Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université Virtuelle de Tunis

UVT
جامعة تونس الافتراضية
UNIVERSITÉ VIRTUELLE DE TUNIS

MEMOIRE

DE STAGE DE FIN D'ETUDE

Pour l'obtention du

MASTERE PROFESSIONNEL

« Nouvelles Technologies des Télécommunications et Réseaux »

Présenté par :
Nizar Saâda

Etude et optimisation d'un backbone IP/MPLS

Soutenu le : 05/02/2014

Devant le jury :

Mr. Khaled Jlassi	**Président**
Mme. Houda Rekaya Houissa	**Membre**
Mme. Ahlem Ben Hsin	**Membre**

www.uvt.rnu.tn
14, Rue Yahia Ibn Omar - 1082 Mutuelleville - Tunis
Tél. : +216 71 28 99 81 / +216 71 89 17 31 - Fax : +216 71 89 26 25

Dédicace

À toute la famille Saâda
À qui je dois tout

Espérant trouver dans ce travail le fruit de vos sacrifices et l'expression de ma profonde gratitude, amour et affection pour la patience et pour tout ce que vous avez consentis pour moi et dont je suis à jamais redevable.

Vous avez tant attendu ce moment que je vous offre avec un très grand plaisir.

Que dieu vous préserve une bonne santé et une longue vie.

À tous les amis et les relatives

En témoignage de ma profonde estime et mon amour, je leur souhaite bonheur et prospérité dans leur vie familiale et professionnelle.

Nulle expression ne peut témoigner ma reconnaissance et mes délicats sentiments envers eux.

Remerciements

Aux termes de ce travail, je tiens à remercier, Mr. Nabil Tirellil, le Directeur des Systèmes, des Réseaux et de la Maintenance, pour sa compréhension et son aide durant toute la période de stage et Mr. Jounaydi Malek pour son aide et ses conseils.

Je tiens également à remercier Mr. Tahar Ezzedine, mon encadreur, pour ses conseils, son assistance et pour le temps qu'il m'a accordé durant la période du stage.

A tous ceux qui ont contribué, de près ou de loin, à la finalisation de ce travail.

Table des matières :

Liste des figures :

Liste des tableaux :

Glossaire :

ADSL: Asymmetric Digital Subscriber Line

API: Application Programming Interface

ATM: Asynchronous Transfer Mode

BS: Bottom of Stack

CBR: Constant Bit Rate

CEF: Cisco Express Forwarding

CNAM: Caisse Nationale d'Assurance Maladie

CNSS: Caisse Nationale de Sécurité Sociale

CoS: Class of Service

CR-LDP: Constraint based Routing-LDP

CSPF: Constraint Shortest Path First

DCI: Data Center Interconnect

DLCI: Data Link Connection Identifier

DV: Distance Vector

ELSR: Edge Label Switching Router

ETTB: Ethernet To The Business

ETTH: Ethernet To The Home

FEC: Forwarding Equivalence Class

FTP: File Transfer Protocol

GMPLS: Generalized MPLS

GNS: Graphical Network Simulator

GNU: General Public License

HTTP: HyperText Transfer Protocol

IETF:	Internet Engineering Task Force
IGP:	Internal Gateway Protocol
IP:	Internet Protocol
IPSec:	Internet Protocol Security
IS-IS:	Intermediate System to Intermediate System
LDP:	Label Distribution Protocol
LFIB:	Label Forwarding Information Base
LIB:	Label Information Base
LS:	Ligne Spécialisée
LSP:	Label Switched Path
LSR:	Label Switching Router
MAC:	Media Access Control
MP-BGP:	Multi-Protocol Border Gateway Protocol
MPLS:	Multi-Protocol Label Switching
NAM:	Network AniMator
NS:	Network Simulator
OS:	Operating System
OSI:	Open System Interconnection
OSPF:	Open Shortest Path First
OTCL:	Object Tools Command Language
PE:	Provider Edge
PoP:	Point of Presence
PPP:	Point to Point Protocol
PRTG:	Paessler Router Traffic Grapher
PXE:	Pre-boot eXecution Environment
QoS:	Quality of Service
RFC:	Request For Comments
RIP:	Routing Information Protocol

RSVP:	Resource ReSerVation Protocol
SNMP:	Simple Network Management Protocol
SPF:	Short Path First
TCL:	Tool Command Language
TE:	Traffic Engineering
TDP:	Tag Distribution Protocol
TDSP:	Two Disjoint Shortest Paths
TT:	Tunisie Telecom
TTL:	Time To Live
VCI:	Virtual Component Interface
VC:	Virtual Circuit
VFI:	Virtual Forwarding Instance
VoIP:	Voice over IP
VPI:	Virtual Path Identifier
VPLS:	Virtual Private LAN Services
VPN:	Virtual Private Network
WAN:	Wide Area Network

Introduction Générale

Avec l'évolution des tailles des entreprises, la croissance des systèmes d'information et la diversification des besoins des applications dans le domaine de transmission de données, la gestion des multiservices s'avère primordiale pour instaurer la notion Qos dans les réseaux.

Certes le développement internet et la simplicité du protocole IP font de celui-ci un protocole presque universel, mais son aspect non connecté implique une difficulté d'intégration de service temps réel qui exigent un certain degré de Qos.

Les technologies réseaux qui ont suivi IP et qui visent à améliorer l'acheminement des données ont certes essayé de remédier au problème de Qos, principalement les réseaux ATM pour le lancement de la transmission des données en temps réel en gérant les classes de trafic.

Mais vu le coût élevé des commutateurs ATM et la difficulté de faire cohabiter ATM avec d'autres technologies réseau, il a fallu concevoir une technologie de commutation qui permettra une meilleure Qos avec la souplesse et la possibilité d'intégration sur différents types de réseaux (Ethernet, FR, ATM…), d'où l'apparition de la technologie de commutation par étiquettes ou MPLS (Multi Protocol Label Switching).

MPLS représente une solution basée sur le principe de commutation de circuit en remédiant au problème de gaspillage des ressources par la gestion des priorités dans le trafic à faire circuler.

Le réseau de télécommunication de la CNSS, basé sur la technologie MPLS, utilise pleinement le backbone MPLS de Tunisie Télécom. Les lignes et les équipements de routage sont loués auprès de Tunisie Télécom.

En effet, durant ce stage effectué au sein de la CNSS, j'ai essayé d'exploiter d'une façon meilleure le réseau MPLS.

Mon projet consiste à simuler un backbone MPLS autonome dans lequel la gestion des équipements de routage et des lignes de télécommunication (louée d'une tierce partie) sera à la charge de la CNSS. Ainsi les critères de QoS seront mis en évidence ce qui peut impliquer une réduction des coûts et une amélioration de la rentabilité du réseau de la CNSS.

Le rapport est composé de trois chapitres : Le premier chapitre représente l'étude de l'existant. Le deuxième chapitre aborde MPLS, ses différents composants et protocoles. Et finalement, le dernier chapitre est consacré à la présentation des étapes détaillées de réalisation.

Chapitre 1

Etude de l'existant

1. Introduction :

Toute installation d'une nouvelle architecture réseau dans un établissement doit être précédée par une phase d'analyse et d'étude de trafic existant pour dégager les problèmes afin de concevoir une nouvelle architecture plus adéquate.

Dans ce chapitre, je vais d'abord présenter l'organisme d'accueil, puis exposer le cahier des charges et enfin décrire l'architecture existante du réseau de la CNSS avec une analyse détaillée.

2. Présentation de l'organisme d'accueil CNSS :

Dans cette partie, je vais présenter la mission et la hiérarchie de la CNSS ainsi que son département informatique.

2.1. Description de la CNSS :

Selon l'article 4 de la loi 60-30 du 14 décembre 1960, la Caisse Nationale de Sécurité Sociale est un établissement public doté de la personnalité civile et de l'autonomie financière et rattaché au ministère des affaires sociales. Son siège est à Tunis et elle regroupe divers bureaux régionaux tout au long du territoire tunisien.

D'après l'article premier de la même loi, la mission de cet organisme consiste à protéger les travailleurs et leurs familles contre les risques susceptibles d'affecter les conditions matérielles et morales de leur existence.

Bénéficiant des régimes de sécurité sociale, le personnel de bureau et le personnel ouvrier rattachés sous quelques formes que ce soit à toutes les personnes morales de droit public ou de droit privé siégeant en Tunisie et qui ne sont pas affiliés à un régime légal de sécurité sociale couvrant les mêmes éventualités surveillées. **[1]**

2.2. L'organisation générale de la CNSS :

La CNSS est placée sous la tutelle du ministre des affaires sociales. Elle est administrée par un conseil d'administration constitué d'un Président Directeur Général et de douze administrateurs.

L'organigramme de la CNSS est composé de plusieurs directions dont la direction centrale des études informatiques, les bureaux régionaux et les polycliniques. Chaque bureau régional comprend plusieurs unités suivantes : l'unité employeur, l'unité prestataire, l'unité contrôle médical, l'unité contrôle comptable et technique, l'unité administrative et l'unité de gestion des comptes au profit de l'Etat. **[T.L]**

Le schéma suivant indique la hiérarchie de la CNSS qui comporte des bureaux, directions…

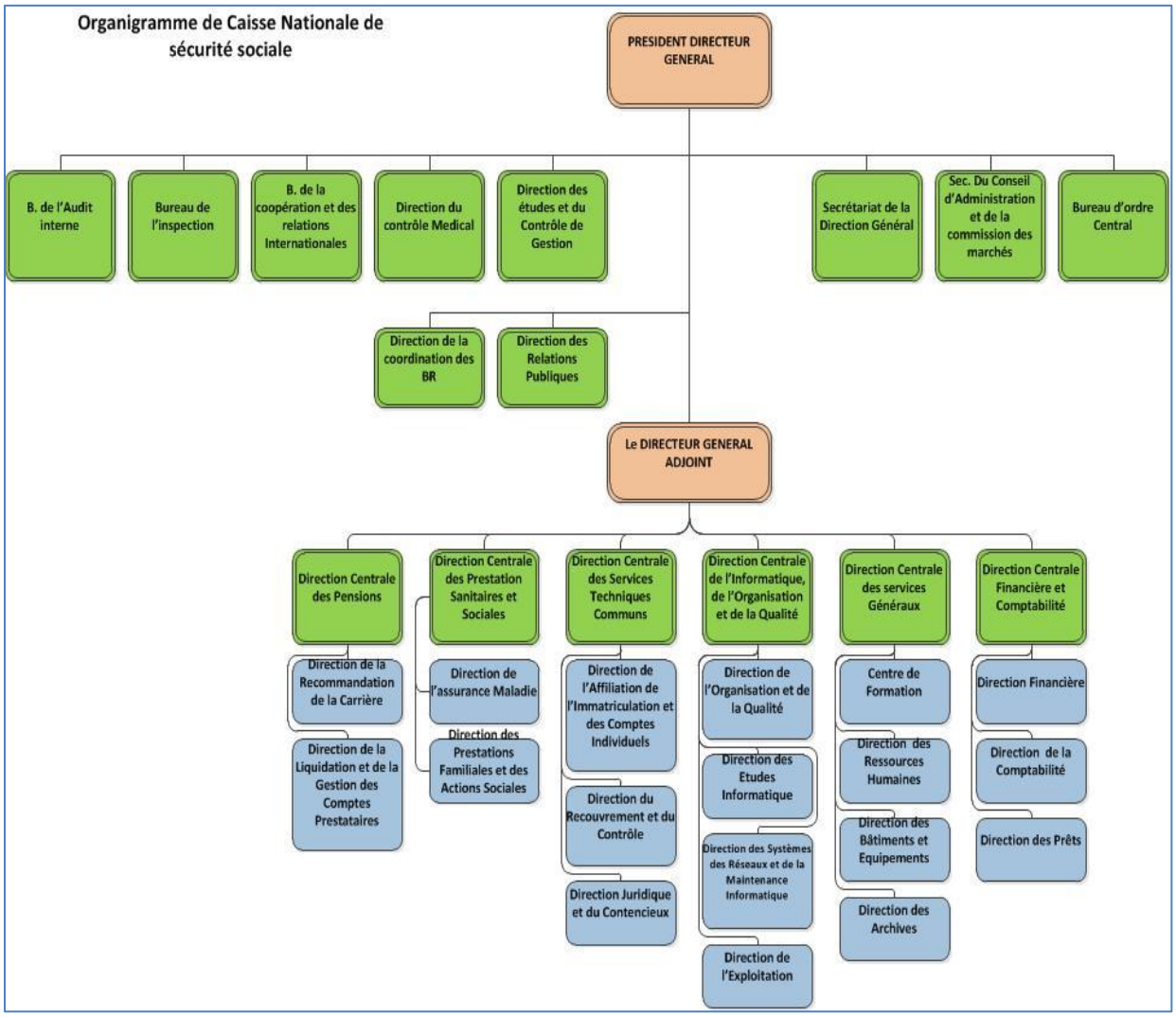


Figure 1.1 : Organigramme de la CNSS

2.3. Présentation du département informatique de la CNSS :

La direction informatique a pour mission de développer le système informatique avec le maximum d'efficacité et d'économie afin de fournir une meilleure qualité de service, d'élaborer un plan général d'informations et de suivre son exécution en collaboration avec les services intéressés.

Elle est chargée de la promotion de l'utilisation des nouvelles techniques de l'informatique et de communication en vue d'améliorer la gestion et la qualité de service.

En plus, elle veille sur la sauvegarde de l'intégrité et la sécurité des données et des programmes de la CNSS conformément à la législation en vigueur.

3. Présentation du projet :

Pour bien expliquer mon projet, je vais le présenter dans son contexte général puis dans son contexte spécifique.

3.1. Architecture globale de la CNSS :

Le réseau se compose de plusieurs sites distants classés en trois catégories :

- (50) Bureaux régionaux.
- (6) Polycliniques.
- (8) Directions.

Ces différents sites sont liés à la Direction Centrale de l'Informatique via le backbone MPLS de la Tunisie Telecom et en utilisant un routage dynamique basé sur le protocole OSPF ainsi qu'un routage statique pour configurer certaines routes spécifiques.

3.2. Architecture WAN de la CNSS :

Les lignes d'interconnexion WAN utilisées au sein de la CNSS sont de trois types :

- Des lignes spécialisées (LS).
- Des lignes ADSL (Backup).
- Des fibres optiques.

Les équipements de routage utilisés sont :

- Des switches Cisco ME 3400G au niveau de la Direction Centrale d'Informatique : Ce sont les premiers commutateurs d'accès optimisés à la fois pour ETTH des services Triple-Play (la voix et la vidéo) et pour ETTB des services VPN. Il fournit une solution de sécurité complète pour Metro Ethernet qui comprend l'accès abonné, switch et la protection du réseau. Ces switches prennent en charge plusieurs images logicielles pour plus de flexibilité dans le modèle de déploiement.

Figure 1.2 : Switche Cisco ME 3400G

- Des routeurs Cisco 1921 au niveau des sites distants : Leur architecture est conçue afin de prendre en charge la phase de l'évolution des sites distants en leur proposant des services de virtualisation et de collaboration multimédia tout en optimisant les coûts d'exploitation. Ces routeurs à services intégrés de deuxième génération sont parés pour l'avenir grâce à des processeurs multi-cœurs, de commutation Gigabit Ethernet et de nouvelles fonctionnalités de contrôle et de surveillance des consommations énergétiques tout en améliorant les performances globales.

Figure 1.3 : Routeur Cisco 1921

3.3. Contexte spécifique de projet :

L'échange de flux entre les sites de la CNSS doit être fiable et performant. La performance de l'architecture est liée à deux critères suivants :

- Les équipements réseaux utilisés : Performance des routeurs et des switchs.
- La performance software : Technologie et protocoles de routage implémentés.

Le choix de la technologie de routage est un facteur indispensable dans l'implémentation de la QoS en particulier pour les paramètres suivants :

- Le choix du meilleur chemin : Utiliser des protocoles de routage basés sur le plus court chemin calculé suivant les poids ou le nombre de sauts.
- La stabilité du réseau : Faire intégrer des coûts basés sur la QoS (délai, disponibilité…).
- L'amélioration des performances (temps de réponse, routage…) du réseau de la CNSS est un objectif principal de la Direction Centrale de l'Informatique.

L'une des solutions efficaces proposées est d'installer un backbone MPLS propre à la CNSS.

4. Cahier des charges :

Dans l'objectif d'améliorer les performances de réseau, je dois implémenter un backbone MPLS au sein de la CNSS. Les objectifs de mon travail sont :

⇨ Etudier le trafic : Mesurer le débit nécessaire pour les différents sites de la CNSS (bureaux régionaux, directions et policliniques).
⇨ Etudier l'architecture existante de la CNSS comme étant un client IP/MPLS chez Tunisie-Telecom : Simuler l'architecture existante et poser les problèmes.
⇨ Déduire le backbone MPLS : Présenter la nouvelle architecture du backbone MPLS à l'aide des résultats de l'étude de trafic.
⇨ Comparaison des deux architectures : Dégager les avantages de la nouvelle architecture par rapport à la première.
⇨ Simulation et implémentation du nouveau réseau MPLS.

5. Architecture existante :

La CNSS utilise le backbone MPLS de TT pour l'acheminement de ses données. Le trafic de ses différents sites traverse les LSRs de l'opérateur pour arriver à la Direction Centrale de l'Informatique.

Chaque site possède une ligne de secours (ADSL) utilisé en cas de panne pour assurer la continuité de service et éviter toute perte de données.

La figure suivante illustre l'architecture existante du réseau de la CNSS avec ses différentes composantes :

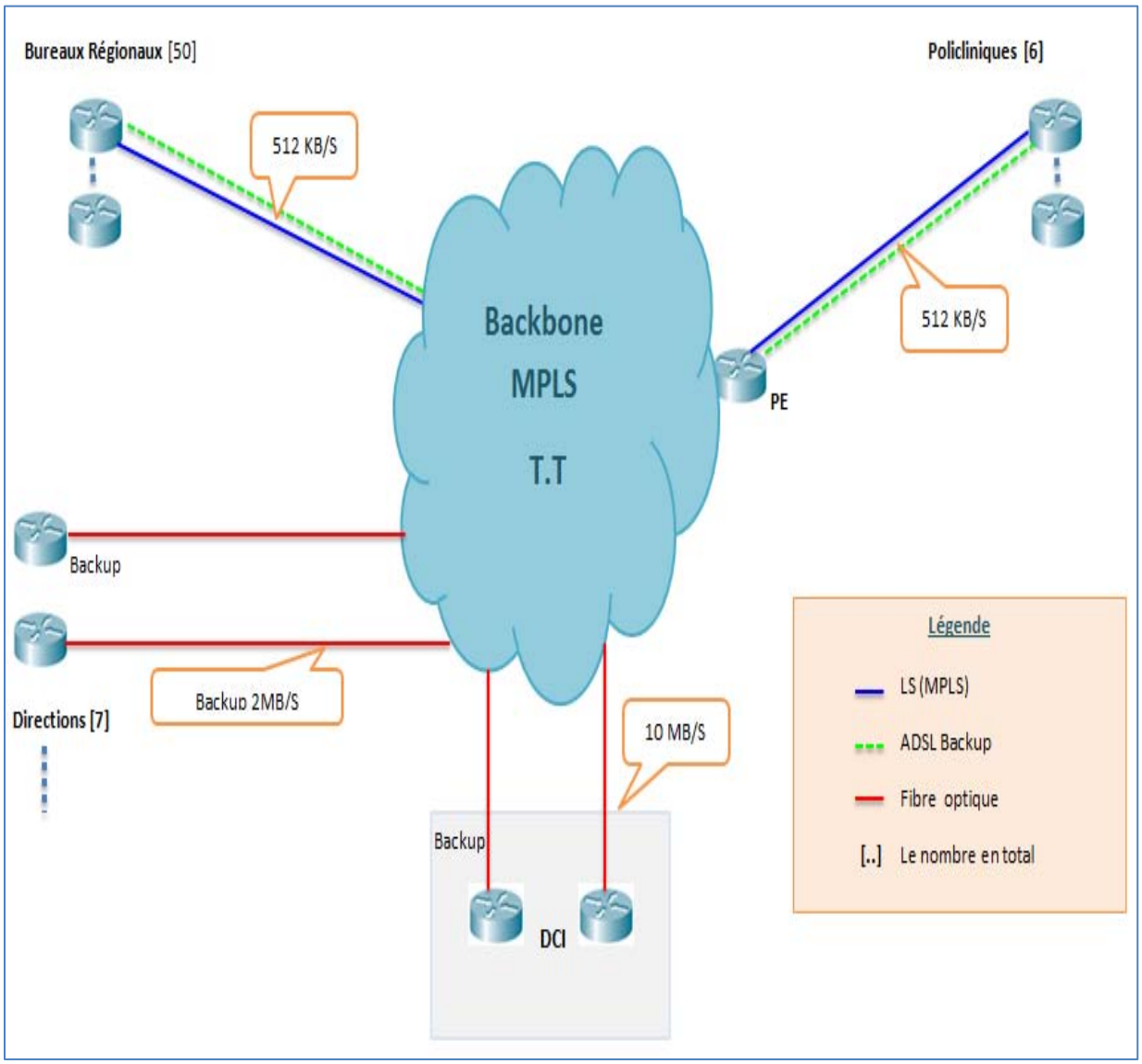


Figure 1.4 : Architecture actuelle de la CNSS

Au cours de la phase d'étude de trafic, le débit utilisé au niveau de chaque site est mesuré pour faciliter la tâche de simulation.

Après avoir activé les agents SNMP dans les différents routeurs, l'outil de calcul de débit utilisé est le logiciel PRTG qui est une interface de programmation d'application permettant d'accéder aux données de surveillance, de manipuler des objets à l'aide de requêtes HTTP, exécuter des capteurs et avis écrits et de personnaliser l'interface web.

J'ai donc dégagé la matrice présentée dans le tableau qui suit :

Site CNSS	*Débit moyen (Kb/s)*	*Site CNSS*	*Débit moyen (Kb/s)*
Ben Arous	300	Hammam Sousse	50
Mannouba	200	Sousse	30
Ariana	300	Monastir	150
Kram	200	Ksar Hellal	50
Bardo	30	Msaken	15
BRTN	200	Kairouan	50
BRTS	200	Kasserine	30
BRTB	200	Mahdia	60
Hammam Lif	150	El jam	100
Menzel Bourguiba	30	Sidi Bouzid	40
Bizerte	250	Jbeniana	10
Ras Djebel	100	Sfax Nord	50
Menzel Temim	300	Sfax Centre	200
Slimen	100	Sakiet Ezzit	200
Nabel	250	Gafsa	100
Zaghouan	50	Metlawi	100
Béja	200	Touzeur	200
Tabarka	50	Gabes	100
Bou Selem	150	Djerba	100
Jandouba	200	Kebili	80
El Kef	200	Mednine	50
Siliana	350	Tataouine	150

Tableau 1.1 : Matrice de débit

Les sites de la CNSS sont distribués aux divers PoPs de TT. Le backbone MPLS de ce fournisseur comprend 18 LSRs dispersés sur tout le territoire du pays. Ces routeurs sont liés par des lignes qui supportent un débit qui varie autour de 3 Gigabits.

Les routeurs de la CNSS sont affectés aux LSRs de TT suivant le tableau ci-dessous :

Nœud TT	*Nœud CNSS*	
	Numéro	*Nom*
0	29	Ras Ejbal
	30	Menzel Temim
	31	Slimen
1	43	Ksar Helal
	44	Mseken
	45	Kairouan
	46	Kasserine
	47	Mehdia
	48	El jam
2	32	Nabeul
	33	Zaghouan
	34	Béja
	35	Tabarka
	36	Bou Selem
4	37	Jendouba
	38	El Kef
	39	Seliana
	40	Hamam Sousse
	41	Sousse
	42	Monastir
5	49	Sidi Bouzid
	50	Jbeniana
	51	Sfax Nord
	52	Sfax Centre
	53	Sakiet Ezzit
	54	Gafsa
	55	Metlawi

Nœud TT	*Nœud CNSS*	
	Numéro	*Nom*
5	56	Tozeur
	57	Gabes
	58	Jerba
12	59	Kebili
	60	Médenine
	61	Tataouine
15	18	Ben Arous
	19	Mannouba
	20	Ariana
	21	Kram
	22	Le bardo
16	23	BRTN
	24	BRTS
	25	BRTB
	26	Hamam Lif
	27	Menzel Bourguiba
	28	Bizerte
17	62	DCI

Tableau 1.2 : Répartition des nœuds CNSS aux LSRs

6. Tests et fiabilité :

Cette phase de test permettra de souligner les points forts et de détecter les anomalies de cette architecture.

