut être implémentée dans la dorsale en utilisant le modèle DiffServ qui est décrit dans [40]. DiffServ est utilisé pour implémenter la QdS dans un réseau IP/MPLS. Cela consiste à classifier et marquer le trafic à l'entrée du réseau. Chaque nœud dans le réseau se base sur le marquage du trafic pour traiter les paquets adéquatement.

L'application de DiffServ à MPLS peut s'effectuer en utilisant deux modèles proposés avec MPLS, soit l'approche *Label-Only-Inferred-PSC LSP* (L-LSP) et l'approche *EXP-Inferred-PSC LSP* (E-LSP) [41].

3.1.4.1 Description de l'approche E-LSP

Un LSP est appelé E-LSP quand la priorité du trafic transporté est déterminée en se basant sur la valeur du champ EXP dans l'étiquette MPLS. Un E-LSP transporte différentes classes de trafic qui partagent la bande passante du lien. Pour un E-LSP, chaque LSR figurant dans le chemin utilise le champ EXP pour effectuer

l'ordonnancement durant le processus de l'acheminement des paquets. Un E-LSP peut transporter un maximum de 8 types de trafics. La figure 3.3 montre un seul LSP qui transporte trois classes de trafic différentes entre deux PEs.

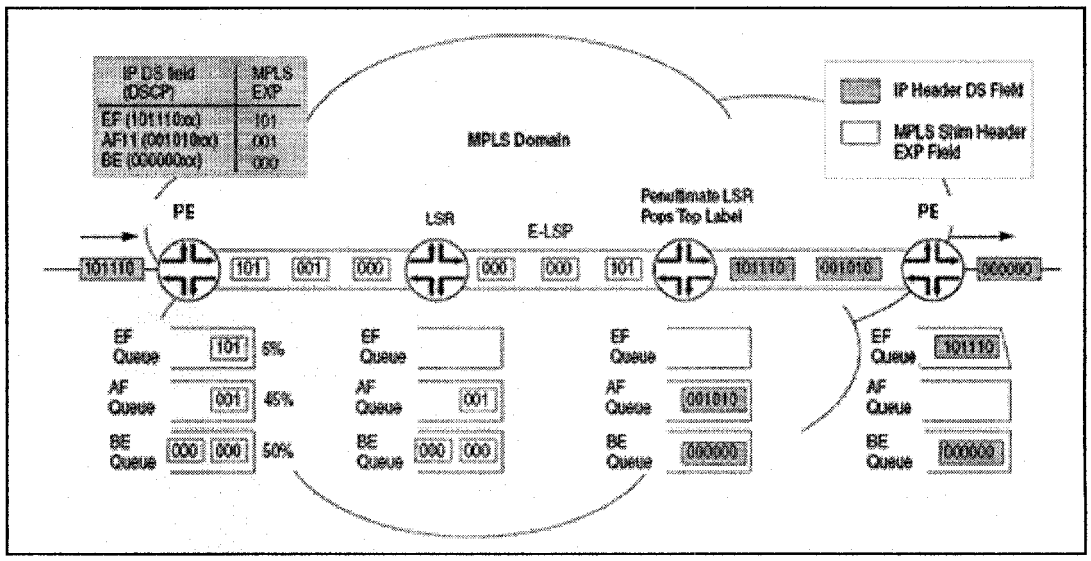


Figure 3.3 ***Le modèle E-LSP [41].***

Dans la figure 3.3, le PE est responsable du traitement des paquets IP qui veulent entrer dans le domaine MPLS. Il doit mettre le champ EXP à la bonne valeur. Ce mécanisme nécessite d'utiliser une équivalence entre la valeur du champ DSCP dans l'entête IP et la valeur du champ EXP dans l'entête MPLS. En effet, les six bits DSCP dans l'entête IP définissent un maximum de 64 valeurs, mais les trois bits EXP définissent un maximum de 8 valeurs. Le PE doit maintenir une table qui définit l'équivalence entre les classes de service DSCP et les huit valeurs du champ EXP. Le tableau suivant présente un exemple d'équivalence DSCP-EXP.

Tableau 3.2

Équivalence DSCP-EXP [41]

DSCP	EXP
CS6 (111000)	111
EF (101110)	101
AF1x	001
AF2x	010
AF3x	011
BE (000000)	000

3.1.4.2 Description de l'approche L-LSP

Dans l'approche L-LSP, les routeurs PEs établissent, entre eux, un LSP par classe de trafic. Pour chaque LSP, on réserve une bande passante en utilisant un protocole de signalisation avec contrainte comme par exemple CR-LDP ou RSVP-TE. Chaque LSP transporte un trafic appartenant à un *Forwarding Equivalency Class* (FEC) spécifique. Chaque équipement LSR, figurant dans le chemin du LSP, utilise l'étiquette MPLS qui lui est associée, pour déterminer le PHB qui doit s'appliquer au paquet durant le processus de son acheminement. Il faut noter que Le champ EXP n'est pas utilisé pour définir la priorité du trafic, mais il peut être utilisé pour déterminer la priorité de rejet du paquet *(drop precedence)*. La figure 3.4 montre trois classes EF, AF1 et BE qui sont acheminés par 3 L-LSPs différents.

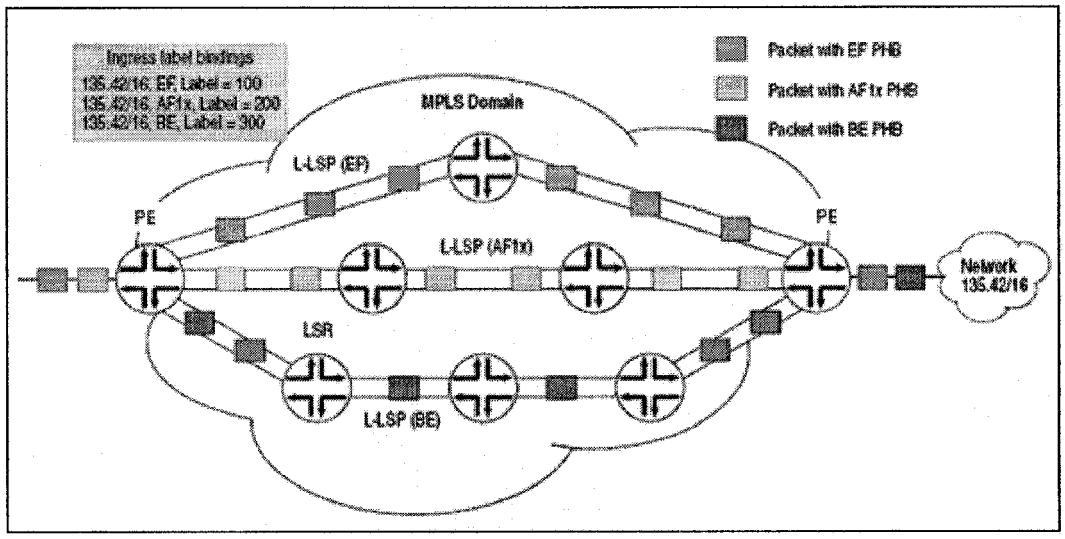


Figure 3.4 ***Le modèle L-LSP [41].***

Cette approche établit une équivalence entre les valeurs du champ DSCP et les LSP utilisés. Chaque LSP est associé à un type de service spécifié dans l'étiquette MPLS du LSP. Dans l'exemple ci-dessus, l'étiquette MPLS 100 est associée au L-LSP (EF), l'étiquette MPLS 200 est associée au L-LSP (AF11, AF12, AF13) et finalement l'étiquette MPLS 300 est associée au L-LSP(BE). Si un paquet, entrant dans l'interface 135.42/16 a une valeur DSCP égale à EF, l'équipement LSR périphérique lui assignera l'étiquette MPLS 100. Le paquet IP sera donc encapsulé dans le L-LSP(EF). Les équipements LSR figurant dans le chemin parcouru par le L-LSP (EF) doivent placer les paquets transportés par ce LSP dans une file d'attente EF.

De façon pratique, les deux approches peuvent donner environ les mêmes résultats. Cependant, l'approche L-LSP comporte quelques désavantages par rapport à l'approche E-LSP. En effet, le fait de ne pas utiliser de mécanisme d'ordonnancement avec L-LSP ne permet pas à cette approche de bénéficier d'une file de type LLQ pour le trafic temps réel. Deuxièmement, pour un chemin donné, un LSP par classe de service doit être établi avec l'approche L-LSP comparativement à un seul avec E-LSP. Le principal avantage

de l'approche L-LSP, par rapport à l'approche E-LSP, est la possibilité de pouvoir définir des priorités de rejet. L'approche E-LSP semble donc la solution la plus avantageuse.

Pour conclure, on a fait un survol de la QdS et de son implémentation dans les services L2VPN. Dans la section suivante, on illustre une problématique reliée au déploiement de la QdS dans les L2VPNs. Par la suite on présente la solution retenue.

3.2 Problématique et méthodologie

La mise en place de la QdS pour L2VPN et L3VPN dans le réseau MPLS est réalisée de la même manière. En effet, les deux solutions sont basées sur l'usage d'une dorsale IP/MPLS et sont implémentées de façon similaire (usage d'une pile d'étiquettes MPLS). La qualité de service est offerte en utilisant les différents mécanismes proposés avec MPLS.

Les sites des clients sont connectés au réseau de l'opérateur via les équipements CEs [4]. La figure 3.5 illustre un exemple de trois sites distants connectés à travers le réseau de l'opérateur. Les services L2VPN ont l'avantage d'utiliser des équipements CEs simples. La simplicité du dispositif CE vient du fait que les informations de routage du client n'ont pas besoin d'être partagées avec les équipements PEs et le trafic passe uniquement par la couche liaison.

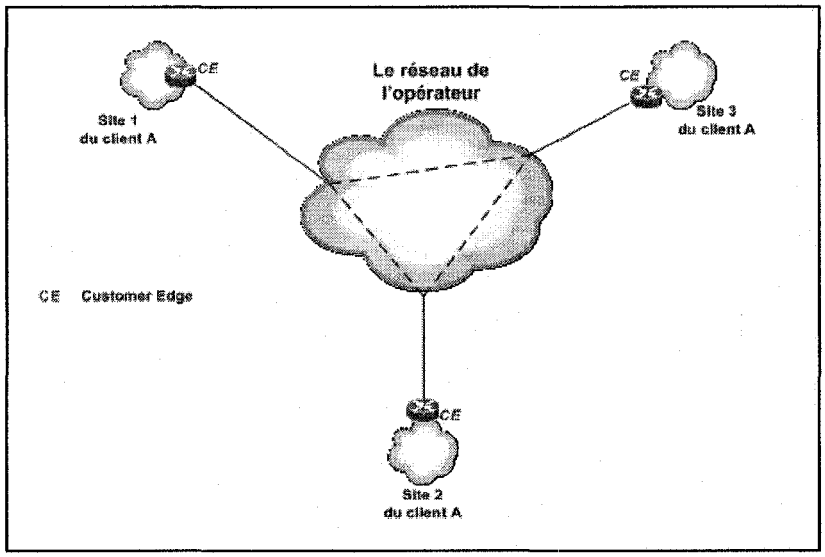


Figure 3.5 ***Équipement CE.***

L'équipement CE peut être utilisé pour connecter plusieurs RPVs. C'est le cas des clients qui utilisent différents RPVs dans leurs sites ou lorsque l'opérateur héberge un seul équipement CE pour plusieurs clients. Nous avons constaté le dernier cas dans le réseau MPLS de Bell Canada.

On a pu voir à travers notre étude que pour utiliser un commutateur comme équipement CE, il faut s'assurer que la qualité de service appliquée au trafic du client permet de répondre aux paramètres du contrat SLA et de maintenir une bonne performance du réseau. En effet, un commutateur est très limité dans l'implémentation de la qualité de service. Un commutateur dispose uniquement dans ses interfaces de sortie de 4 à 8 files d'attentes [37].

Suite à une série de scénarios de tests sur une plateforme réelle de test disponible chez Bell Canada nous avons aboutit à la problématique suivante : Les clients peuvent utiliser les mêmes classes de trafic dans leurs L2VPNs. Donc si plusieurs L2VPNs utilisent le même équipement CE, il faudra contrôler le trafic de chaque client afin de prendre

l'action nécessaire quand le trafic d'un client peut affecter le service offert aux autres clients. Malheureusement le mécanisme d'ordonnancement n'est pas capable de différencier entre les flux de trafic appartenant à différents L2VPNs. Il est capable seulement de distinguer les classes de trafic. Dans ce cas, existe-t-il une solution pour assurer la conformité du trafic de chaque L2VPN connecté à l'équipement CE ?

La solution à ce problème doit garantir les ressources nécessaires pour les différents L2VPNs. Elle doit prévenir le cas où un L2VPN envoie du trafic indésirable et cause la chute des autres L2VPNs. La solution doit limiter les ressources consommées par chaque L2VPN et garantir la disponibilité des ressources pour chaque client afin de remplir les obligations du contrat SLA pour chaque client.

Il faut noter que lorsque l'équipement CE est un routeur, ce problème n'existe pas. En effet, dans un routeur il est possible de séparer les files d'attentes qui traitent le trafic de chaque L2VPN en associant chaque RPV à une sous interface de l'interface globale qui achemine le trafic vers le réseau de l'opérateur [42].

Le présent mémoire propose une solution pour la mise en place de la qualité de service dans l'accès au réseau IP/MPLS de l'opérateur lorsque l'équipement CE est un simple commutateur. La proposition est un algorithme que nous avons appelé "*Dynamic Distributed Token Bucket*". L'algorithme a été développé afin de contrôler le flux de chaque RPV connecté au CE. La solution garantit pour L2VPN le débit définit dans son contrat. Ce débit peut varier s'il y a de la bande passante non utilisée. Le débit du trafic dans un L2VPN est limité à son seuil maximal uniquement lorsque la capacité de la file d'attente de sortie est complètement utilisée. DDTB est basé sur l'algorithme Token Bucket qui sert à vérifier si le trafic des clients est conforme à leurs contrats en évaluant le débit des flux entrants.

3.3 Illustration de la problématique

Cette section a pour but de mettre en évidence la problématique par des scénarios de tests. Ils ont été effectués dans la plateforme de tests qui est montée actuellement dans le laboratoire RETEM chez Bell Canada. La plateforme supporte la technologie AToM, qui est propriétaire à Cisco, pour offrir les services L2VPNs à travers un réseau MPLS.

Le réseau de test est une plate-forme de routeurs Cisco interconnectés par la technologie de niveau 2 Ethernet et implémentant MPLS (RFC 2547). Les PEs sont des routeurs *Cisco 7206*, les LSRs (Ps) sont des routeurs *Cisco 3725*. Les versions des IOS des LERs et des LSRs sont 12.2 et 12.3 respectivement. On observe le trafic à l'aide d'un renifleur *Agilent DNA*.

On commence par une description du scénario de test. Ensuite on présente la configuration du réseau de test. Enfin on analyse les résultats obtenus.

3.3.1 Scénario de test

La figure 3.6 illustre le cas d'utilisation d'un CE commutateur pour connecter plusieurs L2VPNs au réseau MPLS.

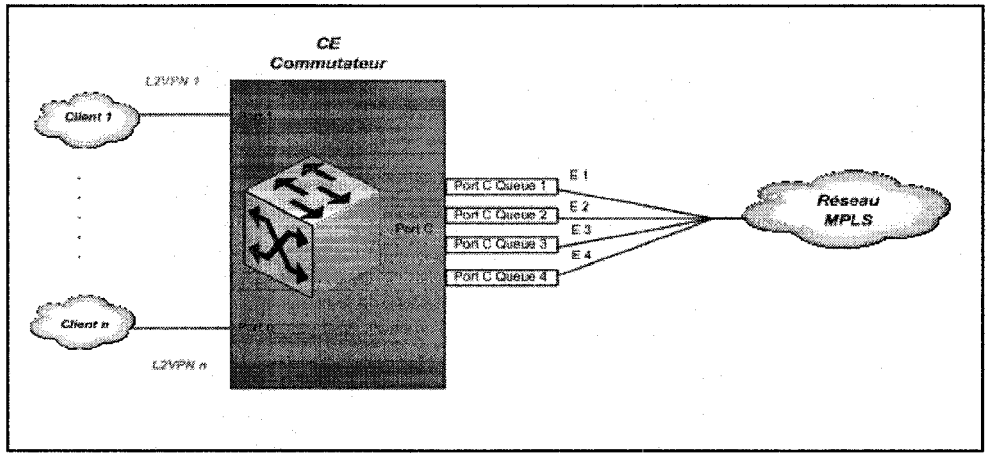


Figure 3.6 ***Plusieurs L2VPNs connectés au commutateur CE.***

Le trafic dans un L2VPN entre dans le commutateur CE et il sort par le port C. Chaque classe de trafic est servie par une file d'attente de sortie du port C. Comme dans la plupart des cas, les ports du commutateur disposent uniquement de quatre files d'attente pour servir le trafic sortant. La première file d'attente du port C, par exemple, a la capacité E1 pour servir les flux qui appartiennent aux classes de trafics qui lui sont assignées.

Supposant qu'il y a du trafic de la même classe dans les L2VPNs, alors les flux de trafic vont être servis par la même file d'attente. Ainsi, il y aura concurrence dans l'utilisation de la bande passante.

3.3.2 Configuration du réseau de test

Dans les tests qui ont été effectués, seulement deux L2VPNs ont été configurés. La figure 3.7 montre le cas de deux L2VPNs utilisant le même commutateur CE pour se connecter au réseau MPLS de l'opérateur.

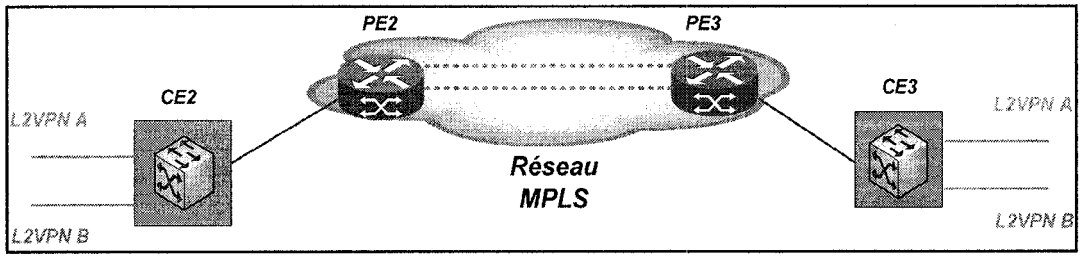


Figure 3.7 ***Deux L2VPNs connectés au même commutateur CE.***

Les flots des deux L2VPNs transitent par la même interface de sortie du commutateur CE2. Lorsqu'il y a congestion, les trafics des deux L2VPNs doivent partager la bande passante de l'interface de sortie. Si, par exemple, les deux L2VPNs contiennent le même type de trafic, il y aura concurrence pour partager la bande passante de la file d'attente dédiée à leurs classes de trafic. Donc il ne sera pas possible de contrôler le trafic de chaque L2VPN.

Les deux L2VPNs ont la même interface de sortie afin que leurs trafics partagent les mêmes files de sortie. La figure 3.8 illustre la configuration du réseau de test.

