

---

---

# Etude expérimentale : déploiement IPv6

- adressage, autoconfiguration, routage, ICMPv6, QoS -

---

---

Rapport de projet SR04  
Grégoire Martinache  
Théo Bocquelet  
Minh Tri Lê  
Augustin De Champs

Université de Technologie de Compiègne  
Génie Informatique - Automne 2016



# Table des matières

<b>1</b>	<b>La couche réseau</b>	<b>3</b>
1.1	Présentation de la couche réseau . . . . .	3
1.1.1	Rôles et services . . . . .	3
1.1.2	Composition . . . . .	4
<b>2</b>	<b>Le routage</b>	<b>5</b>
2.1	Présentation . . . . .	5
2.1.1	Algorithme de routage . . . . .	5
2.2	Les protocoles de routage . . . . .	7
2.2.1	Le protocole RIP (RIPng : Routing Information Protocol) . . .	7
2.2.2	Le protocole OSPF (OSPFv3 : Open Shortest Path First) . . .	8
2.3	Le routage multicast . . . . .	9
2.3.1	Les besoins . . . . .	9
2.3.2	Les difficultés . . . . .	10
2.3.3	Quelles solutions? . . . . .	10
<b>3</b>	<b>Le protocole IPv6</b>	<b>13</b>
3.1	Présentation . . . . .	13
3.1.1	Pourquoi l'IPv6? . . . . .	13
3.1.2	Difficultés rencontrées . . . . .	14
3.2	Types d'adresses IPv6 . . . . .	14
3.2.1	Les adresses réservées . . . . .	14
3.2.2	Les adresses unicast . . . . .	15
3.2.3	Les adresses multicast . . . . .	15
3.2.4	Les adresses anycast . . . . .	16
<b>4</b>	<b>Le protocole ICMPv6</b>	<b>17</b>
4.1	Présentation . . . . .	17
4.1.1	Qu'est-ce-qu'ICMP? . . . . .	17
4.1.2	Quelles sont les nouveautés apportées par ICMPv6? . . . . .	18
4.1.3	Détermination de l'adresse source d'un message . . . . .	18

4.1.4	Traitement des messages ICMPv6 . . . . .	18
4.2	Messages d'erreur . . . . .	19
4.2.1	Type 1 : destination inatteignable . . . . .	19
4.2.2	Type 2 : paquet trop grand . . . . .	19
4.2.3	Type 3 : temps dépassé . . . . .	20
4.2.4	Type 4 : problème de paramètre . . . . .	20
4.3	Messages informatifs . . . . .	20
4.3.1	Type 128 : demande d'écho . . . . .	21
4.3.2	Type 129 : réponse d'écho . . . . .	21
<b>5</b>	<b>La qualité de service</b>	<b>23</b>
5.1	Présentation . . . . .	23
5.2	La qualité de service dans l'IPv6 [1] . . . . .	26
5.2.1	Systèmes de type Integrated Services : le champ Flow Label .	26
5.2.2	Systèmes de type DiffServ : le champ Traffic Class . . . . .	27
5.2.3	Le champ HopByHop . . . . .	28
5.2.4	Absence de NAT . . . . .	28
<b>6</b>	<b>Expérimentation</b>	<b>29</b>
6.1	Connexion au routeur . . . . .	29
6.2	Configuration du routeur . . . . .	29
6.2.1	Configuration du vlan . . . . .	29
6.2.2	Configuration des interfaces . . . . .	30
6.2.3	Configuration IPv6 . . . . .	30
6.2.4	Test de la connexion et adressage IPv6 . . . . .	31
6.2.5	Sauvegarde des différentes configurations du routeur . . . . .	32
6.3	Mise en place des services . . . . .	33
6.3.1	Conférence audio . . . . .	34
6.3.2	Transfert de fichier . . . . .	36
6.3.3	Téléphonie . . . . .	36
6.3.4	Vidéo à la demande, streaming . . . . .	37
6.4	Mise en place de la qualité de service . . . . .	38
6.4.1	Configuration d'une classe . . . . .	38
6.4.2	Configuration d'une politique de traitement des paquets . . .	38
6.5	Test de la qualité de service . . . . .	39
6.5.1	Saturation du réseau . . . . .	39
6.5.2	Test de la QoS par Ping . . . . .	40
6.5.3	Test de la QoS par l'essai . . . . .	42
<b>7</b>	<b>Conclusion</b>	<b>43</b>
	<b>Bibliographie</b>	<b>45</b>



# Chapitre 1

## La couche réseau

### 1.1 Présentation de la couche réseau

#### 1.1.1 Rôles et services

Le rôle de la couche réseau est d'acheminer efficacement des paquets d'une source vers une destination suivant un parcours. Ce parcours peut être composé d'étapes intermédiaires. La couche réseau doit déterminer le meilleur chemin à prendre pour acheminer les paquets et doit donc connaître la topologie du réseau. Les réseaux peuvent avoir des protocoles différents, ce qui est un élément important à prendre en compte lors de la conception de la couche réseau.

Un tel rôle demande un système de routage (2) et d'adressage performant pour diriger les paquets entrants vers leur prochaine destination et doit tenir compte du réseau en lui-même (topologie, panne, surcharge, ...). Il faut donc considérer les problèmes de congestion et plus généralement, la qualité de service (QoS : 5).

La couche réseau doit aussi tenir compte des besoins de l'application (type de service, fiabilité, ...). Selon le besoin, il faut choisir entre le mode non connecté et le mode connecté :

- Soit les paquets sont émis et routés indépendamment les uns des autres : les paquets sont nommés datagramme et le réseau est donc un réseau de datagramme.
- Si une connexion avec le destinataire est établie avant le début de la transmission, cette connexion est appelée circuit virtuel et le réseau est un réseau de circuit virtuel.

Aspect	Datagrammes	Circuits virtuels (CV)
Phase d'établissement	Non nécessaire	Requise
Adressage	Chaque paquet contient les adresses complètes de la source et de la destination.	Chaque connexion requiert des informations d'identification de circuit dans la table de routage.
Routage	Chaque paquet est routé indépendamment.	La route est choisie lors de l'établissement du CV et suivie par tous les paquets.
Impact d'une panne de routeur	Aucun, excepté pour les paquets perdus au moment de l'incident.	Tous les CV passant par le routeur sont supprimés.
Qualité de service (QoS)	Difficile à garantir	Facile à garantir si suffisamment de ressources peuvent être allouées par avance pour chaque CV.
Contrôle de congestion	Difficile.	Facile si suffisamment de ressources peuvent être allouées par avance pour chaque CV

**Table 1.1** – Comparaison de réseaux de datagrammes et de réseaux de circuits virtuels [15]

La couche réseau rend service à la couche transport et utilise la couche liaison. Elle rajoute un en-tête au segment de la couche transport puis fragmente si besoin le paquet pour la couche liaison. Il faut impérativement indépendance entre technologies de routeur et services pour assurer la maintenabilité et compatibilité.

### 1.1.2 Composition

Pour assurer ses rôles, la couche réseau utilise une table de routage pour l'adressage et des algorithmes de routage pour le choix de la prochaine destination.

La couche réseau utilise le protocole IP dans l'Internet qui est un protocole en mode non connecté. IPv4 et IPv6 (3) sont deux versions de ce protocole. Nous étudierons la version IPv6 qui utilise le protocole ICMPv6 (4).

# Chapitre 2

## Le routage

### 2.1 Présentation

Le routage est le mécanisme qui gère les tables de routage et choisit l'itinéraire à suivre dans le réseau afin d'acheminer des paquets. Le routage peut se faire vers un ou plusieurs destinataires (multicast : 2.3) en utilisant un algorithme de routage approprié.

#### 2.1.1 Algorithme de routage

L'algorithme de routage décide du routage à suivre pour les paquets entrants et gère les tables de routage.

L'algorithme de routage doit être apte à s'adapter à toutes variations du réseau (défaillance matérielle et logicielle, modification rapide de la topologie du réseau, congestion). Il doit se montrer efficace au niveau local et global (maximiser l'efficacité globale peut être opposé à maximiser l'efficacité locale) et doit donc faire des compromis.

Un algorithme de routage peut être statique ou dynamique. Dans le cas dynamique, l'algorithme construit la table de routage automatiquement en échangeant des informations au sein du réseau alors que dans le cas statique, la table doit être implémentée et modifiée manuellement.

Nous nous intéresserons aux algorithmes de routage dynamiques, en particulier, le routage par vecteur de distance et le routage par informations d'état de lien.

##### 2.1.1.1 Routage par vecteur de distance (Distance vector)

Le routage par vecteur de distance fonctionne par le principe de collaboration entre routeurs directement voisins. Chaque routeur a une table de routage



contenant l'adresse de l'émetteur et du destinataire et le coût associé (dépend de plusieurs paramètres : nombre de saut, degré de congestion, temps de transit estimé). Les routeurs vont communiquer périodiquement (hello-time) le contenu de leur table de routage (update).