A partir des mesures, j'ai simulé l'architecture existante par « NS2 » qui est un logiciel libre de simulation à événements discrets, largement utilisé dans la recherche académique et dans l'industrie. Et par la suite, j'ai procédé à l'installation des agents suivant le type de trafic à fin de tester sa fiabilité.

Le trafic des données sur le réseau est divisé essentiellement en trois types :

- Trafic oracle.
- Trafic Internet.
- Trafic FTP.

L'installation faite des agents dans les nœuds est conçue dans la figure suivante :

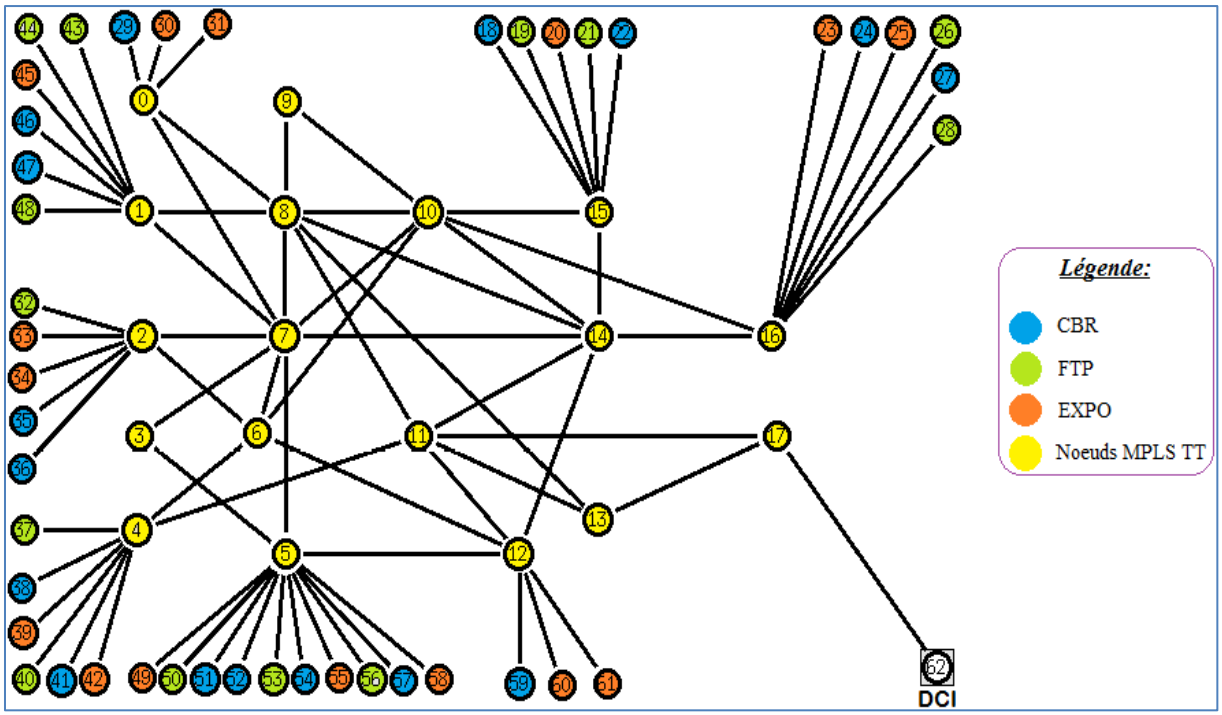


Figure 1.5 : Répartition des agents

La figure précédente montre que j'ai évoqué tous les nœuds dans la simulation. Les différents agents installés se redirigent vers le nœud de la DCI (nœud 62).

Après avoir fixé les types d'agents, j'écris un script TCL de simulation :

Figure 1.6 : Propriétés du script TCL

J'ai commencé par la création de l'instance de simulation et l'ouverture du fichier de traçage « NAM ».

```
# Création d'un simulateur
set ns [new Simulator]

# Création du fichier de trace utilisé par le visualisateur et indication
# à ns de l'utiliser
set nf [open archi1.nam w]
$ns namtrace-all $nf
set f0 [open archi1.tr w]
$ns trace-all $f0
```

Par la suite, j'ai défini la procédure suivante pour fermer les fichiers de traçage, de sortie et exécuter « xgraph » afin d'afficher les résultats et exécuter « NAM » sur le fichier de trace.

```
# Lorsque la simulation sera terminée, cette procédure est appelée pour
# lancer automatiquement le visualisateur
proc finish {} {
        global ns nf f0
        $ns flush-trace
        close $nf
        close $f0
        exec nam archi1.nam &
        # exec xgraph archi1.tr -geometry 800x400 &
        exit 0
}
```

Ensuite, j'ai défini une procédure qui attache un agent UDP avec un nœud créé précédemment.

```
proc attach-expoo-traffic { node sink size burst idle rate } {

        # Création d'une instance de l'objet Simulator
        set ns [Simulator instance]

        # Création d'un agent UDP et attachement au noeud
        set source [new Agent/UDP]
        $ns attach-agent $node $source

        # Création du trafic Expoo et définition des paramètres
        set traffic [new Application/Traffic/Exponential]
        $traffic set packetSize_ $size
        $traffic set burst_time_ $burst
        $traffic set idle_time_ $idle
        $traffic set rate_ $rate

        # Attachement de la source du trafic au générateur de trafic
        $traffic attach-agent $source

        # Connexion de la source au sink
        $ns connect $source $sink

return $traffic
}
```

La fonction suivante permet de créer des agents UDP en précisant une taille de paquet (packetSize_) de 200 ko, le temps moyen d'envoi (burst_time_) de 2s, le temps d'inactivité (idle_time_) de 1s et le débit (rate_) de 100 kbps.

```
set source0 [attach-expoo-traffic $n0 $sink0 200 2s 1s 100k]
```

La commande suivante permet d'associer la paire (origine / destination) :

```
$ns connect $source $sink
```

Pour préciser le protocole de routage utilisé dans la simulation, la commande à placer est la suivante :

```
# Définition du protocole de routage
$ns rtproto DV
```

Par la suite, j'ai fixé les nœuds du réseau MPLS de la manière suivante :

```
# Création du backbone MPLS de T.T
# Création des noeuds de CNSS
$ns node-config -MPLS ON
set n0  [$ns node]
set n1  [$ns node]
set n2  [$ns node]
$ns node-config -MPLS OFF
# Création des bureaux régionaux
set n18  [$ns node]
set n19  [$ns node]
set n20  [$ns node]

set n61  [$ns node]

# Création du noeud62 comme étant le routeur de la DCI
set n62   [$ns node]
```

```
# Nommination des BRs
$n18 label "ben arous"
$n19 label "manouba"
$n20 label "ariana"
$n21 label "le kram"

$n59 label "Kebili"
$n60 label "Mednine"
$n61 label "Tataouine"
$n62 label "DCI"
```

```
# Définition des liens entre les LSRs
$ns duplex-link $n0 $n7 20Mb 10ms DropTail
$ns duplex-link $n0 $n8 20Mb 10ms DropTail

$ns duplex-link $n1 $n7 20Mb 10ms DropTail
$ns duplex-link $n1 $n8 20Mb 10ms DropTail

$ns duplex-link $n14 $n15 20Mb 10ms DropTail
$ns duplex-link $n14 $n16 20Mb 10ms DropTail

$ns duplex-link $n14 $n15 20Mb 10ms DropTail
```

```
# Définition des liens entre la CNSS et la T.T
$ns duplex-link $n17 $n62 512Kb 10ms DropTail

$ns duplex-link $n15 $n18 512Kb 10ms DropTail
$ns duplex-link $n15 $n19 512Kb 10ms DropTail

$ns duplex-link $n4 $n37 512Kb 10ms DropTail
$ns duplex-link $n4 $n38 512Kb 10ms DropTail

$ns duplex-link $n12 $n60 512Kb 10ms DropTail
$ns duplex-link $n12 $n61 512Kb 10ms DropTail
```

Les nœuds de la CNSS sont créés comme suit :

```
set ni [ns node]
```

Pour nommer les nœuds, j'ai utilisé la ligne suivante :

```
$ni label "nom"
```

Les liens et les connexions entre les nœuds sont établis comme indiqué ci-dessous:

```
$ns duplex-link $n0 $n7 20Mb 10ms DropTail
```

En configurant le protocole LDP, j'ai utilisé des couleurs différentes pour les flux de données et le mode de distribution des labels.

```
# Configuration du protocole LDP
for {set i 0} {$i < 18} {incr i} {
        set a n$i
        for {set j [expr $i+1]} {$j < 18} {incr j} {
                set b n$j
                eval $ns LDP-peer $$a $$b
        }
        set m [eval $$a get-module "MPLS"]
        $m enable-reroute "new"
}

$ns ldp-request-color       blue
$ns ldp-mapping-color       red
$ns ldp-withdraw-color      magenta
$ns ldp-release-color       orange
$ns ldp-notification-color  yellow
```

```
# Configuration du mode de distribution des labels
[$n0 get-module "MPLS"] enable-data-driven
[$n1 get-module "MPLS"] enable-data-driven
[$n2 get-module "MPLS"] enable-data-driven
[$n3 get-module "MPLS"] enable-data-driven
[$n4 get-module "MPLS"] enable-data-driven

[$n13 get-module "MPLS"] enable-data-driven
[$n14 get-module "MPLS"] enable-data-driven
[$n15 get-module "MPLS"] enable-data-driven
[$n16 get-module "MPLS"] enable-data-driven
[$n17 get-module "MPLS"] enable-data-driven
```

Puis, j'ai installé les agents en précisant leurs types, leurs caractéristiques, leurs attachements et leurs connexions :

```
# Défintion du lien FTP
# Création d'un agent TCP et son attachement à n19
set tcp19 [new Agent/TCP]
$ns attach-agent $n19 $tcp19

# Création d'une source FTP
set ftp19 [new Application/FTP]
$ftp19 set packetSize_ 500
$ftp19 set Interval_ 0.005

# Connexion
$ftp19 attach-agent $tcp19

# Création d'un agent tcp sink et son attachement à n62
set sink19 [new Agent/TCPSink]
$ns attach-agent $n62 $sink19

# Connexion
$ns connect $sink19 $tcp19
```

```
# Définition des liens CBR
# Création d'un agent UDP et son attachement à n18
set udp18 [new Agent/UDP]
$ns attach-agent $n18 $udp18

# Création d'une source CBR
set cbr18 [new Application/Traffic/CBR]
$cbr18 set packetSize_ 500
$cbr18 set Interval_ 0.005

# Connexion
$cbr18 attach-agent $udp18

# Création d'un agent null
set nul18 [new Agent/Null]
$ns attach-agent $n62 $nul18
$ns connect $nul18 $udp18
```

```
# Définition des liens EXPO
set sink20 [new Agent/LossMonitor]
set sink23 [new Agent/LossMonitor]

set sink60 [new Agent/LossMonitor]
set sink61 [new Agent/LossMonitor]

# Définition des attachements
$ns attach-agent $n62 $sink20
$ns attach-agent $n62 $sink23

$ns attach-agent $n62 $sink60
$ns attach-agent $n62 $sink61

# Création du trafic poissonnier
set ex20 [attach-expoo-traffic $n20 $sink20 200 2s 1s 100k]
set ex23 [attach-expoo-traffic $n23 $sink23 200 2s 1s 200k]

set ex60 [attach-expoo-traffic $n60 $sink60 200 2s 1s 200k]
set ex61 [attach-expoo-traffic $n61 $sink61 200 2s 1s 300k]
```

```
$ftp19 set class_ 1
$ftp21 set class_ 1
```

```
$udp18 set class_ 2
$udp22 set class_ 2
```

```
$ex20 set class_ 3
$ex23 set class_ 3
```

```
# Coloration des flux
$ns color 1 Yellow
$ns color 2 Red
$ns color 3 Orange

# Collection des résultats
#[$n16 get-module "MPLS"] trace-ldp
#[$n16 get-module "MPLS"] trace-mpls
```

Pour établir un trafic poissonnier, j'ai, tout d'abord, créé des agents correspondants aux nœuds générateurs de trafic comme illustré par la suite :

```
set sink20 [new Agent/LossMonitor]
```

Puis, je les ai attachés au nœud de la DCI qui correspond au nœud 62:

```
$ns attach-agent $n62 $sinki
```

Ensuite, j'ai créé les nœuds générateurs de trafic poissonnier « exi » grâce à la procédure suivante :

```
set exi [attach-expoo-traffic $ni $sinki 200 2s 1s 100k]
```

Pour la génération de trafic, j'ai spécifié l'instant de début et de la fin à l'aide de la commande suivante :

```
# Définition des instants de début et de fin de la génération de trafic
$ns at 0.5 "$cbr18 start"
$ns at 0.5 "$cbr22 start"
```

```
$ns at 1.0 "$ftp19 start"
$ns at 1.0 "$ftp21 start"
```

```
$ns at 1.0 "$ex20 start"
$ns at 1.0 "$ex23 start"
```

```
$ns at 50.0 "$ftp19 stop"
$ns at 50.0 "$ftp21 stop"
```

```
$ns at 50.5 "$cbr18 stop"
$ns at 50.5 "$cbr22 stop"
```

```
$ns at 60.0 "$ex20 stop"
$ns at 60.0 "$ex23 stop"
```

J'ai pu aussi voir les tables MPLS dans chaque nœud de backbone en utilisant les lignes qui suivent :

```
# Activation des tables mpls
$ns at 60.0 "[$n15 get-module "MPLS"] pft-dump"
$ns at 60.0 "[$n15 get-module "MPLS"] lib-dump"
$ns at 60.0 "[$n15 get-module "MPLS"] erb-dump"

$ns at 60.0 "[$n0 get-module "MPLS"] pft-dump"
$ns at 60.0 "[$n0 get-module "MPLS"] lib-dump"
$ns at 60.0 "[$n0 get-module "MPLS"] erb-dump"

$ns at 60.0 "[$n17 get-module "MPLS"] pft-dump"
$ns at 60.0 "[$n17 get-module "MPLS"] lib-dump"
$ns at 60.0 "[$n17 get-module "MPLS"] erb-dump"

$ns at 60.0 "[$n2 get-module "MPLS"] pft-dump"
$ns at 60.0 "[$n2 get-module "MPLS"] lib-dump"
$ns at 60.0 "[$n2 get-module "MPLS"] erb-dump"
```

Enfin, j'ai appelé la procédure « finish » et j'ai lancé la simulation :

```
# La simulation va durer 4 secondes
$ns at 60.1 "finish"

# Simulation
$ns run
```

Une fois « NS » a terminé l'exécution, l'outil de visualisation « NAM » montre la figure ci-après :

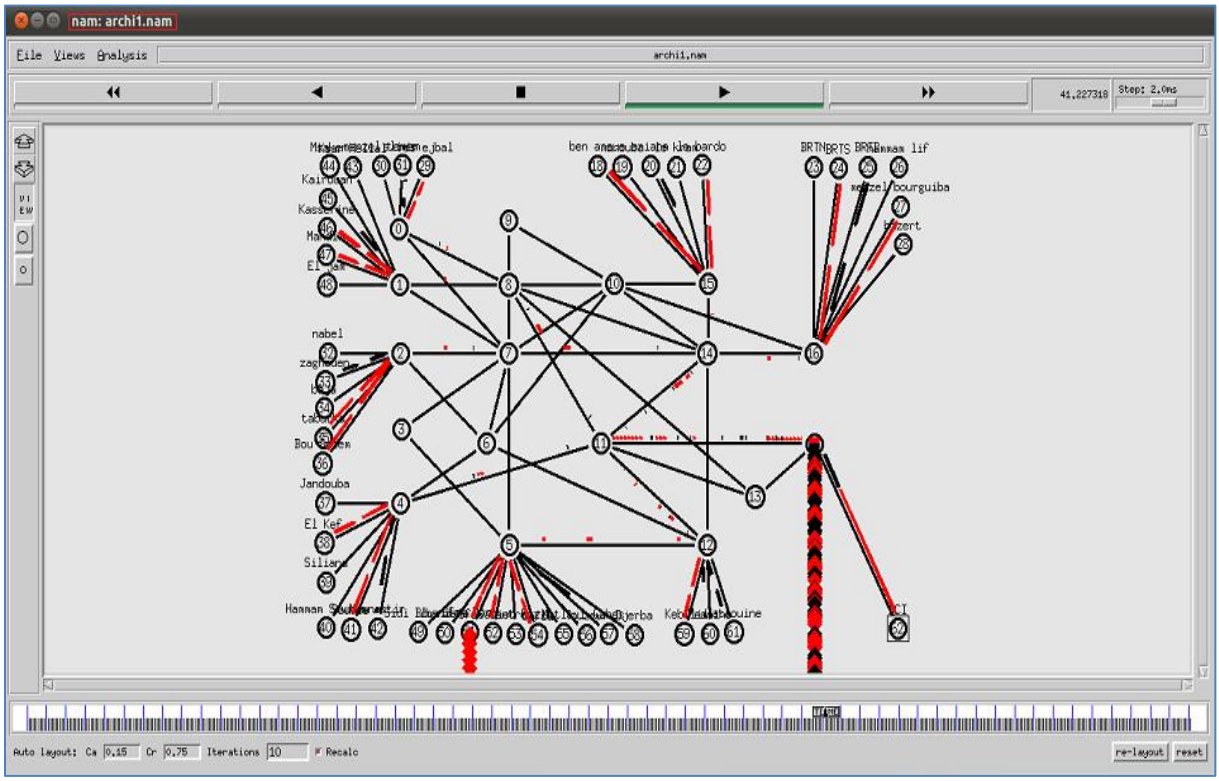


Figure 1.7 : Architecture existante de la CNSS

Après la simulation de l'architecture actuelle de la CNSS, je me suis servi des scripts « awk » pour tester sa fiabilité.

Le test de fiabilité d'un réseau se base sur plusieurs paramètres parmi lesquels :

- ✓ Débit : C'est une mesure de la quantité de données numériques transmises par unité de temps. Il est exprimé en bit par seconde (bit/s, b/s ou bps) ou un de ses multiples.

- ✓ Gigue : C'est la variation de délai de transmission de bout en bout entre des paquets appartenant à un même flux de données. La cause de ce problème peut être due à la différence des chemins empruntés par les paquets dans le réseau, à une congestion ponctuelle du réseau ou encore à un souci d'encapsulation des paquets.

- ✓ Latence : C'est le décalage entre le temps écoulé entre l'envoi d'un paquet et sa réception par le destinataire. Plus la latence est importante plus le transfert est long et sera donc décalé.

- ✓ Perte : C'est la non délivrance d'un paquet de données, la plupart du temps dûe à un encombrement du réseau. Lorsqu'il y a saturation, les mémoires tampons ont besoin de libérer une partie de la bande passante négligeant ainsi certains paquets.

Les scripts « awk » sont exécutés à l'aide des commandes suivantes : (voir **Annexe n°2**)

```
nizar@ubuntu:~$ awk -f temp_reponse.awk < archi1.tr
nizar@ubuntu:~$ awk -f gigue.awk < archi1.tr
nizar@ubuntu:~$ awk -f perte.awk < archi1.tr
```

Je vais me baser sur deux critères lors des tests qui sont la variation de la taille des paquets et la variation de nombre de connexions.

6.1. Variation de la taille des paquets :

Tout d'abord et après avoir fixé le nombre de connexions pour tous les types de trafics, je me suis basé sur la taille des paquets envoyés.

La simulation du trafic CBR donne les résultats suivants :

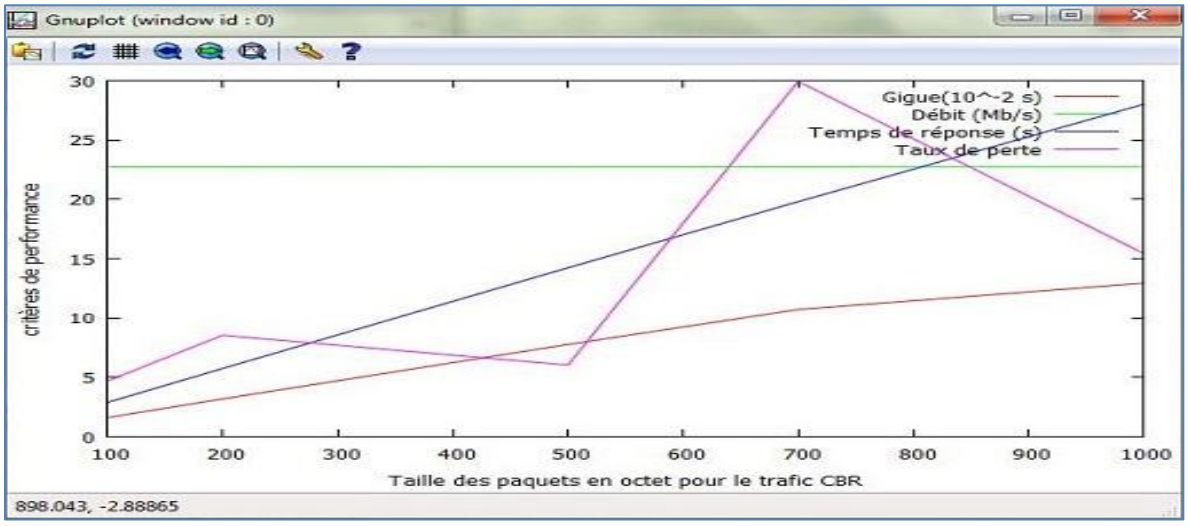


Figure 1.8 : Variation de taille des paquets CBR

⇨ Le débit reste constant tout au long de la simulation par contre les autres paramètres de performance évoluent d'une façon croissante à chaque fois que la taille des paquets augmente.

Ensuite, j'ai effectué une simulation en utilisant uniquement des sources FTP. Les résultats obtenus sont fournis par l'illustration suivante :

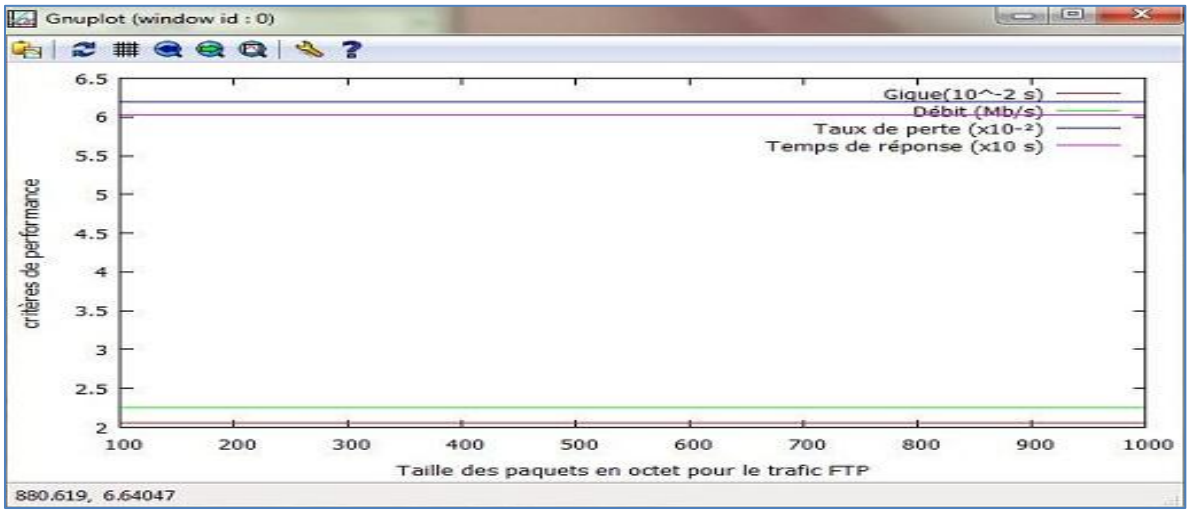


Figure 1.9 : Variation de taille des paquets FTP

⇨ Pour le trafic FTP, les paramètres de performance restent constants tout au long de la simulation.