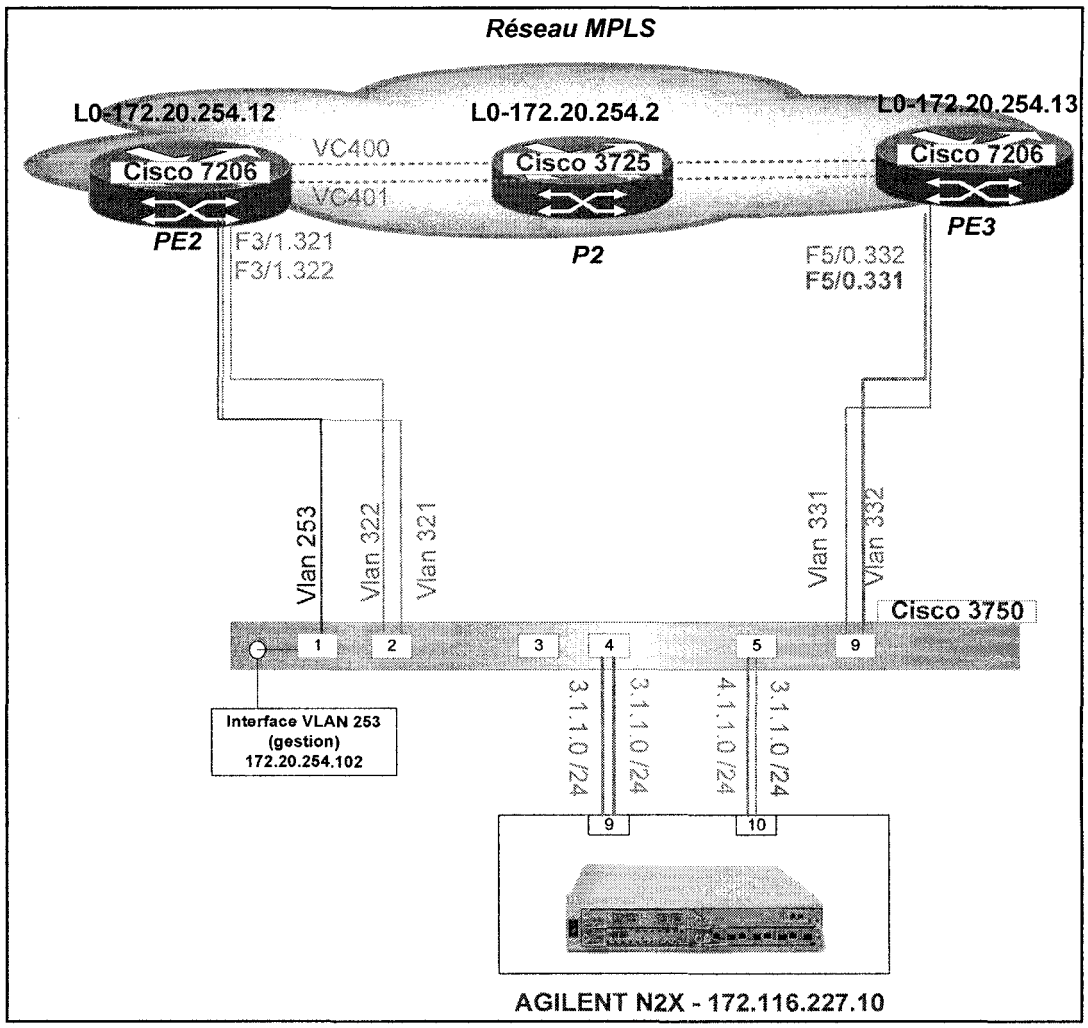


Figure 3.8 ***Montage de test.***

Le commutateur *Cisco 3750* est connecté au PE2 et au PE3. Il est donc à la fois CE2 et CE3. L'interface *VLAN 253* de gestion du commutateur est connectée au PE2. Les ports 2, 4, 5 et 9 sont configurés en mode *Trunk* afin de supporter plusieurs VLANs.

À l'aide du générateur de trafic *Agilent N2X Packets and Protocols*, on peut générer du trafic et le configurer selon nos besoins. Le port 9 de N2X (voir figure 3.8) simule la source de deux L2VPNs. Le premier L2VPN contient deux flux de trafic et le deuxième

contient un seul flux de trafic. Le tableau suivant illustre la configuration des flux dans les L2VPNs.

Tableau 3.3

Configuration de L2VPN et L3VPN

L2VPN A		**L2VPN B**
Flux 1	**Flux 2**	**Flux 3**
- Adresse IP source : 3.1.1.3 - Adresse IP destination : 3.1.1.17 - Port de destination : 10 - Masque sous réseau : 255.255.255. 0 - Marquage DSCP : 40 - VLAN ID : 322 - VC ID : 400	- Adresse IP source : 3.1.1.3 - Adresse IP destination : 3.1.1.17 - Port de destination :10 - Masque sous réseau : 255.255.255. 0 - Marquage DSCP : 8 - VLAN ID : 322 - VC ID : 400	- Adresse IP source : 3.1.1.4 - Adresse IP destination : 3.1.1.18 - Port de destination : 10 - Masque sous réseau : 255.255.255. 0 - Marquage DSCP : 40 - VLAN ID : 321 - VC ID : 401

Le premier flux du L2VPN A est marqué DSCP 40 et le deuxième flux est marqué DSCP 8. Les deux flux ont la même source (3.1.1.3/24) et la même destination (3.1.1.17/24). Les deux flux sont étiquetés par le VLAN ID 322 afin qu'ils appartiennent au L2VPN A dont le VC ID est 400 (voir figure 3.8).

Le flux du L2VPN B (Flux 3) est marqué aussi DSCP 40 et il est étiqueté par le VLAN ID 321. Il a comme source et comme destination 3.1.1.4/24 et 3.1.1.18/24 respectivement. Le VC ID du L2VPN B est 401.

Les ports 9 et 10 de N2X représentent la source et la destination du trafic des L2VPNs. On constate d'après la configuration du réseau, que la source et la destination du trafic

appartiennent au même segment : 3.1.1.0/24. Le trafic passe uniquement par le niveau 2 et il n'y a pas besoin d'effectuer un routage IP entre la source et la destination du trafic.

3.3.2.1 Configuration du commutateur Cisco 3750

Dans l'0, on présente toutes les commandes qui ont été utilisées pour configurer le commutateur Cisco 3750.

Afin de créer un goulot d'étranglement dans le réseau de test, la vitesse de l'interface 2 (voir figure 3.8) du commutateur a été limitée à 10Mbps. L'interface 2 connecte le commutateur au PE2. Elle est l'interface de sortie du trafic des deux L2VPNs.

Le trafic reçu par le commutateur est classifié, ensuite le classificateur doit mettre le trafic de chaque classe dans la file d'attente correspondante. Finalement l'ordonnanceur gère les files d'attentes en cas de gestion et l'ordre de sortie des paquets.

Le trafic sortant d'une interface du commutateur passe forcement par une des quatre files d'attente de sortie (*Egress Queues*). La première file d'attente peut être configurée comme file prioritaire (*Expedit Queue)*. La file prioritaire est servie avant les autres files tant qu'elle n'est pas vide.

Le type d'ordonnancement utilisé est *SRR Shaping and Sharing*. Dans le mode *Sharing*, les files de sortie partagent la bande passante selon les poids qui leurs sont attribués. Chaque file d'attente a une bande passante garantie, mais elle n'est pas limitée à cette dernière. Dans le mode *Shaping*, chaque file d'attente est limitée à la bande passante qui lui est garantie, même si le lien est non utilisé.

3.3.2.2 Configuration des files d'attente de l'interface de sortie 2

La figure 3.9 illustre l'association des différents flux, provenant des deux clients, avec les files d'attente du port de sortie.

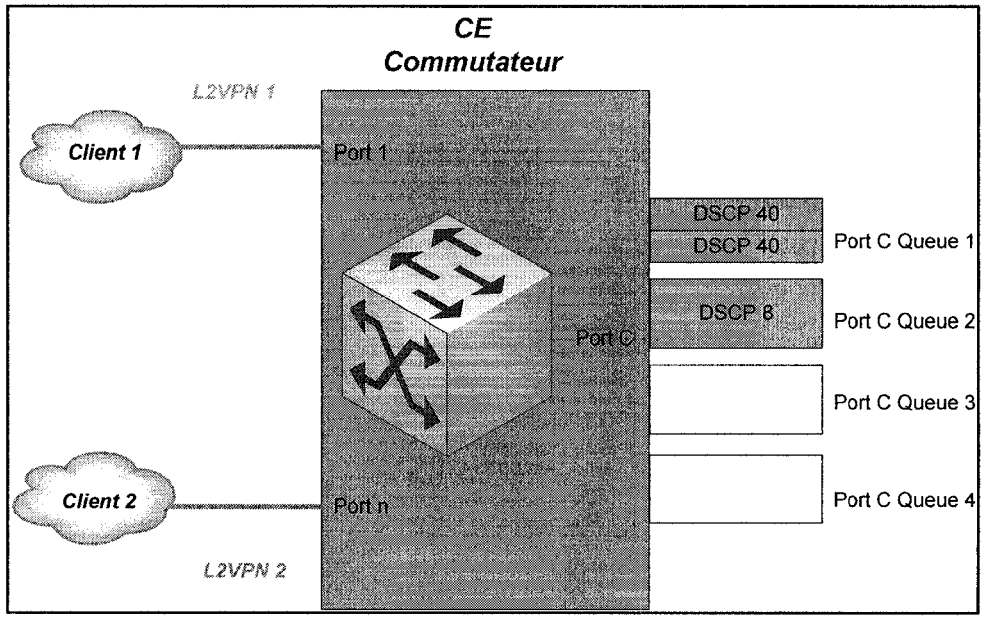


Figure 3.9 ***Association des flux aux files d'attente de sortie.***

Le premier flux de trafic (DSCP 40) du L2VPN A a été associé à la première file d'attente de l'interface de sortie. Le deuxième flux (DSCP 8) du L2VPN A a été associé à la deuxième file d'attente. Le flux du L2VPN B (DSCP 40) est associé aussi à la première file d'attente.

La première file d'attente a été configurée en mode *Shaping*. Les autres files d'attente ont été configurées en mode *Sharing*. La vitesse de l'interface de sortie a été limitée à 10Mbps afin de créer la congestion dans le lien reliant le commutateur au routeur PE2. La taille de la première file d'attente a été configurée à 3.38Mbps. Étant donné que les

trois autres files d'attente sont configurées en mode *Sharing*, la deuxième file d'attente utilisera la bande passante non utilisée par les deux autres files d'attente.

3.3.3 Résultats obtenus

Dans cette section, les résultats obtenus sont présentés et analysés. Deux cas ont été considérés. Dans un premier temps, la QdS n'a pas été activée dans le CE. Ensuite on a activé la QdS afin de comprendre le comportement des flux des L2VPNs dans les deux cas.

Lorsque la qualité de service n'est pas activée dans le commutateur, nous avons obtenus les résultats suivants.

Tableau 3.4

Débits configurés et reçus si la QdS n'est pas activée

	Trafic envoyé (Mbps)			**Trafic reçu (Mbps)**		
	L2VPN A		L2VPN B	L2VPN A		L2VPN B
	Flux 1	Flux 2	Flux 3	Flux 1	Flux 2	Flux 3
Test 1	2	7	1	2	6.81	1
Test 2	4.7	7	1.15	3,78	5.41	0.67

On constate dans le premier test que les flux de la même classe de service (Flux 1/Flux 3) n'ont pas subi une dégradation. Dans le deuxième test, le débit total des deux flux de la même classe de service dépassait la limite de la première file d'attente. Les deux flux ont subit une dégradation importante de leurs débits.

D'autre part le trafic du Flux 2 du L2VPN A a subit dans le premier test une petite dégradation, mais c'est normal car son débit est légèrement plus élevé que la limite

maximale qu'il peut avoir (à peu prêt deux tiers de la bande passante du lien). Dans le deuxième test on voit bien que le débit a subit une dégradation. Bref, lorsque la qualité de service n'est pas activée, il n'y a aucune garantie de bande passante pour les flux L2VPN.

Le tableau suivant illustre les résultats des tests lorsque la qualité de service est activée dans le commutateur.

Tableau 3.5

Débits configurés et reçus si la QdS est activée

	Trafic envoyé (Mbps)			**Trafic reçu (Mbps)**		
	L2VPN A		L2VPN B	L2VPN A		L2VPN B
	Flux 1	Flux 2	Flux 3	Flux 1	Flux 2	Flux 3
Test 1	2	7	1	2	6.81	1
Test 2	4.7	7	1.15	2.93	6.42	0.43

On constate dans le deuxième test que le Flux 2 a bénéficié de la bande passante minimale réservée pour les trois autres files d'attente. Il y a eu concurrence dans le partage de la bande passante entre les deux flux de même classe de trafic (Flux 1 et Flux 3).

La qualité de service a permit de garantir une bande passante maximale pour la première file d'attente qui a été configurée en mode *Shaping*. De même, la configuration de la qualité de service a permis de réserver une bande passante minimale pour les trois autres files d'attente qui ont été configurées en mode *Sharing*.

Lorsque la première file d'attente est utilisée par les deux L2VPNs, il n'est pas possible de contrôler leurs flux. Afin de mettre en évidence la façon dont les deux flux partageaient la bande passante de la file d'attente de sortie, les mêmes tests ont été

effectués en fixant le débit du Flux 3 dans le L2VPN B à 1Mbps et en variant le débit du Flux 1 dans le L2VPN A.de 1 à 10 Mbps.

Tableau 3.6

Variation du débit envoyé pour le flux 1 du L2VPN A

Trafic envoyé (Mbps)			**Trafic reçu (Mbps)**		
L2VPN A		L2VPN B	L2VPN A		L2VPN B
Flux 1	Flux 2	Flux 3	Flux 1	Flux 2	Flux 3
1	7	1	1	7	1
2	7	1	2	6,81	1
3	7	1	2,52	6,42	0,86
4	7	1	2,68	6,42	0,7
5	7	1	2,81	6,42	0,57
6	7	1	2,9	6,42	0,5
7	7	1	3	6,42	0,42
8	7	1	3,02	6,42	0,42
9	7	1	3,03	6,42	0,37
10	7	1	3,05	6,42	0,29

Afin de comprendre comment la concurrence entre les deux L2VPNs prend lieu dans la file d'attente commune, nous avons calculé le pourcentage de chaque flux par rapport au flux total entrant et sortant de la file d'attente. Le tableau suivant illustre les résultats des calculs.

Tableau 3.7

Pourcentages des flux des L2VPNs

Pourcentage du Flux 1 envoyé dans L2VPN A	**Pourcentage du Flux 1 reçu dans L2VPN A**	**Pourcentage du Flux 3 envoyé dans L2VPN B**	**Pourcentage du Flux 3 reçu dans L2VPN B**
50%	50%	50%	50%
66,66%	66,66%	33,33%	33,33%
74,55%	75%	25,44%	25%
79,28%	80%	20,71%	20%
83,13%	83,33%	16,86%	16,66%
85,79%	85,71%	14,79%	14,28%
93,19%	87,5%	12,42%	12,5%
89,34%	88,88%	12,42%	11,11%
98,52%	90%	10,35%	10%
91,42%	90,90%	8,58%	9,09%

La figure 3.10 illustre graphiquement les résultats ci-dessus.

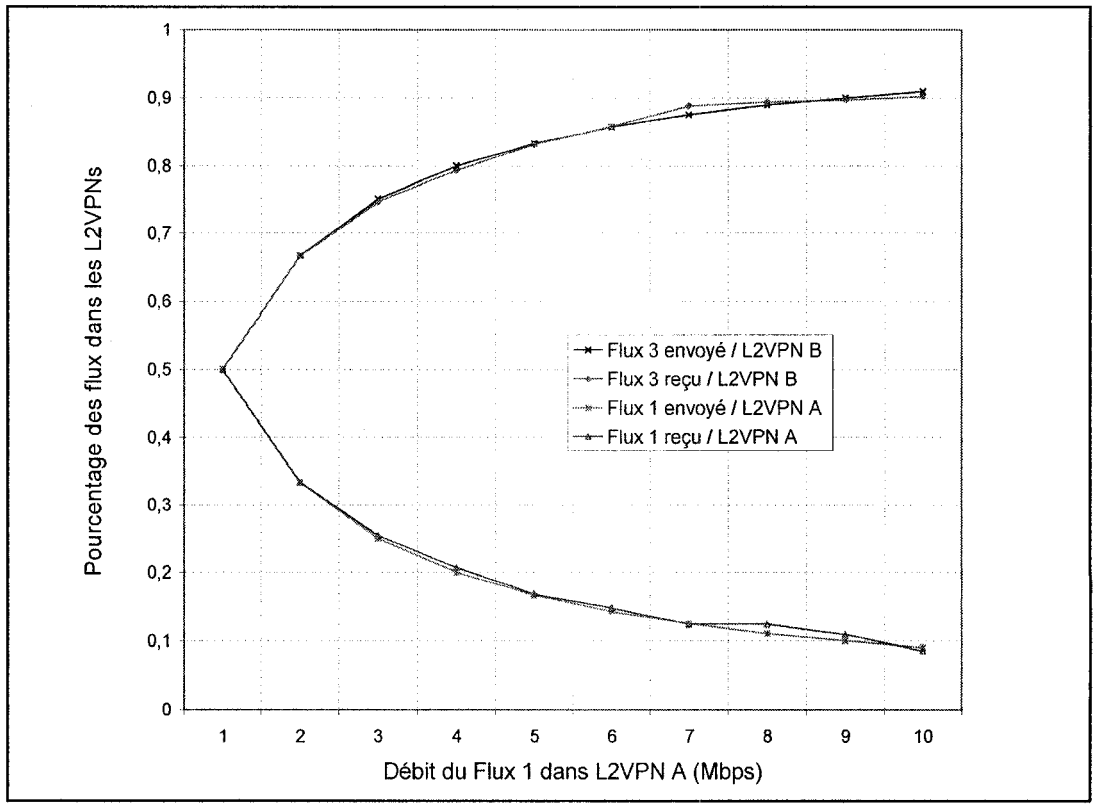


Figure 3.10 ***Pourcentage des flux envoyés et reçus dans les deux L2VPNs.***

Les deux courbes qui correspondent aux pourcentages du flux envoyé et du flux reçu dans le L2VPN B (Flux 3) sont pratiquement identiques. La légère différence entre les deux courbes est due au générateur N2X qui n'affiche pas une valeur stable pour le trafic reçu. Les valeurs utilisées ont été obtenues en faisant une moyenne sur les valeurs affichées par le générateur N2X.

De même, Les deux courbes qui correspondent aux pourcentages du flux envoyé et du flux reçu dans le L2VPN A (Flux 1) sont pratiquement identiques.

Le Flux 1 et le Flux 3 étant les seules flux entrant dans la file d'attente, la somme des pourcentages des deux flux est égale à 1.

On peut conclure que si les deux flux sont servis par la même file d'attente, alors ils partageront la bande passante de la file d'attente selon la quantité de leurs trafics entrants. Si aucun control n'est effectué sur les flux de trafic, la file d'attente sera congestionnée et les flux subiront une dégradation catastrophique.

3.4 Proposition

On suppose que la file d'attente partagée entre les flux L2VPN est de type *sharing*. Le but est d'optimiser l'utilisation des ressources en redistribuant la bande passante non utilisée. On suppose aussi que les paquets qui rentrent dans l'équipement CE sortent uniquement par l'interface qui connecte le CE au PE. Un client est connecté au CE par une seule interface.