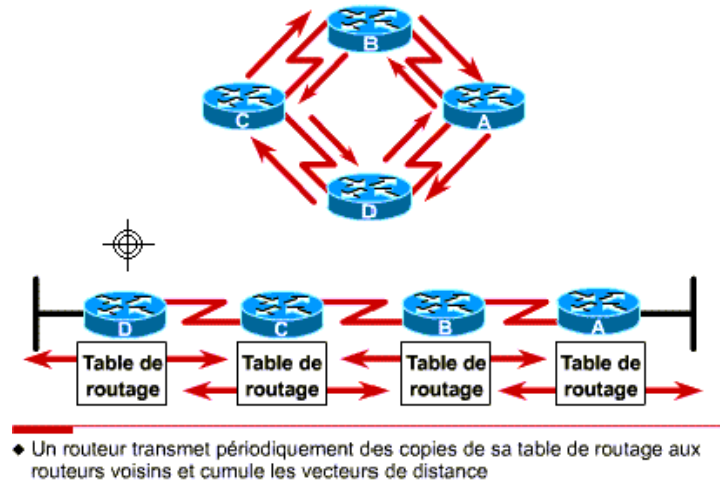


Figure 2.1 – Routage par vecteur de distance [13]

Le routeur va ainsi pouvoir actualiser sa table de routage et prendre en compte les modifications du réseau (nouvelles destinations, coût modifié, panne).

Suivant la destination qu'un paquet doit prendre, le routeur va consulter sa table de routage et emprunter la destination au coût le plus faible. Ainsi, la distance pour arriver à destination est calculée pas à pas et le routeur n'a pas connaissance de la topologie entière du réseau.

Les désavantages des protocoles à vecteurs de distance sont :

- La charge réseau par les updates
- Le délai de convergence : un délai non négligeable est nécessaire pour savoir qu'un voisin est définitivement déconnecté (panne ou non).  
Cet algorithme n'est donc pas efficace pour gérer un réseau à la topologie non stable.
- La latence : un certain délai est nécessaire pour actualiser la table de routage si le réseau est grand.

#### 2.1.1.2 Routage par informations d'état de lien (Link State)

Contrairement à l'algorithme précédent, le routage par informations d'état de lien va chercher à comprendre la topologie complète du réseau, ce qui répond aux limitations du routage vecteur de distance.

Cet algorithme se déroule en cinq points :

- Découverte des voisins du routeur : communique avec les voisins directs afin d'obtenir leur adresse et prise en compte des LAN à diffusion (réseau diffusant une information à tous les routeurs connectés du LAN) où il est seulement nécessaire de connaître une adresse du LAN.
- Définition des coûts de lien : associe une adresse à un coût qui dépend de la distance, du débit et temps de transit estimé afin de trouver le chemin le plus optimal (au coût moindre).
- Élaboration des paquets d'états de lien : Dès que le routeur possède les informations nécessaires sur ses voisins, il transmet un paquet d'états de lien à tous ses voisins qui renseigne l'état du réseau à proximité direct du routeur.  
Le paquet contient l'adresse de l'émetteur, un numéro de séquence, un âge, et l'adresse de ses voisins associés à leur coût. Il faut aussi décider de la fréquence d'envoi de ces paquets (par période fixe ou lors d'une modification du réseau).
- Distribution des paquets d'états de lien : Assure l'envoi et la réception des paquets d'états de lien à tous les routeurs, et rapidement. Cette étape gère les numéros de séquence ainsi que l'âge des paquets afin d'éviter toute duplication.
- Calcul de nouvelles routes : Dès que le routeur a reçu tous les paquets d'états de lien, il va pouvoir représenter le réseau dans son ensemble par un graphe. Ensuite, il suffit d'appliquer l'algorithme de Dijkstra sur le graphe qui va calculer le plus court chemin.

Le principal désavantage de cet algorithme est qu'il est gourmand en capacité mémoire (chaque routeur a connaissance du réseau en entier) et calcul (il faut régulièrement calculer le plus court chemin du graphe). Cependant, il est réactif aux variations du réseau.

## 2.2 Les protocoles de routage

Nous allons traiter deux protocoles de routage intradomaine (protocole de passerelle intérieure) très répandus.

### 2.2.1 Le protocole RIP (RIPng : Routing Information Protocol)

Le protocole de routage RIP se base sur le routage par vecteur de distance (2.1.1.1). De ce fait, il possède les mêmes défauts et fonctionne sous le même principe, mais introduit une limite de coût : 15, afin d'éviter les boucles de routage. RIPng inclut les masques sous-réseau dans les updates pour les réseaux qui en ont besoin (VLMS : Variable Length Masque Subnet).

Il supporte l'IPv6.

L'adresse multicast utilisée est 224.0.0.9.

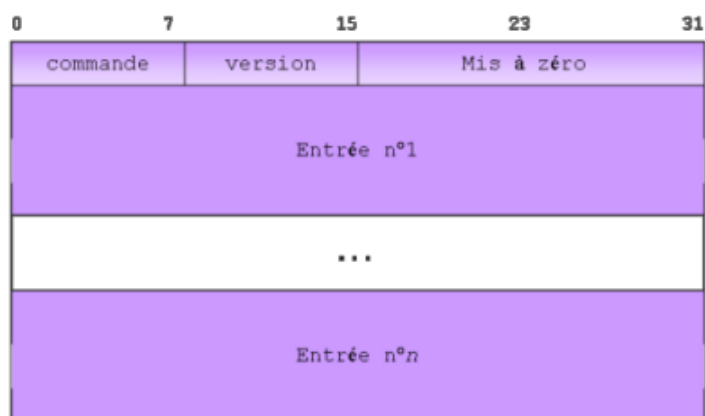


Figure 2.2 – Format d'un paquet RIPng [12]

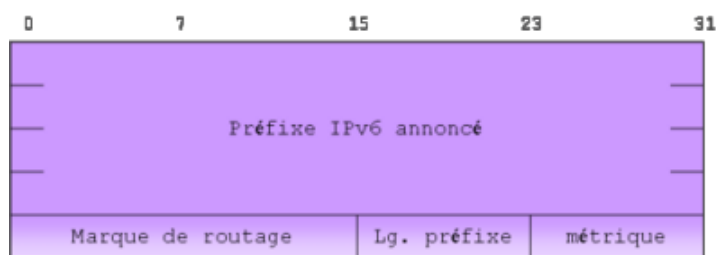


Figure 2.3 – Champ entrée d'un paquet RIPng [12]

Il n'est pas adapté aux grands réseaux.

## 2.2.2 Le protocole OSPF (OSPFv3 : Open Shortest Path First)

Le protocole de routage OSPF se base sur le routage par état de lien (2.1.1.2). Il répond aux défauts du protocole RIP.

Voici les services apportés par le protocole :

- Protocole ouvert donc non propriétaire.
- Routage par type de service : il est possible de spécifier un type de service dans les paquets.
- Permet la division du réseau en aires autonomes et indépendantes entre elles. L'interface entre deux aires est nommée l'aire backbone, ce qui permet d'optimiser le routage en communiquant des résumés de route ainsi que la nature des réseaux IP. Chaque routeur a donc connaissance de la topologie de son aire.
- Gère le routage machine-machine, vers réseaux et sous-réseaux.

- Utilise l’adressage multicast IP (voir 3.2.3).
- Réactif aux variations du réseau (panne, congestion. . .) et optimal : car basé sur le routage par état de lien. Il tient compte du trafic afin de ne pas envoyer tous les paquets sur la meilleure route et de ne pas délaissier les autres routes possibles.
- Peut évaluer une variété de types de coût (nombre de sauts, débit. . .).
- Supporte VLMS (comme RIPng (2.2.1)).
- Supporte l’IPv6 (voir 3).

OSPF utilise cinq types de messages différents :

Type de paquet OSPF	Description
Type 1 Hello	Etablit et maintient les informations de contiguïté ( <i>adjacency information</i> ) avec les voisins.
Type 2 Database Description packet (DBD)	Décrit le contenu des bases de données d'état de liens ( <i>link-state database</i> ) des routeurs OSPF.
Type 3 Link-state request (LSR)	Demande des éléments spécifiques des bases de données d'état de liens ( <i>link-state database</i> ) des routeurs OSPF.
Type 4 Link-state update (LSU)	Transporte les <i>link-state advertisements</i> , les LSA, aux routeurs voisins.
Type 5 Link-state acknowledgment (LSAck)	Accusés de réception des LSA des voisins.

Figure 2.4 – Type de paquet OSPF [9]

Ce protocole est rapide, robuste aux variations du réseau, interopérable et permet de gérer des paramètres utiles à la QoS (5) (répartition du trafic, prise en compte du type de service. . .).

Il possède les mêmes défauts que le routage par état de lien (2.1.1.2), mais la répartition par aire complique la configuration.

## 2.3 Le routage multicast

### 2.3.1 Les besoins

Une application requiert de diffuser des données en continu à un groupe particulier de personnes, comme le streaming vidéo en direct d’un concert et plus généralement, la téléconférence. Dans cet exemple, un émetteur unique transmet un flux d’information à un public visé. Pour ce faire, il faut utiliser le routage multicast.

### 2.3.2 Les difficultés

Toute la difficulté du routage multicast réside dans la diffusion efficace de l'information au groupe concerné.

Cela requiert donc un routage performant.

L'efficacité de la diffusion dépend de deux cas de figure :

- Le groupe est dense : les membres du groupe sont dispersés sur une majeure partie du réseau.
- Le groupe n'est pas dense : beaucoup d'éléments du réseau ne font pas partie du groupe.

### 2.3.3 Quelles solutions ?

Pour communiquer à plusieurs membres particuliers du réseau, plusieurs techniques existent. Elles consistent toutes à obtenir le graphe du groupe de diffusion sans boucles (arbre du groupe) par élagage du graphe du réseau sans boucles (arbre recouvrant).