Concernant le trafic EXPO, les résultats de la simulation sont présentés ci-dessous :

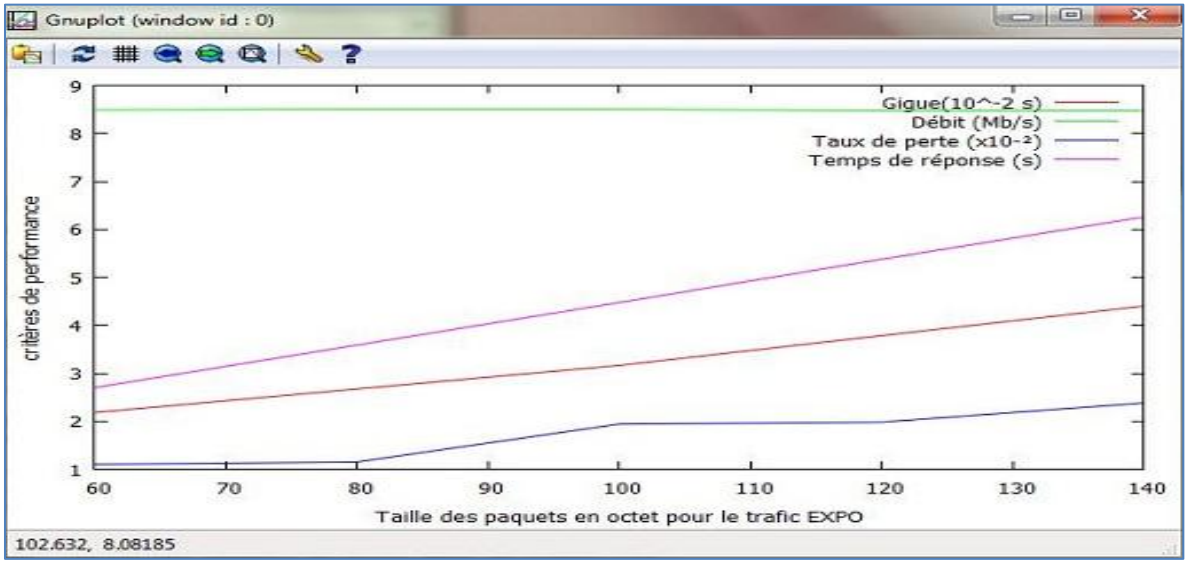


Figure 1.10 : Variation de taille des paquets EXPO

⇨ Le débit reste constant par contre les autres paramètres de performance évoluent d'une façon proportionnelle à l'augmentation de la taille des paquets.

Enfin, la simulation globale de l'architecture existante donne la courbe suivante :

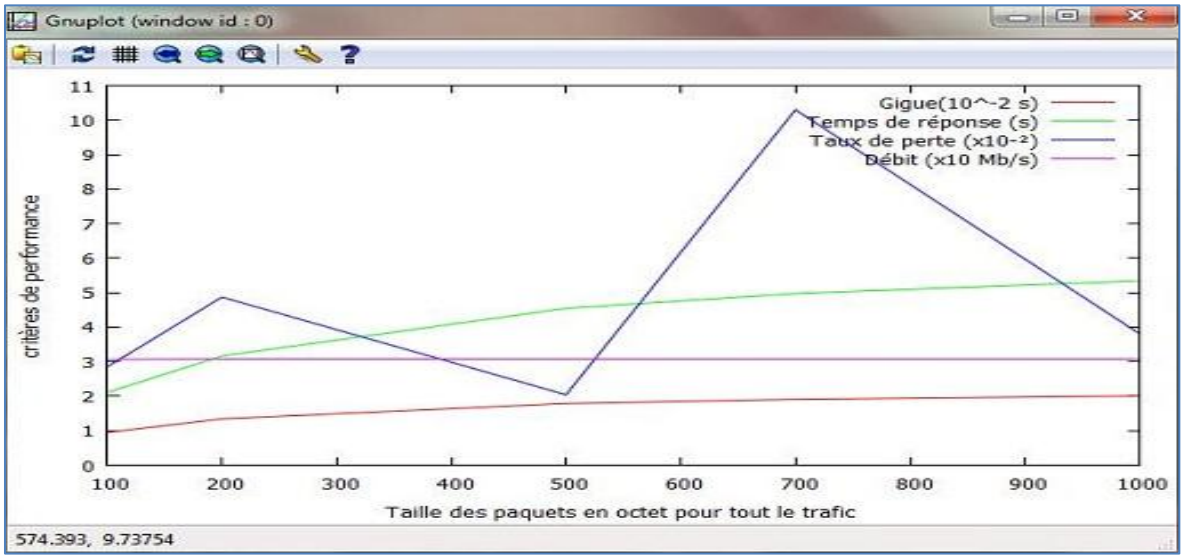


Figure 1.11 : Variation de taille des paquets de trafic global

⇨ Les paramètres gigue et temps de réponse augmentent avec l'augmentation de la taille des paquets alors que le débit reste constant. Aussi, la courbe de taux de perte a la même variation que celle de trafic CBR (voir **Figure 1.8**). Donc, on remarque que le trafic CBR a une influence sur le réseau.

6.2. Variation de nombre de connexions :

Dans cette partie, j'ai varié le nombre de connexions pour chaque type de trafic (CBR, EXPO, FTP).

La figure ci-après présente la variation des paramètres de performance dans le trafic CBR :

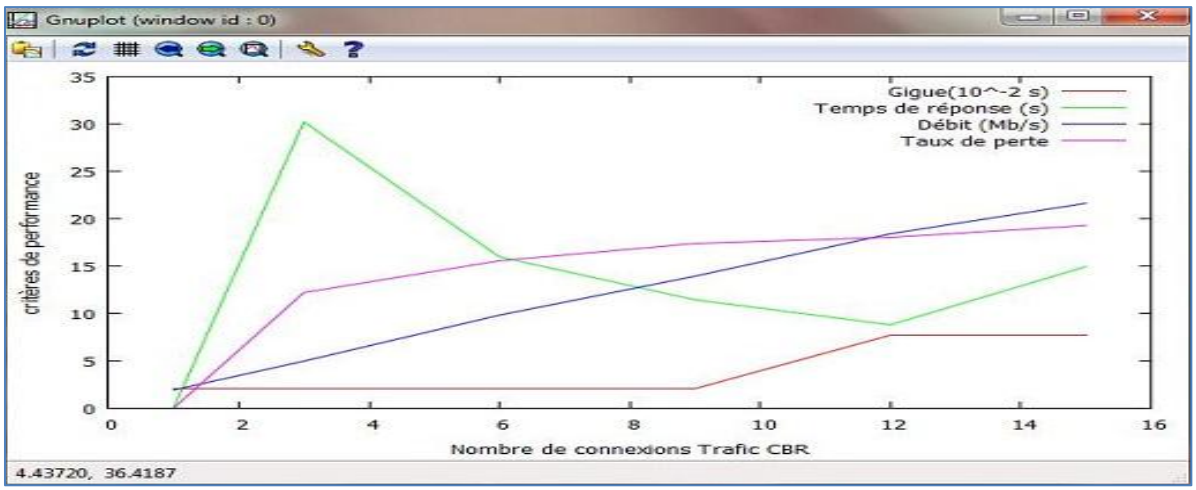


Figure 1.12 : Variation de nombre de connexions de trafic CBR

⇨ Les quatre paramètres présentés sont proportionnels au nombre de connexions établies, c'est-à-dire la densité du réseau entraine la dégradation de sa qualité.

La figure suivante donne une idée sur la variation de performance de trafic FTP :

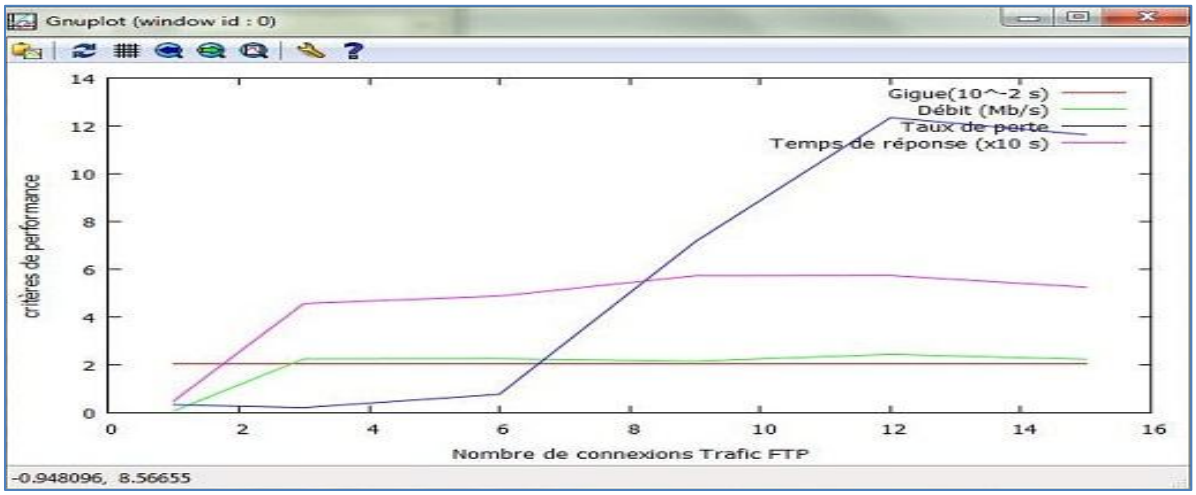


Figure 1.13 : Variation de nombre de connexions de trafic FTP

⇨ Il y a une augmentation au niveau de débit ainsi que le temps de réponse et le taux de perte sauf que la gigue reste constante pour les différents états du réseau.

Concernant le trafic EXPO, la figure suivante montre les résultats obtenus :

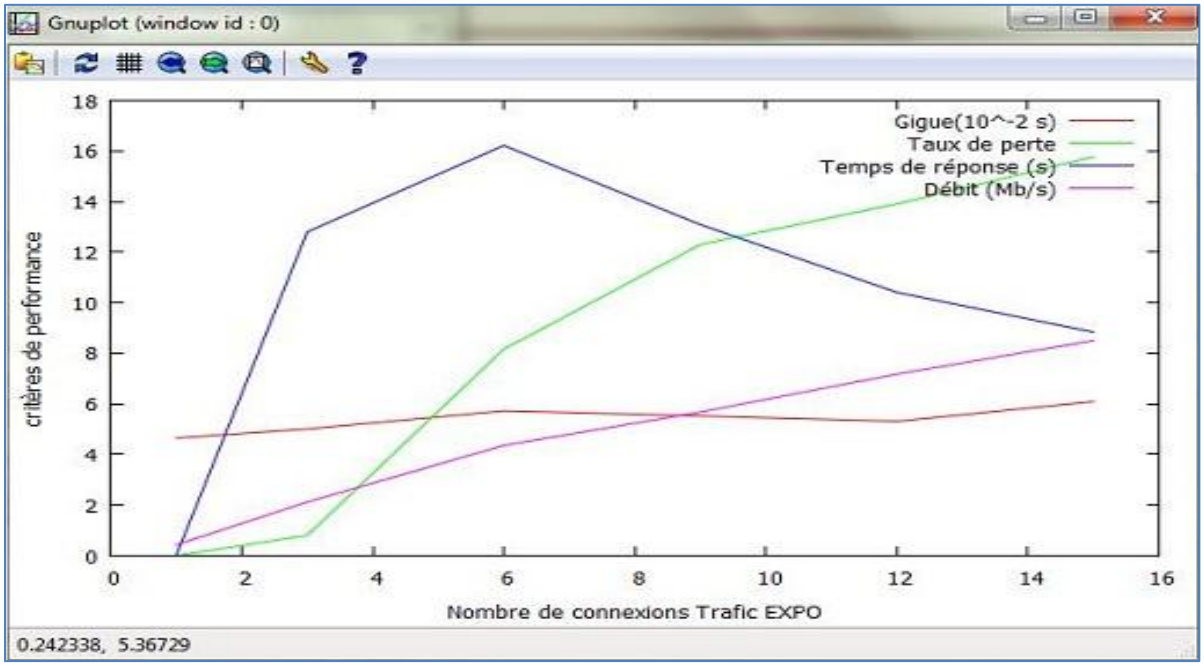


Figure 1.14 : Variation de nombre de connexions de trafic EXPO

⇨ Les quatre paramètres sont croissants alors que le temps de réponse qui a augmenté (environ 16s) puis a descendu à une valeur proche de 8s.

La simulation globale de l'architecture donne les résultats suivants :

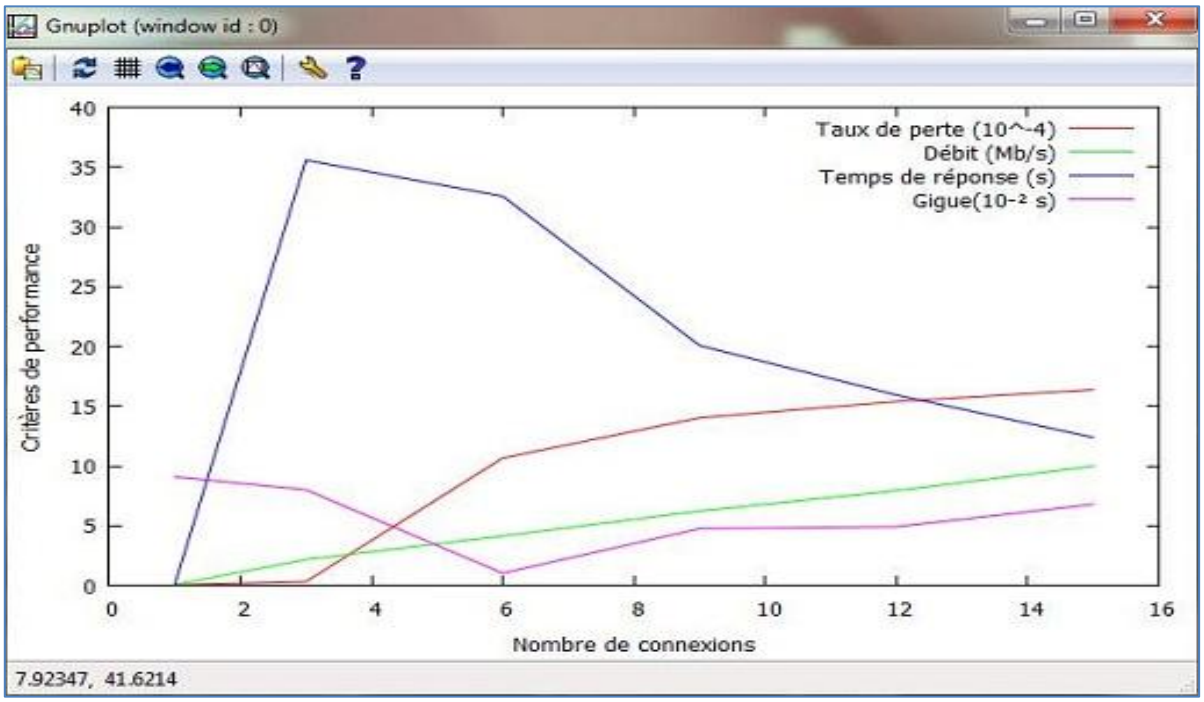


Figure 1.15 : Variation de nombre de connexions de trafic global

⇨ La variation de la gigue, du taux de perte et du débit suit celle des trois trafics et la variation de temps de réponse du réseau suit la variation des trafics CBR et EXPO.

7. Limites de l'architecture existante :

En réalisant cette partie, quelques problèmes émergent du point de vue administration. En fait, les informaticiens de CNSS n'ont pas pu mesurer le débit dans tous les sites car ils n'ont pas les mots de passe de quelques routeurs. Ces derniers sont verrouillés de la part de T.T pour des raisons sécuritaires.

En plus, j'ai remarqué un gaspillage des ressources : La CNSS utilise des LS de 512Kb/s pour toutes les liaisons entre ses sites alors que les besoins de certaines ne dépassent pas 256Kb/s.

Les courbes précédentes montrent que le réseau est fiable mais améliorable.

8. Conclusion :

La simulation de l'architecture avec NS2 a montré ses lacunes au niveau des différents paramètres pris en considération.

Aussi, les courbes générées font apparaitre une faible qualité de service qui est le facteur fondamental pour la migration vers un nouveau backbone MPLS.

Dans le chapitre suivant, le concept MPLS sera abordé d'une manière détaillée.

Chapitre 2

Etude théorique

1. Introduction :

Les techniques employées dans les cœurs de réseaux et les backbones ont subi une grande évolution jusqu'à l'arrivée de la normalisation du protocole MPLS et son développement. Ces réseaux IP/MPLS sont capables de s'adapter aux besoins de forte croissance de l'internet en faisant face aux grandes exigences du trafic professionnel.

Dans cette partie, je vais présenter sommairement la technologie MPLS et ses différents composants.

2. Évolution d'IP vers MPLS :

Le protocole IP est employé en grande partie dans les applications réseaux par les utilisateurs. Au milieu des années 90, il y a eu une augmentation importante de la taille des réseaux, du trafic et l'apparition de nouveaux besoins comme les applications multimédias. **[2]**

Pour transporter les paquets à travers un réseau IP, les routeurs analysent l'adresse de destination dans l'entête avant de les envoyer sur la bonne interface de sortie. Ce processus s'appelle le routage IP et il est réitéré chaque fois que les paquets arrivent sur un routeur. **[3]**

Un réseau IP fonctionne dans un mode non connecté car les paquets constituant un message peuvent emprunter des chemins différents (voir **Figure 2.1**). Le processus de routage prend beaucoup plus de temps et consomme énormément de ressources au niveau de routeur.

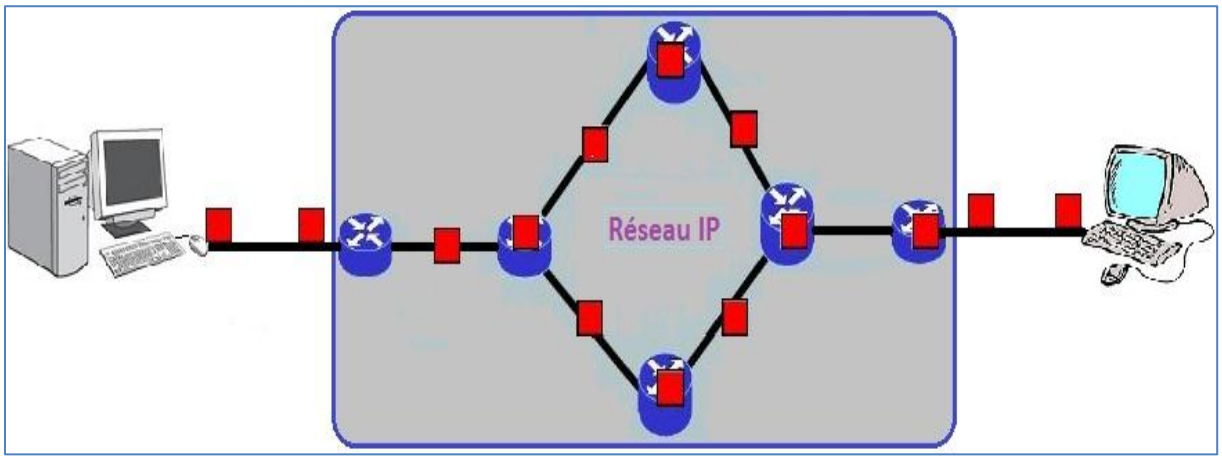


Figure 2.1 : Réseau IP en mode non connecté

Il est nécessaire de trouver une méthode plus efficace pour le routage des paquets. C'est la nouvelle technologie appelée MPLS qui a été mise au point (voir **Figure 2.2**). Son principe de base va être de reprendre les avantages du routage IP et les avantages de la commutation afin de répondre aux besoins de fiabilité et de disponibilité. **[3]**

MPLS est une norme de protocole proposée par l'IETF, l'organisme de normalisation d'Internet pour l'ensemble des architectures et des protocoles de haut niveau. L'idée a été de proposer une norme commune pour transporter des paquets IP sur plusieurs types de réseaux commutés. Il peut s'agir de n'importe quel type de trame de niveau 2 à partir du moment où une référence peut y être incluse. Cette référence est le label utilisé par le protocole MPLS. **[4]**

Avant de traverser un réseau MPLS, il faut d'abord déterminer un chemin à suivre pour aller du nœud d'entrée (Ingress) vers le nœud de sortie (Egress). Par la suite, les données vont être transférées sur le réseau en suivant le chemin prédéterminé.

Pour acheminer les paquets utilisateurs, les nœuds utilisent des labels. A un label d'entrée, correspond un label de sortie et une interface de sortie. Leur succession définit le chemin suivi par l'ensemble des paquets appelé LSP. Les tables de commutation sont calculées à partir d'informations provenant des protocoles de routage IP et celles du protocole de signalisation LDP. **[5]**

MPLS peut être considéré comme un protocole apportant à IP le mode connecté.

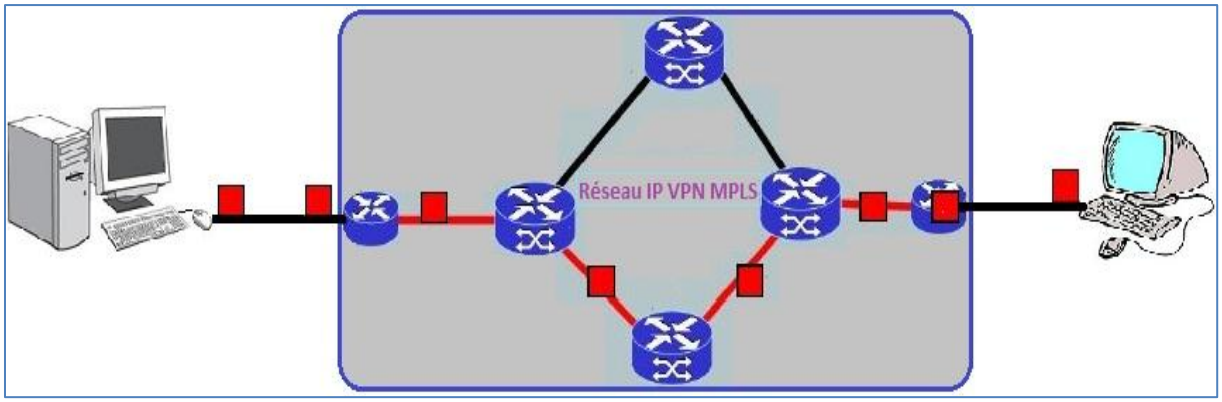


Figure 2.2 : Réseau MPLS en mode connecté

Maintenant, je vais présenter la technologie MPLS qui a permis de transformer le réseau IP en un réseau connecté pour transporter les données des clients.

3. Technologie MPLS :

MPLS est une technologie nouvelle qui utilise des mécanismes de commutation de labels destinés à réduire les coûts du routage. Son intérêt n'est actuellement plus la rapidité de commutation par rapport au routage d'adresse IP mais les services offerts. Un domaine MPLS est composé de deux sortes de routeurs : les LSR et les ELSR. **[2]**

Les LSR sont les routeurs de cœur capables de supporter le MPLS et les ELSR sont des routeurs permettant de faire la transition entre le domaine MPLS et les autres réseaux par exemple les clients IP. **[6]**

La figure suivante montre en détail le rôle des différents composants du réseau MPLS :

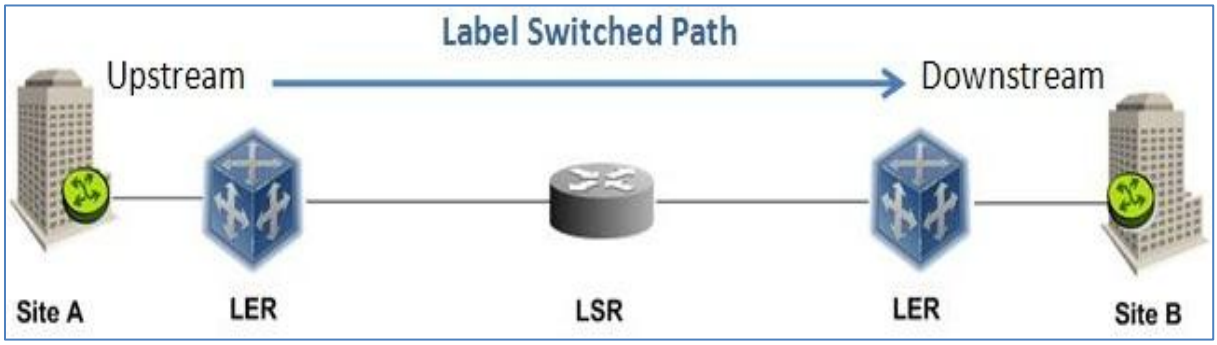


Figure 2.3 : Principe de la commutation de label

- LSR : C'est un routeur de cœur du réseau MPLS qui effectue la commutation sur les labels et qui participe à la mise en place du chemin par lequel les paquets sont acheminés. Lorsque le routeur LSR reçoit un paquet labélisé, il le permute avec un autre de sortie et expédie le nouveau paquet labélisé sur l'interface de sortie appropriée. Le routeur LSR, selon son emplacement dans le réseau MPLS, peut jouer plusieurs rôles à savoir : exécuter la disposition du label (appelé déplacement), marquer l'imposition (appelée poussée) ou marquer la permutation en remplaçant le label supérieur dans une pile de labels avec une nouvelle valeur sortante de label. **[M.M]**

- ELSR : Il s'agit d'un routeur d'accès au réseau MPLS qui gère le trafic entrant dans le réseau MPLS et possédant à la fois des interfaces IP traditionnelles et des interfaces connectées au réseau MPLS.