Une solution possible à ce problème peut être l'utilisation d'une file d'attente pour chaque trafic L2VPN et par classe de service. Malheureusement cette technique n'est pas possible lorsque l'équipement CE est un commutateur. En effet, les interfaces d'un commutateur disposent uniquement de quatre ou huit files d'attentes physiques. Il n'est pas possible d'avoir des files d'attente virtuelles. Donc si les flux de deux L2VPNs passent par la même interface physique, ils partageront les mêmes files d'attente et il y aura une concurrence dans l'utilisation de la bande passante.

Un routeur permet d'utiliser des files d'attentes virtuelles. Et dans le cas de L2VPN on peut associer une file d'attente logique par RPV. Ainsi il n y aura pas de concurrence entre les différents L2VPNs. En effet, il est possible de configurer dans une même interface globale plusieurs sous-interfaces. Sur chaque sous-interface on peut appliquer des politiques de service propres à chaque client L2VPN.

Notre objectif est d'appliquer une méthode de *Policing* capable de redistribuer les ressources disponibles en terme de bande passante entre les clients qui veulent transmettre à un débit supérieur au *Committed Information Rate* (CIR). La solution doit garantir à chaque client L2VPN le débit défini dans son contrat de trafic (SLA). Si, par exemple, un client envoie un flux de trafic qui dépasse son CIR et peut conduire à une congestion et par conséquent une éventuelle dégradation des flux des autres clients.

La solution classique au problème de contrôle des flux est l'implémentation du *Policing* sur le trafic entrant. L'algorithme le plus largement utilisé par les vendeurs dans l'implémentation du *Policing* est le *Token Bucket* (TB) [40].

Le mécanisme *Policing* a pour but d'éviter la congestion à la sortie de l'équipement CE en limitant le débit de chaque trafic au CIR selon le contrat établie. En utilisant ce mécanisme, on perd un grand avantage de l'ordonnancement qui est la redistribution de la bande passante non utilisée d'une classe de trafic aux autres classes de trafic. Si le *Policing* est activé et qu'une classe de trafic a besoin de plus de bande passante, alors, même si les ressources sont disponibles la bande passante résiduelle ne lui sera pas attribuée.

Notre proposition consiste à améliorer l'algorithme décrit dans [43]. Ce dernier propose un modèle dynamique de l'algorithme TB. Il permet d'allouer dynamiquement les ressources de la file d'attente qu'utilisent différents flux de trafic. Cependant, ce modèle ne peut pas garantir aux clients leurs débits CIRs.

Notre contribution consiste à contrôler la distribution des ressources non utilisées de la file d'attente pour garantir aux clients leurs débits CIRs.

Ce mémoire propose l'algorithme *Dynamic Distributed Token Bucket* (DDTB) qui se base sur le modèle dynamique de TB. L'algorithme permet de contrôler et d'assurer la

protection des flux de trafic provenant de différentes sources. Il permet aussi d'optimiser l'utilisation de la file d'attente qui est partagée par les flux de la même classe de trafic.

Le reste de ce chapitre est organisé comme suit : une introduction sur l'algorithme TB est présentée. Ensuite on présente un modèle dynamique qui existe dans la littérature. Par la suite on propose le modèle DDTB qui constitue la contribution du mémoire au modèle dynamique de TB afin de l'améliorer pour répondre à nos besoins et finalement on conclue le chapitre par les résultats des simulations.

3.4.1 Presentation de l'algorithme Token Bucket

Le *Policing* est un mécanisme utilisé dans le déploiement de la qualité de service. Il vérifie si le profile du trafic du client respecte le contrat. Le trafic qui dépasse un débit préalablement fixé est rejeté ou remarqué pour un éventuel rejet plus tard en cas de congestion. Il est appliqué sur le trafic entrant une interface afin de s'assurer que le trafic est conforme au contrat du trafic [37]. La figure 3.11 illustre le fonctionnement de l'algorithme TB.

Les principaux paramètres du *Policing* définis dans le contrat de trafic sont :

a. *Committed Burst size* (Bc) exprime la rafale (en bits) qui peut être transmise pendant l'intervalle de temps Tc.
b. *Excess Burst size* (Be) exprime la rafale (en bits) qui peut être transmise lorsque le Bc est atteint.
c. *Committed Information Rate* (CIR) est le débit (en bits/s) défini dans le contrat pour le trafic conforme (ne dépassant pas Bc).

Lorsque le paramètre Be n'est pas utilisé, le modèle TB est dit simple. Si Be est utilisé, on dit que le modèle TB est double. Dans cette section on présente le modèle général de TB qui est le modèle double, mais dans notre proposition, on utilisera le modèle simple.

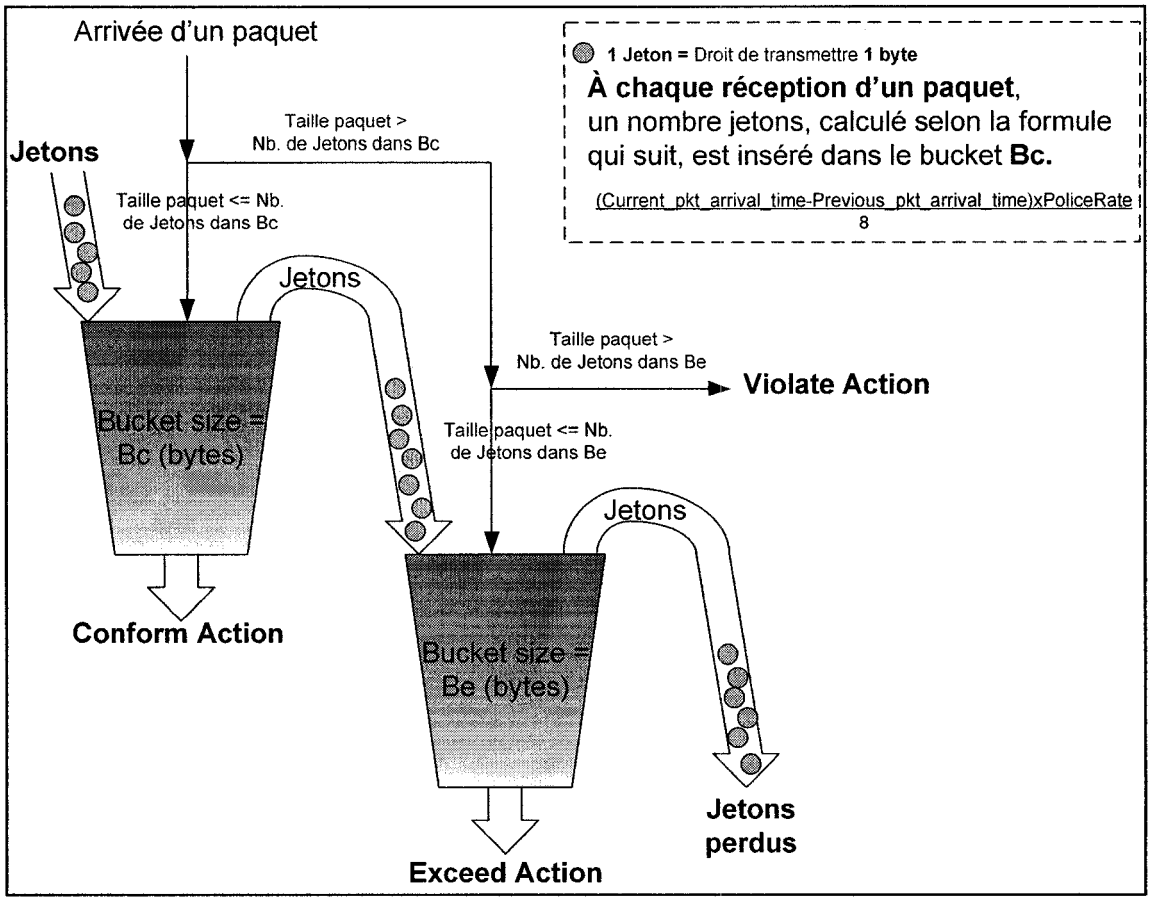


Figure 3.11 ***L'algorithme Token Bucket [39].***

Un jeton est équivalent à un octet. Le seau à jetons Bc (*bucket*) est rempli de jetons (*token*) à chaque fois qu'un paquet est reçu. S'il reste des jetons qui n'ont pas été utilisés au moment de remplissage, ces jetons excédentaires iront dans le seau à jetons Be. Si le seau à jetons Be est complètement rempli, alors les jetons arrivant seront perdus.

Le calcul du nombre de jetons qui sont ajoutés utilise les paramètres suivants :

a. CPAT (*Current Paket Arrival Time*) : le temps d'arrivé du paquet courant.
b. PPAT (*Previous Paket Arrival Time*) : le temps d'arrivé du paquet précédent.

Soit u(t) le taux de remplissage du seau à jetons à l'instant t. Le calcul de ce paramètre est donné par l'équation (3.1) [37].

$$u(t = CPAT) = \frac{(CPAT - PPAT) \times CIR}{8} \tag{3.1}$$

CPAT et PPAT sont en secondes. Le taux de *policing* CIR est en bits par seconde. Pour obtenir le nombre de jetons, il faut diviser par huit.

Dans [37] on trouve aussi la formule (3.2) qui relie le paramètre Bc au CIR.

$$Bc = \frac{CIR}{32} \tag{3.2}$$

L'équation (3.2) peut être interprétée par la formule (3.1). En effet, si on remplace u(t) dans (3.1) par Bc, alors CIR est calculé comme étant le débit d'un flux de trafic équivalant à Bc pendant un temps (CPAT - PPAT) égale à un quart de seconde.

Lorsqu'un paquet arrive, le nombre de jetons équivalents à la taille du paquet sont retirés du seau à jetons Bc ou Be. Comme le montre la figure 3.11, lorsqu'un paquet arrive, il faut vérifier si sa taille ne dépasse pas le nombre de jetons dans le seau à jetons Bc. Si ce n'est pas le cas, alors il est considéré conforme et il subira l'action *Conform Action* qui consiste à accepter le paquet et le transmettre. Cependant, si la taille du paquet excède le nombre de jetons disponible dans le seau à jetons Bc, il faudra vérifier si le nombre de jetons dans le seau à jetons Be est suffisant pour accepter le paquet. Dans ce cas, le paquet est considéré excédentaire et il subira l'action *Exceed Action* qui consiste soit à remarquer le paquet avec une priorité plus basse et le transmettre ou bien transmettre le paquet sans aucune action.

Comme il est expliqué dans [39], si la taille du paquet dépasse le nombre de jetons dans le seau à jetons Be, alors il a violé le contrat de trafic et il subira l'action *Violate Action* qui consiste soit à le rejeter ou le remarquer ou bien de le transmettre. Le type d'action à prendre en cas d'excès ou de violation dépendra du fournisseur de service.

3.4.2 Le modèle dynamique du Token Bucket

Dans [43] les auteurs proposent une modélisation mathématique d'un seau à jetons en se basant sur la théorie du routage dynamique [44]. Les flux de trafic proviennent de différentes sources. Chaque flux de trafic est contrôlé par un seau à jetons. Les flux de trafic acceptés sont dirigés vers une file d'attente partagée. Les ressources résiduelles de la file d'attente sont allouées dynamiquement afin d'optimiser son utilisation. La figure 3.12 illustre le modèle dynamique de TB.

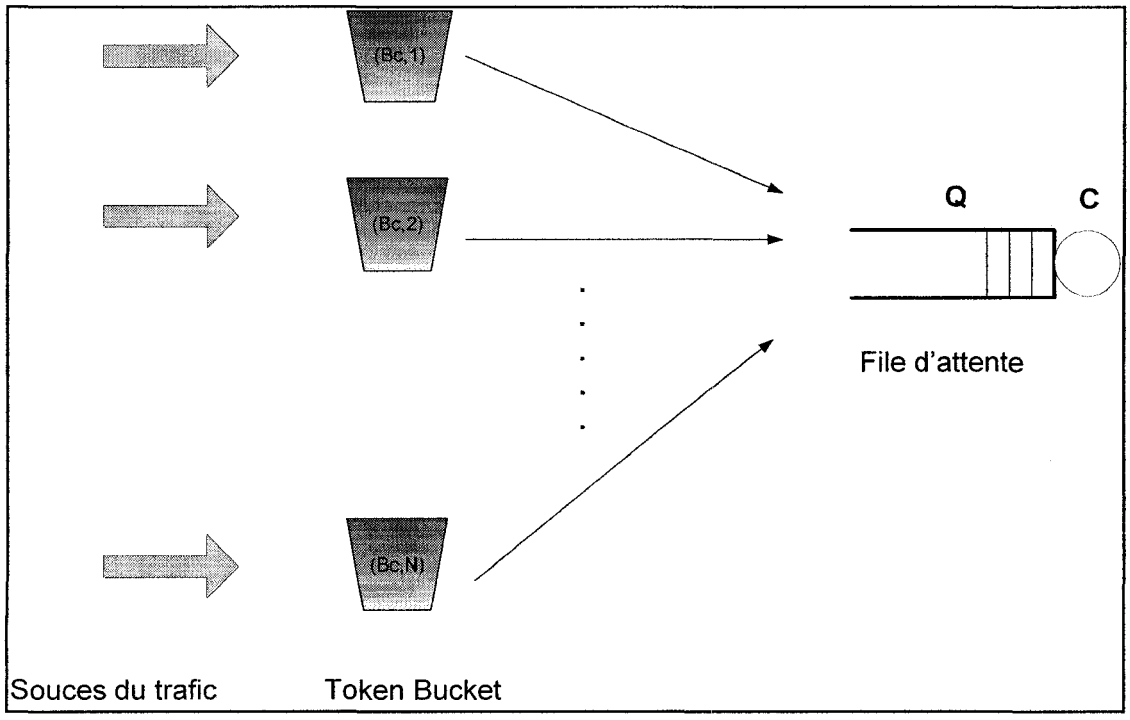


Figure 3.12 ***le modèle dynamique de TB.***

Dans la figure 3.12, le seau à jetons associé au $i^{ème}$ client est noté par (Bc,i) où Bc est la taille du seau à jetons en octets. Les flux des différentes sources utilisent la même file d'attente. La file d'attente est caractérisée par une taille Q et un taux de service C.
Le tableau suivant illustre les paramètres utilisés dans le modèle dynamique de TB proposé dans [43].

Tableau 3.8

Paramètres de l'algorithme du modèle dynamique de TB

Paramètre	**Description**
$V(t_k)$	Le trafic arrivant à l'instant t_k.
$G(t_k)$	Le trafic conforme à l'instant t_k.
$R(t_k)$	Le trafic rejeté à l'instant t_k.
$Q(t_k)$	La quantité de trafic dans la file d'attente.
$q(t_k)$	l'état de la file d'attente à l'instant t_k.
Bc	La taille totale du seau à jeton.
$\rho(t_k)$	La taille du seau à jetons à l'instant t_k.
$u(t_k)$	Le taux d'arrivé des jetons dans l'intervalle de temps $[t_{k-1},t_k)$.
$X(t_k)$	La partie du trafic reçu durant la période $[t_k, t_{k-1})$ qui n'est pas encore servi à l'instant t_k.
$Y(t_k)$	L'espace vide dans la file d'attente à l'instant t_k.
$F(t_k)$	L'espace non utilisé de la file d'attente par le flux de trafic accepté des clients à l'instant t_k.
$\Theta_i(t_k)$	Le nombre de jetons ajoutés dans le seau à jetons à l'instant t_k.

Il est important de noter que dans ce modèle, les jetons à ajouter dans les seaux à jetons sont générés d'une manière différente que dans l'algorithme TB. Le nombre de jetons à

ajouter à l'instant t_k est donné par le paramètre $\Theta_i(t_k)$. le reste de cette section présente l'algorithme qui calcule $\Theta_i(t_k)$ en fonction des autres paramètres.

On a choisi d'utiliser le même modèle de trafic adopté par [43]. Le trafic reçu $V(t_k)$ est observé à l'instant t_k pour k=1, 2, 3, ... K. La taille de la trame reçue est donnée par le paramètre $V(t_k)$. L'intervalle de temps $[t_k, t_{k-1})$ a une durée fixe et petite durant laquelle uniquement un seul paquet peut arriver. La figure 3.13 illustre les paramètres du modèle dynamique de TB.

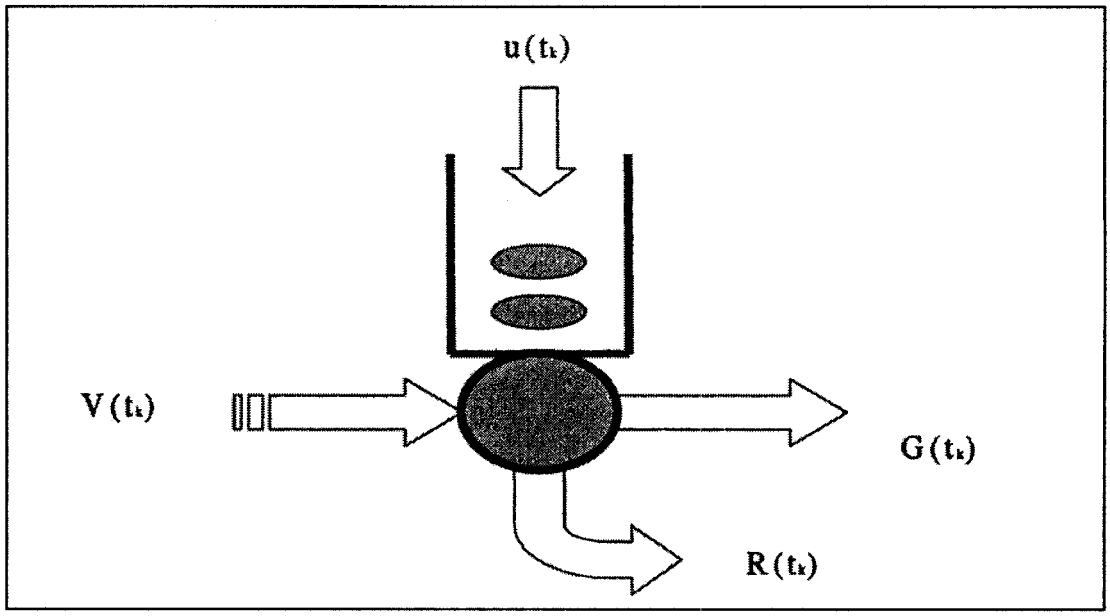


Figure 3.13 ***Le modèle du seau à jetons [43].***

Dans le reste de ce chapitre, les symboles suivants sont utilisés :

a. Le minimum entre les deux variables a et b est noté par $\{a \wedge b\}$
b. Le maximum entre les deux variables a et b est noté par $\{a \vee b\}$
c. La fonction I retourne un booléen quand son argument est vrai ou faux :
   - I(s) = 1, si s est vrai
   - I(s) = 0, si s est faux

Le trafic rejeté est calculé par l'équation suivante :

$$R(t_k) = V(t_k) - G(t_k) \tag{3.3}$$

Le trafic conforme qui est accepté par le seau à jetons peut être calculé en utilisant l'équation (3.4)

$$G(t_k) = V(t_k) I\{V(t_k) \leq \rho(t_{k-1}) + \{u(t_k) \wedge [Bc - \rho(t_{k-1})]\}) \tag{3.4}$$

L'argument de la fonction I dans l'équation (3.4) est la condition que la taille du paquet reçu ne soit pas plus large que le nombre de jetons dans le seau à jetons. Le nombre des jetons dans le seau à jetons à l'instant t_k peut être calculé en utilisant l'équation d'équilibre (3.5).

$$\rho(t_k) = \rho(t_{k-1}) + \{u(t_k) \wedge [Bc - \rho(t_{k-1})]\} - G(t_k) \tag{3.5}$$

Le premier terme ($\rho(t_{k-1})$) dans l'équation représente le nombre de jetons qui n'ont pas été utilisés à l'instant t_{k-1}. Le deuxième terme représente le nombre de jetons ajoutés dans le seau à jetons pendant la période $[t_k, t_{k-1})$. Le dernier terme représente le nombre de jetons consommés à l'instant t_k par le trafic conforme.