#### 2.3.3.1 Avec un routage par état de lien

Dans ce cas, la topologie complète du réseau est connue par chaque routeur. Il est ainsi facile de construire un graphe contenant le réseau sans les éléments qui ne sont pas membres du groupe ainsi que leurs liens.

Cela permet d'obtenir un graphe spécifique au groupe de diffusion.

#### 2.3.3.2 Avec un routage par vecteur de distance (DVMRP : Distance Vector Multicast Routing Protocol)

Cette solution se base sur le protocole RIP (2.2.1).

En trois étapes :

- Un premier message pour le groupe est envoyé sur le réseau. Le champ TTL (time to live) du message restreint la zone de diffusion de ce dernier.
- Tous les routeurs recevant ce message, mais ne faisant pas partie du groupe multicast et n'étant pas liés directement à un membre renvoient un message PRUNE à l'émetteur. Ce message précise qu'un routeur n'est pas membre du groupe et ne veut plus recevoir de message de ce groupe.
- À partir de la réponse de chacun des récepteurs, il est ainsi possible de construire un arbre du groupe multicast.

L'arbre obtenu ne contient que les liaisons utiles pour diffuser un message sur le groupe multicast. Il est donc efficace.

Cependant, le problème est que cela nécessite une grande capacité de travail pour chaque routeur, en particulier dans les grands réseaux. En effet, il faut constituer

et mémoriser un arbre pour chaque routeur membre du groupe dans un réseau de taille supérieur ou égal.

### 2.3.3.3 Avec la technique du cœur de l'arbre (CBT : Core-based tree)

Dans ce cas, un seul arbre est calculé par groupe et chaque routeur membre en a connaissance.

Le principe est de décider d'un routeur racine (membre du groupe) : le cœur. L'arbre obtenu par cette technique est le CBT.

Le cœur doit d'abord constituer le CBT. Chaque membre envoie un paquet vers le cœur, ce qui permet d'obtenir la route entre le cœur et chaque membre du groupe. L'arbre du groupe a donc le cœur comme racine et tous ses membres en branches et feuilles.

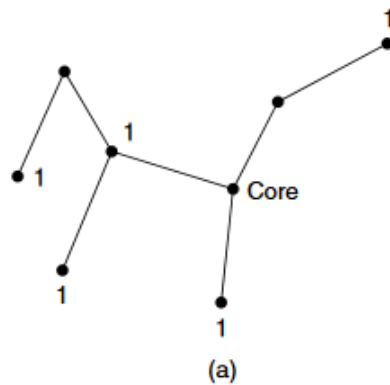


Figure 2.5 – Core-based tree

Le choix de la racine est critique pour la performance.

Cette technique est très optimale si le cœur est au milieu de l'arbre, et donc à distance plus ou moins égale de chaque membre du groupe. Elle se montre utile dans le cas où le groupe est dispersé dans le réseau.

De plus, seul un arbre est nécessaire pour chaque membre du groupe et ceux qui ne sont pas membres du groupe ne travaillent pas. Ainsi, on gagne en capacité mémoire et calcul.



## Chapitre 3

# Le protocole IPv6

### 3.1 Présentation

#### 3.1.1 Pourquoi l'IPv6 ?

Le protocole IPv6 succède au protocole IPv4. Celui-ci a été conçu par l'Internet Engineering Task Force (IETF) pour surmonter les manques de l'IPv4. L'IPv6 apporte notamment :

- Beaucoup plus d'adresses IP possibles (avec un maximum de 667 millions de milliards d'appareils connectés sur chaque millimètre carré de la surface de la Terre!).

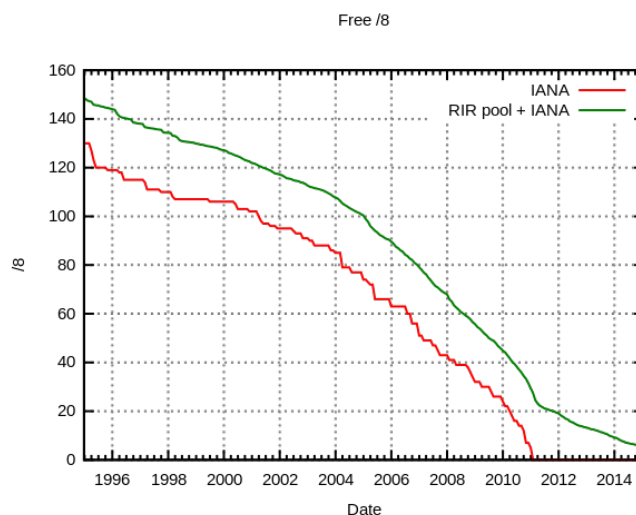


Figure 3.1 – Quantité d'adresse IPv4 disponibles au cours du temps

- L'implémentation directe de la Quality of Service (QoS 5);



- L'implémentation d'un protocole de sécurité (IPSec);
- Le multicast;
- La simplification de l'entête des paquets.