  Ce routeur ELSR d'entrée, exécute les fonctions de l'imposition de label et de l'expédition d'un paquet à destination du réseau MPLS. A la sortie du réseau MPLS, il exécute les fonctions de déplacement (disposition) de label et la transmission de paquet IP au destinataire. **[M.M]**

- LSP : C'est un chemin pour un paquet de données dans un réseau basé sur MPLS ou une séquence de labels à chaque nœud du chemin allant de la source à la destination. Les LSP sont établis avant la transmission des données ou à la détection d'un flot qui souhaite traverser le réseau. Il est unidirectionnel et le trafic de retour doit donc prendre un autre LSP. **[M.M]**

- FEC : Il représente un groupe de paquets ayant les mêmes propriétés. Tous les paquets d'un tel groupe reçoivent le même traitement au cours de leur acheminement. Dans le réseau MPLS, la transmission de paquets s'effectue par l'intermédiaire de classes d'équivalence FEC.

  Contrairement aux transmissions IP classiques, un paquet est assigné à une FEC une seule fois lors de son entrée sur le réseau. Les FEC sont basés sur les besoins en termes de service pour certains groupes de paquets ou même un certain préfixe d'adresses. **[M.M]**

- LDP : C'est un protocole permettant d'apporter aux LSR les informations d'association des labels dans un réseau MPLS. Il est utilisé pour associer les labels aux FEC pour créer des LSP. Les sessions LDP sont établies entre deux éléments du réseau MPLS qui ne sont pas nécessairement adjacents. Il construit la table de commutation des labels sur chaque routeur et se base sur le protocole IGP pour le routage. **[M.M]**

- Upstream and Downstream : Ce sont les deux modes de distribution des labels utilisés par le protocole LDP dans un réseau MPLS : Upstream pour le mode ascendant et Downstream pour le mode descendant. **[M.M]**

4. Principes MPLS :

Le principe de base de MPLS est la commutation des labels. C'est un protocole situé au niveau 2,5 du modèle OSI car il associe les protocoles des couches 2 et 3. (voir **Figure 2.4**)

MPLS rend le concept de commutation générique car il peut fonctionner sur tout type de protocole de niveau 2. **[7]**

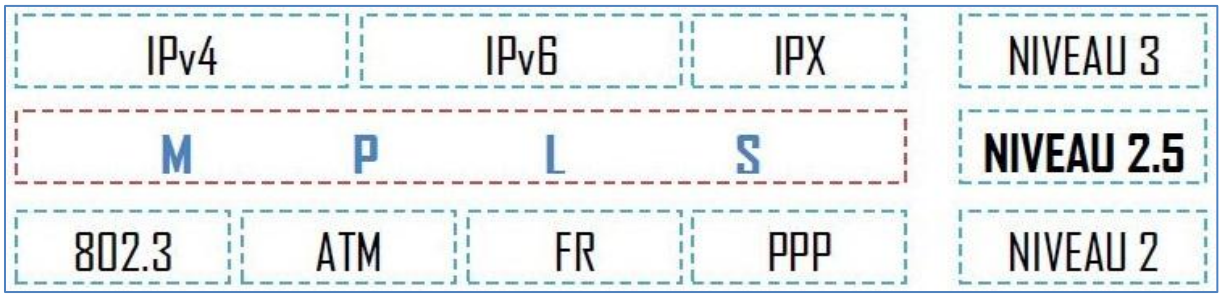


Figure 2.4 : MPLS au niveau des couches

Les routeurs MPLS, à l'intérieur du backbone, permutent les labels tout au long du réseau jusqu'à destination sans consultation de l'entête IP et la table de routage. La commutation MPLS est une technique orientée connexion.

Une transmission des données s'effectue sur un chemin LSP et chaque routeur MPLS, LSR possède une table de commutation associant un label d'entrée à un label de sortie. **[9]**

La table de commutation est rapide à parcourir dans le but d'accroître la rapidité de commutation sur label par rapport à la table de routage du réseau IP. Le chemin suivi est déterminé par le routeur d'entrée ELSR du réseau MPLS. Il aiguille les paquets labélisés sur un chemin LSP donné selon différents critères : adresse destination, qualité de service, applications, etc. **[M.M]**

Les RFCs utilisés pour le protocole MPLS sont les suivants : « RFC3031 », « RFC2702 » et « RFC2547 ». Les labels spécifiques VPN ne sont imposés sur les paquets IP qu'une seule fois en périphérie du réseau MPLS au niveau du routeur d'entrée appelé « Ingres ELSR ». A ce niveau, un calcul est effectué sur le paquet afin de lui affecter un label spécifique. Ce label est supprimé à l'autre extrémité par le routeur de sortie du backbone appelé « Egress ELSR ». **[10]**

Le résumé du mécanisme se présente comme suit : le « Ingress ELSR » reçoit les paquets IP, réalise une classification des paquets dans un FEC en fonction du réseau de destination ou QoS, y assigne un label VPN et un label MPLS puis transmet les paquets labélisés au réseau MPLS. **[R.L]**

En se basant uniquement sur les labels, les routeurs LSR du réseau MPLS commutent les paquets labélisés jusqu'au routeur de sortie « Egress LSR » qui supprime les labels et remet les paquets à leur destination finale. **[10]**

Voici une démonstration du principe du MPLS sur le schéma ci-dessous :

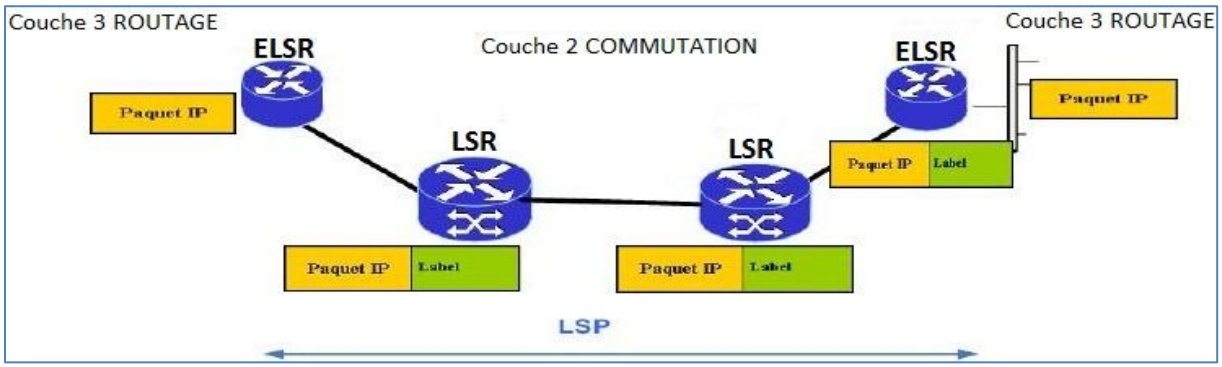


Figure 2.5 : Flux MPLS

5. Label :

Les labels sont des simples nombres entiers de 4 octets (32 bits) insérés entre les entêtes des couches 2 et 3 du modèle OSI. Un label a une signification locale entre deux routeurs LSR adjacents et mappe le flux de trafic entre le LSR amont et le LSR aval. **[8]**

A chaque bond le long du chemin LSP, un label est utilisé pour chercher les informations de routage (Next Hop, interface de sortie). Les actions à réaliser sur le label sont les suivantes: insérer, permuter et retirer. **[10]**

Un label MPLS se présente sous cette forme :

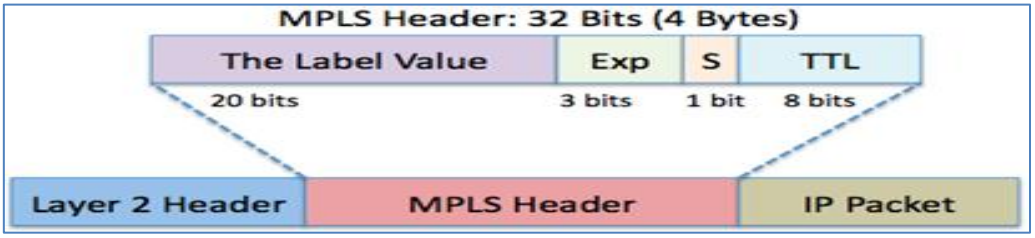


Figure 2.6 : Détails d'un label MPLS

La signification des différents champs est donnée comme suit :

- Label (20 bits) : Valeur du label.
- Exp (3 bits) : Classe du service du paquet.
- BS (1 bit) : Indicateur de fin de pile (égal à 1 s'il s'agit du dernier label).
- TTL (8 bits) : Durée de vie du paquet (évite les doublons).

Un label peut être mis en œuvre dans les différentes technologies ATM, Frame Relay, PPP et Ethernet (Encapsulation). Pour les réseaux Ethernet, un nouveau champ appelé « SHIM » a été introduit entre les couches 2 et 3 comme l'indique le schéma suivant :

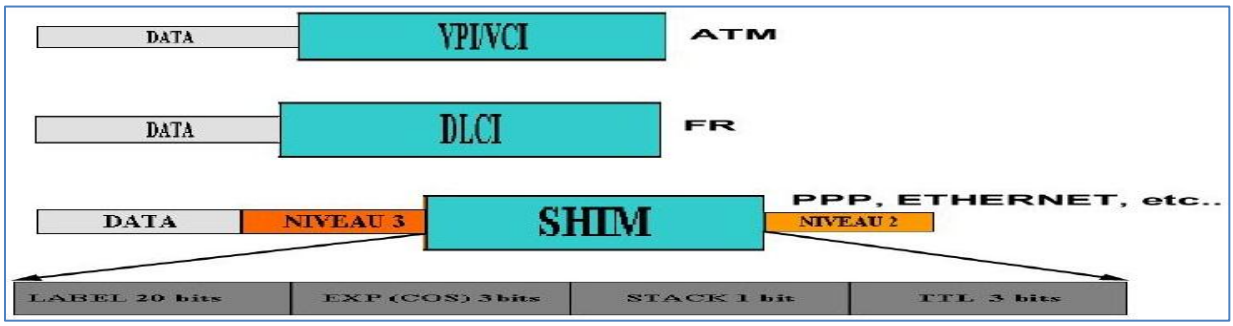


Figure 2.7 : Encapsulation pour ATM, Frame Relay, etc.

La technologie MPLS repose sur la technique de la commutation de label. Pour cela, chaque paquet traversant le réseau doit donc être capable de transporter un label. Il existe deux façons de réaliser le transport des labels dans un réseau MPLS :

- La première solution de transport de labels est celle appliquée aux protocoles de la couche 2 qui peuvent transporter des labels à l'intérieur même de leur entête (c'est le cas des protocoles ATM et Frame Relay). Dans le cas du protocole ATM, le label sera transporté dans le champ « VPI/VCI » de l'entête et dans le cas du Frame Relay, c'est le champ « DLCI » qui sera affecté à cette tâche. **[6]**

- Pour les protocoles ne pouvant pas utiliser cette méthode, le label sera transporté dans le champ « SHIM » qui sera inséré entre l'entête de la couche liaison et l'entête de la couche réseau. Cette technique permet de supporter la technique de commutation de label sur n'importe quel protocole de la couche de liaison de données. **[6]**

Les labels peuvent être associés à un chemin, à une destination, à une source, à une application, à un critère de qualité de service ou à une combinaison de ces différents éléments.

Pour conclure, on peut dire que le routage IP est considérablement enrichi sans pour autant voir ses performances dégradées étant donné que les paquets sont encapsulés dans l'entête MPLS et acheminés utilisant les mécanismes de commutation de niveau 2.

6. Architecture du protocole MPLS :

L'architecture MPLS est composée de deux plans principaux pour la commutation dans le réseau backbone :

- Plan de contrôle : Il permet de créer et de distribuer les routes et les labels. Ainsi, il contrôle des informations de routage, de commutation et de distribution des labels entre les périphériques adjacents.
- Plan de données : Il est connu également sous le nom de « Forwarding Plane » et permet de contrôler la transmission des données en se basant sur la commutation des labels.

Les différents composants MPLS sont sus-indiqués :

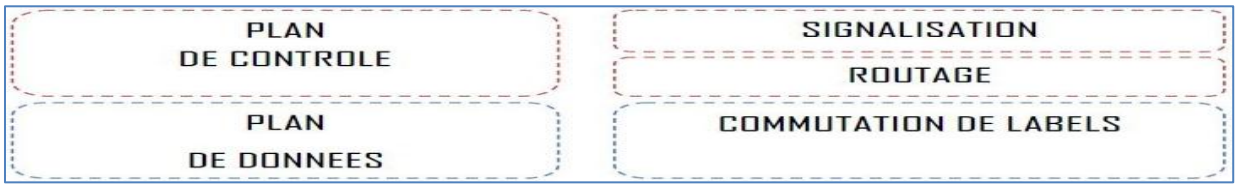


Figure 2.8 : Architecture logicielle MPLS

6.1. Plan de contrôle :

Il est composé d'un ensemble des protocoles de routage classique et de signalisation. Il est chargé de la construction, du maintien et de la distribution des tables de routage et de commutation. Pour ce faire, le plan de contrôle utilise des protocoles de routage classique tels qu'IS-IS ou OSPF afin de créer la topologie des nœuds du réseau MPLS et des protocoles de signalisation spécialement développés pour le réseau MPLS comme LDP, MP-BGP (utilisé par MPLS-VPN) ou RSVP (utilisé par MPLS-TE). **[M.M]**

Dans un réseau MPLS, il existe deux méthodes pour créer et distribuer les labels : « Implicit routing » et « Explicit routing ». Ces deux méthodes sont celles utilisées pour définir les chemins LSP dans le réseau MPLS.

6.1.1. La méthode « Implicit Routing » :

C'est la méthode de base choisie par l'IETF pour les réseaux MPLS. C'est un modèle orienté-contrôle fondé sur la topologie du réseau. Dans ce cas, les labels sont créés à l'issue de l'exécution des protocoles de routage classique.

Il existe également la distribution implicite des labels aux routeurs LSR. Cette distribution est réalisée grâce au protocole LDP. Les labels sont spécifiés selon le chemin « Hop By Hop » défini par le protocole de routage interne classique IGP dans le réseau. **[M.M]**

Chaque routeur LSR doit donc mettre en œuvre un protocole de routage interne de niveau 3 comme le protocole OSPF et les décisions de routage sont prises indépendamment les unes des autres. **[11]** (voir **Figure 2.9**)

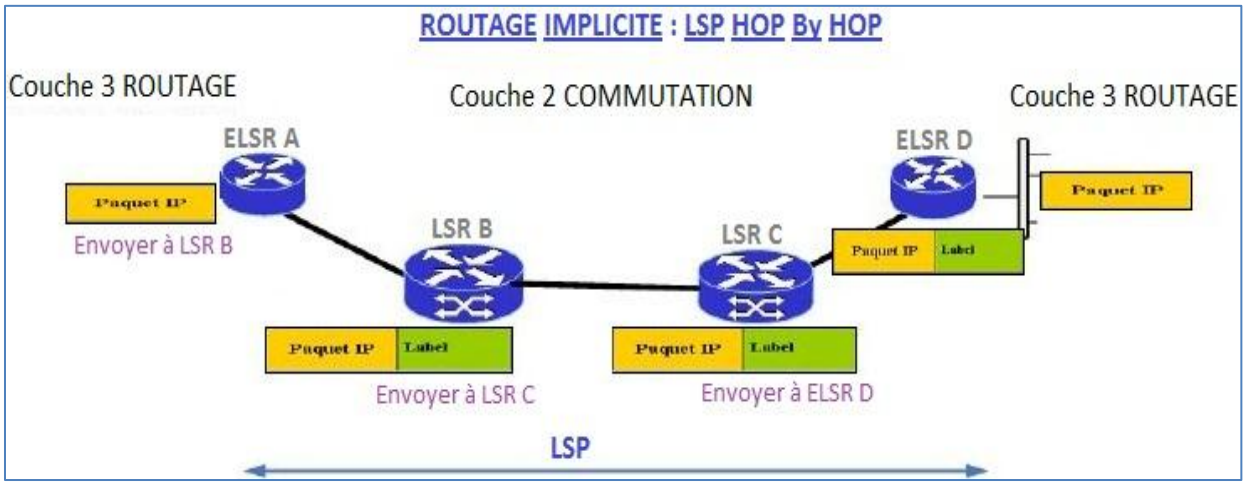


Figure 2.9 : Routage implicite des labels

La distribution des labels fait partie du plan de contrôle et particulièrement du plan de signalisation.

A l'origine, « Cisco Systems » a développé son propre protocole de distribution de label qu'on appelle TDP pour les réseaux backbones MPLS. De nos jours, ce protocole est très peu utilisé et les réseaux MPLS utilisent plutôt le protocole de distribution des labels appelé LDP. **[12]**

6.1.2. La méthode « Explicit Routing » :

Cette méthode explicite est fondée sur les requêtes (REQUEST-BASED) et consiste à ne construire une route que lorsqu'un flux de données est susceptible de l'utiliser.

Avec cette méthode, le routeur Ingress ELSR choisit le chemin de bout en bout au sein du réseau MPLS. Dans ce cas, la création des labels est déclenchée lors de l'exécution d'une requête de signalisation comme RSVP par exemple. **[2]** (voir **Figure 2.10**)

Cette méthode est utilisée pour CR-LDP (CR-LDP=LDP+TE) et RSVP-TE. Et, le LSP n'est plus déterminé à chaque bond contrairement au routage implicite.

Ce qui permet MPLS de faire du « Trafic Engineering » afin d'utiliser efficacement les ressources du réseau et d'éviter les points de forte congestion en répartissant le trafic sur l'ensemble du réseau.

Ce mécanisme permet à un opérateur de faire du TE en imposant au réseau des contraintes sur les flux du point source jusqu'au point destination. Ainsi, des routes, autres que le plus court chemin, peuvent être utilisées. **[10]**

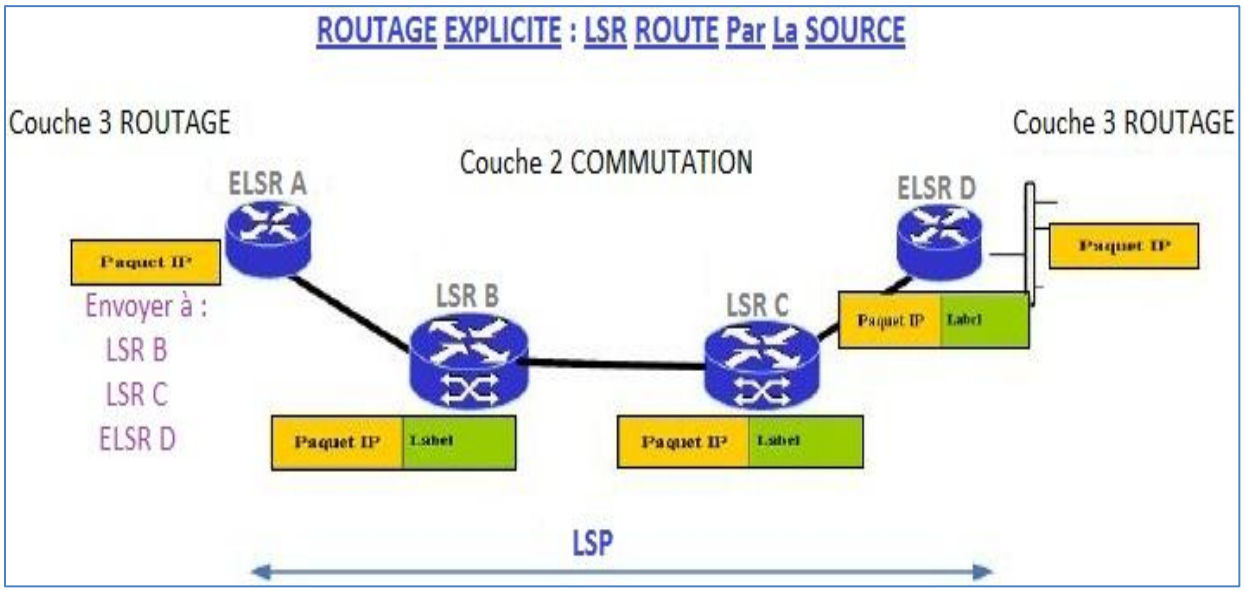


Figure 2.10 : Routage explicite des labels

Je vais présenter par la suite les différents protocoles utilisés dans le plan de contrôle commençant par le protocole LDP et le protocole OSPF.

6.1.3. Protocole de distribution des labels :

LDP est un ensemble de procédures par lesquelles un routeur LSR en informe un autre des affectations faites des labels.

Dans un réseau MPLS, deux routeurs LSR sont liés au label de distribution (peer) lorsqu'ils utilisent un LDP pour échanger leurs affectations. Ce protocole LDP est bidirectionnel utilisé dans le backbone MPLS. **[11]**

Les routeurs LSR se basent sur l'information de label pour commuter les paquets labellisés. Chaque routeur LSR, lorsqu'il reçoit un paquet labellisé, utilise le label local pour déterminer l'interface et le label de sortie.

Il est donc nécessaire de propager les informations sur ces labels à tous les routeurs LSR. Pour cela, des protocoles de distribution sont employés pour l'échange des labels entre les routeurs LSR.

L'autre objectif du protocole LDP est l'établissement des chemins appelés « Label Switch Path » sur le réseau MPLS. Il définit un ensemble de procédures et de messages permettant l'échange des labels et la découverte dynamique des nœuds adjacents grâce aux messages échangés par UDP. **[13]**

Il définit aussi une suite de procédures et de messages utilisés par les routeurs LSR pour s'informer mutuellement de la correspondance (mapping) entre les labels et le flux. (voir **Figure 2.11**)

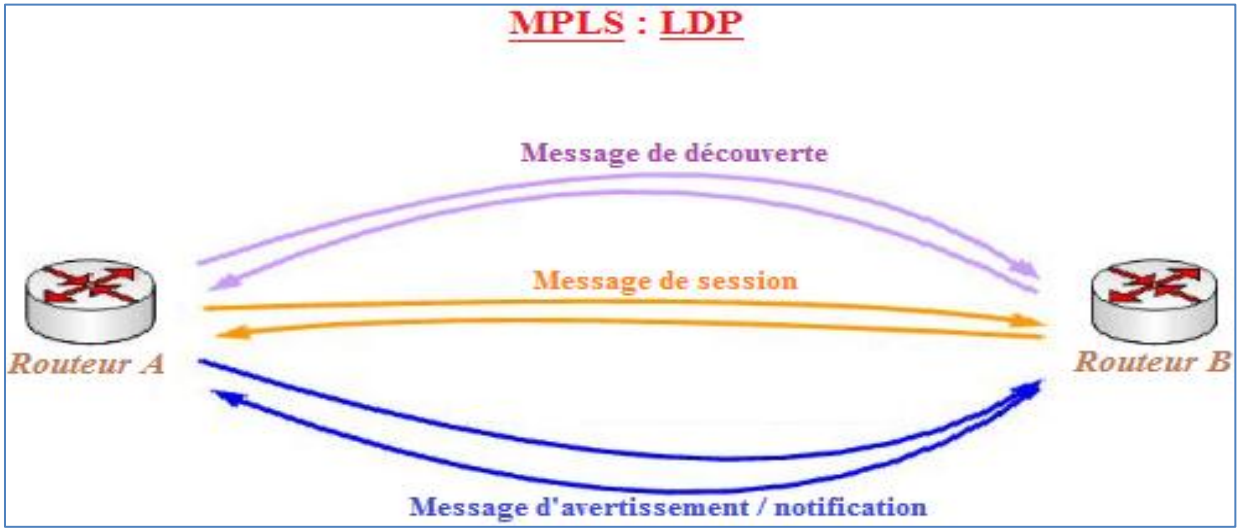


Figure 2.11 : Principe de fonctionnement d'un LDP

Le schéma suivant montre le mécanisme d'établissement d'une connexion LDP pour les annonces de labels :

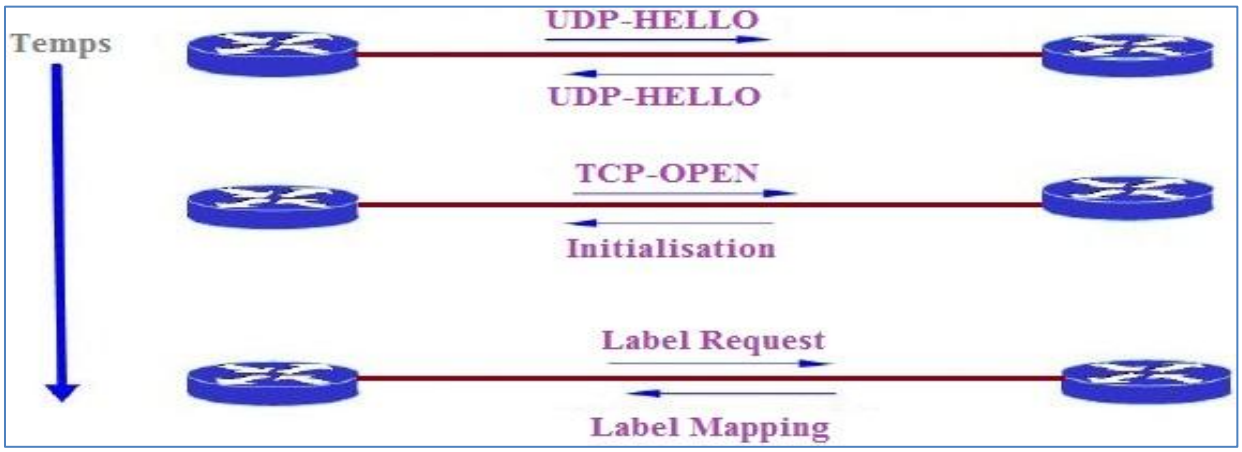


Figure 2.12 : Etablissement d'une connexion LDP

LDP s'appuie sur les protocoles de routage IP dont il utilise la table de routage. Il définit un ensemble de procédures et de messages pour l'échange des labels entre les routeurs LSR. Chaque fois qu'une destination IP (ou une FEC) découverte, LDP est sollicité pour obtenir un label pour cette destination. **[M.M]**

Pour distribuer des labels dans le réseau MPLS, il existe deux modes utilisés par le protocole LDP :

6.1.4. Différents modes de distribution de labels :

- Mode DownStream Unsolicited :

Le fonctionnement de premier mode de distribution est schématisé dans la figure ci-dessous :

Figure 2.13 : Fonctionnement de mode « Downstream Unsolicited »

Dans ce mode, dès qu'un routeur LSR a associé un label à une FEC, il informe automatiquement tous ses voisins de cette opération pour augmenter le trafic dû à la signalisation sur le réseau.