Afin d'aboutir à un modèle dynamique, il faut être capable de mesurer à chaque instant t_k l'état de la file d'attente. Soit q(t) l'état de la file d'attente à l'instant t. À l'instant t_k le paramètre $q(t_k)$ peut être déterminé par l'équation (3.6).

$$q(t_k) = X(t_k) + \left\{ \left[\sum_{i=1}^{n} G_i(t_k) \right] \wedge Y(t_k) \right\} \tag{3.6}$$

$X(t_k)$ est la partie du trafic reçu durant la période $[t_k, t_{k-1})$ qui n'est pas encore servi à l'instant t_k. Le terme $Y(t_k)$ représente l'espace vide dans la file d'attente à l'instant t_k. Le

deuxième terme dans l'équation (3.6) représente le trafic transmis à la file d'attente durant la même période $[t_k, t_{k-1})$. Il est obtenu par le minimum entre la somme des trafics acceptés des différentes sources et l'espace vide dans la file d'attente à l'instant t_k. Les équations (3.7) et (3.8) calculent $X(t_k)$ et $Y(t_k)$ respectivement.

$$X(t_k) = \{[q(t_{k-1}) - C\tau] \vee 0\} \tag{3.7}$$

$$Y(t_k) = Q - X(t_k) \tag{3.8}$$

Où τ indique la durée de la période $[t_k, t_{k-1})$.

Afin d'optimiser l'utilisation de la file d'attente, l'espace non utilisé par le trafic reçu par la file d'attente à l'instant t, sera distribué entre les TBs associés aux sources de trafic. L'espace non utilisé de la file d'attente par le trafic accepté des clients à l'instant t_k est noté $F(t_k)$. Ce paramètre est donné par l'équation (3.9)

$$F(t_k) = Q - q(t_k) \tag{3.9}$$

À l'instant t_k, l'espace résiduel de la file d'attente $F(t_k)$ doit être distribué aux seaux à jetons proportionnellement aux demandes des sources de trafic (tailles des paquets reçus). On définit le paramètre $\Theta_i(t_k)$ comme étant la part du $i^{ème}$ client de l'espace résiduel $F(t_k)$ à l'instant t_k. l'équation (3.10) calcule $\Theta_i(t_k)$.

$$\Theta_i(t_k) = \frac{V(t_k) \cdot F(t_k)}{\sum_{i=1}^{n} V(t_k)} \tag{3.10}$$

Pour conclure, le modèle dynamique de TB, permet de générer dynamiquement des jetons selon l'espace libre de la file d'attente et la demande des clients. La distribution

des jetons est faite selon le volume de trafic généré. Cela conduit certainement à la monopolisation de la file d'attente par les flux qui génèrent un volume important. Donc ce modèle ne peut pas garantir aux clients leurs débits CIRs. En se basant sur cette remarque nous avons développé l'algorithme DDTB qui constitue l'adaptation du modèle dynamique de l'algorithme TB à notre problématique.

3.4.3 Dynamic Distributed Token Bucket (DDTB)

L'algorithme DDTB a pour but de contrôler les flux de trafic provenant de différentes sources en prenant en considération la bande passante non utilisée. Le but est de maximiser les débits des flux acceptés des clients tout en garantissant à chaque client un débit minimum égal à son CIR.

Dans DDTB, les paquets conformes aux contrats, sont acceptés et envoyés à la file d'attente. Les paquets qui ne sont pas conformes peuvent aussi être acceptés à condition que la file d'attente puisse les accepter et que les bits des flux de trafic des autres clients ne soient pas dégradés.

Cette solution mesure les ressources non utilisées par les différents flux de trafic et effectue leurs redistributions sur les clients qui veulent transmettre à un débit supérieur au CIR. Le tableau suivant illustre les paramètres utilisés dans l'élaboration de l'algorithme DDTB.

Tableau 3.9

Paramètres de l'algorithme DDTB

Paramètre	Description
tokens_available	Le nombre des jetons disponibles dans le seau à jetons
pk_len	La taille du paquet reçu.
CIR_i	Le débit CIR pour le $i^{ème}$ client.
throughput_received_i	Le débit du flux de trafic reçu du $i^{ème}$ client.
throughput_accepted_i	Le débit du flux de trafic accepté pour le $i^{ème}$ client.
threshold_i	Un seuil défini pour limiter le trafic accepté du $i^{ème}$ client.
service_rate	Le taux de service de la file d'attente.

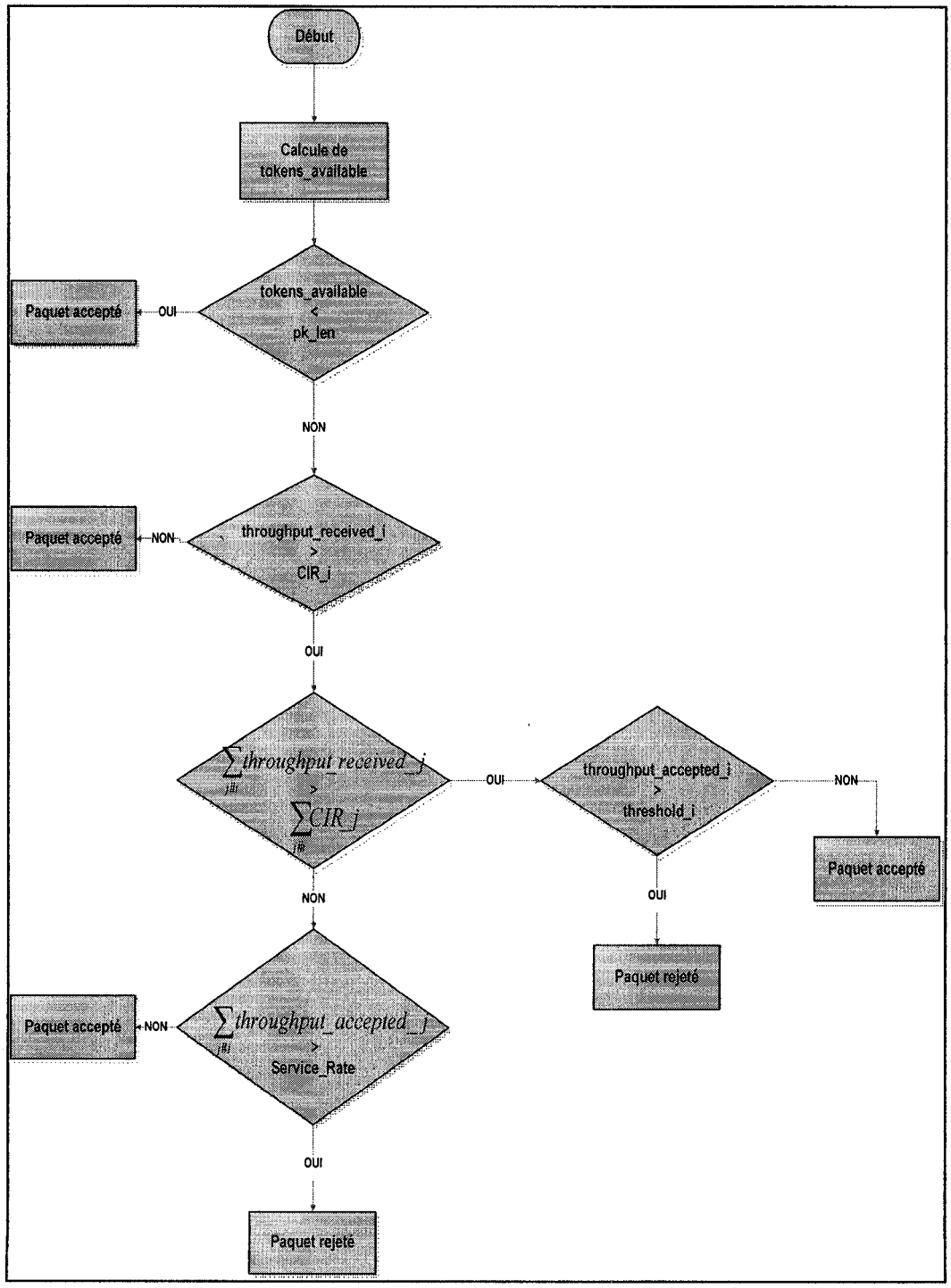


Figure 3.14 ***L'algorithme DDTB.***

Comme le montre la figure 3.14, lorsqu'un paquet est reçu, sa taille est comparée au nombre de jetons disponibles dans le seau à jetons (*tokens_available*). Lorsque la taille du paquet est plus petite que *tokens_available,* on vérifie si le débit du trafic reçu du client ne dépasse pas CIR. Si ce n'est pas le cas, alors il est accepté. Sinon, le paquet a violé le contrat de trafic et il est considéré non conforme.

Dans DDTB, si un paquet reçu n'est pas conforme, il ne sera pas automatiquement rejeté, mais il peut être accepté si la file d'attente a les ressources nécessaires pour le servir sans dégrader les flux des autres clients. Pour accepter les paquets non-conformes, deux cas possibles se présentent :

a. La somme des débits des flux de trafic reçus des autres sources ne dépassent pas la somme de leurs CIRs. Dans ce cas, les paquets entrant non-conformes seront acceptés tant que le débit du flux total accepté des différentes sources ne dépasse pas le taux de service de la file d'attente.
b. La somme des débits des flux de trafic reçus des autres clients dépassent la somme de leurs CIRs. Lorsque les flux entrant dépassent leurs CIR, il est important de limiter le trafic accepté de chaque flux afin d'éviter la congestion. Au lieu de limiter chaque client à son CIR, dans DDTB on propose de limiter le flux du ième client au seuil donné par le paramètre *threshold_i* et calculé par l'équation (3.11).

$$threshold\_i = CIR\_i + \frac{service\_rate - \sum_j CIR\_j}{\sum_j CIR\_j} \cdot CIR\_i \qquad (3.11)$$

La différence entre le taux de service de la file d'attente et la somme des CIRs est distribuée entre les clients selon leur CIRs. Comme le montre la figure 3.15, le débit accepté du ième client sera plus élevé que le CIR. Ceci est l'avantage de l'algorithme DDTB qui a pour but de maximiser l'utilisation de la file d'attente.

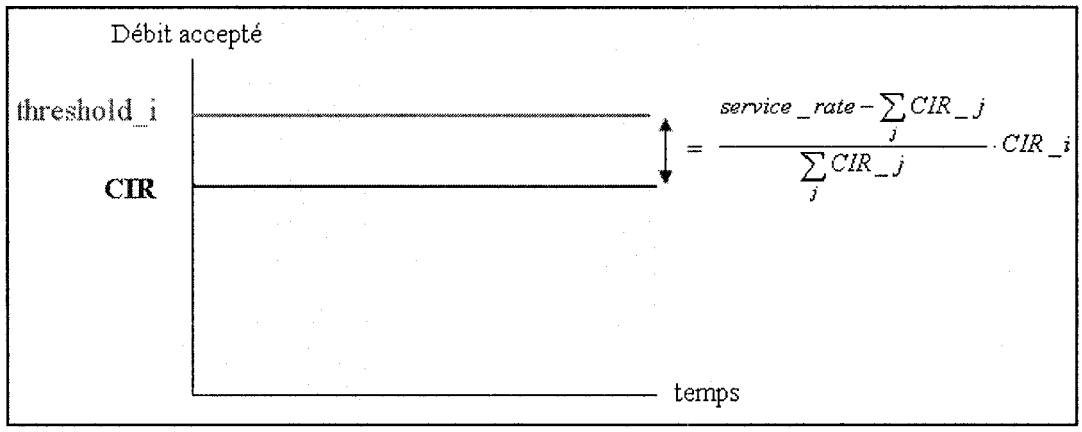


Figure 3.15 ***La différence entre le threshold_i et le CIR.***

3.5 Simulation et analyse des résultats

L'algorithme DDTB a été codé et simulé à l'aide de OPNET. Le but de la simulation est de montrer l'intérêt de l'utilisation de l'algorithme dans le contrôle des flux des clients L2VPNs transportant la même classe de trafic et partageant la même file d'attente. Le modèle OPNET, de l'algorithme DDTB, est illustré dans la figure 3.16.

La simulation a été effectuée en variant la valeur du débit du trafic entrant dans chaque L2VPN et obtenir le débit du trafic accepté et transmis à la file d'attente. On peut aussi évaluer le nombre des paquets acceptés et le nombre des paquets rejetés pour chaque client L2VPN.

Le modèle implémente les éléments suivants :

a. La source de trafic pour chaque client L2VPN.
b. Le processus TB qui simule le comportement de l'algorithme DDTB.
c. Une file d'attente FIFO partagée par les clients.
d. Le processus *sink* qui permet de détruire les paquets qu'il reçoit et ainsi libérer la mémoire occupée par les paquets.

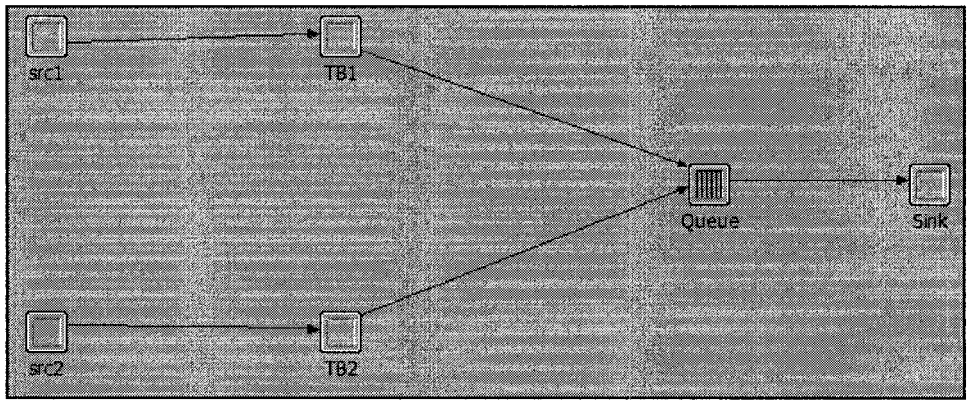


Figure 3.16 ***Simulation OPNET.***

3.5.1 Processus TB

Le comportement de l'algorithme DDTB est simulé par le processus TB. Les paquets acceptés sont envoyés dans la file d'attente. Si les ressources ne sont pas suffisantes pour accepter un paquet il sera automatiquement détruit. Le processus fait aussi l'enregistrement des statistiques sur le débit, les paquets et les jetons. Le processus TB est illustré dans la figure 3.17.

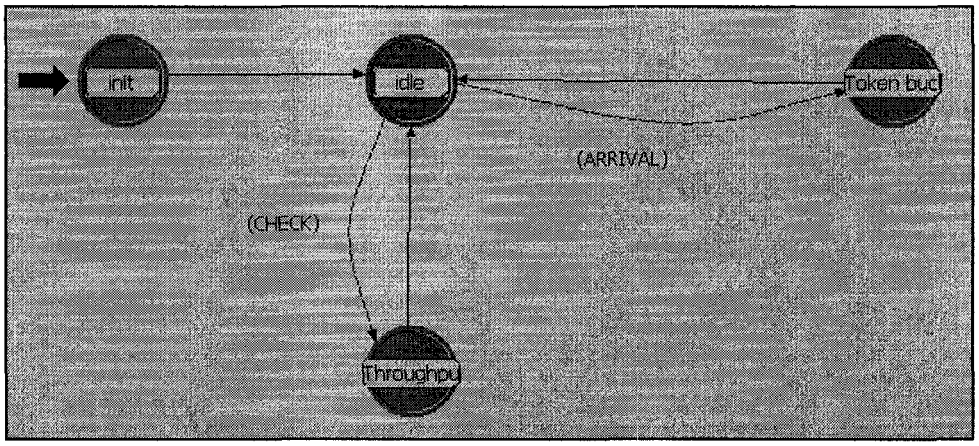


Figure 3.17 ***Processus TB.***

Le code de chaque état se trouve dans 0. L'état initial *init* est le point d'entrée du processus. Il permet d'initialiser les paramètres de la simulation et les statistiques. L'état non-forcé *idle* est un état intermédiaire qui attend une des deux interruptions suivantes :

a. *ARRIVAL* – Un paquet provenant de la source pour changer d'état et aller vers l'état suivante *Token_Bucket*.
b. *CHECK* – L'interruption est déclenchée à chaque seconde afin de calculer les statistiques des débits des flux de trafic reçus des sources et les débits des flux de trafic acceptés par les TBs.

Tant que les interruptions mentionnées précédemment ne sont pas reçues, l'état boucle sur lui-même.

L'état, *Token Bucket*, implémente l'algorithme DDTB. D'abord, il reçoit un paquet et incrémente la variable *packet_count* qui indique le nombre des paquets reçus. Ensuite il met à jour la valeur du paramètre *throughput_receive* qui compte la taille totale des paquets reçus pendant une seconde. Par la suite, il calcule le nombre de jetons qui seront ajoutés (*tokens_added*) dans le seau à jetons en se basant sur l'espace vide de la file d'attente. Cette valeur est enregistrée dans la statistique des jetons ajoutés

(*tokens_added_stat*). Le nombre de jetons disponibles (*tokens_available*) dans le seau à jetons est ensuite mis à jour.

Comme il est indiqué dans l'algorithme DDTB, le processus se base sur la taille du paquet reçu et le nombre des jetons disponibles dans le seau à jetons pour décider d'accepter le paquet ou bien de le détruire. Lorsqu'un paquet est accepté, le compteur *packet_accepted,* qui compte le nombre des paquets acceptés par le seau à jetons, est incrémenté. Le compteur *throughput_accepted,* qui compte la taille totale des paquets acceptés pendant une seconde, est aussi mis à jour en lui ajoutant la taille du paquet accepté. Mais lorsqu'un paquet est rejeté, le compteur *paquet_rejected* est incrémenté.

L'état, *Throughput,* a été mis en place afin d'enregistrer les statistiques des débits des flux de trafic reçus des sources et les débits des flux de trafic acceptés par les seaux à jetons. Après chaque seconde :

a. La valeur du paramètre *throughput_received* est enregistrée dans la statistique du débit reçu *throughput_received_stat* et ensuite sa valeur est mise à zéro.
b. De même, la valeur du paramètre *throughput_accepted* est enregistrée dans la statistique *throughput_accepted_stat* et ensuite sa valeur est mise à zéro.

Nous avons aussi défini les paramètres P1 et P2 pour la première source et la deuxième source respectivement. Ces paramètres prennent la valeur 2 lorsque le débit du flux reçu de la source dépasse le débit de son CIR, sinon ils prennent la valeur 1. Ceci permet au processus de déterminer si une source de trafic a dépassé son CIR ou non.

3.5.2 Scénarios de tests et simulations

Dans cette section on présente les résultats des simulations pour 4 scénarios de tests différents.