### 3.1.2 Difficultés rencontrées

L'IPv6 est incompatible avec l'IPv4 rendant compliqué son déploiement sur Internet. Les hôtes doivent donc proposer une *double pile*, c'est à dire que les appareils disposent à la fois d'une adresse IPv4 et d'une IPv6.

```
enp7s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.42 netmask 255.255.255.0 broadcast 192.168.2.255
    inet6 fe80::4ab9:ab2:77dd:ee4c prefixlen 64 scopeid 0x20<link>
    ether 82:00:00:00:00:00 txqueuelen 1000 (Ethernet)
    RX packets 1888088 bytes 2454882205 (2.2 GiB)
    RX errors 0 dropped 3 overruns 0 frame 0
    TX packets 1052324 bytes 86004366 (82.0 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Figure 3.2 – Exemple de double pile sur mon ordinateur

L'IPv6 étant une technologie encore jeune, les entreprises doivent également éduquer l'ensemble du personnel informatique ralentissant et compliquant la procédure de migration de IPv4 à IPv6 dans les grandes entreprises en plus de la rendre chère.

## 3.2 Types d'adresses IPv6

Préfixe	Description
::/8	Adresses réservées
2000::/3	Adresses unicast routables sur Internet
fc00::/7	Adresses locales uniques
fe80::/10	Adresses locales lien
ff00::/8	Adresses multicast

Table 3.1 – Les différents types d'adresses IPv6.

*NB : On utilise la notation CIDR pour l'IPv6, de la même façon qu'avec l'IPv4.*

Nous allons désormais expliciter les différents types d'adresses ainsi que leurs utilités.

### 3.2.1 Les adresses réservées

Certaines adresses IPv6 sont réservées pour une utilisation particulière : il en existe 13[10]. On pourra noter les plus importantes :

Adresse	Description
::/128	Adresse non spécifiée. Elle correspond à 0.0.0.0 en IPv4.
::1/128	Boucle locale. Elle correspond à 127.0.0.1 en IPv4.
2002::/16	6to4.[16]. Mécanisme permettant d'envoyer des paquets IPv6 sur un réseau ne supportant que l'IPv4.

Table 3.2 – Quelques adresses IPv6 réservées.

### 3.2.2 Les adresses unicast

À la grande différence de l'adressage IPv4, il n'y a plus d'adresse de Broadcast (diffusion) en IPv6. On pourra retrouver les adresses Unicast, Multicast et Anycast.

L'adresse Unicast sert à définir un hôte particulier. Un paquet émis avec cette adresse de destination n'est remis qu'à la machine ayant cette adresse IPv6.

IPv6 inclut deux assignations différentes d'adresses unicast :

- adresse unicast globale ;
- adresse lien local.

En IPv6, les adresses unicast se présentent ainsi :

Champ	Préfixe	Sous-réseau	Interface
Bits	48	16	64

Table 3.3 – Le format d'une adresse unicast. [6]

Les adresses en lien local sont toujours dans *fc00::/7*.

### 3.2.3 Les adresses multicast

L'adresse de Multicast concerne un ensemble d'hôtes appartenant à un même groupe de diffusion. Un paquet émis avec cette adresse de destination est remis à l'ensemble des machines concernées par cette adresse.

En IPv6, les adresses multicast se présentent ainsi :

Champ	Préfixe	Drapeau	Portée	Groupe
Bits	8	4	4	112

Table 3.4 – Le format d'une adresse multicast. [3]

### 3.2.4 Les adresses anycast

L'adresse Anycast est ni plus ni moins de l'adressage multicast, à la différence qu'un paquet émis avec cette adresse de destination ne sera remis qu'à un seul membre du groupe.

Pour l'instant, une seule adresse anycast est définie et est réservée au routeur mais dans l'avenir, d'autres pourraient être définies.

La construction d'une adresse IPv6 anycast d'un sous-réseau (la seule définie actuellement) concatène le préfixe du sous réseau de l'interface et la valeur nulle pour la dernière partie de l'adresse.

<i>n bits</i>	<i>112 - nbits</i>
Préfixe de sous-réseau	ID du groupe

**Table 3.5** – Le format d'une adresse anycast.

## Chapitre 4

# Le protocole ICMPv6

### 4.1 Présentation

#### 4.1.1 Qu'est-ce-qu'ICMP ?

ICMP : Internet Control Message Protocol

ICMPv6 est l'un des protocoles fondamentaux d'Internet. Il permet aux routeurs d'envoyer des messages d'erreurs ou de contrôle à d'autres routeurs ou machines. Ces messages contiennent des informations sur les problèmes rencontrés lors de la transmission des paquets.

Quelques exemples :

- rapporter des erreurs trouvées dans le traitement de paquets
- effectuer des diagnostics

Les datagrammes ICMP sont transportés à l'intérieur de datagrammes IPv6 dans lequel un en-tête d'extension peut aussi être présent. Un message ICMP est identifié par sa valeur 58 positionnée dans le champ Next Header de l'en-tête IPv6. Le protocole ICMPv6 est implémenté au niveau de la couche 3.

Type	Code	Checksum
	Message Body	

**Table 4.1** – Format d'un message ICMPv6

Champ *type* : indique le type du message. Sa valeur détermine le format des données.

Champ *code* : fournit des informations supplémentaires sur le type du message.

Champ *checksum* : utilisé pour détecter une éventuelle corruption des données.

### 4.1.2 Quelles sont les nouveautés apportées par ICMPv6 ?

Quelques nouveautés :

- suppression de certains types de messages devenus obsolètes
- renforcement de la sécurité des messages ICMP avec l'apparition du système d'authentification (Authentication Header) et de confidentialité (Encapsulating Security Payload Header) de IPv6
- ICMPv6 reprend des fonctionnalités qui étaient à la base implémentées dans d'autres protocoles pour IPv4, par exemple la découverte de voisin qui était effectuée par ARP

On peut aussi noter que le protocole ICMPv6 n'est pas seulement utilisé pour rapporter des erreurs mais implémente par exemple le mécanisme de découverte de route utilisé pour configurer les hôtes. Il est aussi utilisé pour le management des adresses multicast.

### 4.1.3 Détermination de l'adresse source d'un message

Un noeud envoyant un message ICMPv6 doit déterminer à la fois les adresses IPv6 de source et de destination de l'en-tête IPv6 avant de calculer la somme de contrôle. Si le noeud a plus d'une adresse unicast, il doit choisir l'adresse source du message de la manière suivante :

- si le message est une réponse à un message envoyé à l'une des adresses unicast du noeud, l'adresse source de la réponse doit être la même.
- si le message est une réponse à un message envoyé en multicast, ou anycast, d'un groupe dont le noeud est membre, l'adresse de la réponse doit appartenir au groupe.

### 4.1.4 Traitement des messages ICMPv6

Les règles suivantes sont appliquées pour traiter les messages ICMPv6 :

- 1 : les messages d'erreur inconnus doivent être passés à la couche supérieure ayant causé le datagramme responsable de l'erreur
- 2 : les messages informatifs inconnus sont jetés
- 3 : un noeud IPv6 doit limiter le nombre de messages d'erreur qu'il envoie
- 4 : un message d'erreur ne doit jamais être envoyé en réponse à :
  - un message d'erreur
  - un message de redirection
  - un paquet destiné à une adresse multicast IPv6
  - un paquet dont l'adresse source identifie plusieurs noeuds

## 4.2 Messages d'erreur

On identifie les messages d'erreur grâce à la présence d'un 0 en bit de poids fort pour le type, cela représente donc les types de 0 à 127.

Types de messages d'erreur ICMPv6 :

- 1 : Destination inatteignable
- 2 : Paquet trop grand
- 3 : Temps écoulé
- 4 : Problème de paramètre
- 100 : Expérimentation privée
- 101 : Expérimentation privée
- 127 : Réservé pour la création de nouveaux messages d'erreur

Ces messages permettent d'effectuer des diagnostics, de savoir pourquoi le paquet n'a pas été acheminé, ou pourquoi il a été rejeté. Ils sont en rapport avec les problèmes de livraison des paquets IP.

Une fois reçu, ces messages sont délivrés soit aux processus utilisateurs, soit à un protocole de transport (exemple : TCP).

### 4.2.1 Type 1 : destination inatteignable

Les messages de ce type sont utilisés pour indiquer qu'un paquet a pu ne pas être délivré à son destinataire soit à cause d'un problème rencontré sur le chemin, soit à cause d'un manque d'intérêt du destinataire.

Ce message n'utilise pas le champ message body.

Champ *code* :

- 0 : Pas de route vers la destination
- 1 : Communication avec la destination interdite administrativement (firewall)
- 2 : La destination est hors de portée
- 3 : Adresse inatteignable
- 4 : Port inatteignable
- 5 : La politique de filtrage estime que le paquet n'est pas conforme
- 6 : Rejet de la route vers la destination

### 4.2.2 Type 2 : paquet trop grand

Ce message utilise le champ message body pour y indiquer le MTU du noeud posant problème.

Champ *code* : il est mis à 0 par l'émetteur et ignoré par le récepteur.

Champ *MTU* : contient la valeur du MTU du noeud qui a rejeté le paquet

### 4.2.3 Type 3 : temps dépassé

Ce message est envoyé par un routeur lorsque la durée de vie d'un paquet est dépassée.

Ce message n'utilise pas le champ message body.

Champ *code* :

0 : Le "hop limit" est dépassé

1 : Temps de réassemblage des fragments dépassé

### 4.2.4 Type 4 : problème de paramètre

Ce message ICMP est envoyé quand un paramètre dans un paquet reçu est invalide.

Ce message utilise le champ message body pour y stocker un pointeur.

Champ *code* :

0 : Champ d'en-tête erroné

1 : En-tête suivant non reconnaissable

2 : Option IPv6 non reconnue

Champ *pointeur* : identifie l'octet du paquet où l'erreur a été détectée. Le pointeur pointera au delà de la fin du paquet ICMPv6 si le champ erroné se trouve au delà de ce qui peut tenir dans un message d'erreur ICMPv6.

## 4.3 Messages informatifs

On identifie les messages informatifs grâce à la présence d'un 1 en bit de poids fort pour le type, cela représente donc les types de 128 à 255.

Types de messages informatifs ICMPv6 :

128 : Demande d'écho

129 : Réponse à un écho

200 : Expérimentation privée

201 : Expérimentation privée

255 : Réservée pour la création de nouveaux messages informatifs

Les messages d'écho sont par exemple utilisés par l'utilitaire "ping", permettant de savoir si une machine est présente sur le réseau. Ils servent à tester la connexion entre deux points.

Type	Code	Checksum
Identifiant	Numéro de séquence	
Données		

**Table 4.2** – Format des message de demande et réponse d'écho []

#### 4.3.1 Type 128 : demande d'écho

Champ *identifiant* : l'identifiant aide à faire correspondre les réponses d'écho aux demandes.

Champ *numéro de séquence* : le numéro de séquence aide à faire correspondre les réponses d'écho aux demandes.

Champ *données* : contient des données à transmettre

#### 4.3.2 Type 129 : réponse d'écho

Champ *identifiant* : contient l'identifiant du message correspondant à la demande d'écho.

Champ *numéro de séquence* : contient le numéro de séquence du message correspondant à la demande d'écho.

Champ *données* : contient des données provenant du message correspondant à la demande d'écho.

Les messages de réponse d'écho doivent être passés au noeud qui a créé une demande d'écho.

On peut noter qu'il n'y a pas de limitation à la quantité de données que l'on peut fournir dans des messages de demande d'écho ou de réponse.





## Chapitre 5

# La qualité de service

### 5.1 Présentation

Les besoins de chaque flux de données peuvent être caractérisés par quatre principaux paramètres : la fiabilité (reliability) , le temps d'acheminement (delay), la gigue (jitter), et la bande passante (bandwidth).

Fiabilité : quel est le taux de perte des paquets dans le réseau ?

Temps d'acheminement : quel est le temps d'acheminement des paquets de la source à la destination ?

Gigue : avec quelle régularité les paquets arrivent-ils au destinataire ?

Bande passante : quelle quantité d'information peut être transmise en un temps donné (débit) ?

Les besoins pour chaque paramètre dépendent de l'application :

Application	Fiabilité	Temps d'acheminement	Gigue	Bande passante
Email	Important	Faible	Faible	Faible
Transfert de fichier	Important	Faible	Faible	Moyen
Navigation web	Important	Moyen	Faible	Moyen
Connexion distante	Important	Moyen	Moyen	Faible
Audio à la demande	Faible	Faible	Important	Moyen
Vidéo à la demande	Faible	Faible	Important	Important
Téléphonie	Faible	Important	Important	Faible
Vidéoconférence	Faible	Important	Important	Important

**Table 5.1** – Exemples des besoins de différentes applications. [15]

C'est pourquoi il est important de faire de la qualité de service en fonction du type d'application visée.

Il existe différentes techniques pour obtenir une bonne QoS :

- Surapprovisionnement (overprovisioning) :  
Il s'agit de disposer de plus de ressources (espace de stockage, bande passante, débit...) que nécessaire, afin que tous les paquets puissent transiter sans la moindre attente. Cette solution est simple mais extrêmement coûteuse à mettre en place. D'autres solutions moins coûteuses permettent généralement d'atteindre les objectifs de Qualité de Service fixés.
- La mise en mémoire tampon (buffering) :  
Il s'agit de stocker les paquets d'un flux chez le destinataire avant de les délivrer. Cette solution permet de réduire la gigue (on peut contrôler la vitesse d'arrivée des paquets au destinataire) mais augmente le temps d'acheminement de la source à la destination.
- La régulation de flux (traffic shaping) :  
Il s'agit ici d'imposer à l'émetteur de transmettre ses paquets à une vitesse régulière (constante). Cela permet de diminuer la congestion dans le réseau (on empêche une machine d'envoyer une grande quantité de paquets en une seule fois).  
Dans un protocole en mode connecté, l'émetteur et le récepteur peuvent aussi convenir des paramètres des flux de données à échanger lors de l'établissement de la connexion. Lorsque le destinataire reçoit un flux, et si le flux respecte la convention établie, le destinataire s'engage à transmettre les données dans les temps. Cette technique s'appelle la limitation du flux (traffic policing).
- L'algorithme du seau percé (leaky bucket, Turner, 1986) :  
Dans un seau percé rempli d'eau, peu importe la quantité d'eau dans le seau, le débit sortant ne varie pas ; et lorsque le seau est plein, l'eau excédentaire est perdue. Dans cet algorithme, les paquets arrivants sont stockés dans une structure de file à taille fixée. A chaque tick d'horloge et si la file n'est pas vide, un paquet est défilé et envoyé. Lorsque la file est pleine, un paquet arrivant est jeté. Cet algorithme permet de fluidifier des échanges irréguliers ou limiter le débit dans un réseau.
- L'algorithme du seau à jetons (token bucket) :  
Cet algorithme est similaire à l'algorithme du seau percé mais plus flexible. Le seau percé contient un certain nombre de jetons. Un jeton est généré chaque intervalle de temps jusqu'à un certain seuil maximum du nombre de jetons total. Chaque paquet peut consommer un jeton pour être immé-

diatement transmis. Lorsqu'il n'y a plus de jetons disponibles, les paquets sont stockés comme dans l'algorithme du seau percé mais ne seront jamais jetés (a priori). Cet algorithme permet de mieux réguler le trafic lorsque de grande quantité de données arrivent à un routeur.

— La réservation de ressource :

L'idée ici est de prévoir un chemin pour un certain flux de données (on crée une sorte de circuit virtuel), et de réserver toutes les ressources nécessaires sur cette route afin de garantir une bonne qualité de service pour ce flux passant par cette route. On pourra réserver de la bande passante, de l'espace de stockage ou encore de la capacité CPU dans les routeurs par exemple.

— Le contrôle d'admission :

Lorsqu'un routeur reçoit un flux, il doit décider selon ses ressources disponibles immédiatement, de l'accepter ou de le rejeter. Cependant cette décision est compliquée car elle dépend de nombreux paramètres donnés (ou non) par l'émetteur. De plus, tous les routeurs sur le chemin de l'émetteur au destinataire peuvent avoir leurs propres contraintes.

Une solution est que l'émetteur transmette au destinataire les paramètres qu'il voudrait négocier (dont il aurait besoin) : nous les nommerons spécifications du flux. Tous les routeurs sur le chemin peuvent les examiner et les modifier si besoin mais seulement en réduisant le flux. Lorsque les spécifications ont été examinées par toutes les machines du chemin, elles sont adoptées : l'émetteur devra respecter ces spécifications lors de l'émission. Ainsi chaque routeur le long du chemin peut réserver les ressources nécessaires correspondantes.

— Le routage proportionnel :

Au lieu de transmettre un flux arrivant, sur le meilleur chemin comme dans les algorithmes de routage classiques, il est possible de diviser ce flux et de le transmettre sur plusieurs chemins différents. Une méthode simple est de diviser le flux proportionnellement aux capacités des liens sortants.

— L'ordonnancement de la transmission des paquets (packet scheduling) :

Un problème dans un routeur recevant plusieurs flux de données peut être qu'un flux utilise toutes les capacités du routeur et ainsi mette les autres flux dans une situation de famine. Il peut donc être intéressant d'implémenter des algorithmes d'ordonnancement des paquets arrivant dans un routeur.

L'algorithme du Fair queueing proposé par John Nagle en 1987 peut être une première solution : on associe une structure de file à chaque flux de

données. Puis on défile et on transmet un paquet par file, file après file. Ainsi chaque flux est géré de manière équitable par le routeur.

Le problème de cet algorithme est qu'il est impossible de gérer des priorités différentes pour les flux. La solution est l'algorithme du Weighted fair queueing : une priorité (un poids) est affectée à chaque flux et on traite pour chaque file un nombre de paquets (ou un nombre d'octets) proportionnel à la priorité de la file.

## 5.2 La qualité de service dans l'IPv6 [1]

Deux champs de l'en-tête IPv6 peuvent être utilisés pour la qualité de service : le champ Traffic Class (RFC 2474) et le champ Flow label, selon le type de système. On peut distinguer deux types de système :

- Integrated Services : le but est de réserver les ressources de bout en bout d'une connexion
- Differentiated Services (DiffServ) : la réservation de ressource se fait routeur par routeur, pas de réservation de ressources de bout en bout

### 5.2.1 Systèmes de type Integrated Services : le champ Flow Label

Ici le but est de réserver les ressources de bout en bout, donc chaque routeur doit avoir des critères différents pour traiter chaque flux. Un routeur doit analyser chaque paquet pour savoir à quel flux il appartient. La même politique de QoS sera appliquée à tous les paquets d'un flux. L'entête IPv6 dispose pour cela du champ Flow Label.

Le champ Flow label(codé sur 20 bits) et le champ Adresse Source identifient de manière unique un groupe de paquets (un flux). Ainsi, tous les paquets du même flux sont traités d'une manière identique par les routeurs, et donc plus rapidement.

Le Flow Label est assigné par le noeud source du flux. Il peut y avoir plusieurs flux entre deux noeuds en parallèle. Le Flow Label doit être choisi au hasard dans l'intervalle [00001 ;FFFFF].

Lors d'une communication, dans un premier temps une phase de réservation de ressources est effectuée par le protocole RSVP (Ressource Reservation Protocol). Puis il y a transmission des paquets de données dans laquelle le Flow Label est utilisé pour classer le paquet transmis.

La transmission se déroule comme suit :

- Connexion établie de A vers B
- B envoie une requête RSVP vers A : tous les routeurs sur le chemin réservent les ressources demandées si disponibles
- A renvoie un acquittement RSVP vers B
- A transmet ses paquets de données à B

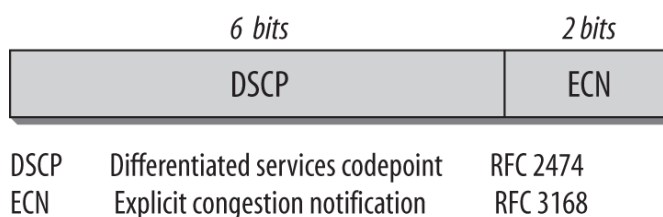
L'intérêt d'utiliser le champ Flow Label comme marqueur de flux est que les routeurs n'ont plus à décoder la charge du paquet pour déterminer le flux auquel appartient le paquet. Le traitement et le routage des paquets est ainsi accéléré pour une QoS particulière.

### 5.2.2 Systèmes de type DiffServ : le champ Traffic Class

Ici, le mécanisme est différent, le traitement se fait routeur par routeur (Per Hop Behavior : PHB). Ce type de système est plus simple à mettre en oeuvre et donc plus utilisé aujourd'hui.

Le fonctionnement est le suivant : la source marque le paquet en positionnant certains bits du marqueur DSCP (Differentiated Service Code Point). Les routeurs recevant ce paquet le traite comme prioritaire sur ceux n'ayant aucun marquage ou des valeurs de DSCP inférieures.

Le champ Traffic Class est aussi appelé DS field (Differentiated Service field) : il remplace le champ Type of Service de l'IPv4. Il est codé sur 8 bits. Il permet de définir une classe ou une priorité au paquet pour un traitement spécifique.



**Figure 5.1** – Structure du champ Traffic Class. [7]

Les 6 premiers bits définissent la valeur du DSCP qui détermine la gestion des priorités du trafic ou per-hop behavior (PHB) de chaque routeur.

Le PHB détermine comment les paquets seront transmis par le routeur. Le PHB par défaut est codé par la suite : 000000. Il correspond au best-effort forwarding behavior c.a.d. que le réseau transmettra ces paquets sans aucune priorité et utilisera les ressources disponibles pour les transmettre.

Groupe	Format du Codepoint	Politique d'affectation
1	xxxxx0	Usage standard
2	xxxx11	Usage expérimental ou local
3	xxxx01	Usage expérimental ou local ; Ou en complément du groupe 1

**Table 5.2** – Groupes de Codepoints.

3 groupes de Code Points sont définis : xxxxx0 pour des PHB standardisés, xxxx11 pour un usage expérimental ou local et xxxx01 pour un usage expérimental ou local ou en supplément du groupe xxxxx0 si il est entièrement utilisé.

Les 2 derniers bits du champ Traffic Class sont utilisés pour la notification explicite de congestion (ECN). Ces bits sont présents dans IPv4 et IPv6.

### **5.2.3 Le champ HopByHop**

Le champ HopByHop dans l'entête des options peut aussi être utilisé pour transporter un message d'alerte routeur (Multicast Listener Discovery message, RSVP message....). De cette manière, le paquet peut être traité plus rapidement car il est inutile d'analyser des entêtes d'autres protocoles de plus haut niveau par les routeurs intermédiaires pour détecter que c'est un paquet message.

### **5.2.4 Absence de NAT**

La Network Address Translation (NAT) n'existe plus en IPv6 car suffisamment d'adresses IP sont disponibles et donc la NAT n'est plus nécessaire. De cette manière, la QoS n'est plus arrêtée par la NAT et peut s'étendre sur des réseaux plus importants.

## Chapitre 6

# Expérimentation

La partie expérimentation de notre projet concerne principalement le test de la qualité de service (QoS). Néanmoins, nous avons dû apprendre à utiliser le routeur et à le configurer avant de pouvoir réaliser nos tests.

### 6.1 Connexion au routeur

Pour configurer le réseau nous avons dû nous connecter au routeur via l'utilitaire Putty par le port serial<->USB. De cette manière, on accède à l'interface de configuration Cisco en ligne de commande.

### 6.2 Configuration du routeur

#### 6.2.1 Configuration du vlan

Avant de connecter nos machines, il faut créer un vlan (Virtual Local Area Network) pour pouvoir utiliser IPv6. En effet, il est impossible d'activer directement l'IPv6 sur les interfaces FastEthernet 1 à 7 avec le modèle de routeur dont nous disposons.

On crée ici un vlan appelé "vlan 2" :