- Mode DownStream On Demand :

Le fonctionnement de deuxième mode de distribution des labels est expliqué dans le schéma suivant :

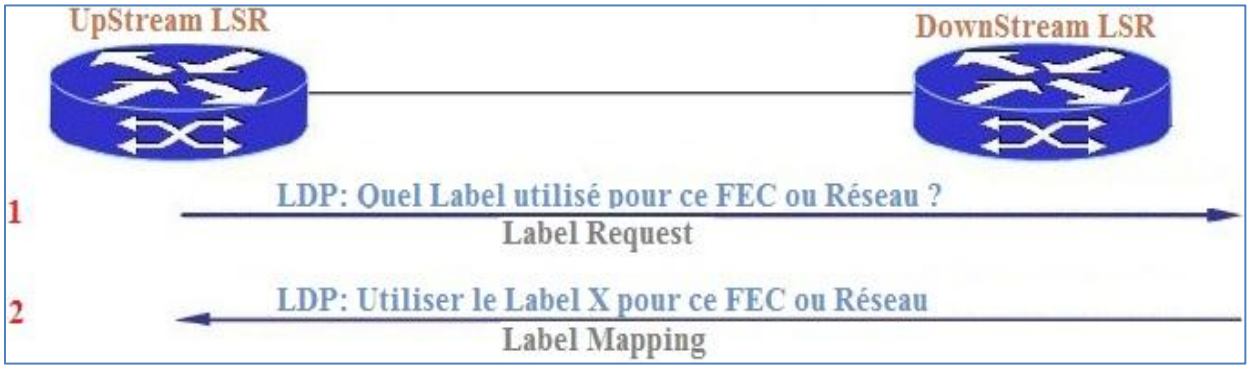


Figure 2.14 : Fonctionnement du mode « Downstream on demand »

Dans ce mode, le routeur amont « UpStream » demande au routeur aval « DownStream » de lui fournir le numéro de label à associer à un FEC particulier. Le routeur « UpStream » envoie le trafic vers le routeur « DownStream », lors du passage d'un paquet qui n'est pas encore associé à une FEC, le routeur « UpStream » va demander l'association d'un label pour cette FEC au routeur suivant (le DownStream LSR sur le schéma précédent). **[2]**

Ce mode est utilisé par le protocole RSVP-TE. Le routeur LSR désirant envoyer un paquet doit d'abord faire une demande de label à son voisin.

Maintenant, j'aborde le protocole OSPF qui est le protocole IGP utilisé dans le backbone MPLS. C'est un protocole à état de lien et très utilisé.

6.1.5. Le protocole de routage OSPF :

- Présentation du protocole OSPF :

Il a été conçu au sein de l'IETF à la fin des années 80 pour résoudre les principaux défauts du protocole RIP et entre autres le temps de convergence. Actuellement, ce temps est d'environ d'une minute avec l'utilisation d'un protocole tel qu'OSPF.

Il est donc certain que ce protocole a permis de réduire considérablement le temps de convergence mais pas suffisamment pour certaines applications pour lesquelles ce temps est de l'ordre d'une minute ou plus. **[14]**

Dans ce cadre, une solution complémentaire a été apportée au protocole OSPF qui consiste à calculer préalablement un chemin de secours disjoint du premier chemin utilisé pour chaque destination possible sur le réseau. C'est l'algorithme TDSP qui est chargé de cette mission. **[15]**

- Caractéristiques et fonctionnement du protocole OSPF :

Ce protocole a deux caractéristiques essentielles :

- Il est ouvert (l'Open de OSPF), son fonctionnement peut être connu de tous.
- Il utilise l'algorithme SPF, plus connu sous le nom d'algorithme de Dijkstra, afin d'élire la meilleure route vers une destination donnée.

OSPF fait partie de la seconde génération de protocole de routage (Link State Protocol). Il est beaucoup plus complexe que RIP mais ses performances et sa stabilité sont supérieures. Il utilise une base de données distribuées qui permet de garder en mémoire l'état des liaisons. Ces informations forment une description de la topologie du réseau et de l'état de l'infrastructure. **[14]**

Pour bien comprendre son fonctionnement, il est nécessaire de s'intéresser aux notions suivantes :

- Notion de système autonome : C'est un ensemble de réseaux qui utilise un protocole de routage commun et qui dépend d'une autorité administrative unique. OSPF est un protocole de routage intra-domaine, c'est-à-dire qu'il ne diffuse les informations de routage qu'entre les routeurs appartenant à un même système autonome. **[14]**

- Notion de zone (area) : Un système autonome géré par le protocole OSPF est divisé en plusieurs zones de routages qui contiennent des routeurs et des hôtes. Cette division introduit le routage hiérarchique. Chaque zone possède sa propre topologie et ne connaît pas les topologies des autres zones du système autonome. **[14]**

  La « zone backbone » est une zone particulière constituée de plusieurs routeurs interconnectés et devant être le centre de toutes les zones. Autrement dit, toutes les zones doivent être connectées physiquement au backbone.

Pour mieux interpréter cette notion des zones, voici un schéma illustratif :

Figure 2.15 : Organisation d'OSPF selon les zones

On peut voir sur le schéma précédent que le système autonome est découpé en trois zones plus le backbone. Les routeurs de la zone 1 ne connaissent pas les routeurs de la zone 2 ni ceux de la zone 3. De même, la zone 1 ne connaît pas la topologie des zones 2 et 3.

L'intérêt de définir des zones est de limiter le trafic de routage, de réduire la fréquence des calculs du plus court chemin par l'algorithme SPF et d'avoir une table de routage plus petite, ce qui accélère la convergence de celle-ci. **[16]**

6.2. Plan de contrôle :

Le plan de données permet de transporter les paquets labélisés à travers le réseau MPLS en se basant sur les tables de commutations. Il correspond à l'acheminement des données en accolant l'entête SHIM aux paquets arrivant dans le domaine MPLS. **[2]**

Le plan de contrôle est indépendant des algorithmes de routages et d'échanges des labels et utilise une table de commutation appelée LFIB pour transférer les paquets labélisés avec les bons labels.

Cette table est remplie par les protocoles d'échange de label comme le protocole LDP. A partir des informations de labels apprises par LDP, les routeurs LSR construisent deux tables la LIB et la LFIB. **[M.M]**

De manière générale, la LIB contient tous les labels appris des voisins LSR, tandis que la LFIB est utilisée pour la commutation proprement dite des paquets labélisés. La table LFIB est un sous-ensemble de la base LIB.

6.2.1. Table TIB :

La première table construite par le routeur MPLS est la table LIB. Elle contient pour chaque sous réseau IP (ou réseau destinataire) la liste des labels affectés par les routeurs LSR voisins. **[O.F]**

6.2.2. Table LFIB :

A partir de la table LIB et de la table de routage IP du réseau interne au backbone, chaque routeur LSR construit une table LFIB qui sera utilisée pour commuter les paquets labélisés. Dans le réseau MPLS, chaque sous-réseau IP est appris par un protocole IGP qui détermine le prochain saut (Next Hop) pour l'atteindre. **[3]**

Donc pour atteindre un sous-réseau IP donné, le routeur LSR choisit le label d'entrée de la table LIB qui correspond à ce sous-réseau IP et sélectionne comme label de sortie le label annoncé par le routeur voisin (correspondant au Next Hop) déterminé par le protocole IGP (plus court chemin). **[3]**

6.2.3. Transmission des données :

A l'entrée du réseau MPLS, les paquets IP se voient insérés un label par le routeur d'entrée Ingress ELSR. Ces paquets labélisés sont ensuite commutés vers le cœur du réseau selon leur numéro de label.

Ensuite, les routeurs MPLS du cœur de réseau (les LSR) commutent les paquets labélisés jusqu'au routeur de sortie (Egress ELSR) par changement de labels à chaque nœud. **[M.M]**

Un routeur LSR, recevant un paquet labélisé, se base sur la table LFIB pour transiter le paquet. A partir d'un label d'entrée (label local), il en déduit l'interface et le label de sortie (Outgoing interface et Outgoing tag ou VC) pour faire suivre les paquets. **[2]**

Lors de l'arrivée du paquet au dernier routeur « Egress ELSR », ce dernier va retirer le label et transmettre le paquet à sa couche de niveau 3 (sous-réseau IP) qui va se charger du routage classique du paquet.

Les opérations d'ajout et de suppression des labels sont primordiales car elles vont conditionner l'interopérabilité des réseaux à commutation de labels avec les autres types de réseaux basés sur IP.

6.3. Les applications de la technologie MPLS :

Les principales applications de la technologie MPLS sont les réseaux privés virtuels (VPN), la qualité de service (QoS) et l'ingénierie de trafic (TE).

6.3.1. VPN/MPLS :

Les réseaux privés virtuels basés sur la technologie MPLS simplifient considérablement le déploiement des services VPN par rapport aux VPN traditionnels.

En plus, les services de VPN/MPLS sur IP sont le moyen le plus souple et le plus économique pour interconnecter un ensemble de sites. L'architecture mise en place par MPLS est très sollicitée par les opérateurs réseaux fournissant plusieurs clients car elle permet de partager des équipements physiques. **[M.M]**

IPSec est une autre approche permettant de mettre en œuvre des VPN sur le réseau IP. L'IPSec privilégie la sécurisation des flux d'informations par encryptage des données alors que MPLS se concentre plutôt sur la gestion de la qualité de service et la priorité des flux. **[10]**

Le problème de sécurité dans MPLS/VPN est minimal dans le cas d'un réseau propriétaire (non Internet). Cependant, si cette garantie n'est pas suffisante, il existe des solutions qui permettent d'utiliser en même temps MPLS et IPSec et ainsi de construire des VPN disposant des avantages des deux approches en même temps : la souplesse de MPLS et la sécurisation d'IPSec. **[18]**

6.3.2. Quality Of Service :

Avec la technologie MPLS et la définition des classes disposant chacune d'un niveau de priorité, il est possible de garantir une qualité de service adaptée à chacun des flux utilisant la solution VPN/MPLS.

La QoS est un élément crucial pour un réseau d'opérateur. En effet, l'opérateur doit pouvoir garantir à ses clients le transport de leurs flux en garantissant différentes contraintes, comme par exemple : Débit minimal garanti, Débit maximal, Latence, Gigue. **[18]**

Ainsi cette solution permet de véhiculer la voix sur IP (VoIP) et de mettre en place des applications de visioconférence dans des conditions excellentes sur des réseaux VPN/MPLS à forts taux d'utilisation.

6.3.3. Traffic Engineering :

Tout comme la qualité de service, le « Traffic Engineering » est un élément crucial pour un réseau d'opérateur. Comme sus-indiqué, il existe deux modes pour l'établissement des LSP sur un réseau MPLS.

Dans le cadre du routage implicite, le chemin sera défini selon l'IGP. Par conséquent, le chemin de base sélectionné sera par défaut celui qui contient le moins de sauts.

Dans un réseau MPLS, le TE permet d'optimiser l'utilisation des ressources d'un réseau afin d'éviter la congestion. C'est la prise en compte de la bande passante disponible sur un lien lors des décisions de routage qui rend possible cette optimisation. Pour cela, il faut utiliser le protocole « Source Routing » pour le configurer. Ainsi, pour sa mise en place dans un réseau, l'opérateur doit utiliser un protocole de routage particulier qui doit implémenter l'algorithme CSPF. **[18]**

C'est cet algorithme qui permet le choix d'une route en fonction des paramètres comme par exemple le débit disponible sur un lien. Des évolutions des protocoles de routages existant comme OSPF-TE ou ISIS-TE ont été développés afin d'implémenter l'algorithme CSPF. **[13]**

Dans un réseau MPLS, le respect de ces contraintes lors des décisions de routage est fait grâce à la présence d'un protocole de routage implémentant l'algorithme CSPF. Enfin, la réservation de la bande passante éventuelle, qui doit être faite sur les routeurs, est très souvent faite grâce au protocole RSVP-TE. **[18]**

6.4. Évolutions MPLS :

Les principales évolutions du protocole MPLS sont :

6.4.1. GMPLS :

C'est la première extension du MPLS. Le concept de cette technologie est d'étendre la commutation aux réseaux optiques.

Le GMPLS met en place une hiérarchie dans les différents supports de réseaux optiques. Il permet donc de transporter les données sur un ensemble de réseaux hétérogènes en encapsulant les paquets successivement à chaque entrée dans un nouveau type de réseau. **[17]**

Ainsi, il est possible d'avoir plusieurs niveaux d'encapsulations selon le nombre de réseaux traversés. Le label correspond à ce réseau est conservé jusqu'à la sortie du réseau.

GMPLS reprend le plan de contrôle de MPLS en l'étendant pour prendre en compte les contraintes liées aux réseaux optiques. En effet, il va ajouter une brique de gestion des liens à l'architecture MPLS. Cette brique comprend un ensemble de procédures utilisées pour gérer les canaux et les erreurs rencontrées. **[3]**

6.4.2. VPLS :

VPLS définit un service de VPN au niveau de la couche 2. Son but est de simuler un réseau LAN à travers l'utilisation d'un réseau MPLS classique. Là encore, la plus grande partie des traitements va s'effectuer sur les PE tout comme les VPNs de niveau 3. Chaque PE maintient une table d'adresses MAC appelée table VFI.

A ce niveau-là, le mapping des FEC s'effectue directement par rapport aux adresses MAC et non les adresses IP. Le principe est similaire à la commutation classique de niveau 2 : Une trame arrive sur un PE qui consulte sa table VFI pour vérifier l'existence de l'adresse et la commute si trouvée. Le cas échéant, le PE, qui émule ce commutateur, va envoyer la trame sur tous les ports logiques relatifs à l'instance VPLS concernée. **[12]**

Le principe est exactement similaire aux VPNs de niveau 3 mise à part le fait que tout se passe au niveau 2. Le VPLS est encore à l'état de test à l'IETF et sa norme (le protocole de communication et les algorithmes) n'est donc pas encore définitive.

7. Conclusion :

Le protocole MPLS semble intéressant pour l'avenir en tant que technique fédératrice et de nombreux travaux sont menés pour faciliter les choix à faire.

Grâce à ses mécanismes de commutation de labels avancés et sa simplicité de mise en place sur des réseaux déjà existants, le MPLS est devenu une technologie phare de demain alliant souplesse, évolutivité et performance pour un coût réduit.

Dans le dernier chapitre, je vais essayer d'améliorer le réseau actuel du CNSS en optimisant la capacité de ses liaisons inter-sites.

Chapitre 3

Réalisation

1. Introduction :

Après l'analyse de l'état actuel du réseau de la CNSS et l'étude théorique de la technologie MPLS, je propose une nouvelle architecture qui parait adéquate et plus optimisée.

Pour bien dégager les avantages de cette nouvelle architecture, je vais la simuler avec NS2 et la tester avec les mêmes paramètres utilisés auparavant.

2. Conception et présentation :

Ce nouveau réseau pourra remplacer l'existant et résoudre ses problèmes. Il se compose de cinq PoPs choisis par région (Tunis, Sousse, Sfax, Gafsa et Jendouba) qui sont liés entre eux par des fibres optiques. Tous les BR sont liés aux PoPs de TT par des LS en cuivre.

Cette nouvelle architecture est illustrée comme suit :

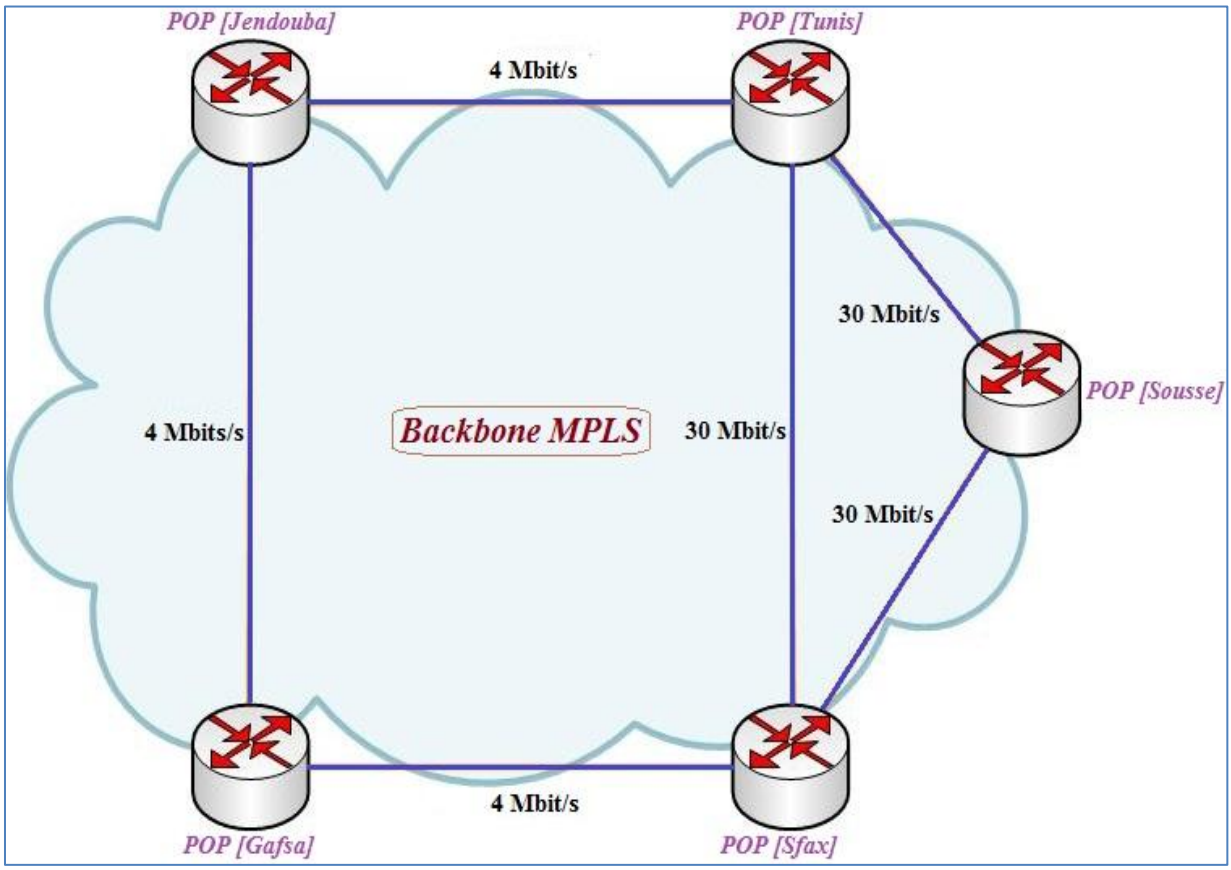


Figure 3.1 : La nouvelle architecture de la CNSS

Pour chaque site de la CNSS, j'ai optimisé le débit et j'ai obtenu les valeurs citées dans le tableau suivant :

POP	Sites CNSS		
	Nom	Débit Moyen	Débit Estimé
Gafsa	Gafsa	220	256
	Melawi	300	512
	Tozeur	200	256
	Kebili	200	256
	Total Gafsa	*920*	*1280*
Jendouba	Beja	200	256
	Tabarka	100	256
	Bou Selem	150	256
	Jendouba	200	256
	El Kef	200	256
	Siliana	350	512
	Kasserine	120	256
	Total Jendouba	*1320*	*2048*
Tunis	Pension	800	1024
	Beyrout	2900	4096
	20 bis	1100	2048
	8 bis	600	1024
	Centre Formation	300	512
	117 Avenue Liberté	200	256
	Micro filmage	3000	4096
	Omrane	300	512
	Khadra	350	512
	Bizerte	300	512
	Ben Arous	300	512
	Manouba	200	256
	Ariana	300	512
	Le Kram	200	256
	Le bardo	300	512
	BRTN	200	256

POP	Sites CNSS		
	Nom	Débit Moyen	Débit Estimé
Tunis	BRTS	200	256
	Hammam Lif	150	256
	Menzel Bourguiba	150	256
	Bizerte	250	256
	Ras Ejbal	150	256
	Menzel Temim	300	512
	Slimen	150	256
	Nabeul	250	256
	Zaghouan	300	512
	Total Tunis	*13250*	*19712*
Sfax	Sfax	400	512
	Jbeniana	100	256
	Sfax nord	250	256
	Sfax centre	250	256
	Sakiet Ezzit	240	256
	Gabes	100	256
	Djerba	210	256
	Medenine	200	256
	Tatounine	180	256
	Total Sfax	*1930*	*2560*
Sousse	Sousse	400	512
	Hamam Sousse	140	256
	Sousse	190	256
	Monastir	240	256
	Ksar Hlel	220	256
	Mseken	160	256
	Kairouan	280	512
	Mahdia	160	256
	El Jam	100	256
	Sidi Bouzid	150	256
	Total Sousse	*1630*	*2304*

Tableau 3.1 : Répartition de la nouvelle architecture selon le débit

3. Simulation et test :

L'estimation du débit au niveau des sites a permis de faire une simulation de son nouveau backbone. Alors, j'ai donné des types de liens différents au sein et en dehors des nœuds MPLS de la CNSS.