3.5.2.1 Choix des paramètres

Dans OPNET on s'est limité à deux sources de trafic afin de simplifier le modèle de simulation. Il est possible de spécifier différents types de distribution pour la taille des paquets ainsi que l'intervalle de temps entre les paquets en provenance des sources. Les paramètres ont été définis comme étant *PROMOTED*. Cela signifie qu'ils peuvent être tous modifiés à partir du nœud racine. La figure 3.18 illustre la fenêtre *ATTRIBUTES* qui s'affiche pour entrer les valeurs des paramètres de simulations.

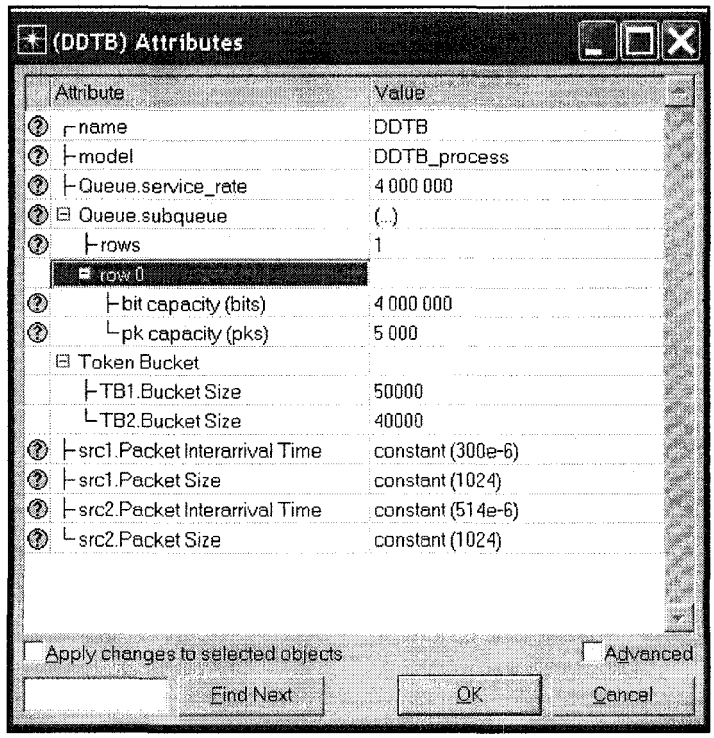


Figure 3.18 ***Définition des valeurs des paramètres des simulations.***

On voit dans la figure 3.18 que la taille de la fille d'attente est définit à 4 Mbits. Le taux de service est fixé à 4 Mbps. L'intervalle de temps entre les paquets suit la distribution "constant". On peut contrôler le débit du flux de trafic provenant des sources en variant la valeur de l'intervalle de temps entre les paquets. Le débit du trafic provenant des sources est inversement proportionnel à la valeur de l'intervalle de temps séparant deux paquets. Comme il est indiqué dans le modèle DDTB, la taille des paquets suit la distribution "poisson" avec une moyenne de 1024 bits.

Dans cet exemple, l'intervalle de temps a été défini à 300us et 514us pour la première source et la deuxième source successivement. Vu la taille des paquets, le flux de trafic généré sera d'une moyenne de 3.41Mbps et 2Mbps pour la première source et la deuxième source successivement.

La taille du seau à jetons a été fixée à 50000 octets et 40000 octets pour la première source et la deuxième source respectivement. Le débit CIR peut alors être calculé en utilisant l'équation (3.2). La valeur du débit CIR est de 1.60 Mbps et 1.28 Mbps pour la première source et la deuxième source respectivement.

Dans le reste de cette section on présente les résultats des tests de simulations. On a choisi 4 scénarios pour simuler le comportement de DDTB dans les différents cas possibles.

3.5.2.2 Les flux de trafic des deux clients ne dépassent pas leurs CIRs

Dans ce scénario, on génère pour les deux sources un flux de trafic qui a un débit de 1 Mbps. Les résultats des simulations sont illustrés dans la figure 3.19 et la figure 3.20.

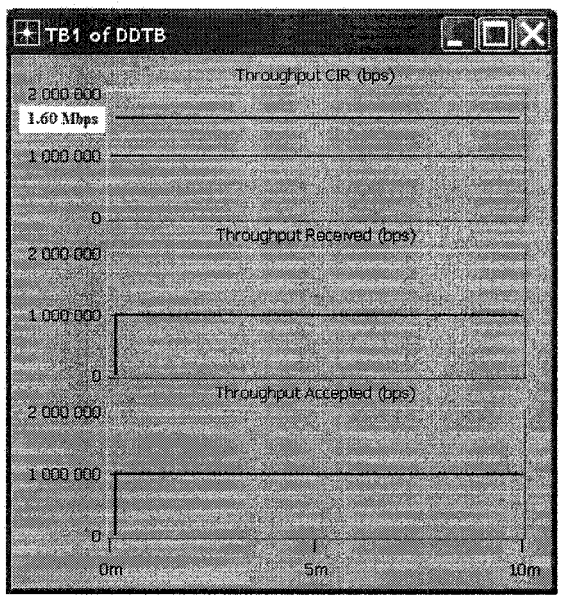


Figure 3.19 ***Résultats pour le client 1.***

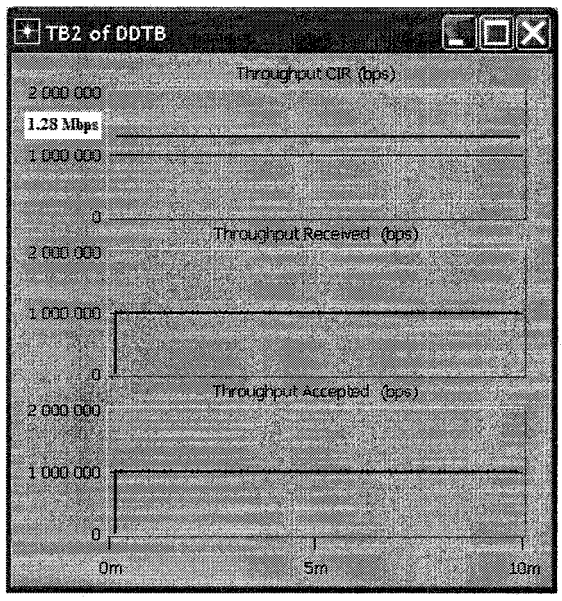


Figure 3.20 ***Résultats pour le client 2.***

On constate que pour les deux clients, le trafic accepté est identique au trafic reçu. En effet, la file d'attente a un taux de service plus élevé que le débit du flux reçu des deux clients. Ainsi il reste toujours de l'espace vide dans la file d'attente qui permettra de générer des jetons pour les deux clients. En plus, le débit du trafic des deux clients était moins élevé que leurs CIR, donc les paquets étaient toujours acceptés.

3.5.2.3 Seulement le flux de trafic du deuxième client dépasse son CIR

Dans ce scénario, le premier client génère un flux de trafic dont le débit est de 1 Mbps. Le deuxième client génère un flux de trafic dont le débit est de 2 Mbps. Les résultats obtenus sont illustrés dans la figure 3.21 et la figure 3.22.

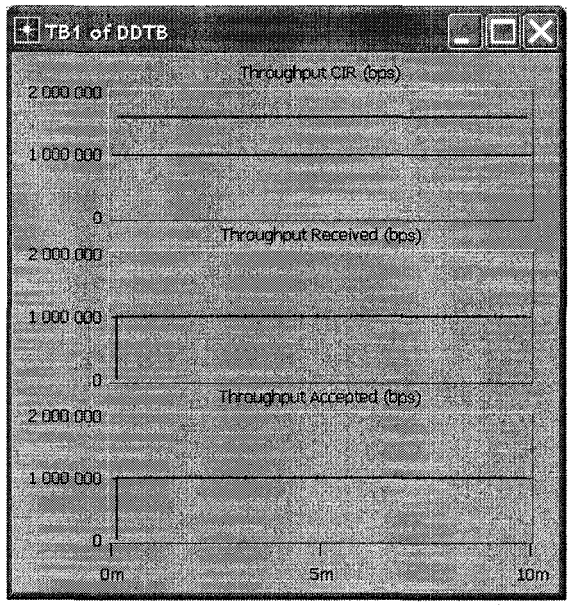


Figure 3.21 ***Résultats pour le client 1.***

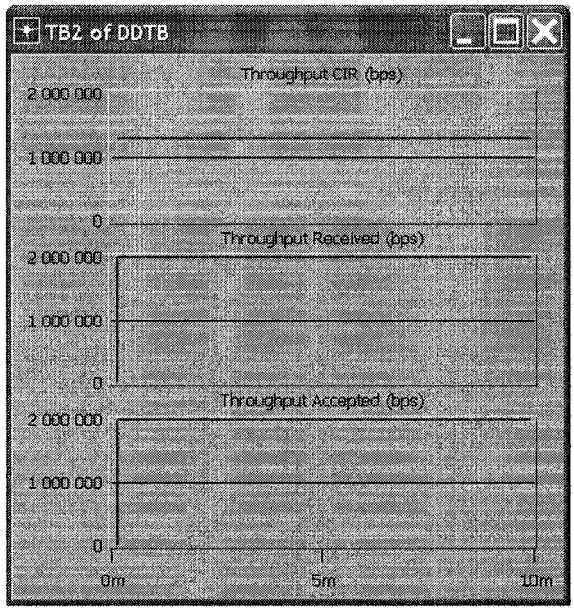


Figure 3.22 ***Résultats pour le client 2.***

Le flux de trafic de la première source est inférieur à son CIR (1.6Mbps), donc il a été accepté au complet sans aucune dégradation. Le flux de trafic généré par la deuxième source dépasse son CIR (1.28Mbps).

Comme il a été expliqué dans 0, vu que le premier client ne dépasse pas son CIR, les paquets sont acceptés tant que la bande passante n'est pas utilisée au complet. Dans cet exemple le débit total du flux entrant est inferieur au taux de service de la file d'attente, le débit du client 2 a été accepté sans aucune dégradation.

3.5.2.4 Les flux de trafic des deux clients dépassent leurs CIRs

Le but de ce scénario est de s'assurer que lorsque les flux de trafic des deux clients violent leurs contrats, l'algorithme DDTB empêche la congestion de la file d'attente et garantit à chaque client au moins son débit CIR.

Le premier client génère un flux de trafic dont le débit est de 2.572 Mbps qui est supérieur à son CIR (1.6 Mbps). Le deuxième client génère un flux de trafic dont le débit est de 2 Mbps qui est supérieur à son CIR (1.28 Mbps). Les résultats obtenus en utilisant l'algorithme DDTB, sont illustrés dans la figure 3.23 et la figure 3.24.

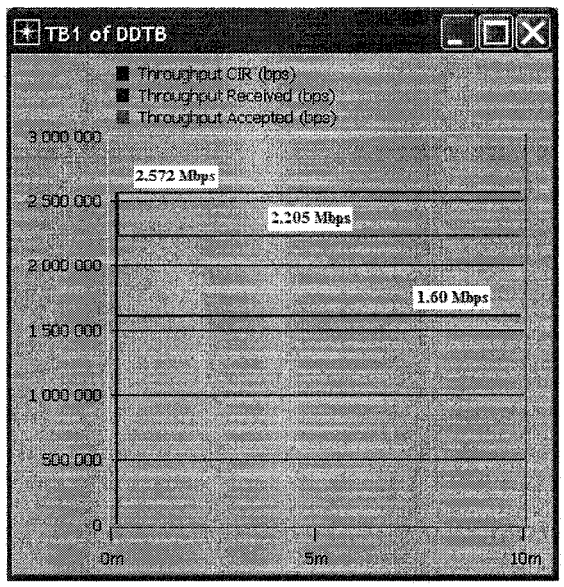


Figure 3.23 ***Résultats pour le client 1.***

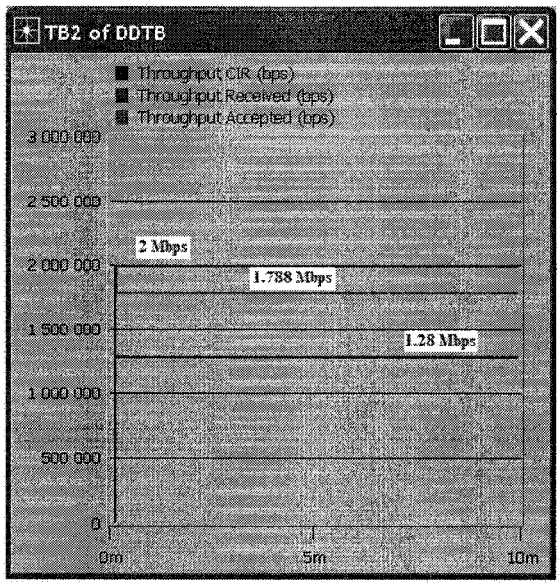


Figure 3.24 ***Résultats pour le client 2.***

Le débit total des deux flux entrant est à peu prêt de 4.5 Mbps qui est supérieur au taux de service de la file d'attente (4 Mbps). Dans ce cas, DDTB limite le flux de trafic de chaque client au seuil définit dans l'équation (3.11). La valeur du seuil est de 2.205 Mbps et 1.788 Mbps pour la première source et la deuxième source respectivement.

On constate une différence considérable entre les valeurs des CIRs et les débits des flux acceptés. Ainsi le résultat montre bien que DDTB est capable d'empêcher une congestion lorsque le flux total des sources de trafic dépasse la capacité de la file d'attente tout en garantissant au minimum leurs débits CIR.

3.5.2.5 La deuxième source génère le trafic après un certain délai

La première source génère un flux de trafic de 3.417 Mbps. La deuxième source ne génère pas de trafic pendant les premières 5 minutes et ensuite elle génère un flux de

trafic de 2 Mbps. Les résultats de la simulation de ce scénario sont illustrés dans la figure 3.25 et la figure 3.26.

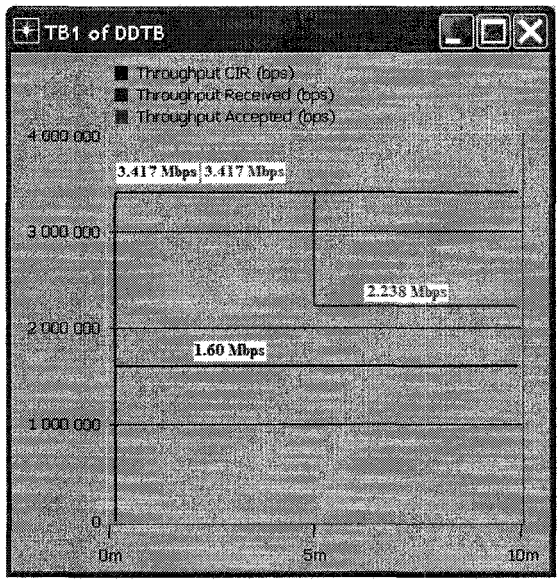


Figure 3.25 ***Résultats pour le client 1.***

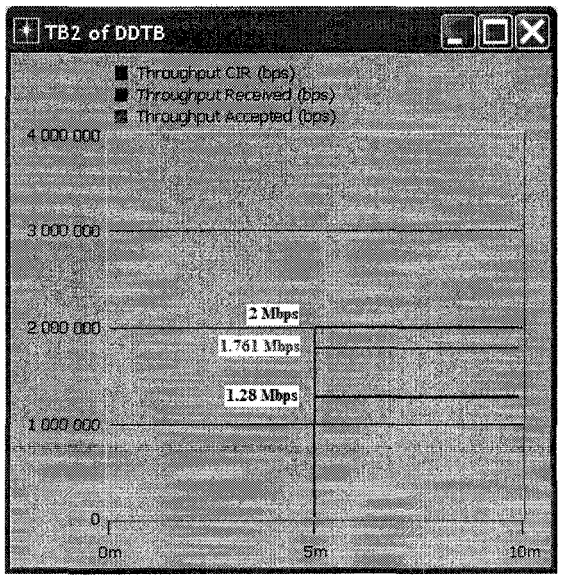


Figure 3.26 ***Résultats pour le client 2.***

On constate que pendant les premières 5 minutes, le flux de trafic de la première source a été accepté sans aucune dégradation. En effet, au début de la simulation, la deuxième source ne génère pas de trafic, donc, comme il est indiqué dans 0, les paquets qui ne sont pas conformes de la première source sont acceptés tant que le débit du flux accepté n'excède pas la bande passante. Or, le débit du flux entrant de la première source (3.417Mbps) est inférieur au taux de service de la file d'attente (4Mbps), donc le flux de la première source est accepté sans aucune dégradation.

Cependant, après 5 minutes, la deuxième source a commencé la génération de trafic. À cet instant, l'algorithme DDTB lui a garanti ses ressources en dégradant un peu le débit du flux accepté de la première source et en garantissant toujours son CIR.

Le flux généré par la deuxième source dépasse son CIR, donc, comme il a été expliqué dans la section 0, les paquets non-conformes des deux sources sont acceptés tant que le débit du flux accepté de chaque source ne dépasse pas son seuil. L'équation (3.11) permet de calculer les valeurs des seuils *threshold_1* et *threshold_2* qui sont égales à 2.228 Mbps et 1.761 Mbps pour la première source et la deuxième source respectivement.

3.5.2.6 Comparaison avec les résultats obtenus sur le banc d'essai.

Dans ce scénario de test on a effectué les mêmes tests qui ont été réalisés pour illustrer la problématique. Le but est de montrer la pertinence de la solution en prouvant l'amélioration introduite par l'algorithme DDTB.

Comme dans l'illustration de la problématique, le flux de trafic généré par la première source de trafic était fixé à 1Mbps et le flux de trafic généré par la deuxième source de trafic variait de 1Mbps à 10Mbps.

Le flux de trafic provenant de la première source est appelé Flux 1 et le flux de la deuxième source est appelé Flux 3, comme dans la problématique.

Les résultats obtenus sur le banc d'essai (tableau 3.6) ont montré une dégradation catastrophique des deux flux entrant Flux 1 et Flux 3. Nous avons vu que la bande passante utilisé par un flux de trafic est proportionnel à sa quantité.

Les résultats obtenus en utilisant DDTB sont illustrés dans le tableau suivant.

Tableau 3.10

Résultats obtenus pour DDTB

Trafic généré (Mbps)		**Trafic accepté (Mbps)**	
L2VPN A Flux 1	L2VPN B Flux 3	L2VPN A Flux 1	L2VPN A Flux 3
1	1	1	1
2	1	2	1
3	1	3	1
4	1	2.238	1
5	1	2.238	1
6	1	2.238	1
7	1	2.238	1
8	1	2.238	1
9	1	2.238	1
10	1	2.238	1

Le flux de la deuxième source (Flux 3) est généré à un débit inférieur à son CIR (1.28Mbps). On peut constater que Flux 3 n'a subit aucune dégradation.

Dans les deux premiers scénarios, on a vu que tant la somme des flux de trafic reçus des deux sources ne dépassent pas le taux de service de la file d'attente, le trafic est accepté des deux sources. Cependant, lorsque le débit du Flux 1 dépasse 4Mbps, la file d'attente risque d'être congestionnée si aucun contrôle n'est effectué sur les flux de trafic entrants. En effet le taux de service de la file d'attente est uniquement de 4Mbps.

Lorsque le débit du flux généré de la deuxième source (Flux 1) varie entre 4Mbps et 10Mbps, son débit accepté a été limité au seuil défini par l'algorithme DDTB pour ce trafic. Ainsi, la congestion a été évitée et le trafic du Flux 3 n'a pas subit une dégradation.