```
vlan database
vlan 2
```



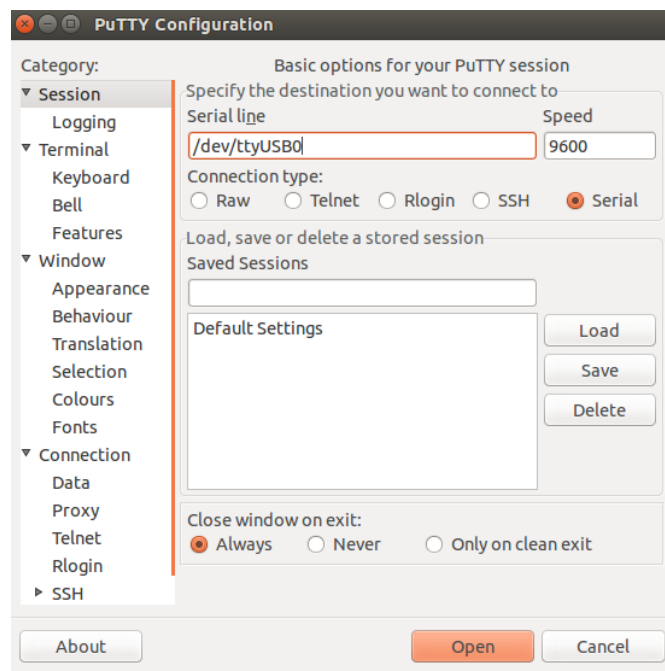


Figure 6.1 – Connexion de notre ordinateur au routeur

## 6.2.2 Configuration des interfaces

On configure ensuite les différentes interfaces du routeur que l'on va utiliser (fastEthernet 1 à 3 pour notre expérimentation).

Exemple de la configuration de l'interface 1 :