La répartition des agents aux différents nœuds est établie comme indiquée : (voir **Figure 3.2**)

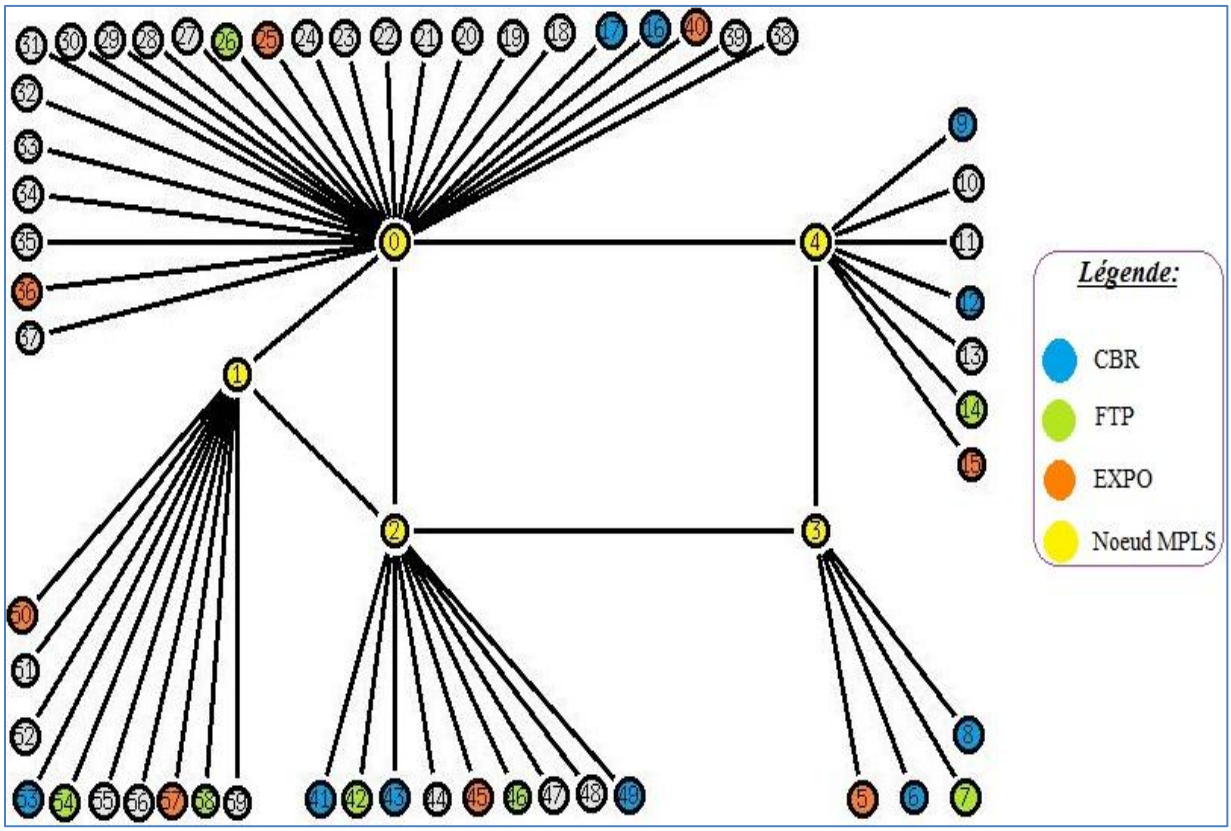


Figure 3.2 : Répartition des agents dans la nouvelle architecture

Afin de mieux expliquer la figure précédente, le tableau ci-dessous indique la direction des trafics générés et la nature des différents agents :

Nœud MPLS	*Trafic /Agent*	*Nœud source*	*Nœud destination*
0	CBR	16	17
	FTP	26	54
	EXPO	36	57
1	CBR	53	43
	FTP	54	58
	EXPO	50	40

2	CBR	41	49
	FTP	46	42
	EXPO	45	5
3	CBR	6	8
	FTP	7	14
	EXPO	5	15
4	CBR	9	12
	FTP	14	7
	EXPO	15	25

Tableau 3.2 : Répartition des agents aux nœuds dans la nouvelle architecture

Le script TCL de la nouvelle architecture est semblable à celui de la première (voir **Annexe°2**). L'outil de visualisation a donné comme résultat la figure suivante :

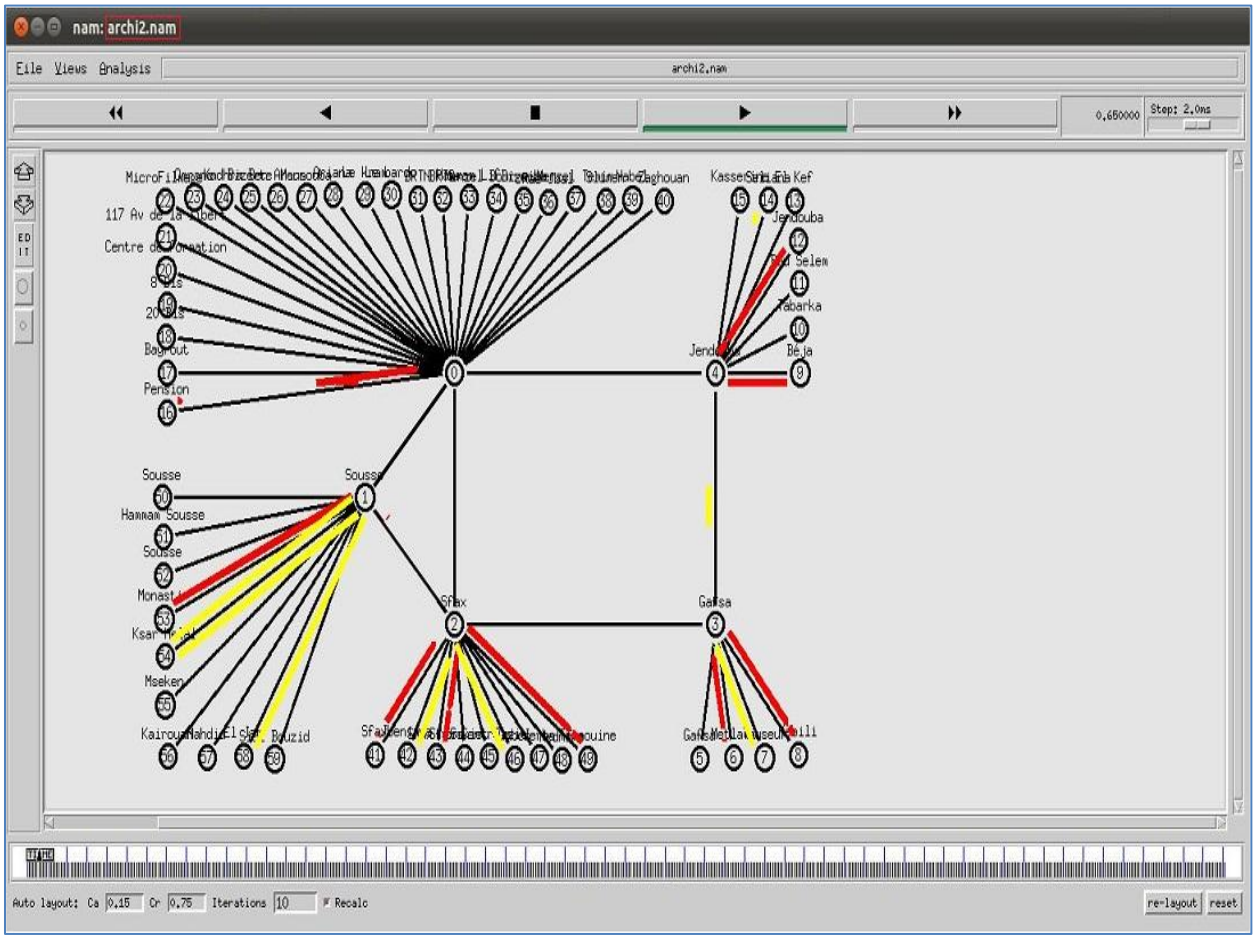


Figure 3.3 : Backbone MPLS propre de la CNSS

J'ai suivi les mêmes étapes de test en utilisant la variation de nombre de connexions et la variation de la taille des paquets comme critères de test.

3.1. Variation de la taille des paquets :

Commençant par le trafic CBR, les résultats sont présentés dans la figure suivante :

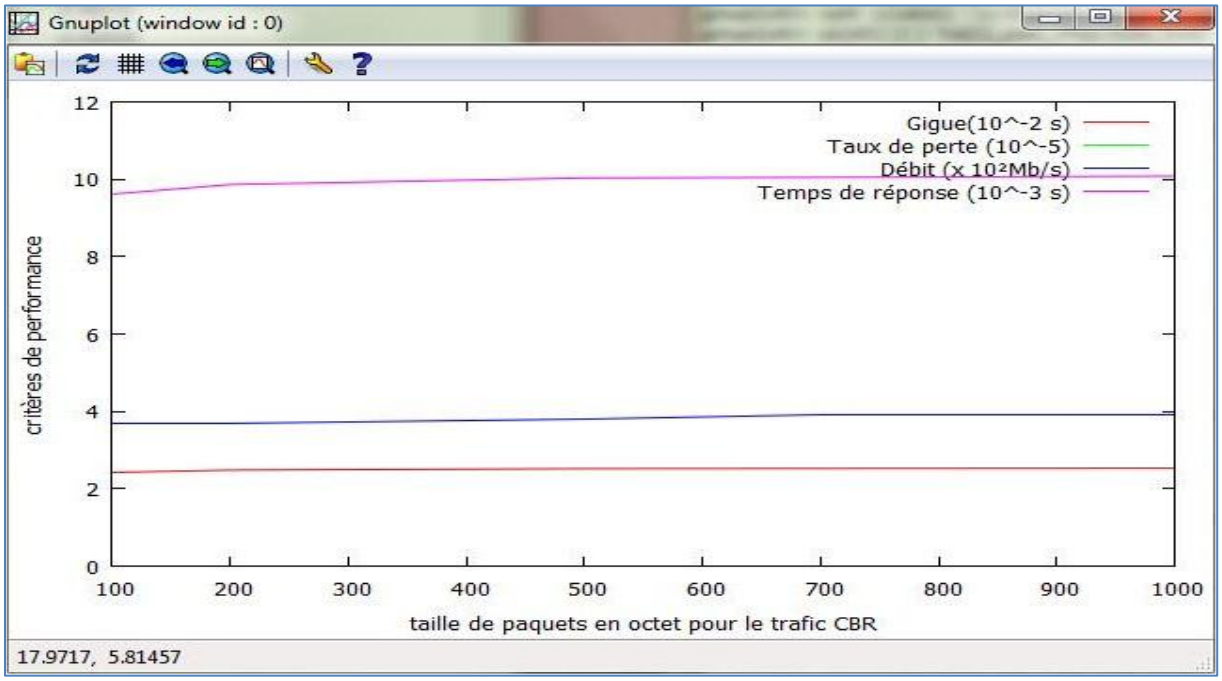


Figure 3.4 : Variation de taille des paquets CBR (1)

⇨ Les quatre paramètres de performance restent stables même si on varie la taille des paquets sans qu'il y est de perte de paquets.

Pour le trafic FTP, les résultats sont indiqués dans la figure suivante :

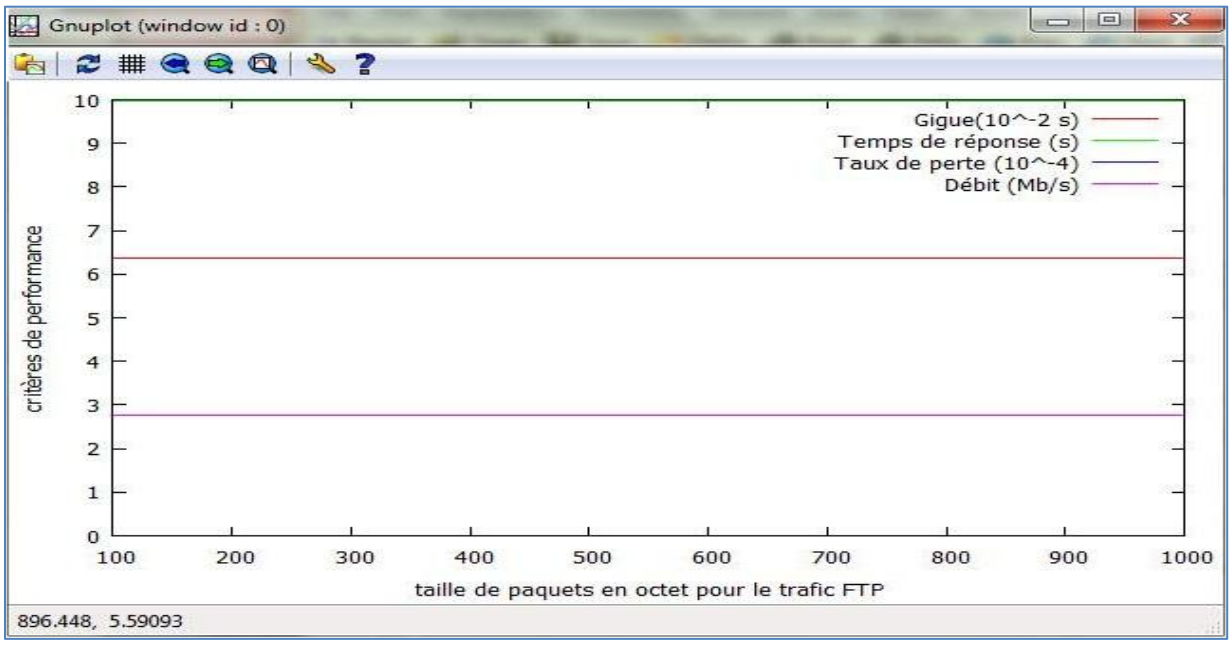


Figure 3.5 : Variation de taille des paquets FTP (1)

⇨ Il y a une stabilité des paramètres de performance et une absence de perte.

La figure qui suit montre les résultats de simulation de trafic EXPO :

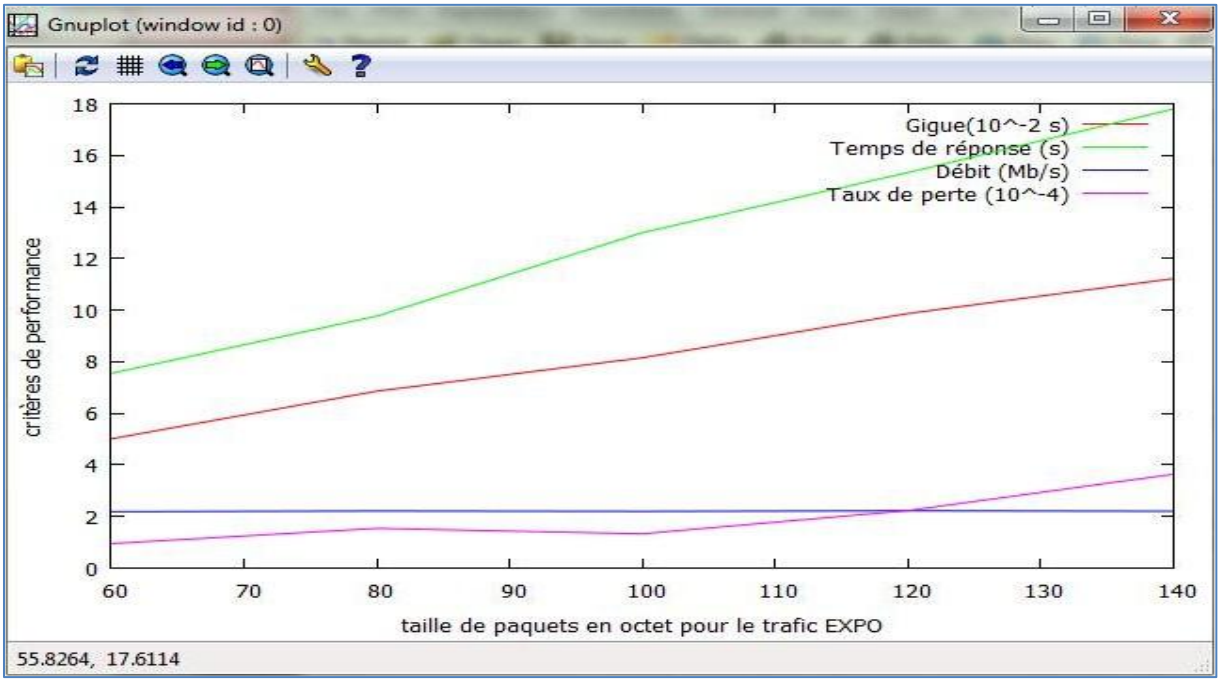


Figure 3.6 : Variation de taille des paquets EXPO (1)

⇨ Il y a une stabilité de débit ainsi qu'une augmentation des valeurs des autres paramètres proportionnellement à la taille des paquets émis.

La simulation totale du nouveau backbone et du test de performance donne la courbe suivante :

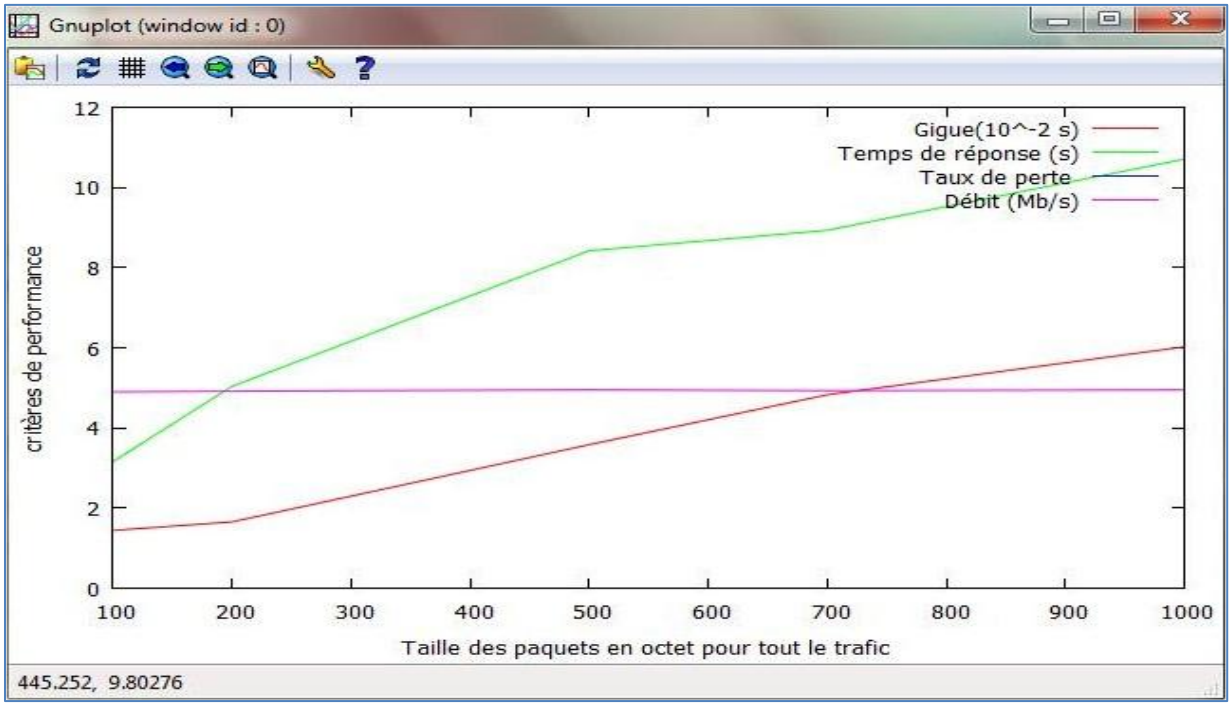


Figure 3.7 : Variation de taille des paquets de trafic global (1)

⇨ Il y a une influence du trafic EXPO sur la performance du nouveau backbone.

3.2. Variation de nombre de connexions :

Les résultats de test de trafic CBR sont portés sur la figure suivante :

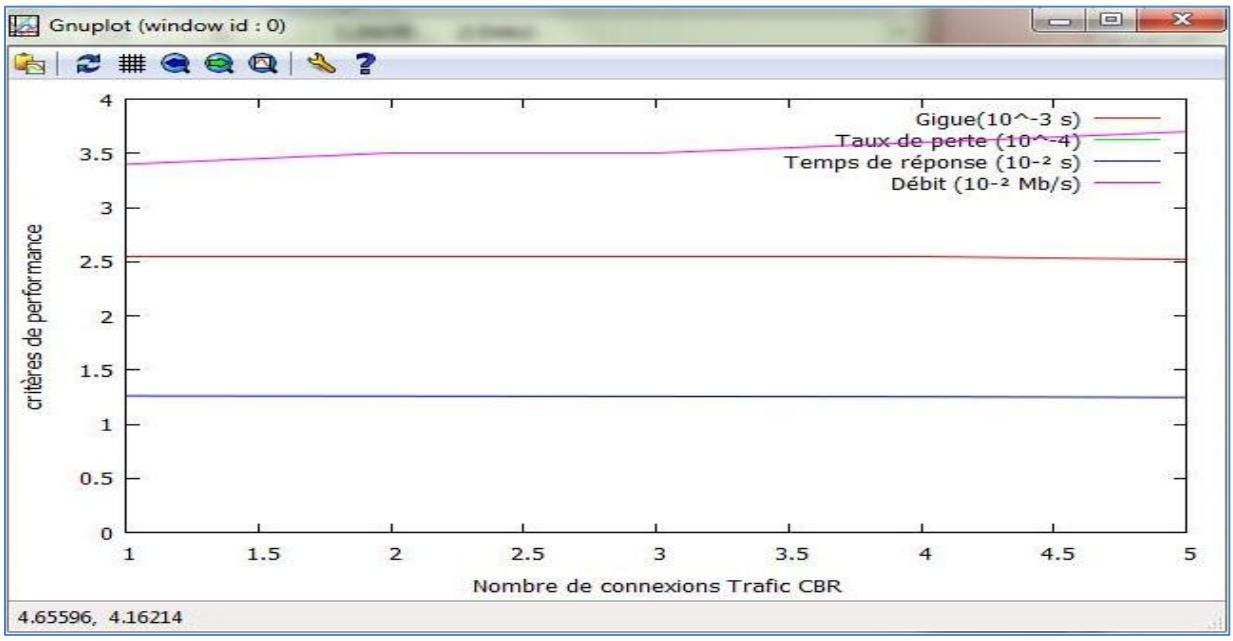


Figure 3.8 : Variation de nombre de connexions de trafic CBR (1)

⇨ Il y a une stabilité de la gigue et du temps de réponse, une croissance de débit et une absence de perte de paquets.

La variation des paramètres de FTP est illustrée comme suit :

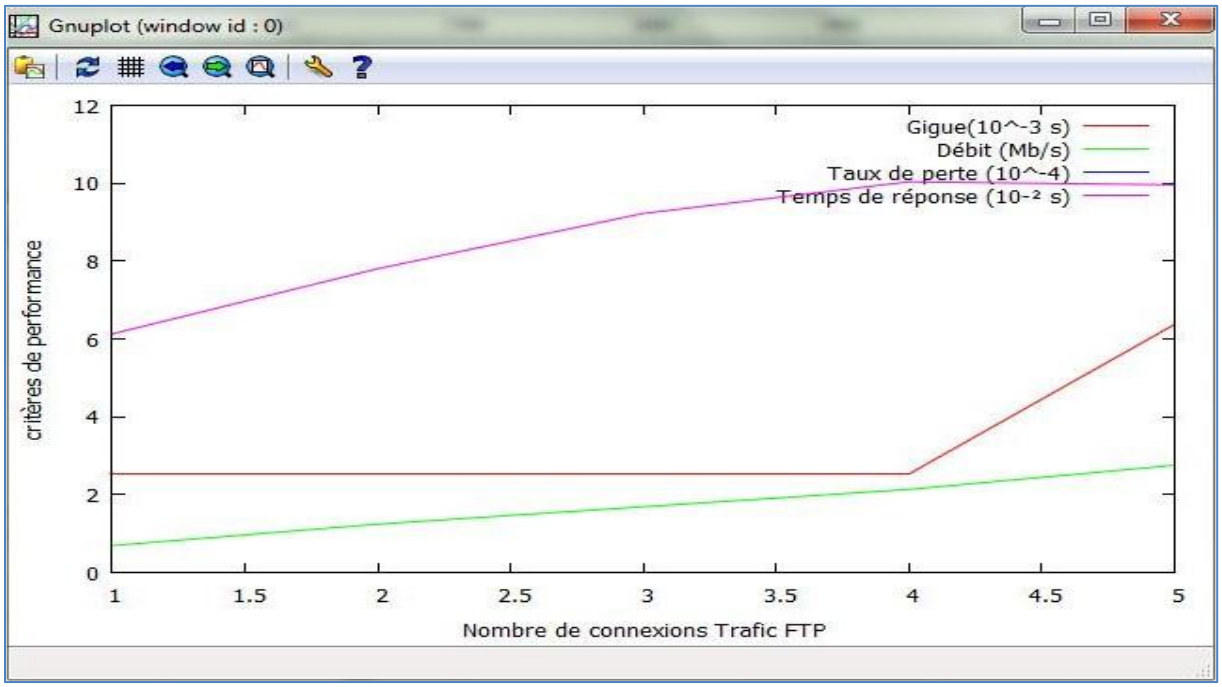


Figure 3.9 : Variation de nombre de connexions de trafic FTP (1)

⇨ Il y a une croissance des valeurs de la gigue, de temps de réponse et de débit cependant le taux de perte reste nul.