La figure 3.27 et la figure 3.28 illustrent les résultats obtenus, lorsque le débit du flux de trafic de la première source est de 5Mbps et 10Mbps respectivement.

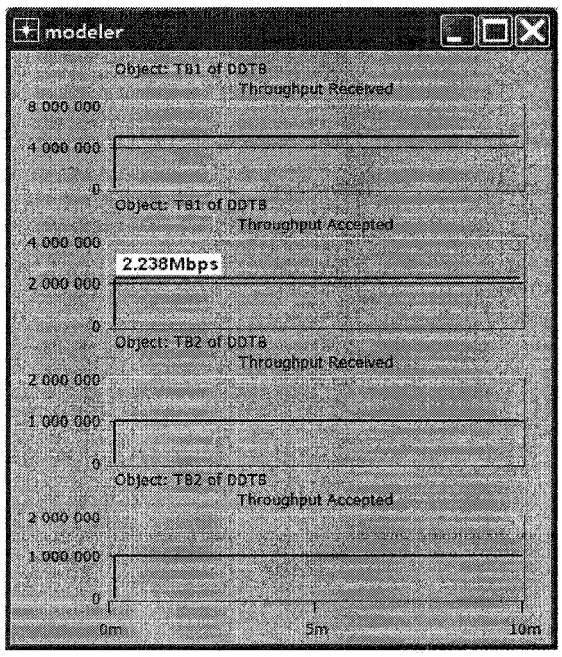


Figure 3.27 ***Résultats lorsque Flux 1 reçu a un débit de 5Mbps.***

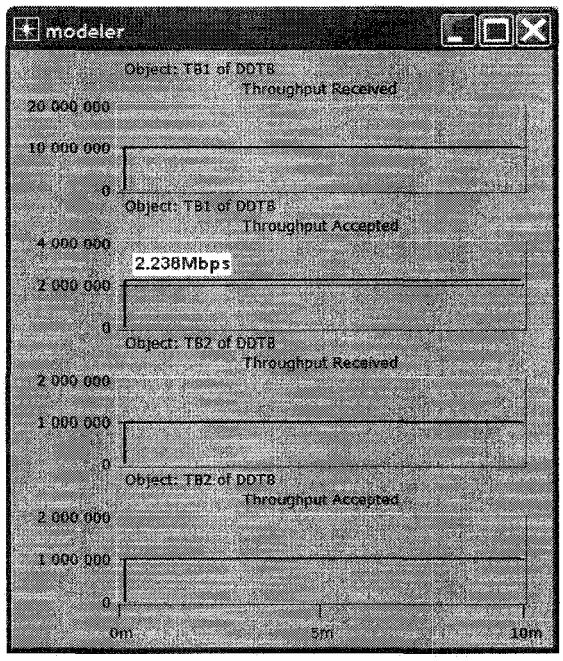


Figure 3.28 ***Résultats lorsque Flux 1 reçu a un débit de 10Mbps.***

A travers les différents scénarios de tests, on a montré que l'algorithme répond bien à notre problématique. L'algorithme DDTB assure une utilisation efficace des ressources et contrôle le débit du flux accepté de chaque client. Les ressources résiduelles sont allouées dynamiquement afin d'optimiser leurs utilisations.

Cette solution mesure les ressources non utilisées par les différents flux de trafic et effectue leurs redistributions sur les clients qui veulent transmettre à un débit supérieur à leurs CIRs. Il est très avantageux de pouvoir décaler l'utilisation de la bande passante en se basant sur le temps ou les jours. Par exemple, une institution financière pourrait avoir besoin de sa bande passante uniquement pendant les heures des affaires.

CONCLUSION

Dans ce mémoire fut étudié et analysé l'application de MPLS dans l'établissement des réseaux privés virtuels. Nous avons vu qu'il existe deux approches pour l'implémentation des réseaux virtuels privés à travers une infrastructure IP/MPLS, L2VPN et L3VPN. Chaque approche offre une solution qui répond de manière différente aux besoins de la connectivité des clients.

Afin de répondre le mieux aux besoins des clients souhaitant connecter leurs réseaux locaux distants, il est nécessaire de considérer les points forts et les points faibles de chaque approche. Dans ce but, nous avons comparé les deux solutions en se basant sur des critères vitaux de type dimensionnement, déploiement, gestion et maintenance, type de trafic transporté, mise à l'échelle et coût.

Une étude théorique des performances des L2VPNs et des L3VPNs a été réalisée afin de mesurer la différence entre les performances des deux technologies. Cette étude nous a confirmé que les performances des services L2VPN sont limitées à cause de la taille des entêtes ajoutés dans le processus d'encapsulation des paquets IP. La différence des performances entre les deux approches est plus visible pour les petits paquets. Étant donné que la communication voix utilise des petits paquets, alors nous avons calculé la différence entre la bande passante consommée par le trafic voix dans un L2VPN et dans un L3VPN en utilisant différents CoDecs de voix. La solution L3VPN est plus favorable pour une entreprise qui aimerait déployer de la voix sur IP.

D'autre part, un des avantages des services L2VPN est la possibilité d'utiliser des commutateurs comme CEs pour connecter les réseaux des clients au réseau MPLS de l'opérateur. Nous avons pu voir a travers notre étude que pour utiliser un commutateur comme équipement CE, la mise en place de la qualité de service n'est pas garantie si plusieurs client sont connectés au même CE.

Notre problématique a donc été de concevoir une solution qui peut garantir à chaque client L2VPN les paramètres définis dans son contrat de trafic (SLA) avec l'opérateur. Donc si un client génère beaucoup de trafic qui peut créer une congestion, la solution doit l'empêcher afin que ce trafic ne puisse pas nuire aux trafics des autres clients. On souhaitait aussi que la solution soit aussi capable de garantir à chaque client au moins son débit CIR et lui permettre de transmettre plus si la bande passante disponible le permettrait.

Finalement, pour répondre à notre problématique nous avons proposé l'algorithme DDTB qui est une amélioration au modèle dynamique de l'algorithme TB. L'algorithme a été simulé à l'aide du logiciel OPNET. Les résultats obtenus ont montré que l'algorithme contrôle bien les flux des clients transportant la même classe de trafic et partageant la même file d'attente. L'algorithme développé répond aux contraintes et caractéristiques spécifiées.

ANNEXE 1

ÉTABLISSEMENT D'UN PW EN UTILISANT ATOM

L'objet de cette annexe est de décrire en détail les mécanismes réseau mis-en œuvre lorsqu'un lien AToM est activé, en particulier les échanges de message LDP, protocole de signalisation et distribution des étiquettes MPLS dans le réseau MPLS.

Cette description est faite dans le contexte du laboratoire de test utilisé et décrit précédemment dans ce mémoire. Nous avons procédé par une observation poussée du fonctionnement de la technologie AToM. Les figures qui suivent ont été obtenues à l'aide d'un renifleur placé dans le réseau MPLS disponible chez Bell Canada. Une trace a été relevée à chaque étape de la configuration du lien AToM.

5.1 Activation d'un lien AToM

Le réseau de test est illustré dans la figure 3.27, Notez que CE2 et CE3 se trouvent dans le même segment.

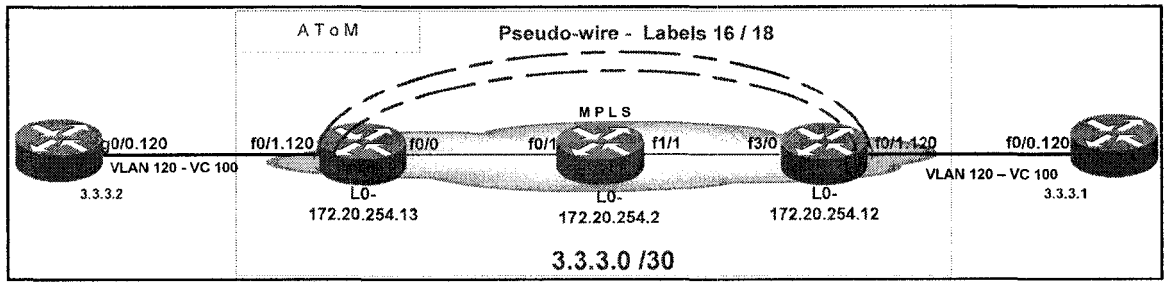


Figure 3.29 ***Réseau de test.***

Le réseau de test est une plate-forme de routeurs Cisco interconnectés par différentes technologies de niveau 2 (ATM et Ethernet) et implémentant MPLS [5]. Les PEs sont

des routeurs *Cisco 7206*, les équipements LSR sont des routeurs *Cisco 3725*. Les versions des IOS des routeurs PEs et des routeurs LSR sont 12.2 et 12.3 respectivement. On observe le trafic à l'aide d'un renifleur *Agilent DNA*. On suppose que l'on veuille activer un PW entre les routeurs PE3 et PE2 et que l'on commence par configurer PE3. Les commandes de configuration du *Ethernet over MPLS VLAN mode* sont présentées dans le tableau suivant.

Tableau 3.11

Configuration des routeurs PEs et CEs

Routeurs	**Commandes**
PE2	interface FastEthernet2/0.120 Encapsulation dot1Q 120 no ip address no cdp enable xconnect 172.20.254.12 100 encapsulation mpls
CE2	interface GigabitEthernet0/0.120 Encapsulation dot1Q 120 ip address 3.3.3.2 255.255.255.252
PE3	interface FastEthernet0/1.120 Encapsulation dot1Q 120 no ip address no cdp enable xconnect 172.20.254.13 100 encapsulation mpls
CE3	interface FastEthernet0/1.120 Encapsulation dot1Q 120 ip address 3.3.3.1 255.255.255.252

Avec cette configuration, le nuage MPLS se comportera comme un seul segment (3.3.3.0/30) reliant CE2 et CE3 de manière transparente pour le réseau du client. Le VC

100 est de type Ethernet, il correspond au VLAN 120. Le pseudo-wire encapsule les trames Ethernet appartenant au VC 100.

La commande `xconnect` permet d'établir un *Emulated VC* entre PE2 et PE3. Les interfaces `f2/0.120` et `f0/1.120` sont, alors, configurées pour supporter *Ethernet over MPLS* (EoMPLS). Les trames Ethernet sont encapsulées dans MPLS et acheminées au PE correspondant en utilisant son adresse loopback. On désactive `cdp` pour la connexion PE2-PE3. PE2 n'a pas besoin de `cdp` pour découvrir PE2 puisque la commande **`xconnect`** permet d'établir une connexion virtuelle entre PE2 et PE3 à travers le nuage MPLS. On constate dans la figure 3.28 que PE2 répond à ce message par un autre message d'initialisation LDP.

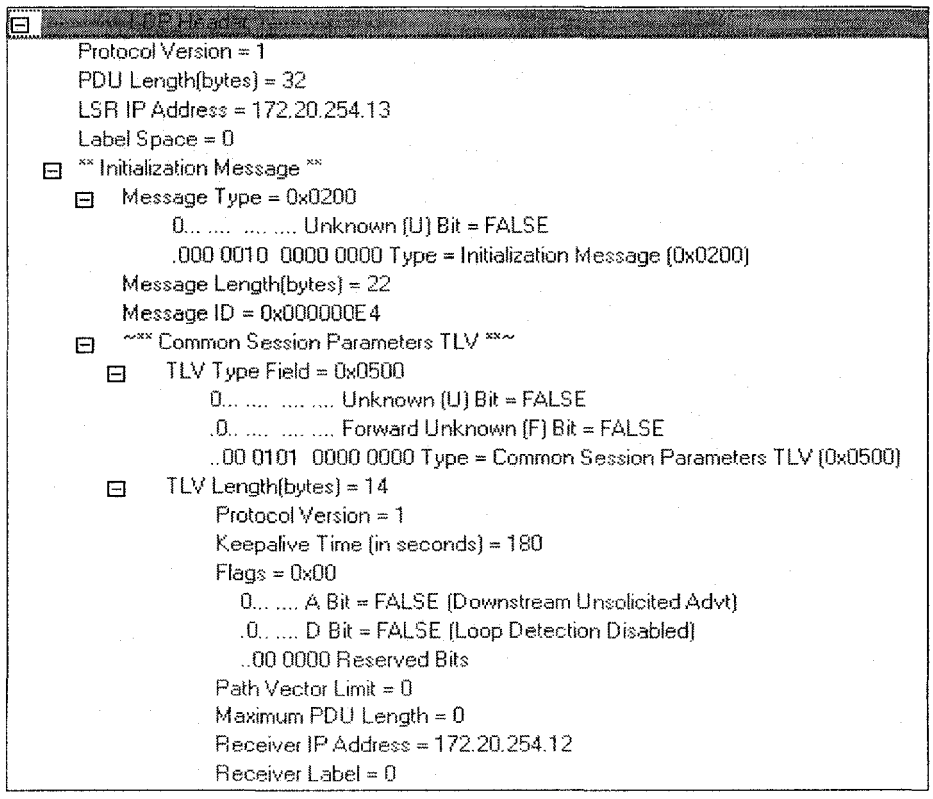


Figure 3.30 ***Message LDP d'initialisation de PE3 vers PE2.***

La figure 3.29 illustre le message envoyé par PE2 vers PE3. Il s'agit d'un autre message d'initialisation.

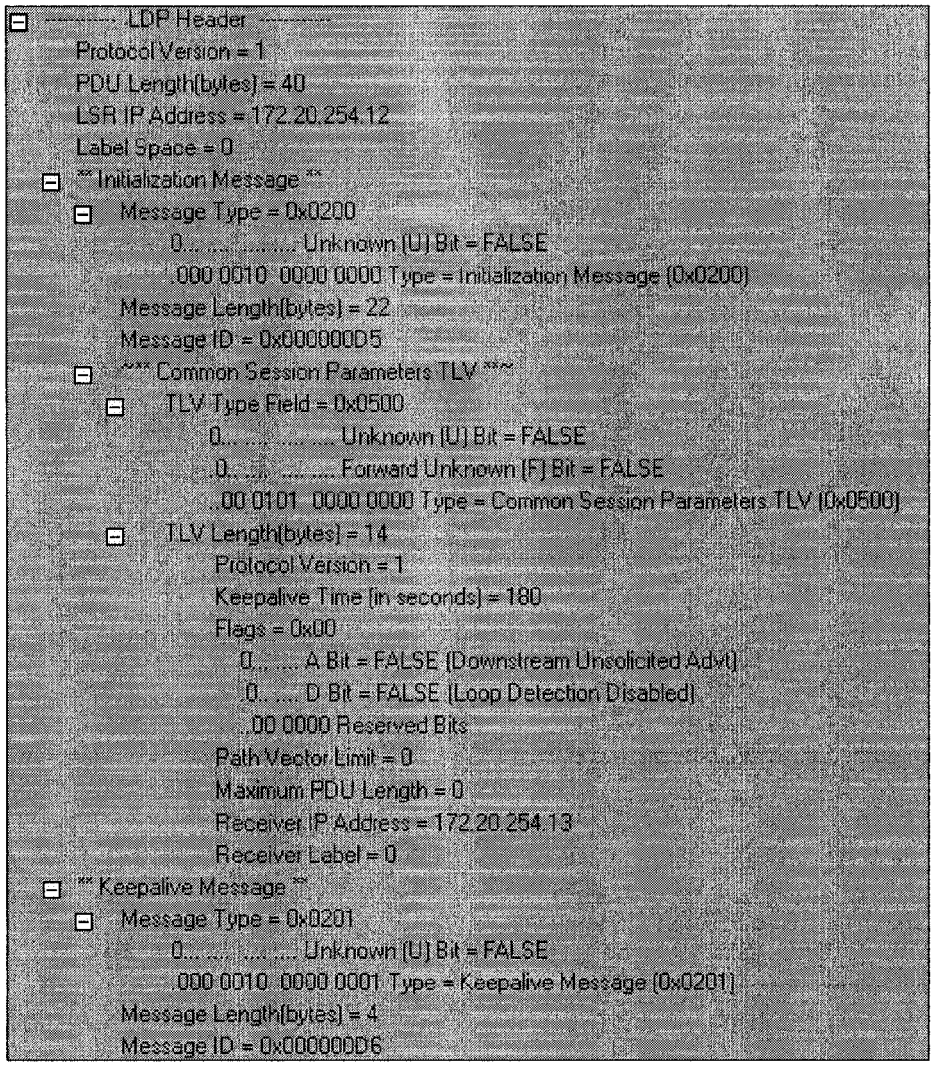


Figure 3.31 ***Message d'initialisation LDP de PE2 vers PE3.***

La figure 3.30 affiche le message *KeepAlive* envoyé par PE3 vers PE2.

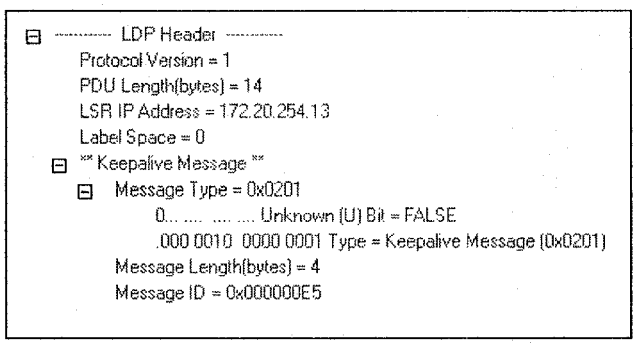


Figure 3.32 ***Message KeepAlive LDP de PE3 vers PE2.***

PE2 signale ensuite les adresses de ses interfaces globales physiques et LoopBack0 à PE3 afin d'établir un lien LDP. Ces informations sont incluses dans un message TLV.

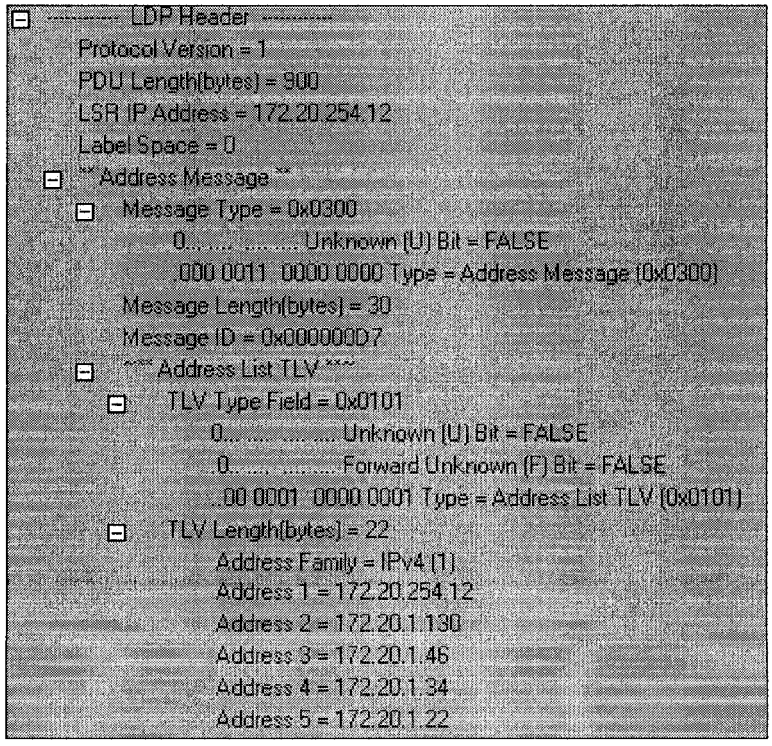


Figure 3.33 ***Message LDP de PE2 vers PE3.***

PE2 envoie ensuite un *Label Mapping Message* qui a pour but d'informer PE3 des Labels à utiliser pour chaque FEC. Dans ce cas, ce sont des *Prefix*. Ils correspondent aux sous réseaux auxquels appartiennent les interfaces physiques.