```
interface FastEthernet 1
switchport mode access
switchport access vlan 2
```

On précise que l'interface est prête à recevoir des informations avec "mode access" et qu'elle est liée au vlan 2.

## 6.2.3 Configuration IPv6

On passe en mode configuration pour l'interface du vlan 2 et on lui attribue une adresse IPv6.

On affecte à chaque machine du vlan 2 une adresse générée à partir du préfixe 2001:DB8:0:1::/64 et de l'adresse MAC de la machine (option eui-64).

Configuration :

```
interface vlan 2
ipv6 address 2001:DB8:0:1::/64 eui-64
```

```

ipv6 enable
no ip address
exit
ipv6 unicast-routing

```

À noter que l'on a désactivé l'adressage IPv4 avec la commande suivante afin d'être certain que nos services utilisent l'IPv6 :

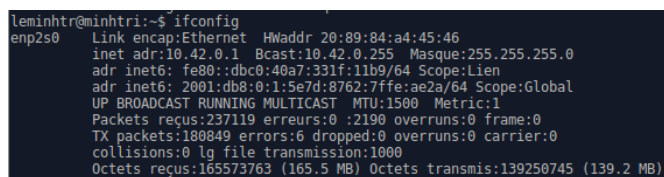
```
no ip address
```

#### 6.2.4 Test de la connexion et adressage IPv6

Une fois que la configuration IPv6 est effectuée, on connecte nos ordinateurs sur les interfaces puis on teste la connexion.

Tout d'abord on vérifie qu'une adresse IPv6 a été affectée à chaque machine connectée au routeur. Pour cela on utilise la commande suivante :

```
ifconfig
```



```

leminhtr@minhtri:~$ ifconfig
enp2s0: Link encap:Ethernet  HWaddr 20:89:84:a4:45:46
        inet addr:10.42.0.1  Bcast:10.42.0.255  Masque:255.255.255.0
        adr inet6: fe80::dbc0:40a7:331f:11b9/64 Scope:Lien
        adr inet6: 2001:db8:0:1:5e7d:8762:7ffe:ae2a/64 Scope:Global
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        Packets reçus:237119 erreurs:0 :2190 overruns:0 frame:0
        TX packets:180849 errors:6 dropped:0 overruns:0 carrier:0
        collisions:0 lg file transmission:1000
        Octets reçus:165573763 (165.5 MB) Octets transmis:139250745 (139.2 MB)

```

Figure 6.2 – Résultat de la commande ifconfig

On peut voir une adresse IPv6 link-local en FE80 :, qui a été automatiquement affectée à la machine par défaut. On a également l'adresse unicast global commençant par 2001 :DB8 :0 :1 (préfixe) suivi d'une adresse dérivée de l'adresse MAC de la machine.

L'adresse multicast FF02 ::1 est aussi disponible par défaut, toutes les machines du LAN écoutent à cette adresse. L'adresse FF02 ::2 désigne quant à elle tous les routeurs du LAN.

On teste également la connexion entre le routeur et l'ordinateur en effectuant un ping (requête ICMP echo).

La connexion fonctionne correctement car aucun paquet n'est perdu !

On peut également tester un ping sur l'adresse multicast FF02 ::1, en précisant l'interface de sortie, de cette manière dans un terminal :

```
ping6 -I eth0 FF02::1
```

Cela permet de connaître tous les clients connectés au routeur.

```

up
*Jan 1 01:24:42.147: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet
et1, changed state to up
Router#show ne
Router#nei
Router#
Router#ping ipv6 2010:ab8:0:1:82fa:5bff:fe15:1ac
% Unrecognized host or address, or protocol not running.

Router#ping ipv6 2010:ab8:0:1:82fa:5bff:fe15:1ac
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2010:AB8:0:1:82FA:5BFF:FE15:1AC, timeout is 2
seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/4 ms
Router#ping ipv6 2010:ab8:0:1:f017:d4c1:6e35:9d01
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2010:AB8:0:1:F017:D4C1:6E35:9D01, timeout is 2
seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/4 ms
Router#

```

Figure 6.3 – On ping la machine depuis le routeur

```

augustin@augustin-W840AU:~$ ip -6 ne
neigh netns
augustin@augustin-W840AU:~$ ip -6 neigh
2010:ab8:0:1::1 dev eth0 lladdr 00:1a:a1:55:62:ce router REACHABLE
fe80::21a:a1ff:fe55:62ce dev eth0 lladdr 00:1a:a1:55:62:ce router DELAY
augustin@augustin-W840AU:~$ ping6 2010:ab8:0:1::1
PING 2010:ab8:0:1::1(2010:ab8:0:1::1) 56 data bytes
64 bytes from 2010:ab8:0:1::1: icmp_seq=1 ttl=64 time=0.399 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=2 ttl=64 time=0.438 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=3 ttl=64 time=0.424 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=4 ttl=64 time=0.333 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=5 ttl=64 time=0.405 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=6 ttl=64 time=0.400 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=7 ttl=64 time=0.408 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=8 ttl=64 time=0.415 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=9 ttl=64 time=0.423 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=10 ttl=64 time=0.411 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=11 ttl=64 time=0.384 ms
64 bytes from 2010:ab8:0:1::1: icmp_seq=12 ttl=64 time=0.414 ms
^C
--- 2010:ab8:0:1::1 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 10998ms
rtt min/avg/max/mdev = 0.333/0.404/0.438/0.032 ms
augustin@augustin-W840AU:~$

```

Figure 6.4 – Ping depuis la machine vers le routeur

## 6.2.5 Sauvegarde des différentes configurations du routeur

### 6.2.5.1 Problème avec le chargement de la configuration de démarrage

Nous avons rencontré un problème avec le chargement de la startup-config lorsqu'on démarre notre routeur. En effet, le mode de paramétrage du routeur faisait qu'au démarrage, la configuration n'était pas chargée depuis startup config. Nous avons donc corrigé ce problème de la manière suivante.

Le paramètre de configuration registre est un code hexadécimal qui permet de changer le comportement du routeur au démarrage.[4] Notamment :

- comment le routeur démarre (ROMmon, NetBoot)
- avec quelles options (ignorer la configuration de démarrage, désactiver les messages de démarrage)

— la vitesse de la console (en bauds)

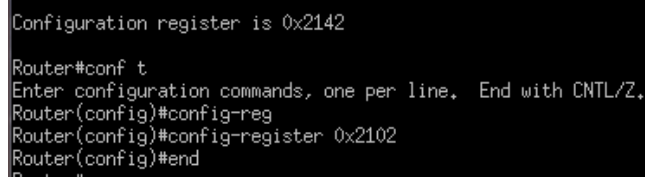
La commande :

```
show version
```

permet de connaître le code du paramètre de configuration registre (dernière ligne du résultat).

Le code de paramètre de configuration registre présent à l'origine était le 0x2142 qui configure le routeur pour ignorer la configuration de démarrage (startup-config). Nous avons redéfini le code de paramètre de configuration registre à sa valeur par défaut sortie d'usine : 0x2102, ce qui permet de charger la configuration de démarrage au démarrage.