La courbe de performance du trafic EXPO est représentée comme suit :

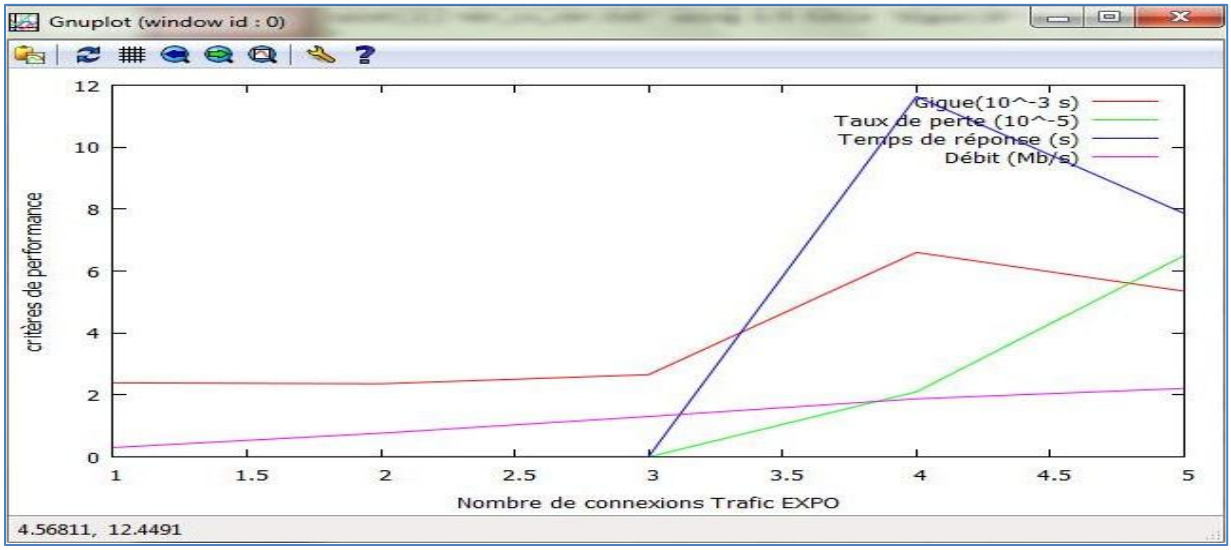


Figure 3.10 : Variation de nombre de connexions de trafic EXPO (1)

⇨ La gigue est constante, le temps de réponse et le taux de perte sont nuls. Au-delà de trois connexions, ces paramètres évoluent proportionnellement avec le nombre de connexions. Le débit évolue dès le début de la simulation en fonction de nombre de connexions.

Enfin, la simulation totale du nouveau backbone MPLS donne la courbe de performance suivante :

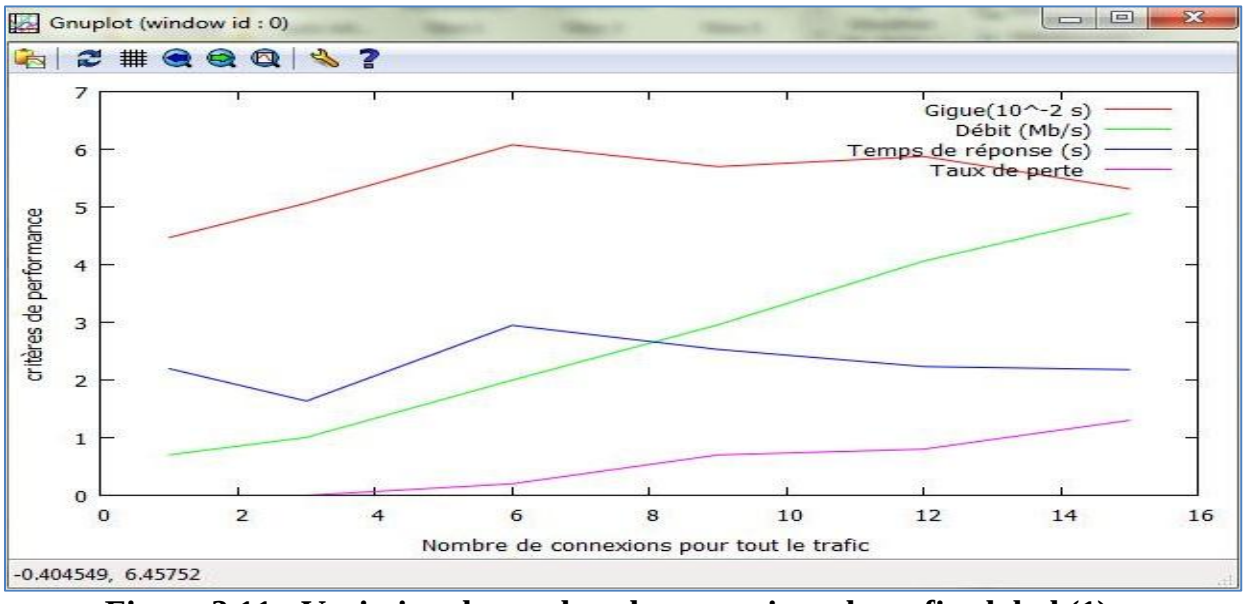


Figure 3.11 : Variation de nombre de connexions de trafic global (1)

⇨ Le taux de perte est faible (il commence à augmenter dès la troisième connexion), le temps de réponse tend à être stable (environ 2.5 s), la gigue varie entre 4.10^{-2} et 6.10^{-2} s et le débit croit en fonction de nombre de connexions.

4. Comparaison des résultats :

Dans ce cadre, j'ai mis deux valeurs différentes (une pour l'architecture existante et une pour la nouvelle) pour chaque paramètre de performance afin de les comparer.

4.1. Perte des paquets :

Le tableau montre le taux de perte en fonction de la taille des paquets :

Taille de paquet (Octet)	*Taux de perte*	
	Architecture existante	*Nouvelle architecture*
100	0.00285	0.00007
200	0.00487	0.000168
500	0.0204	0.000015
700	0.103	0.00038
1000	0.0381	0

Tableau 3.3 : Statistiques de perte en fonction de taille de paquets

Le tableau présente le taux de perte en fonction de nombre de connexions :

Nombre de connexion	*Taux de perte*	
	Architecture existante	*Nouvelle architecture*
1	0.00038	0
3	0.003028	0
6	0.106523	0.000072
9	0.140162	0.000057
12	0.153843	0.000008

Tableau 3.4 : Statistiques de perte en fonction de nombre de connexions

4.2. Débit :

Le tableau indique le débit selon la variation de la taille des paquets :

Taille de paquet (Octet)	*Débit (Mb/s)*	
	Architecture existante	*Nouvelle architecture*
100	2.85	4.899
200	4.87	4.917

500	2.04	4.952
700	10.3	4.929
1000	3.81	4.950

Tableau 3.5 : Statistiques de débit en fonction de taille de paquets

Le tableau illustre le débit en fonction de nombre de connexions :

Nombre de connexion	***Débit (Mb/s)***	
	Architecture existante	***Nouvelle architecture***
1	0.065	0.702
3	2.158	1.005
6	4.134	1.997
9	6.212	2.956
12	7.925	4.061

Tableau 3.6 : Statistiques de débit en fonction de nombre de connexions

4.3. Gigue :

Ce tableau exprime la gigue selon la variation de la taille des paquets :

Taille de paquet (Octet)	***Gigue ($x10^{-2}$ s)***	
	Architecture existante	***Nouvelle architecture***
100	9.4752	8.4418
200	13.4106	1.654
500	17.8621	3.582
700	19.0689	4.826
1000	20.0857	6.023

Tableau 3.7 : Statistiques de gigue en fonction de taille de paquets

Ce tableau illustre la gigue en fonction de nombre de connexions :

Nombre de connexion	***Gigue ($x10^{-2}$ s)***	
	Architecture existante	***Nouvelle architecture***
1	9.078	6.4715
3	8.00	5.0663

6	10.20	6.07827
9	4.745	5.70142
12	4.903	5.87679

Tableau 3.8 : Statistiques de gigue en fonction de nombre de connexions

4.4. Temps de réponse :

Le tableau montre le temps de réponse en fonction de la taille des paquets :

Taille de paquet (Octet)	***Temps de réponse (sec)***	
	Architecture existante	***Nouvelle architecture***
100	31.592262	21.08976
200	50.382914	31.63459
500	84.210824	45.43837
700	89.325680	49.72560
1000	107.072699	53.44255

Tableau 3.9 : Statistiques de temps de réponse en fonction de taille de paquets

Ce tableau présente le temps de réponse selon le nombre de connexions :

Nombre de connexion	***Temps de réponse (sec)***	
	Architecture existante	***Nouvelle architecture***
1	16.352544	0.047727
3	29.483610	35.57487
6	25.304599	32.55907
9	22.324981	20.06811
12	21.794024	15.93005

Tableau 3.10 : Statistique de temps de réponse en fonction de nombre de connexions

4.5. Analyse des résultats :

La nouvelle architecture est plus avantageuse du point de vue performance. En plus, l'administration de backbone sera plus facile car on n'a plus besoin de demander à un fournisseur de service une permission d'accès pour effectuer une simple tâche de maintenance ou de supervision.

5. Implémentation et configuration :

Au cours de cette phase, j'ai implémenté le backbone MPLS à l'aide de GNS3 (voir **Page 61**) et j'ai utilisé des routeurs Cisco 3640 (128Mb de RAM).

Aussi, j'ai ajouté deux clients pour les deux routeurs de Sousse et Jendouba : un pour la CNSS et un autre pour la CNAM.

Etant donné que le réseau CNAM est connecté à celui de la CNSS et qu'il utilise quelques applications propres à la CNSS, j'ai choisi la CNAM comme exemple afin de montrer l'extensibilité de ma solution.

5.1. Adressage :

La répartition des adresses IP est fixée dans le tableau qui suit :

Routeurs	*Interface*	*Adresse IP*	*Adresse Loopback*
R1	E0/0	192.168.1.12	1.1.1.1
	E1/0	192.168.2.13	
	E2/0	192.168.3.14	
R2	E0/0	192.168.4.12	2.2.2.2
	E1/0	192.168.5.13	
	E2/0	192.168.6.14	
	E3/0	192.168.7.15	
R3	E0/0	192.168.8.12	3.3.3.3
	E1/0	192.168.9.13	
	E2/0	192.168.10.14	
R4	E0/0	192.168.11.12	4.4.4.4
	E1/0	192.168.12.13	
	E2/0	192.168.13.14	
	E3/0	192.168.14.15	
R5	E0/0	192.168.15.12	5.5.5.5
	E1/0	192.168.16.13	

Tableau 3.11 : Répartition des adresses IP sur les routeurs du backbone

La figure suivante montre mon backbone MPLS implémenté sous GNS3 :

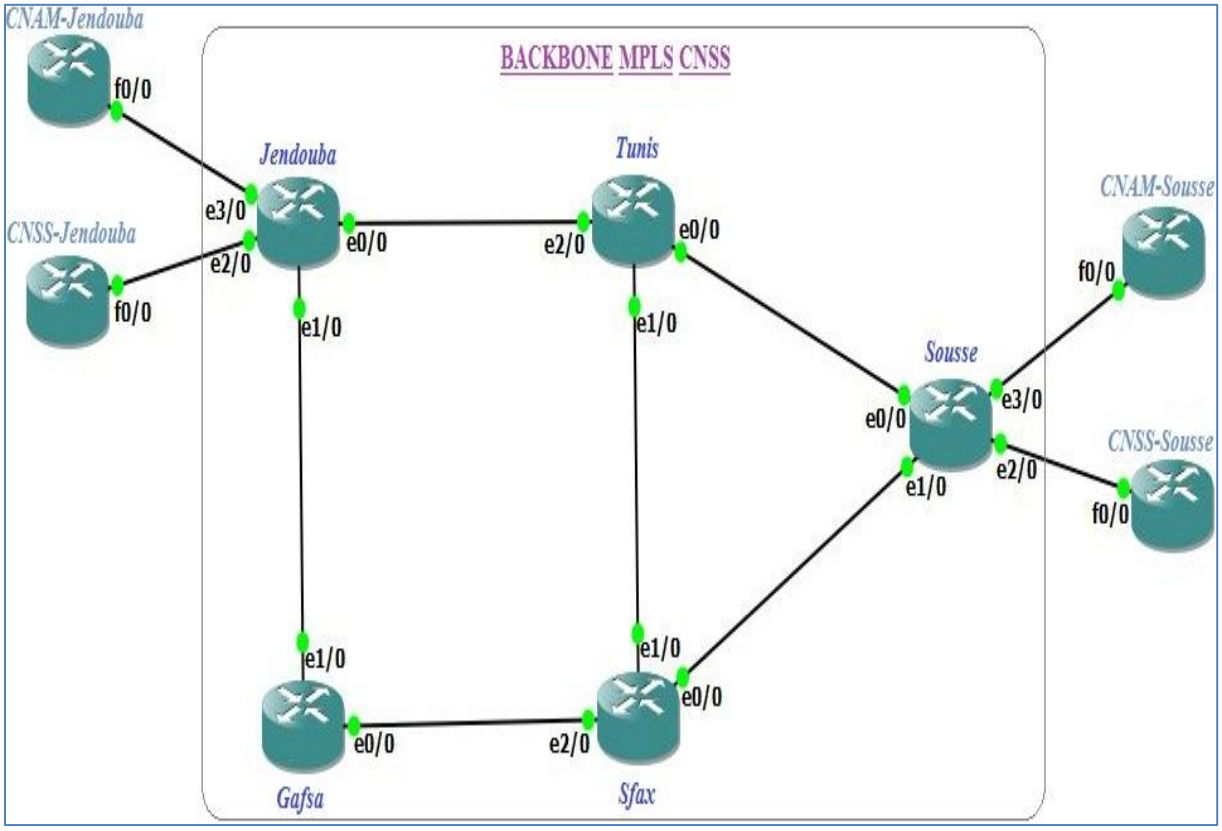


Figure 3.12 : Topologie de backbone MPLS/CNSS

5.2. Configuration :

5.2.1. Outil d'implémentation GNS3 :

GNS3 est un simulateur de réseau graphique qui permet de simuler des réseaux complexes.

Pour fournir des simulations complètes et précises, GNS3 est fortement lié à :

- Dynamips : est un émulateur IOS Cisco.
- Dynagen : est une extrémité avant à base de texte pour Dynamips.
- Qemu : est un émulateur de machine source et virtualiseur.
- VirtualBox : est un logiciel de virtualisation libre et puissant.

GNS3 est un excellent outil complémentaire à des véritables laboratoires pour les ingénieurs réseau, les administrateurs… **[19]**

Il peut également être utilisé pour des fonctionnalités expérimentales de Cisco IOS pour vérifier les configurations qui doivent être déployées plus tard sur des vrais routeurs.

5.2.2. Etapes de configuration :

Dans cette partie, je vais indiquer la liste des différentes étapes pour configurer MPLS sur un backbone IP :

- Configuration de CEF :

La première opération à effectuer pour utiliser MPLS est d'activer CEF comme méthode de commutation sur tous les routeurs du backbone. Il se configure avec la commande suivante :

```
#sh ip cef
```

- Configuration d'un IGP :

Dans mon projet, j'ai utilisé OSPF comme protocole de routage IGP. La configuration d'OSPF au niveau console est la suivante :

```
#router ospf 100
#network 10.0.0.0 0.255.255.255 area 0
```

- Configuration de LDP :

Pour permettre à un routeur d'établir une adjacence LDP avec un voisin sur une interface donnée, cette dernière doit être configurée avec la commande qui suit :

```
#mpls ip
```

- Configuration des VRF :

Le concept de VRF permet à un opérateur de créer plusieurs tables de routage dans un même routeur. Ces tables sont étanches entre elles et chacune est généralement associée à un client. Une même adresse IP peut être affectée plusieurs fois à différentes interfaces car celles-ci sont placées dans des VRF différentes.

J'ai créé les instances VRF associées :

```
(config)#ip vrf CNSS-Jend
(config-vrf)#rd 1:1
(config-vrf)#ip vrf CNAM-Jend
(config-vrf)#rd 1:2
(config-vrf)#exit
```

Puis, j'ai configuré les interfaces du routeur principal :

```
(config)#interface E3/0
(config-if)#ip vrf CNAM-Jend
(config-vrf)#exit
(config)#interface E3/0
(config-if)#ip address 192.168.14.15 255.255.255.0
```

- Configuration MP-BGP des routeurs PE :

La configuration d'un routeur PE, pour échanger des routes VPNv4, se présente sous la forme suivante :

```
(config)#router bgp 65000
(config-router)#neighbor 1.1.1.1 remote-as 65000
(config-router)#neighbor 1.1.1.1 update-source Loopback0
(config-router)#neighbor 3.3.3.3 remote-as 65000
(config-router)#neighbor 3.3.3.3 update-source Loopback0
(config-router)#neighbor 4.4.4.4 remote-as 65000
(config-router)# neighbor 4.4.4.4 update-source Loopback0
(config-router)#neighbor 5.5.5.5 remote-as 65000
(config-router)#neighbor 5.5.5.5 update-source Loopback0
(config-router)#no auto-summary
(config-router)#!
(config-router)#address-family vpnv4
(config-router-af)#neighbor 1.1.1.1 activate
(config-router-af)#neighbor 1.1.1.1 send-community both
(config-router-af)#neighbor 3.3.3.3 activate
(config-router-af)#neighbor 3.3.3.3 send-community both
(config-router-af)#neighbor 4.4.4.4 activate
(config-router-af)#neighbor 4.4.4.4 send-community both
(config-router-af)#neighbor 5.5.5.5 activate
(config-router-af)#neighbor 5.5.5.5 send-community both
(config-router-af)#exit-address-family
```

On remarque la présence d'une section supplémentaire par rapport à une configuration BGP traditionnelle introduite par la commande « address-family vpnv4 ». Cette partie de la configuration BGP contient tous les voisins tournant MP-BGP.

Pour pouvoir l'ajouter dans la configuration VPNv4, un voisin doit être préalablement déclaré dans la configuration globale de BGP (commande « remote-as »). Pour éviter qu'un voisin ne soit actif à la fois pour BGP et MP-BGP, la ligne « no neighbor <neighbor> activate » doit être insérée globalement.

Naturellement, il est possible pour un routeur d'être simultanément actif pour BGP et MP-BGP. Par exemple, le BGP traditionnel servira à propager les routes Internet aux routeurs PE, tandis que MP-BGP servira à la propagation des routes VPN.

➢ Les configurations des routeurs sont présentées en détails. (voir **Annexe°3**)

6. Conclusion :

La nouvelle architecture offre une QoS nettement meilleure. Elle améliore la rentabilité du réseau de la CNSS et facilite la tâche d'administration et la gestion de réseau.

Conclusion Générale

Mon projet de fin d'études est un travail réalisé au sein de la CNSS ayant pour objectif l'amélioration du réseau de télécommunication et l'optimisation de l'administration en installant un backbone MPLS/CNSS.

L'implémentation de la maquette du backbone est le résultat des étapes qui suivent :

- ✓ Etude de l'architecture existante (trafic, critères de performance).
- ✓ Conception de la nouvelle architecture.
- ✓ Simulation et tests.

Ce projet est la première étape dans le processus d'évolution du réseau de la CNSS afin de suivre le progrès technologique.

Ce travail servira la CNSS dans ses futurs projets de migration vers son propre réseau MPLS comme étant la première référence documentaire basée sur des résultats remarquables de tests menés sur des données réelles.

Cette expérience professionnelle au sein de la CNSS m'a permis d'avoir un esprit d'équipe, d'être appliqué et de découvrir le milieu professionnel. Aussi, j'ai appris comment gérer le stress, la routine et les contraintes quotidiennes du travail.

Pour conclure, cette expérience m'a permis d'explorer mes connaissances théoriques acquises à l'Université Virtuelle de Tunis durant les deux dernières années.

Annexes

Annexe°1 : Outils utilisés

Dans cette partie, je vais décrire les outils utilisés :

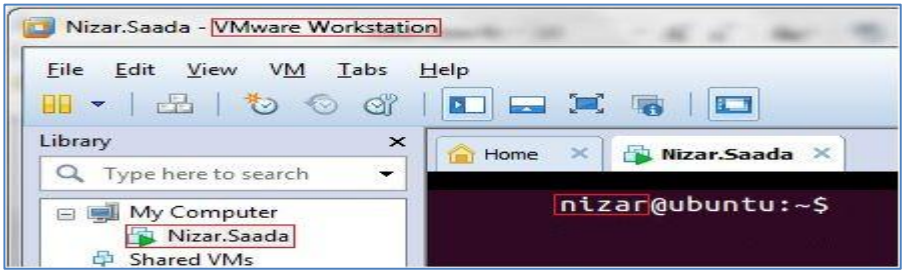
VMware Workstation 9.0.2 :

C'est un puissant logiciel de création et d'utilisation de machines virtuelles qui permet aux développeurs et aux administrateurs des systèmes de révolutionner le développement, les tests et le déploiement des logiciels dans leurs entreprises. **[20]**

Distribué depuis plus de cinq ans et plus de douze fois récompensé par la presse, VMware Workstation permet de développer et de tester sur un seul et même ordinateur les applications-serveurs les plus complexes fonctionnant sous Microsoft Windows, Linux ou NetWare. **[21]**

Doté de fonctions essentielles comme la mise virtuelle en réseau, les captures instantanées, le glissement-déplacement des dossiers et le support virtuel de PXE, VMware Workstation est devenu l'outil indispensable des développeurs en informatique et des administrateurs-système.

« VMware Workstation 9.0.2 » est une version de maintenance qui corrige quelques problèmes connus.

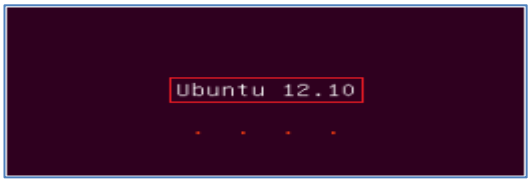


Ubuntu 12.10 :

C'est un système d'exploitation libre, gratuit, sécurisé et convivial, qui peut aisément remplacer ou cohabiter avec un système actuel (Windows, MacOS, GNU/Linux..). **[22]**

L'Ubuntu permet naviguer sur Internet, lire et écrire des courriers, créer des documents, des présentations et des feuilles de calculs et de gérer une bibliothèque multimédia...etc.

« Ubuntu 12.10 », nom de code « The Quantal Quetzal », est la dix-septième version d'Ubuntu sortie le 18 octobre 2012. Cette version amorce un nouveau méta-cycle de développement de quatre versions et sera maintenue pour une durée de 18 mois, soit jusqu'avril 2014. **[23]**



- **NS2 :**

Le simulateur du réseau NS2 est un outil logiciel de simulation de réseaux informatiques. Il est principalement bâti avec les idées de la conception par objets, de réutilisation du code et de modularité. **[M.D]**

NS2 est écrit en C++ et utilise le langage OTCL dérivé de TCL. A travers OTCL, l'utilisateur décrit les conditions de la simulation : la topologie du réseau, les caractéristiques des liens physiques, les protocoles utilisés, les communications qui ont lieu. **[24]**

La simulation doit d'abord être saisie sous forme de fichier que NS va utiliser pour produire un fichier contenant les résultats. Mais l'utilisation de l'OTCL permet aussi à l'opérateur la création de ses propres procédures (par exemple s'il souhaite enregistrer dans un fichier l'évolution d'une variable caractéristique du réseau au cours du temps). **[24]**

NS2 contient les fonctionnalités nécessaires à l'étude des algorithmes de routage unicast ou multicast, des protocoles de transport, de réservation, des services intégrés et des protocoles d'application. **[M.D]**

De plus, ce simulateur possède déjà une palette de systèmes de transmission, d'ordonnanceurs et de politiques de gestion de files d'attente pour effectuer des études de contrôle de congestion. **[S.K]**

Ce logiciel utilise des scripts écrits en langage « TCL » et un outil de visualisation appelé « NAM » pour simuler et visualiser la maquette de réseau.

NAM :

C'est un outil de visualisation qui présente deux intérêts principaux : représenter la topologie d'un réseau décrit avec NS2 et afficher temporellement les résultats d'une trace d'exécution NS2.

Par exemple, il est capable de représenter des paquets TCP ou UDP, la rupture d'un lien entre nœuds ou encore de représenter les paquets rejetés d'une file d'attente pleine. Ce logiciel est souvent appelé directement depuis les scripts TCL pour NS2 pour visualiser directement le résultat de la simulation. **[25]**

Il a aussi la fonction de gèrer l'affichage des nœuds lorsqu'on n'a pas décrit leurs dispositions.