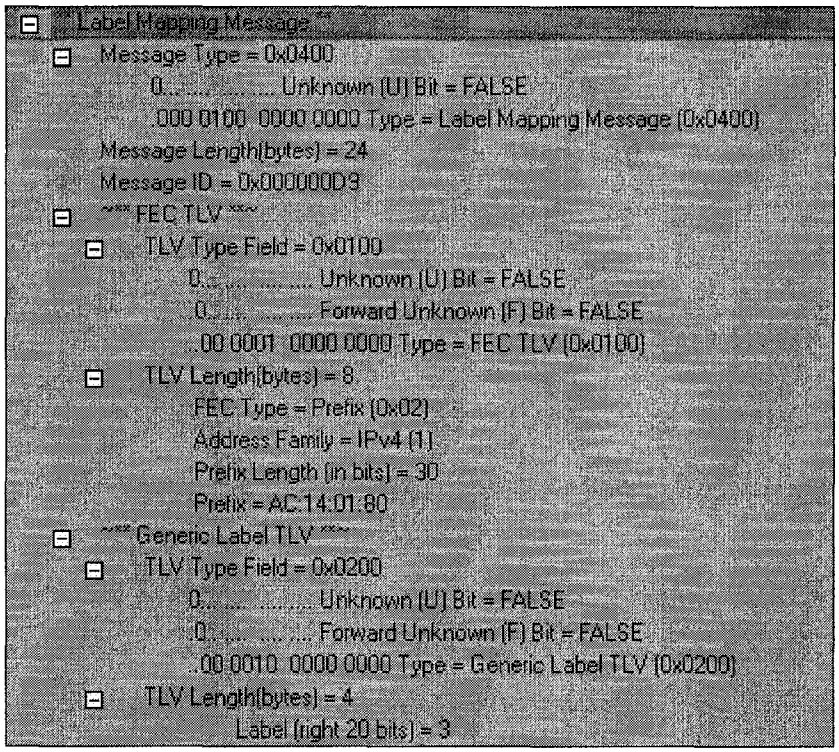


Figure 3.34 ***Label Mapping Message LDP de PE2 vers PE3.***

Un *Label Mapping Message* est envoyé pour chaque FEC de PE2. Celui-ci correspond à l'adresse réseau (*Prefix*) 172.20.1.128 /30. Ce sous réseau est directement connecté sur l'interface *f3/1* de PE2 (172.20.1.130). On voit également que la valeur de l'étiquette MPLS contenue dans ce message est 3, c'est une valeur réservée signifiant « NULL-IMPOSITION », c'est à dire qu'aucune l'étiquette MPLS n'est réellement attribué à ce préfix. Il en est de même pour tous les autres. En effet, le lien AToM n'étant pas configuré du coté de PE2, aucun lien LDP ne peut être établit. PE3 envoie donc un message d'erreur. La figure 3.33 illustre le message d'erreur LDP envoyé de PE3 vers PE2.

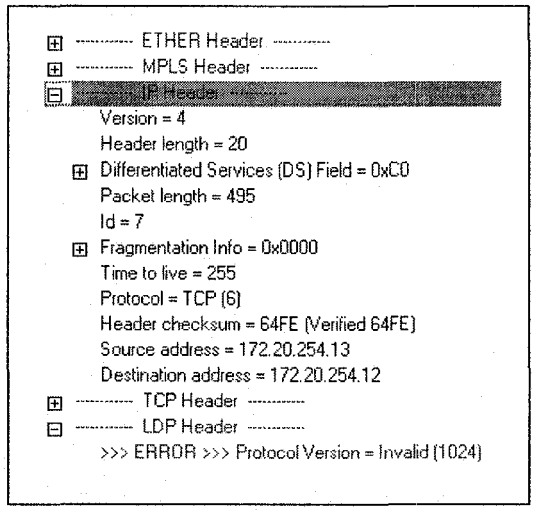


Figure 3.35 ***Message d'erreur de PE3.***

PE2 répond également par un message d'erreur LDP. Le message d'erreur est illustré dans la figure suivante.

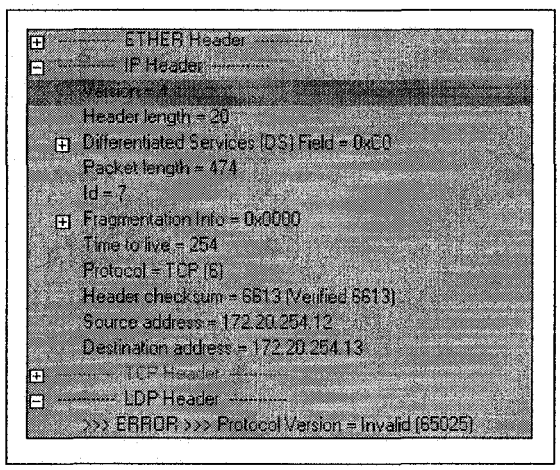


Figure 3.36 ***Message d'erreur de PE2.***

Il faut noter que les messages échangés ont eu lieu juste après la configuration de AToM au niveau de PE3. Les messages d'erreur obtenus a la fin sont dus au fait que l'on n a

pas encore configuré AToM sur PE2. La commande « `xconnect 172.20.254.13 120 encapsulation mpls` » configure AToM sur PE2. La commande déclenche l'initialisation d'une connexion TCP puis d'une session LDP avec la destination PE3 dont l'adresse *loopback* est 172.20.254.13. Maintenant que les deux routeurs sont configurés, PE2 envoie un *LDP Label Maping* Message pour informer PE3 de l'étiquette MPLS à utiliser avec le VC 120 (voir figure 3.35). Ici c'est l'étiquette MPLS 16 qui est stocké dans la table d'information relative au VC 120. Une autre étiquette MPLS est ensuite rajoutée pour le NEXT-HOP alors que l'étiquette MPLS 16 permet uniquement aux routeurs PEs d'identifier des paquets provenant du VC 120.

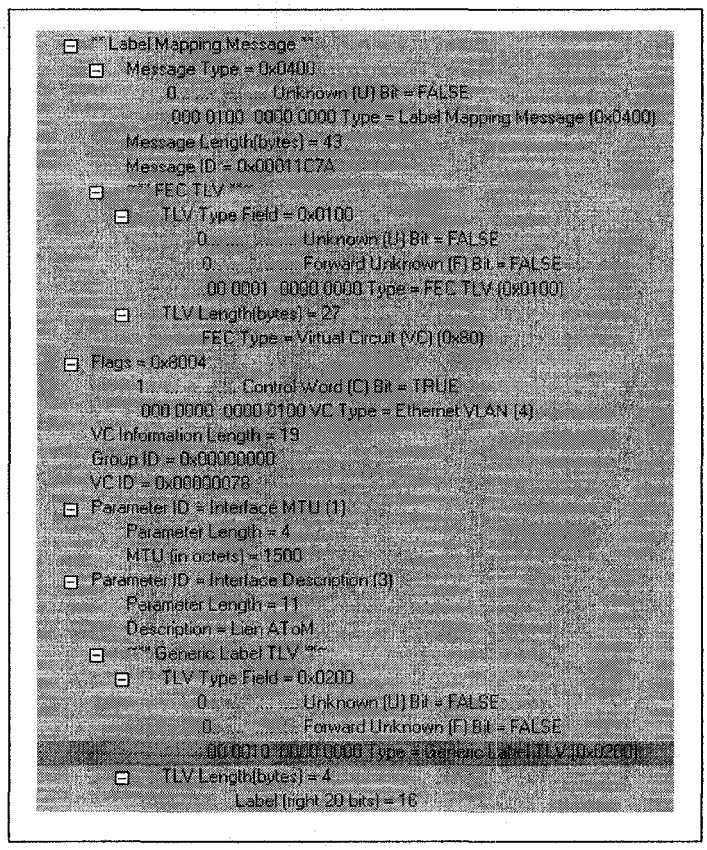


Figure 3.37 ***Label Maping Message envoyé de PE2 vers PE3.***

5.2 Désactivation d'un tunnel

Sur PE2, on désactive une extrémité du tunnel AToM et on observe les paquets entre P2 et PE3. On voit une trame LDP en provenance de PE2 vers PE3 qui l'averti que le tunnel est brisé. Il indique le numéro du VC (78 en base 16 soit 120 en base 10) et l'étiquette MPLS qui lui correspondait afin que PE3 l'efface de sa table.

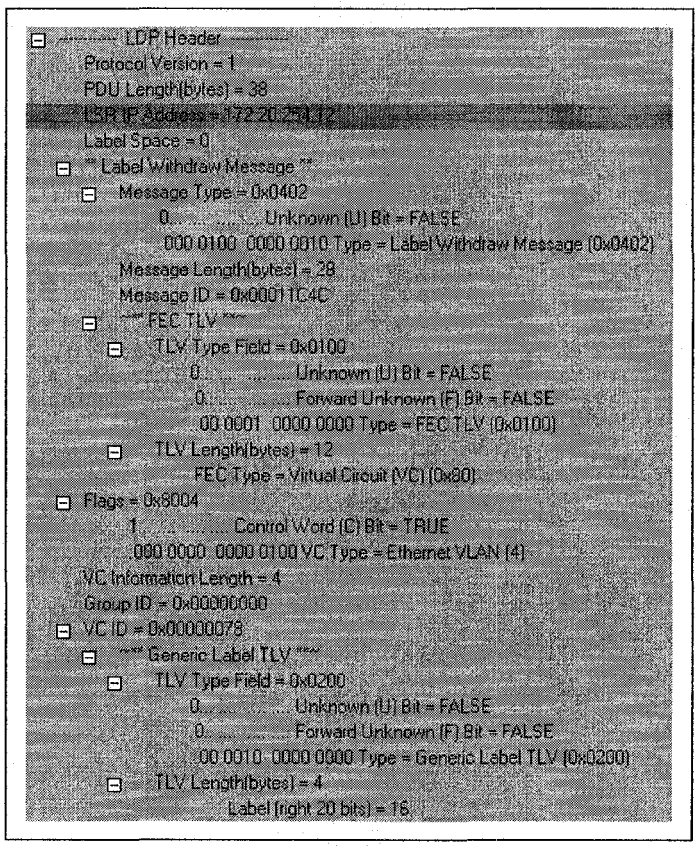


Figure 3.38 ***Désactivation du tunnel AToM.***

ANNEXE 2

COMMANDES DE LA QdS DANS DISCO 3750 CATALYST [45]

Tableau 3.12

Configuration des seuils

Commande	Description
mls qos queue-set output *qset-id* **buffers** *allocation1 ... allocation4*	Allouez des tampons à un *queue-set.*
mls qos queue-set output *qset-id* **threshold** *queue-id drop-threshold1 drop-threshold2 reserved-threshold maximum-threshold*	Configurez les seuils WTD, garantissez la disponibilité de tampons et configurez l'allocation de la mémoire maximale pour le faire la *queue-set* (quatre files de sortie par port).
interface *interface-id*	Spécifiez le port de la circulation du trafic et entrez dans le mode de la configuration de l'interface.
queue-set *qset-id*	Faire un *mapping* entre le numéro de port et la queue-set. Pour qset-id, entrez le ID de la queue-set spécifié dans l'étape 2. La gamme est de 1 à 2. La valeur par défaut est 1.
show mls qos interface [*interface-id*] **buffers**	Vérification

Tableau 3.13

Calcul de la taille totale des entêtes dans L2VPN

Commande	Description
mls qos srr-queue output dscp-map queue *queue-id* **threshold** *threshold-id dscp1...dscp8* ou bien **mls qos srr-queue output cos-map queue** *queue-id* **threshold** *threshold-id cos1...cos8*	Faire un *mapping* entre les valeurs DSCP et les valeurs CoS et la file de sortie ainsi que le seuil ID. Par défaut, les valeurs DSCP 0-15 sont en *mapping* avec la file d'attente 2 et le seuil 1. les valeurs DSCP 16-31 sont en *mapping* avec la file d'attente 3 et le seuil 1. les valeurs DSCP 32-39 et 48-63 sont en *mapping* avec la file d'attente 4 et le seuil 1. les valeurs DSCP 40-47 sont en *mapping* avec la file d'attente 1 et le seuil 1. Par défaut, les valeurs CoS 0-1 sont en *mapping* avec la file d'attente 2 et le seuil 1. les valeurs CoS 2-3 sont en *mapping* avec la file d'attente 3 et le seuil 1. les valeurs CoS 4, 6 et 7 sont en *mapping* avec la file d'attente 4 et le seuil 1. la valeur CoS 5 est en *mapping* avec la file d'attente 1 et le seuil 1.
show mls qos maps	Vérification

Tableau 3.14

Configuration de SRR Shaped Weights

Commande	Description
interface *interface-id*	Spécifiez le port de la circulation du trafic pour entrer dans le mode de la configuration de l'interface.
srr-queue bandwidth shape *weight1* *weight2 weight3 weight4*	Par défaut, le weight1 est mis à 25; les weight2, weight3 et weight4 sont mis à 0, et ces files sont configurées en mode *shared.* Pour weight4 du weight3 du weight2 du weight1, entrez les poids pour contrôler le pourcentage du port qui est en mode *shaped.* Le ratio inverse (1/weight) contrôle la bande passante pour cette file. Séparez chaque valeur avec un espace. Les valeurs doivent être dans la gamme de 0 à 65535. Si vous configurez un poids de 0, la file correspondante opère dans mode *shared.*Le poids spécifié par la commande **srr-queue bandwidth shape** est ignoré, et les poids qui sont spécifiés avec le **srr-queue bandwidth share** sont pris en considération. Quand vous configurez des files dans le même *queue-set*, assurez-vous que vous configurez en mode *shaped* la file d'ordre le plus bas. Le mode *shaped* outrepasse le mode *shared.*

Tableau 3.15

Configuration de SRR Shared Weights

Commande	Description
srr-queue bandwidth share *weight1 weight2 weight3 weight4*	Assigner des poids SRR aux files de sortie. Par défaut, tous les quatre poids sont 25 (1/4 de la bande passante sont alloués à chaque file). Pour *weight4 weight3 weight2 weight1*, entrez les poids pour contrôler le ratio de la fréquence dans laquelle l'Ordonnanceur SRR envoie des paquets. Séparez chaque valeur avec un espace. La gamme des valeurs est de 1 à 255.
show mls qos interface *interface-id* **queueing**	Vérification

Tableau 3.16

Configuration du Expedite Queue

Commande	Description
mls qos	Activer la QdS dans le commutateur.
interface *interface-id*	Spécifier le port de la sortie et entrez la mode de la configuration de l'interface.
priority-queue out	Activer la file d'attente prioritaire qui est désactivée par défaut.

Tableau 3.17

Limiter la bande passante sur l'interface de sortie

Commande	Description
interface *interface-id*	Spécifier le numéro de port et entrez dans le mode de la configuration de l'interface.
srr-queue bandwidth limit *weight1*	Spécifier le pourcentage de la vitesse de port à qui le port devrait être limité. La gamme est 10 à 90.
show mls qos interface [*interface-id*] **queueing**	Vérification
interface *interface-id*	Spécifier le numéro de port et entrez dans le mode de la configuration de l'interface.
srr-queue bandwidth limit *weight1*	Spécifier le pourcentage de la vitesse de port à qui le port devrait être limité. La gamme est 10 à 90.

ANNEXE 3

TEST DE LA FRAGMENTATION DANS L3VPN

Cette section présente les tests de la fragmentation des paquets IP dans L3VPN. On a vu que L2VPN ne fragmente pas les paquets plus larges que le MTU du réseau. Il est important de vérifier si Layer-3 VPN supporte cette option ou non. La figure 3.37 illustre le montage utilisé pour le test de la fragmentation.

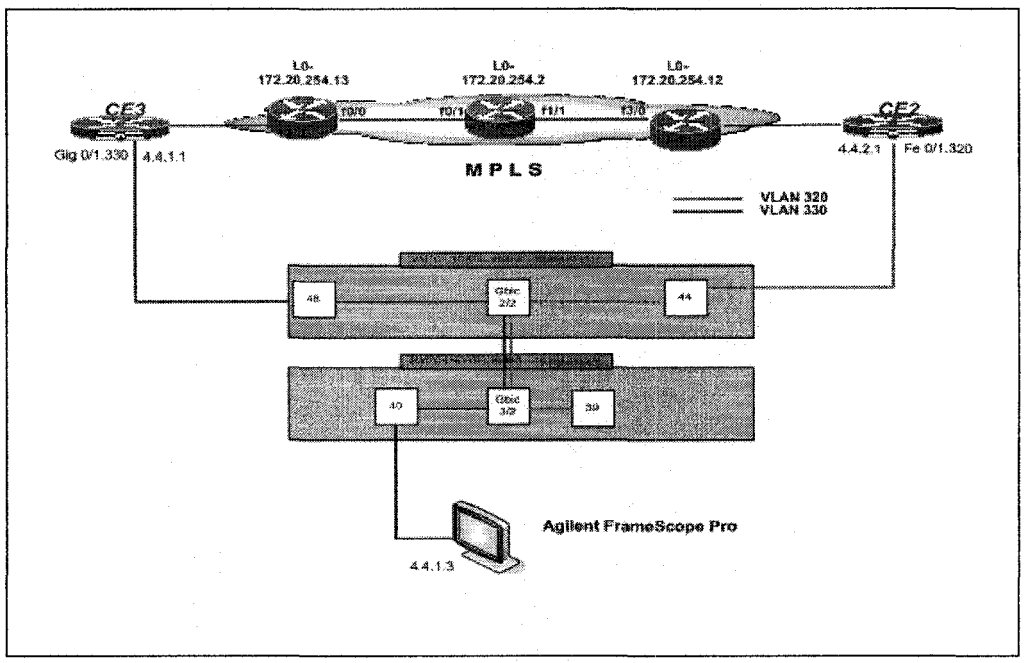


Figure 3.39 ***Montage de test de la fragmentation.***

Dans ce test on génère quelques trames qui dépassent le MTU prédéfini. On utilise un renifleur *Agilent* pour vérifier si les trames Ethernet sont transmises dans le réseau MPLS ou bien sont détruites comme dans le cas de Layer 2 VPN. On utilise l'outil *FrameScop Pro* de *Agilent* pour générer le trafic. Il est possible grâce à cet outil, de générer des trames en définissant leurs nombres et leurs tailles.

Le MTU dans le réseau est de 1500 octets. Le *FrameScope Pro* génère une trame de 1526 octets et on teste si le PE la fragmentera en deux fragments. La figure 3.38 illustre une trace prise dans le réseau MPLS par un renifleur *Agilent DNA* entre PE2 et PE3. Elle montre le premier fragment qui est de taille 1498 octets. Il contient 1456 octets de données. La figure 3.39 montre la deuxième trace prise par le renifleur. La trace montre le deuxième fragment qui est de taille 64 octets. Dans ce fragment il y a seulement 10 octets de données, le reste de la trame est occupé par des bits de bourrage.