```
show version
conf t
config-register 0x2102
```



```
Configuration register is 0x2142
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#config-reg
Router(config)#config-register 0x2102
Router(config)#end
Router#
```

Figure 6.5 – Modification du paramètre de configuration

### 6.2.5.2 Sauvegardes et changements de configurations

Nous avons sauvegardé différentes configurations du routeur (avec et sans QoS) dans des fichiers sur la carte compact flash du routeur, par exemple dans le fichier SR04QOS :

```
copy running-config SR04QOS
```

Pour changer de configuration (passer d'une configuration sans QoS à une configuration avec QoS par exemple), on copie le fichier de configuration de la carte dans la startup-config puis on recharge la configuration du routeur :

```
copy SR04QOS startup-config
reload
```

## 6.3 Mise en place des services

Pour tester la qualité de service, nous avons mis en place quelques services pour tester la connexion. Nous nous sommes concentrés sur les logiciels de voix

sur IP puis par la suite, nous avons fait du streaming vidéo.

Note : Les services ont été mis en place sur un ordinateur utilisant ArchLinux (gestion de service : systemctl). Les configurations des serveurs et les procédures d'installation ne sont pas présentées ici car hors-sujet.

### 6.3.1 Conférence audio

Nous avons installé et hébergé un serveur Murmur sur une de nos machines. Murmur permet d'héberger des salons de discussions (conversation vocale et écrite) du logiciel Mumble. Mumble est un logiciel permettant la voix sur IP, et permet donc de communiquer oralement à distance.

Il faut d'abord se connecter au serveur Murmur que nous avons créé. L'adresse est l'adresse IPv6 de l'ordinateur faisant tourner le serveur.

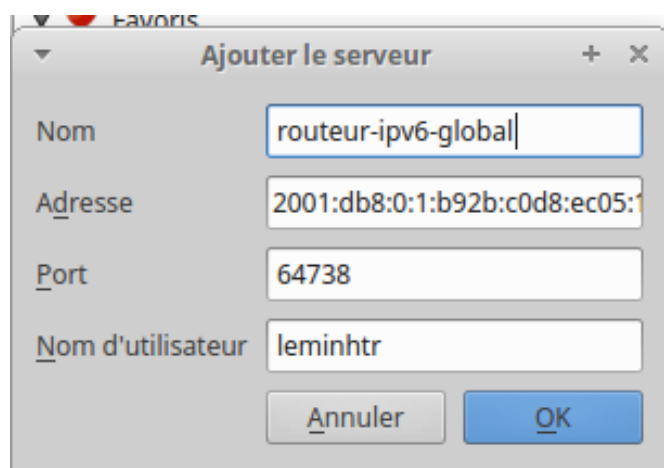


Figure 6.6 – Connexion au serveur Murmur dans Mumble

Il est également possible d'activer/désactiver la QoS sur Mumble, fonctionnalité utile pour nos tests.

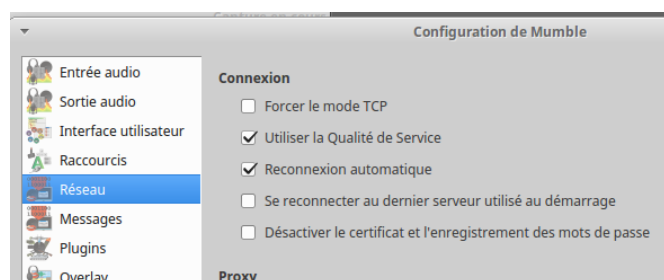


Figure 6.7 – Option QoS de Mumble

Enfin, nous avons réussi à discuter par message écrit et VoIP (une bouche est

rouge quand une personne parle par VoIP)

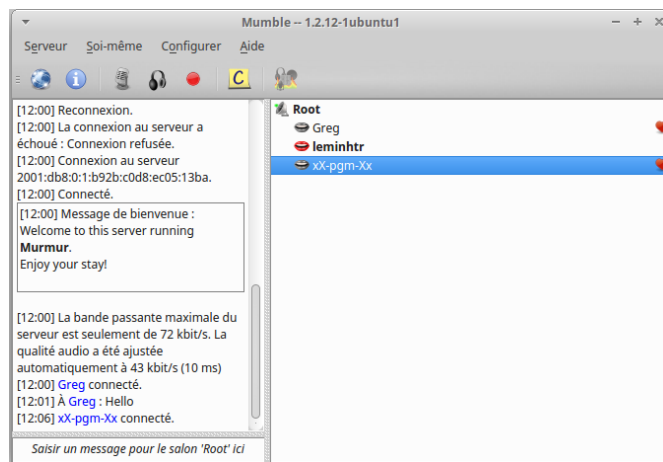


Figure 6.8 – Salon de discussion Mumble (3 personnes)

### 6.3.2 Transfert de fichier

Nous avons installé et hébergé un serveur SSH pour le transfert de fichier en SFTP (openSSH).

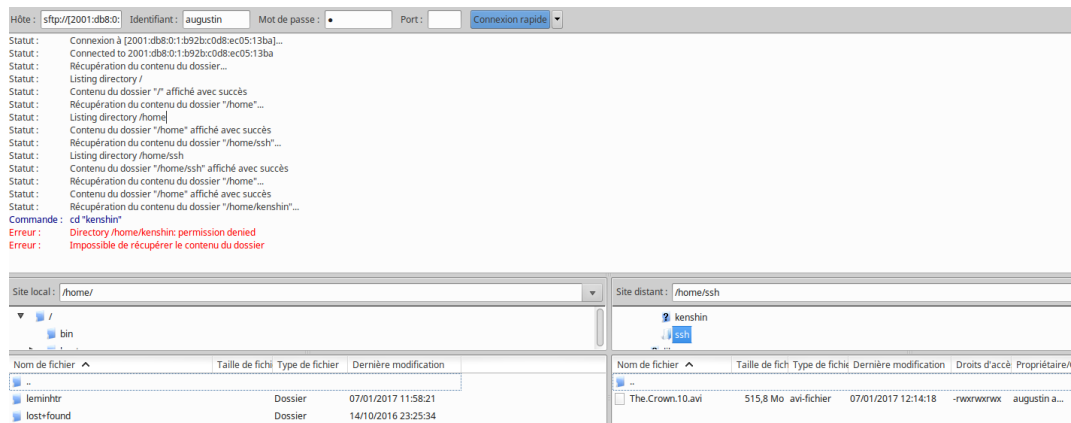


Figure 6.9 – Configuration et accès au serveur SSH

### 6.3.3 Téléphonie

Bien que nous ayons déjà pu tester la VoIP avec Mumble, nous voulions voir ce que cela donnait avec des téléphones. Nous avons donc emprunté deux téléphones utilisant la VoIP fournis par le département du GI.

Le téléphone possède deux méthodes d'alimentation possible : PoE (Power over Ethernet) ou par alimentation classique (Transformateur 220v -> 36V). Le POE permet de transmettre à un équipement à la fois des données, mais aussi son alimentation électrique via le câble Ethernet, deux des quatre paires du câble sont allouées à l'alimentation électrique. Cette technique permet d'éviter l'installation d'un double réseau (IP et électrique). Cependant, le routeur ne dispose pas de l'option facultative permettant le support de la PoE. L'alimentation s'est donc faite via transformateur.

La mise en place du téléphone fut assez longue : nous avons dû lire la documentation technique pour téléphone. La mise en place de 4 serveurs (TFTP, FTP, HTTP, SIP) ainsi qu'une configuration assez complète, notamment pour la gestion du protocole SIP. Nous avons appris par ce biais à mettre en place une architecture VOIP complète qui nous a permis de passer des appels entre tous nos différents appareils : téléphone portable (via Linphone), ordinateur (via Ekiga) et bien sûr, via le téléphone IP.

### 6.3.4 Vidéo à la demande, streaming

Nous avons implémenté un serveur HTTP afin de pouvoir faire du streaming vidéo.