- Pour installer les outils « ns », « nam » et « xgraph » sous « Ubuntu 12.10 », il suffit de taper des simples commandes dans le terminal :

```
nizar@ubuntu:~$ sudo apt-get install ns2
[sudo] password for nizar:
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

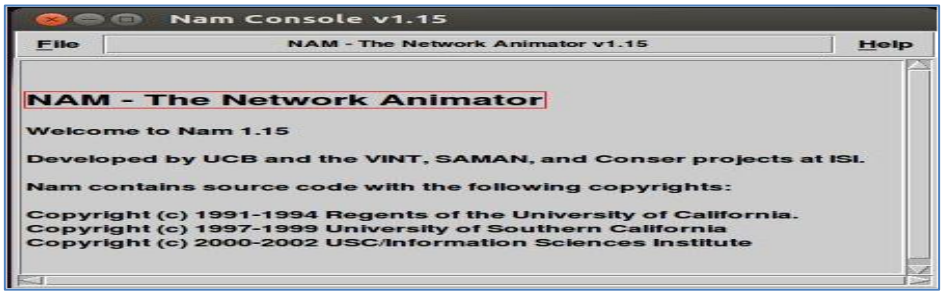
```
nizar@ubuntu:~$ sudo apt-get install nam
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  nam
```

```
nizar@ubuntu:~$ sudo apt-get install xgraph
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  ygraph
The following NEW packages will be installed:
  xgraph
```

- On tape « ns » pour voir % et « nam » pour afficher la fenêtre de démarrage :

```
nizar@ubuntu:~$ ns
% exit
nizar@ubuntu:~$ nam
```

- La fenêtre suivante affirme la réussite de mon installation :



TCL :

C’est un langage de commande comme le Shell Unix mais qui sert à contrôler les applications. Il offre des structures de programmation telles que les boucles, les procédures ou les notions de variables. Il y a deux principales façons de s’en servir :

- ✓ Comme un langage autonome interprété.
- ✓ Comme une interface applicative d'un programme classique écrit en C ou C++.

En pratique, cet interpréteur se présente sous la forme d'une bibliothèque de procédures C qui peut être facilement incorporée dans une application. Cette application peut utiliser les fonctions standard du langage TCL mais également ajouter des commandes à l'interpréteur. [**L.M]**

AWK :

AWK est un langage de traitement de lignes disponible sur la plupart des systèmes Unix et sous Windows.

Il est principalement utilisé pour la manipulation des fichiers textuels pour des opérations de recherches, de remplacement et de transformations complexes. Cet utilitaire a été créé dans le but de remplacer les commandes « grep » et « sed ».

Aujourd'hui, cet utilitaire est toujours utilisé du fait de sa ressemblance avec le langage C, de sa souplesse et de sa présence sur la majorité des systèmes d'exploitation Unix. Il est encore utilisé en administration système et dans les scripts Shell en tant que commande. **[25]**

- ➢ Pour vérifier la version d’Awk, on exécute la commande suivante :

Gnulpot :

C’est un logiciel qui sert à produire des représentations graphiques de fonctions numériques ou de données, en deux ou trois dimensions. Ce programme fonctionne sur de nombreux ordinateurs et systèmes d'exploitation. **[26]**

Annexe n°2 : Scripts Awk

Script 1 :

temp_reponse.awk

```
BEGIN {
total_time = 0;
total_packets = 0;
}
{
action = $1;
time = $2;
src = $3;
dst = $4;
name = $5;
size = $6
flow_id = $8;
src_address = $9;
dst_address = $10;
seq_no = $11;
packet_id = $12;
if (action == "+") {
packets[packet_id] = time;
} else if (action == "-") {
total_packets = total_packets + 1;
total_time = total_time + (time - packets[packet_id]);
}
}
END {
mean = total_time / total_packets * 1000;
printf "%1f", mean;
}
```

Script 2 :

gigue.awk

```
BEGIN {
        for (i in send) {
                send[i] = 0
        }
        for (i in recv) {
               recv[i] = 0
        }
        delay = avg_delay = 0
 }

 {
        # Tracer le format de ligne: normal
        if ($2 != "-t") {
                event = $1
                time = $2
                if (event == "+" || event == "-") node_id = $3
                if (event == "r" || event == "d") node_id = $4
                flow_id = $8
                pkt_id = $12
        }
        # Tracer le format de ligne: nouveau
        if ($2 == "-t") {
                event = $1
                time = $3
                node_id = $5
                flow_id = $39
                pkt_id = $41
        }
```

```
        # Stocker les packets à temps envoyé
        if (flow_id == flow && node_id == src && send[pkt_id] == 0 &&
 (event == "+" || event == "s")) {
                send[pkt_id] = time
                #printf("send[%g] = %g\n",pkt_id,time)
        }
       # Stocker les packets à temps d'arrivé
        if (flow_id == flow && node_id == dst && event == "r") {
                recv[pkt_id] = time
                #printf("\t\trecv[%g] = %g --> delay[%g] =%g
\n",pkt_id,time,pkt_id,recv[pkt_id]-send[pkt_id])
        }
 }

 END {
        # Calculer le délai moyen
        for (i in recv) {
                if (send[i] == 0) {
                        printf("\nErreur %g\n",i)
                }
                delay += recv[i] - send[i]
                num ++
        }

       printf("%10g ",flow)
        if (num != 0) {
                avg_delay = delay / num
        } else {
                avg_delay = 0
        }
        printf("%10g\n",avg_delay*1000)
}
```

- **Script 3 :**

perte.awk ×

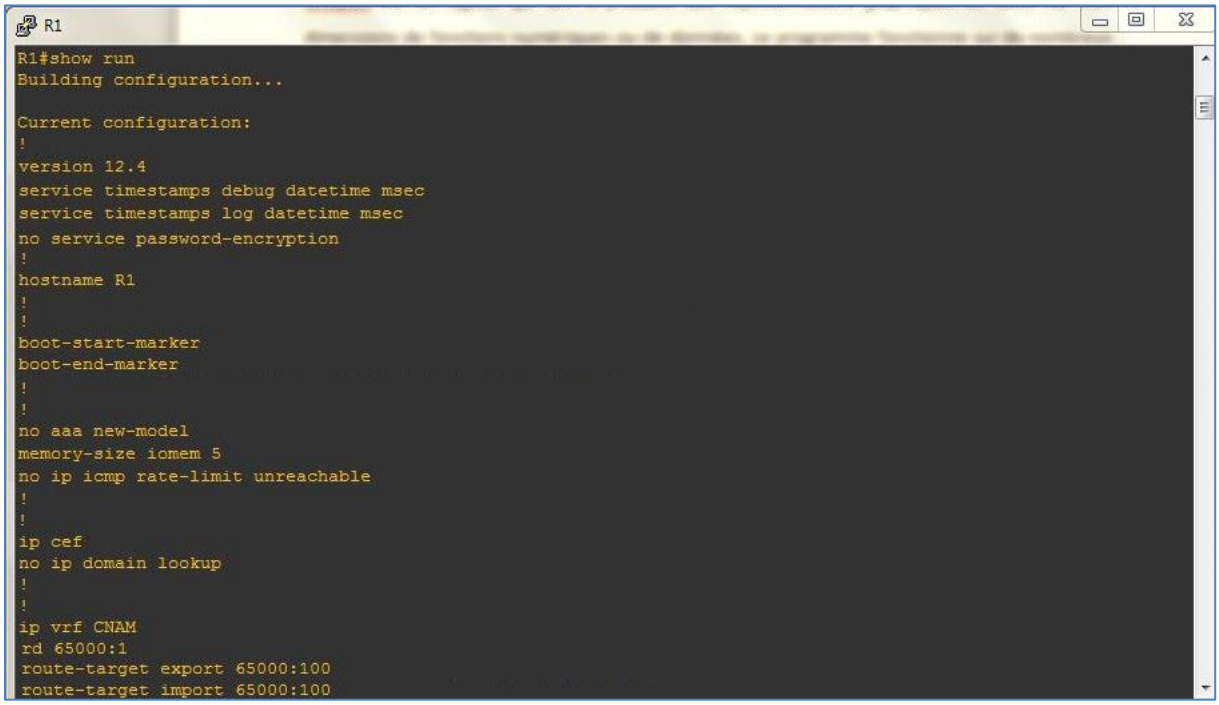
```
BEGIN {
# Données de la simulation: début à 0, pas de 100ms
record_time = 0.0;
time_interval = 0.100;
received_bytes = 0;
lost_packets = 0;
total_received_bytes = 0;
total_lost_packets = 0;
total_loss_bytes = 0;
loss_bytes = 0;
total_bytes = 0;
}
{
action = $1;
time = $2;
src = $3;
dst = $4;
name = $5;
size = $6
flow_id = $8;
src_address = $9;
dst_address = $10;
seq_no = $11;
packet_id = $12;
if ((action == "r" || action == "d")){ #&& src == "2" && dst == "3") {
# Si "received time" appartient à un nouvel intervalle de temps

if (time > (record_time + time_interval)) {
total_received_bytes = total_received_bytes + received_bytes;
total_loss_bytes = total_loss_bytes + loss_bytes;
```

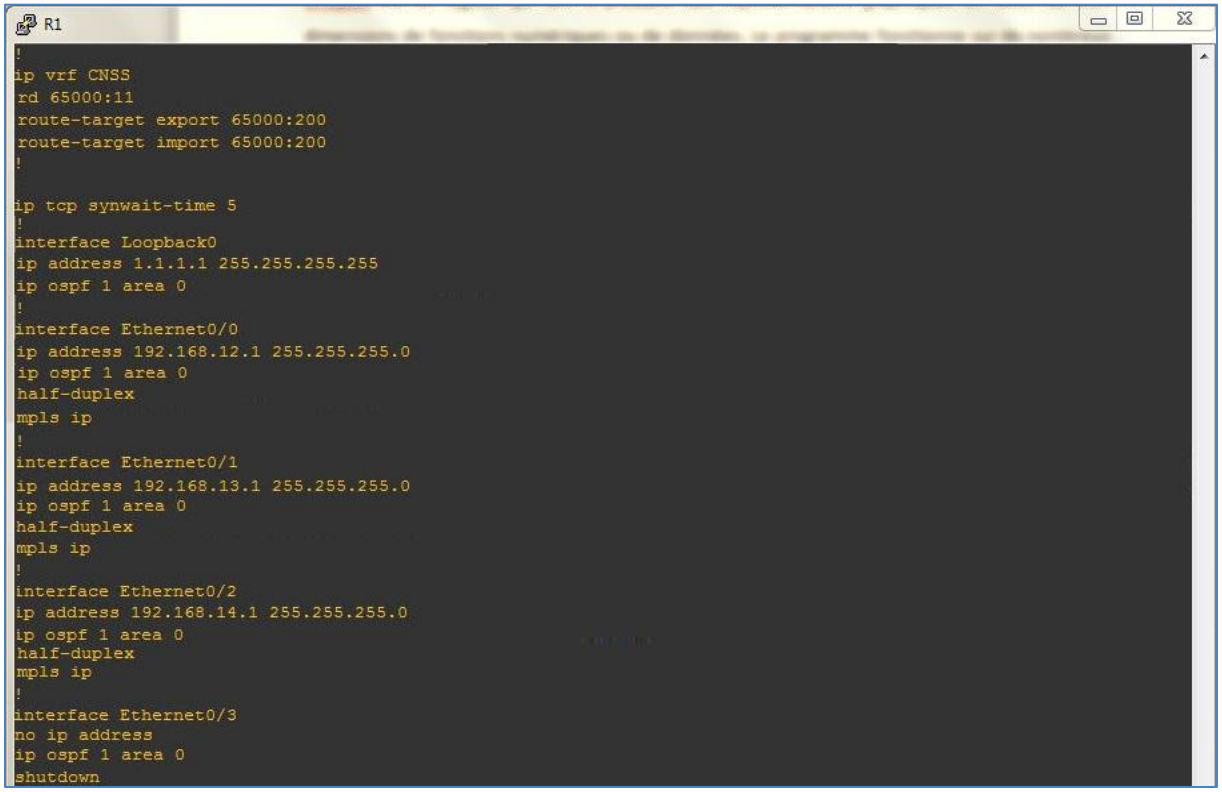
```
total_lost_packets = total_lost_packets + lost_packets;
record_time = record_time + time_interval;
total = received_bytes * 8 / time_interval /1000000;
total_lost = lost_packets;
printf "%f %f %f\n", record_time, total_lost, total > "total.dat";
while (time > (record_time + time_interval)) {
record_time = record_time + time_interval;
printf "%f %f %f\n", record_time, 0.0, 0.0 > "total.dat";
}
# Si "received packet" arrive au prochain intervalle de temps
{
action = $1;
time = $2;
src = $3;
dst = $4;
name = $5;
size = $6
flow_id = $8;
src_address = $9;
dst_address = $10;
seq_no = $11;
packet_id = $12;
if ((action == "r" || action == "d")){ #&& src == "2" && dst == "3") {
# Si "received time" appartient à un nouvel intervalle de temps
if (action == "r") {
received_bytes = size;
lost_packets = 0;
loss_bytes = 0; # La ligne que j'ai ajoutée -------------
} else if (action == "d") {
loss_bytes = size;
received_bytes = 0;
lost_packets = 1;
}
} else {
# Si "rcv_time" appartient encore à cet intervalle de temps
if (action == "r") {
received_bytes = received_bytes + size;
loss_bytes = 0; # La ligne que j'ai ajoutée---------
} else if (action == "d") {
lost_packets = lost_packets + 1;
loss_bytes = loss_bytes + size;
}
}
}
}
END {
total_bytes = total_loss_bytes + total_received_bytes;
average_throughput = total_received_bytes * 8 / record_time / 1000000;
taux_de_perte = total_loss_bytes / total_bytes ;
printf "Debit Moyen: %1.3f (Mbits)\n", average_throughput;
printf "Nombre de paquets perdus: %1f\n", total_lost_packets;
printf "Taux de perte: %1f\n", taux_de_perte;
}
```

Annexe n°3 : Configuration du Backbone

Remarque : Les routeurs R1, R2, R3, R4 et R5 sont configurés de la même manière et les clients de backbone MPLS de la CNSS (Jendouba et Sousse) ont la même implémentation.

Configuration de routeur R1 :





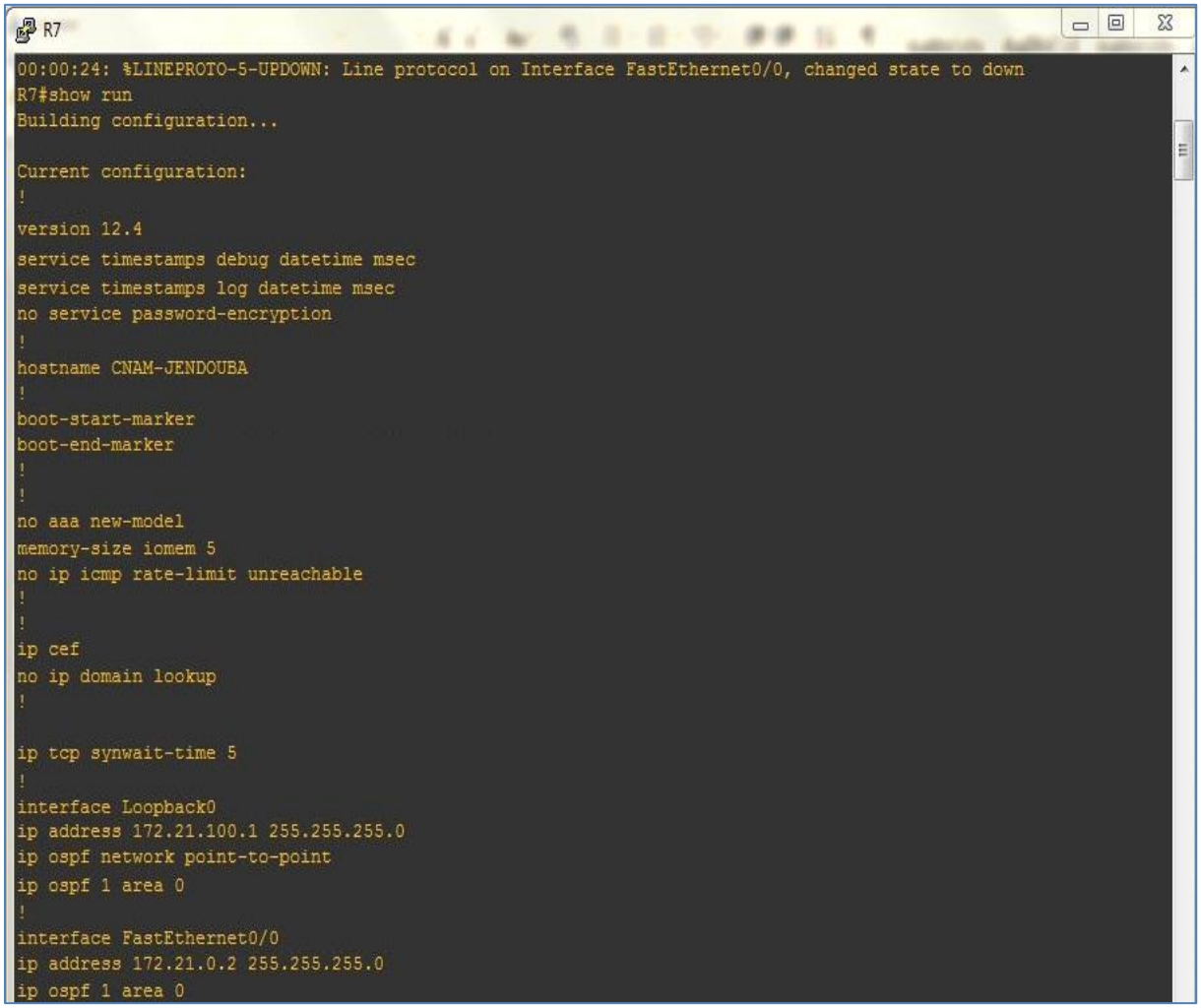
R1

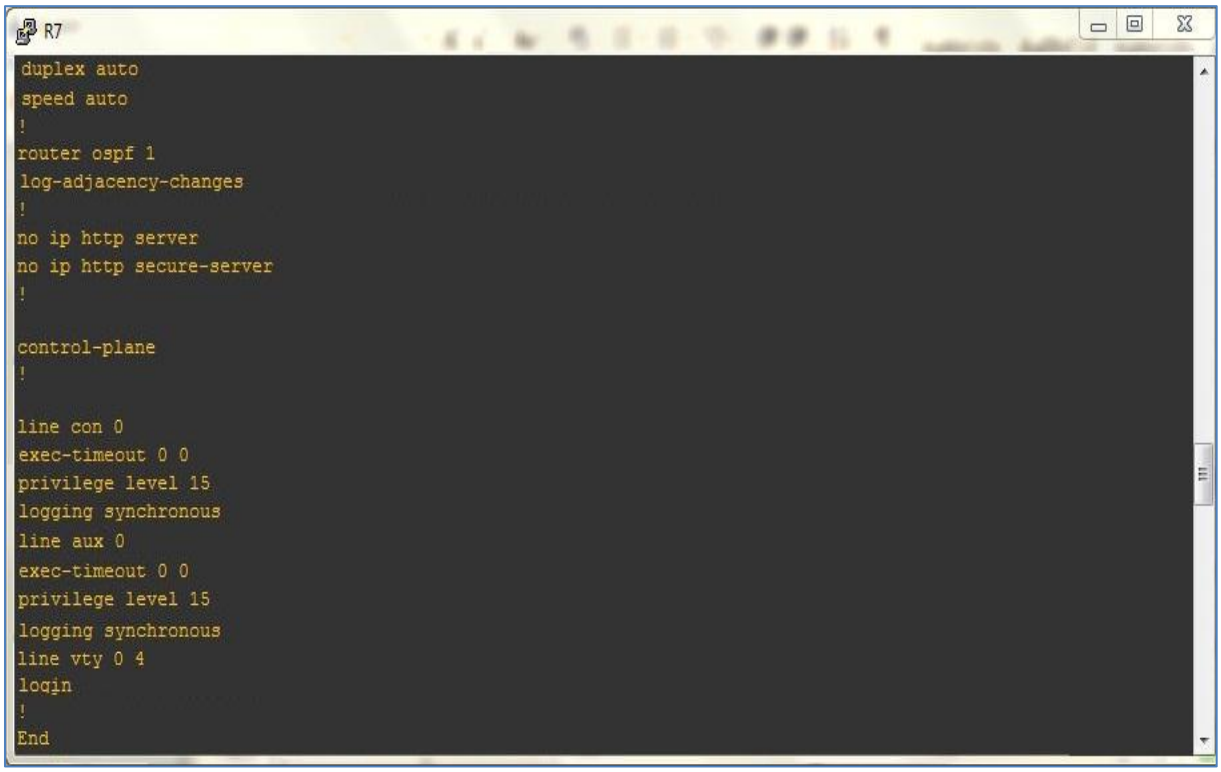
```
half-duplex
mpls ip
!
router ospf 1
log-adjacency-changes
!
router bgp 65000
no synchronization
bgp log-neighbor-changes
neighbor 2.2.2.2 remote-as 65000
neighbor 2.2.2.2 update-source Loopback0
neighbor 3.3.3.3 remote-as 65000
neighbor 3.3.3.3 update-source Loopback0
neighbor 4.4.4.4 remote-as 65000
neighbor 4.4.4.4 update-source Loopback0
neighbor 5.5.5.5 remote-as 65000
neighbor 5.5.5.5 update-source Loopback0
no auto-summary
!
address-family vpnv4
neighbor 2.2.2.2 activate
neighbor 2.2.2.2 send-community both
neighbor 3.3.3.3 activate
neighbor 3.3.3.3 send-community both
neighbor 4.4.4.4 activate
neighbor 4.4.4.4 send-community both
neighbor 5.5.5.5 activate
neighbor 5.5.5.5 send-community both
exit-address-family
!
address-family ipv4 vrf CNSS
no synchronization
```

R1

```
exit-address-family
!
address-family ipv4 vrf CNAM
no synchronization
exit-address-family
!
no ip http server
no ip http secure-server
!

control-plane
!

line con 0
exec-timeout 0 0
privilege level 15
logging synchronous
line aux 0
exec-timeout 0 0
privilege level 15
logging synchronous
line vty 0 4
login
!
End
```

Configuration CNAM Jendouba:





Webographie :

[1]: http://www.humanforcetunisie.com/Bibli/Loi-cnss-tunisie.php#_Toc72041853

[2]: http://fr.scribd.com/doc/80012946/Mpls-Cisco-by-Amichoo-Istanet

[3]: http://www.memoireonline.com/03/11/4293/Mise-en-oeuvre-dun-coeur-dereseau-IPMPLS.html

[4]: http://fr.scribd.com/doc/76647205/MPLS-GMPLS

[5]: http://www.guill.net/index.php?cat=3&pro=3&wan=5

[6]: http://wapiti.telecomlille1.eu/commun/ens/peda/options/ST/RIO/pub/exposes/exposesrio2002ttnfa03/Bordessoules-Bret/Site/MPLS.htm

[7]: http://www.invocom.et.put.poznan.pl/~invocom/C/INT/tcpip/fr/ch5.html

[8]: http://www.frameip.com/vpn/

[9]: http://www.memoireonline.com/12/10/4148/Mise-en-place-dune-architecture-VPN-MPLS-avec -gestion-du-temps-de-connexion-et-de-la-bande-passan.html

[10]: http://www.frameip.com/mpls/

[11]: http://www.frameip.com/mpls-cisco/

[12]: http://www-igm.univ-mlv.fr/~dr/XPOSE2006/marot/

[13]: http://www-igm.univ-mlv.fr/~dr/XPOSE2007/ykarkab_MPLS/

[14]: http://www.bonnefoy.eu/25_OSPF

[15]: http://www.dissertationsgratuites.com/dissertations/Microprocesseur-8086/495133.html

[16]: http://www.inetdoc.net/guides/zebra.ospf/zebra.ospf.area.html

[17]: http://mpls-rt.wikispaces.com/

[18]: http://www.memoireonline.com/09/13/7405/m_Conception-et-deploiementde - la-technologie-MPLS-dans-un-reseau-metropolitain11.html

[19]: http://cyril-k.blogspot.com/2011/03/gns3-tutoriel-d-installation.html

[20]: http://www.ciao.fr/VMware_Workstation_5_Licence__766460

[21]: http://www.microaffaires.com/product.php?id_product=2028

[22]: http://doc.ubuntu-fr.org/ubuntu_distribution

[23]: http://www.infothema.fr/forum/index.php?topic=181.0

[24]: http://y-baddi.developpez.com/tutoriels/ns2/

[25]: http://fr.wikipedia.org/wiki/Awk

[26]: http://fr.wikipedia.org/wiki/Gnuplot

Bibliographie :

[T.L]: Textes législatifs et règlementaires de sécurité sociale dans le secteur privé ; Ministère des affaires sociales, de la solidarité et des tunisiens à l'étranger ; République Tunisienne ; 2007

[M.M]: Mouadh MENSI ; Etude et mise en place d'un Backbone ADSL à base MPLS en utilisant la technologie VPN ; Projet de Fin d'Année en Ingénierie Réseaux Informatiques et Télécoms ; 2011

[R.L]: Rami LANGAR ; Mécanismes de Gestion de la Mobilité et Evaluation de Performance dans les Réseaux Cellulaires tout-IP ; Thèse Doctorat ; 2006

[O.F]: Oussama FOUDHAILI ; Analyse des performances de MPLS en terme de Traffic Engineering dans un réseau multiservice ; Projet de Fin d'Etudes en Ingénierie des Réseaux ; 2004

[M.D]: Mariam DAWOUD ; Analyse du protocole AODV ; DEA Informatique ; 2005

[S.K]: Sabeur KAMMOUN ; Implémentation de la QoS sur un protocole de routage (multicast) Ad hoc ; Projet de Fin d'Etudes en Ingénierie des Réseaux Mobiles ; 2005

[L.M]: Layouni MAJID ; Simulator Network 2 ; 2013