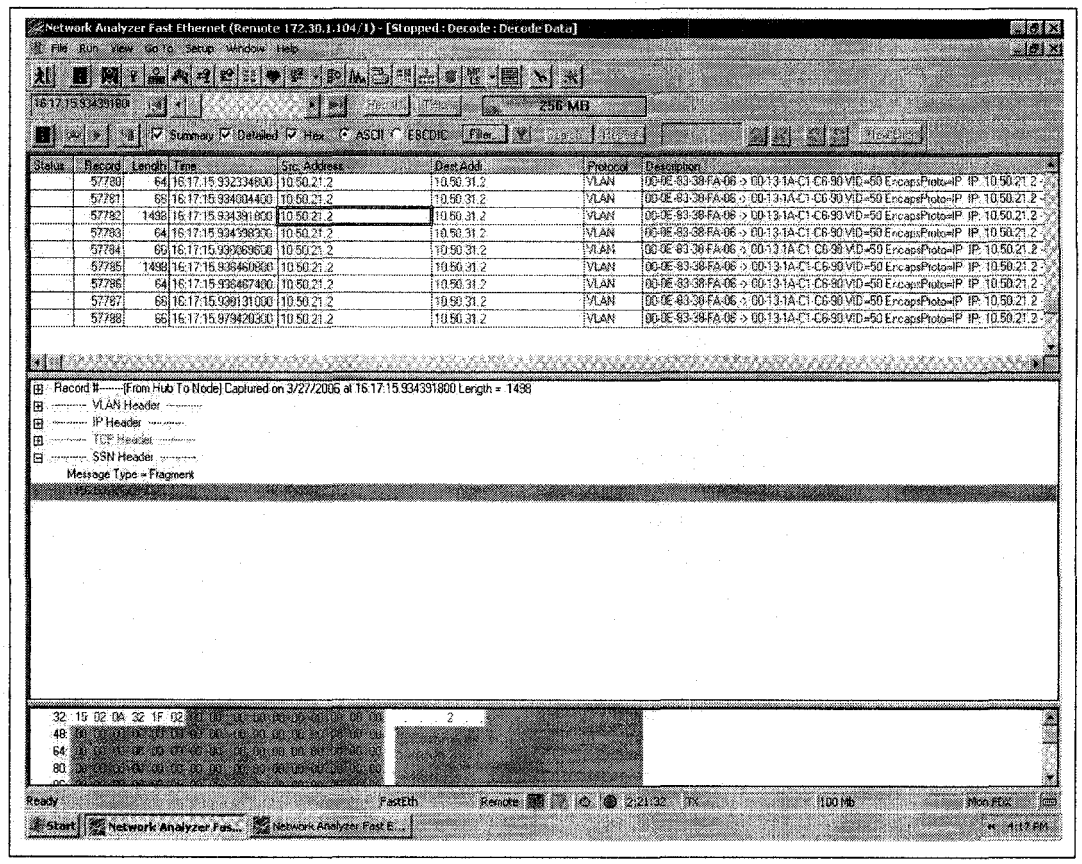


Figure 3.40 ***Le premier fragment.***

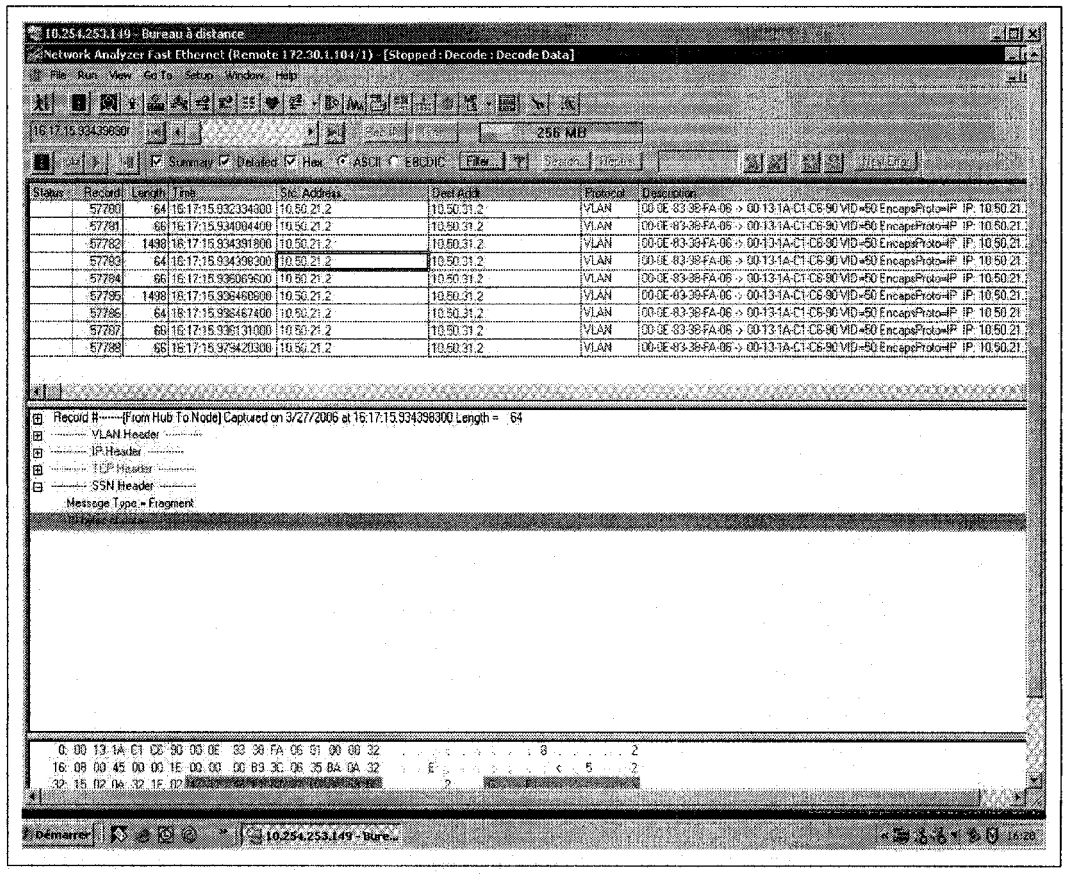


Figure 3.41 ***Le deuxième fragment avec des bits de bourrage.e.***

ANNEXE 4

CODES DES PROCESSUS DU MODÈLE OPNET

8.1 Code de l'état initial *init*

```
packet_count_stat = op_stat_reg("Packet Count", OPC_STAT_INDEX_NONE,
OPC_STAT_LOCAL);
packet_accepted_stat = op_stat_reg("Packet Accepted",
OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
packet_rejected_stat = op_stat_reg("Packet Rejected",
OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
tokens_available_stat = op_stat_reg("Tokens Available",
OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
tokens_residual_stat = op_stat_reg("Tokens Residual",
OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
tokens_added_stat = op_stat_reg("Tokens Added", OPC_STAT_INDEX_NONE,
OPC_STAT_LOCAL);
throughput_received_stat = op_stat_reg("Throughput Received",
OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
throughput_accepted_stat = op_stat_reg("Throughput Accepted",
OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
throughput_CIR_stat = op_stat_reg("Throughput CIR",
OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
// Recuperation de la valeur de l'attribut Bucket Size
node_objid  = op_id_from_name (0, OPC_OBJTYPE_NDFIX, "DDTB");
TB1_objid = op_id_from_name (node_objid, OPC_OBJTYPE_PROC, "TB1");
op_ima_obj_attr_get (TB1_objid, "Bucket Size", &bucket_size);
//Initialisation du parametre tokens_residual
tokens_residual = bucket_size;
//Calcule de CIR
throughput_CIR = bucket_size*32;
//Parametre X pour l'interuption
X = op_sim_time()+10;
```

8.2 Code de l'état *Token_Bucket*

```
//Aller chercher le pointeur du packet
pkptr = op_pk_get(op_intrpt_strm());
++packet_count; //On incrémente le compteur packet_count qui
calcule le nombre des paquets reçus
op_stat_write(packet_count_stat, packet_count);

//Calcule de la taille du paquet en bits
pk_len1 = op_pk_total_size_get (pkptr);
throughput_received = throughput_received + pk_len1;

//Calcule de la somme des tailles des paquets recus par les TBs
node_objid  = op_id_from_name (0, OPC_OBJTYPE_NDFIX, "DDTB");
TB2_objid = op_id_from_name (node_objid, OPC_OBJTYPE_PROC, "TB2");
pk_len2_ptr = (OpT_Packet_Size *) op_ima_obj_svar_get (TB2_objid,
"pk_len2");
pk_len_total = pk_len1 + *pk_len2_ptr;

//Calcule du nombre des jetons ajoutés
queue_objid = op_id_from_name (node_objid, OPC_OBJTYPE_QUEUE,
"Queue");
free_bit_ptr = (double *) op_ima_obj_svar_get (queue_objid,
"free_bit");
queue_threshold = (double *) op_ima_obj_svar_get (queue_objid,
"queue_threshold");
throughput_received_ptr = (OpT_Packet_Size *) op_ima_obj_svar_get
(TB2_objid, "throughput_received");
throughput_accepted_ptr = (OpT_Packet_Size *) op_ima_obj_svar_get
(TB2_objid, "throughput_accepted");
throughput_CIR_ptr = (OpT_Packet_Size *) op_ima_obj_svar_get
(TB2_objid, "throughput_CIR");
p_ptr=(OpT_Packet_Size *) op_ima_obj_svar_get (TB2_objid, "p2");
p2=*p_ptr;
```

```
free_bit = *free_bit_ptr;


threshold = throughput_CIR+((4000000-throughput_CIR-
*throughput_CIR_ptr)*throughput_CIR/(throughput_CIR+*throughput_CIR
_ptr));
max_allowable_tokens = free_bit*pk_len1/pk_len_total;
if(max_allowable_tokens >= pk_len1)
max_allowable_tokens = pk_len1;

if(max_allowable_tokens >= tokens_residual)
      tokens_added = max_allowable_tokens - tokens_residual;
else
      tokens_added = 0;
op_stat_write(tokens_added_stat,tokens_added);

//Calcule du nombre des jetons disponibles dans le bucket
tokens_available = tokens_added + tokens_residual;
if (tokens_available >= bucket_size)
tokens_available = bucket_size; //les jetons qui depassent la
taille du bucket sont rejetés
op_stat_write(tokens_available_stat, tokens_available);
if ( tokens_available < pk_len1 )
{     op_pk_destroy (pkptr); // On détruit le paquet
      tokens_residual = tokens_available; // On ne tranche pas la
taille du paquet de la taille du bucket
      ++packet_rejected;
      op_stat_write(packet_rejected_stat, packet_rejected);
}
else
{     if (p1<2)
      {     tokens_residual = tokens_available - pk_len1;
            if (*queue_threshold >= 0.9)
               op_pk_send_delayed (pkptr, 0, 1);
            else
```

```
                op_pk_send(pkptr, 0); // On envoit le paquet vers la
file d'attente
                ++packet_accepted;
                throughput_accepted  = throughput_accepted +
pk_len1;
                op_stat_write(packet_accepted_stat,
packet_accepted);
            }
      else
      {     if(p2<2)
            {if (throughput_accepted + *throughput_accepted_ptr >
3900000)
                        {
            op_pk_destroy (pkptr);
            ++packet_rejected;
            op_stat_write(packet_rejected_stat, packet_rejected);
      }
            else
            { tokens_residual = tokens_available - pk_len1;
            if (*queue_threshold >= 0.9)
                  op_pk_send_delayed (pkptr, 0, 1);
            else
                  op_pk_send(pkptr, 0); // On envoit le paquet vers
la file d'attente
                  ++packet_accepted;
                  throughput_accepted = throughput_accepted +
pk_len1;
                  op_stat_write(packet_accepted_stat,
packet_accepted);
             }
         }
            else
             {if (throughput_accepted >= threshold)
                  { op_pk_destroy (pkptr);
                   ++packet_rejected;
```

```
                    op_stat_write(packet_rejected_stat,
packet_rejected);
                  }
               else
                  { tokens_residual = tokens_available - pk_len1;
                    if (*queue_threshold >= 0.9)
                         op_pk_send_delayed (pkptr, 0, 1);
                    else
                        op_pk_send(pkptr, 0); //On envoit le paquet
vers la file d'attente
                  ++packet_accepted;
                  throughput_accepted  = throughput_accepted +
pk_len1;
                  op_stat_write(packet_accepted_stat,
packet_accepted);
                  }
                     }
            }}    op_stat_write(tokens_residual_stat,
tokens_residual);
```

8.3 Code de l'état *Throughput*

```
if(throughput_received <= throughput_CIR)
      p1=1;
else
      p1=2;

//Enregistrement des statistiques pour les débits

op_stat_write(throughput_received_stat, throughput_received);
op_stat_write(throughput_accepted_stat, throughput_accepted);
op_stat_write(throughput_CIR_stat, throughput_CIR);

//Réinitialisation des compteurs de débits
throughput_received = 0;
throughput_accepted = 0;
```

BIBLIOGRAPHIE

[1] I. pepelnjak and J. Guichard, *MPLS AND VPN ARCHITECTURES*: Cisco Press, 2001.

[2] B. Davie and Y. Rekhter, *MPLS: Technology and Applications*: Morgan Kaufmann, 2000.

[3] E. Rosen, D. Tappan, G. Fedorkow, Y. Rekhter, D. Farinacci, T. Li, and A. Conta, "RFC 3032 : MPLS Label Stack Encoding," 2001.

[4] A. Nagarajan, "RFC 3809: Generic Requirements for Provider Provisioned Virtual Private Networks (PPVPN)," 2004.

[5] E. Rosen and Y. Rekhter, "RFC 2547: BGP/MPLS VPNs," 1999.

[6] P. Knight and B. Gleeson, "Internet Draft: Network based IP VPN Architecture Using Virtual Routers, draft-ietf-l3vpn-vpn-vr-03.txt," 2006.

[7] L. Andersson and E. C. Rosen, "Internet Draft: Framework for Layer 2 Virtual Private Networks (L2VPNs)," 2004.

[8] X. Xiao, D. McPherson, and P. Pate, "RFC 3916: Requirements for Pseudo-Wire Emulation Edge-to-Edge (PWE3)," 2004.

[9] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and B. Thomas, "RFC 3036 : LDP Specification," 2001.

[10] L. Martini, E. Rosen, G. Heron, and N. El-Aawar, "RFC 4448: Encapsulation Methods for Transport of Ethernet Over MPLS Networks," 2005.

[11] H. Shah and E. Rosen, "Internet Draft : IP-Only LAN Service (IPLS) Draft-ietf-l2vpn-ipls-06.txt," 2006.

[12] M. Lasserre and V. Kompella, "Internet Draft: Virtual Private LAN Services over MPLS - draft-ietf-l2vpn-vpls-ldp-07.txt," 2005.

[13] RiverstoneNetworks, "Scalability of Ethernet Services Networks," 2004.

[14] L. Lobo and U. Lakshman, *MPLS Configuration on Cisco IOS Software*, vol. 1. Indianapolis, Indiana, 2005.

[15] Y. Rekhter and P. Gross, "RFC 1268: Application of the Border Gateway Protocol in the Internet," 1991.

[16] Y. Rekhter and T. Li, "RFC 1771 : A Border Gateway Protocol 4 (BGP-4)," 1995.

[17] E. Rosen, W. Luo, B. Davie, and V. Radoaca, "Internet-Draft : Provisioning, Autodiscovery, and Signaling in L2VPNs - draft-ietf-l2vpn-signaling-08.txt," 2006.

[18] L. Zier, W. Fischer, and F. Brockners, "Ethernet-Based Public Communication Services: Challenge and Opportunity," *IEEE Communications Magazine*, 2004.

[19] J. Ash, B. Goode, J. Hand, and R. Zhang, "RFC 4247 : Requirements for Header Compression over MPLS," 2005.

[20] H. G. Perros, *Connection-Oriented Networks: SONET/SDH, ATM, MPLS and Optical Networks*: John Wiley & Sons, 2005.

[21] D. Minoli, *Voice over MPLS - PLANNING AND DESIGNING NETWORKS*: McGraw-Hill Telecom, 2002.

[22] M. Allman, V. Paxson, and W. Stevens, "RFC 2581 : TCP Congestion Control," 1999.

[23] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "RFC 2574 : An Architecture for Differentiated Services," 1998.

[24] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, "RFC 2702 : Requirements for Traffic Engineering Over MPLS," 1999.

[25] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "RFC 2205 : Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification," 1997.

[26] B. Tremblay, "ALGORITHMES DE GESTION DYNAMIQUE DES RESSOURCES," in *Génie logiciel et TI*. Montréal: École de technologie supérieure, 2006.

[27] A. Kankkunen, G. Ash, A. Chiu, J. Hopkins, J. Jeffords, F. L. Faucheur, B. Rosen, D. Stacey, A. Yelundur, and L. Berger, "Internet Draft : VoIP over MPLS Framework <draft-kankkunen-vompls-fw-01.txt>," 2000.

[28] R. Cherukuri and T. Walsh, "Voice over MPLS – Bearer Transport Implementation Agreement," *MPLS Forum 1.0*, 2001.

[29] D. Wright, "Voice over MPLS compared to voice over other packet transport technologies," *IEEE Communications Magazine*, 2002.

[30] B. Davie, J. Lawrence, K. McCloghrie, E. Rosen, G. Swallow, Y. Rekhter, and P. Doolan, "RFC 3035 : MPLS using LDP and ATM VC Switching," 2001.

[31] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RFC 3550 : RTP A Transport Protocol for Real-Time Applications," 2003.

[32] S. Casner and V. Jacobson, "RFC 2508 : Compressing IP/UDP/RTP Headers for Low-Speed Serial Links," 1999.

[33] J. Postel, "RFC 791 : INTERNET PROTOCOL," 1981.

[34] K. Kompella, "Internet Draft : Layer 2 VPNs Over Tunnels - draft-kompella-l2vpn-l2vpn-01.txt," 2006.

[35] J. Mogul and S. Deering, "RFC 1063: Path MTU Discovery," 1990.

[36] P. Savola, "RFC 4459 : MTU and Fragmentation Issues with In-the-Network Tunneling," 2006.

[37] W. Odom and M. J. Cavanaugh, *IP Telephony self Study, Cisco QOS Exam Certification Guide*, Second ed: Cisco Press, 2005.

[38] M.-A. Breton and N. Manen, *Rapport des tests de validation des mécanismes de QoS - Projet Bell Canada*, 2006.

[39] M. Breton, "Développement d'un système de surveillance des mécanismes de qualité de service dans le contexte des réseaux de prochaine génération," École de technologie supérieure. Mémoire de maîtrise, Montréal 2006.

[40] S. Shenker, C. Partridge, and R. Guerin, "RFC 2212 : Specification of Guaranteed Quality of Service," 1997.

[41] C. Semeria, "Supporting Differentiated Service Classes : Multiprotocol Label Switching (MPLS)," 2002.

[42] M. E. HACHIMI, M. Breton, and M. Bennani, "QoS for MPLS-based Virtual Private Networks," presented at Conférence Internationale sur les NOuvelles TEchnologies de la REpartition, NOTERE, 2007.

[43] N. U. Ahmed, L. Barbosa, and Q. Wang, "Systems Approach to Modeling the Token Bucket Algorithm in Computer Networks," *Mathematical Problems in Engineering*, vol. 8(3), 2002.

[44] N. U. AHMED, T. DABBOUS, and Y. W. LEE, "Dynamic routing for computer queueing networks," *International journal of systems science*, 1988.

[45] Cisco, "Catalyst 3750 Switch Software Configuration Guide-Cisco IOS Release 12.2(25) SE," 2004.