```
vlc -vvv '/home/ssh/film.avi' --ipv6 --sout '#  
transcode{vcodec=h264,acodec=mpga,ab=128,channels=2,  
samplerate=44100}:http{mux=ffmpeg{mux=flv},dst=:8080/SR04}'  
--ttl 12 --sout-all
```

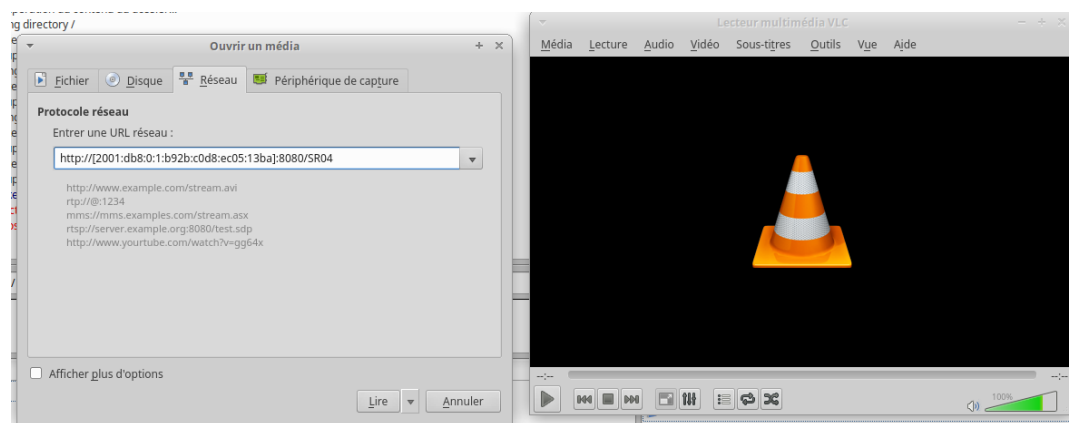


Figure 6.10 – Configuration de l'accès au streaming

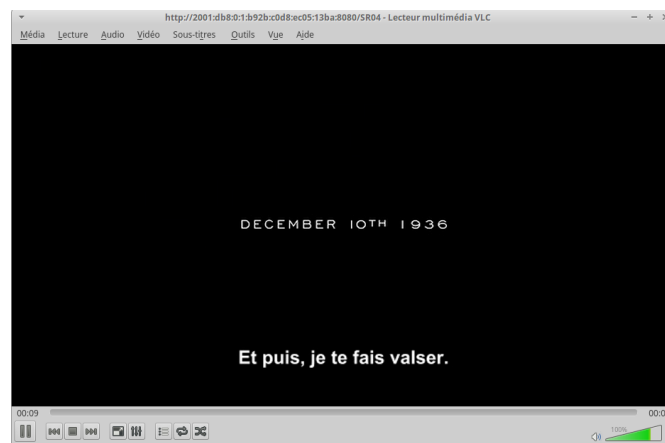


Figure 6.11 – Streaming d'une vidéo

Au cours de nos différents essais de streaming, nous avons également essayé le protocole RTP (Real-time Transport Protocol) et son homologue gérant la qualité de service : le protocole RTCP (RTP Control Protocol). RTCP sert à s'assurer de la qualité de service et permet de synchroniser les flux. Pour utiliser ces protocoles,



nous avons dû créer et utiliser un groupe multicast.

## 6.4 Mise en place de la qualité de service

Pour mettre en place notre qualité de service, nous nous sommes basés sur les services différenciés (DS - Differentiated Services), expliqués section 5.2.2. Ce type de QoS utilise le champ Traffic Class d'un paquet IPv6 afin de préciser le DSCP (Differentiated Services Code Point). Le DSCP est un code permettant de classer et de gérer les paquets d'un réseau suivant une certaine priorité. Ainsi, on peut demander au routeur d'accélérer au maximum le passage de certains paquets en précisant le code DSCP "EF" dans leur champ Traffic Class. Cette QoS est gérée au niveau de chaque routeur, suivant leur politique de traitement des paquets. Lorsqu'aucun code DSCP n'est entré, le routeur applique une politique par défaut : "best effort".

Ci-dessous se trouvent les instructions permettant d'implémenter la qualité de service.

### 6.4.1 Configuration d'une classe

Supposons que l'on veuille donner la priorité aux paquets échangés entre un serveur Http et un ordinateur, sur le port 80.

En utilisant la commande ci-dessous, on crée un groupe d'accès appelé 100. Ce groupe va référencer les paquets provenant du port 80 (eq 80) de n'importe quelle source (premier any) vers n'importe quelle destination (second any).

```
access-list 100 permit tcp any any eq 80
```

On crée ensuite une classe nommée "http". Une classe permet de représenter un ensemble de paquets sur lesquels on va appliquer une politique de traitement. La commande "match..." permet donc de spécifier les conditions d'appartenance à la classe. Ici, il faut que le paquet soit dans le groupe 100.

```
class-map http  
match access-group 100
```

### 6.4.2 Configuration d'une politique de traitement des paquets

Une fois que l'on connaît les paquets sur lesquels appliquer la QoS, il ne reste plus qu'à préciser la politique de traitement. Ici, on crée une politique nommée "acceshttp" qui s'appliquera à la classe "http". On précise ensuite la priorité que l'on donne aux paquets avec "set dscp".

```
policy-map acceshttp
```



Pendant la saturation du réseau, nous avons tenté de passer un appel VOIP sur Mumble. La latence était importante et la qualité mauvaise. Le besoin de la QoS se faisait ressentir.

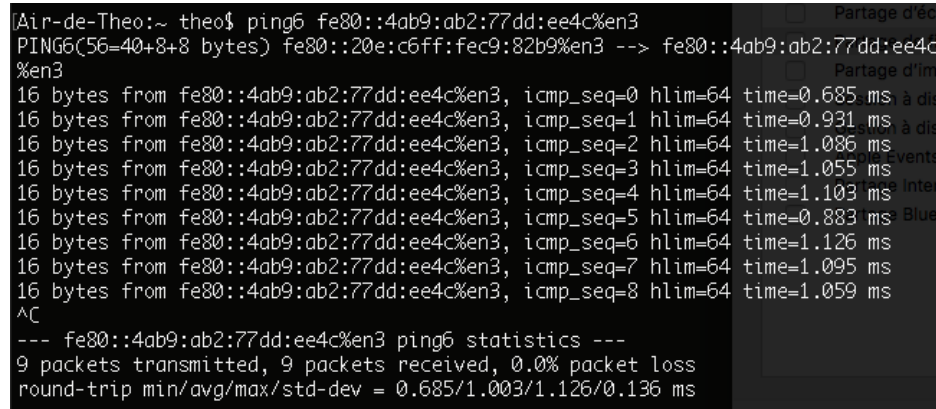
### 6.5.2 Test de la QoS par Ping

Pour prouver le bon fonctionnement de la QoS, nous avons trouvé une méthode se servant de l'utilitaire ping. Nous avons ensuite observé les paquets sur Wireshark.

La commande suivante permet d'envoyer un paquet en spécifiant le ToS (Type of Service) souhaité. Le ToS est un code comprenant le DSCP. On peut l'obtenir par la formule  $DSCP * 4 = TOS$ , ou par des tables de conversion comme celle-ci : [5].

```
ping6 -Q <ToS> <adresse IP>
ping6 -Q 0x68 fe80::4ab9:ab2:77dd:ee4c%en3
```

L'idée du test est donc d'activer la QoS au niveau du routeur et de charger le réseau en envoyant un fichier très lourd par exemple. On envoie ensuite un ping pour vérifier que la valeur DSCP entrée n'est pas perdue en cours de route. On peut voir cela avec Wireshark, qui repère le champ correspondant au DSCP et la priorité fixée. On peut également vérifier l'impact de la QoS entre un ping normal et un ping spécifiant le DSCP. On utilise la priorité EF, Expedited Forwarding, améliorant le temps d'acheminement du paquet.



```
Air-de-Theo:~ theo$ ping6 fe80::4ab9:ab2:77dd:ee4c%en3
PING6(56=40+8+8 bytes) fe80::20e:c6ff:fec9:82b9%en3 --> fe80::4ab9:ab2:77dd:ee4c%en3
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=0 hlim=64 time=0.685 ms
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=1 hlim=64 time=0.931 ms
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=2 hlim=64 time=1.086 ms
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=3 hlim=64 time=1.055 ms
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=4 hlim=64 time=1.103 ms
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=5 hlim=64 time=0.883 ms
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=6 hlim=64 time=1.126 ms
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=7 hlim=64 time=1.095 ms
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=8 hlim=64 time=1.059 ms
^C
--- fe80::4ab9:ab2:77dd:ee4c%en3 ping6 statistics ---
9 packets transmitted, 9 packets received, 0.0% packet loss
round-trip min/avg/max/std-dev = 0.685/1.003/1.126/0.136 ms
```

Figure 6.13 – Envoie d'un ping sans QoS

Petite remarque, la commande est ici ping6 -Z car elle a été réalisée sur Mac OS.

En moyenne le ping met 1.003 ms pour arriver sans QoS et 0.683 ms avec QoS.

Cela représente donc une amélioration de 68% ( $0.683/1.003$ ) du temps d'acheminement.

```
Air-de-Theo:~ theo$ ping6 -z 184 fe80::4ab9:ab2:77dd:ee4c%en3
PING6(56=40+8+8 bytes) fe80::20e:c6ff:fec9:82b9%en3 --> fe80::4ab9:ab2:77dd:ee4c%en3
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=0 hlim=64 time=0.398 ms tclass=184
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=1 hlim=64 time=0.570 ms tclass=184
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=2 hlim=64 time=0.747 ms tclass=184
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=3 hlim=64 time=0.756 ms tclass=184
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=4 hlim=64 time=0.716 ms tclass=184
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=5 hlim=64 time=0.776 ms tclass=184
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=6 hlim=64 time=0.641 ms tclass=184
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=7 hlim=64 time=0.814 ms tclass=184
16 bytes from fe80::4ab9:ab2:77dd:ee4c%en3, icmp_seq=8 hlim=64 time=0.729 ms tclass=184
^C
--- fe80::4ab9:ab2:77dd:ee4c%en3 ping6 statistics ---
9 packets transmitted, 9 packets received, 0.0% packet loss
round-trip min/avg/max/std-dev = 0.398/0.683/0.814/0.122 ms
```

Figure 6.14 – Envoi d'un ping avec QoS

On a choisi les codes DSCP correspondants au téléphone VoIP que nous avons afin de passer le Per-Hop Behavior des paquets de notre téléphone en AF, alors que tous les autres paquets étaient en default. La téléphonie VoIP devient donc prioritaire.

Pour le code DSCP : EF, le Per-Hop Behavior est EF (Expedited Forwarding). Le routeur met en priorité les ressources nécessaires pour réduire les délais, la gigue, les pertes par un service de bout en bout. Cela est très utile pour assurer le trafic temps-réel des applications telles que la VoIP ou le streaming vidéo. Il y a cependant une limite concernant le nombre de paquets marqué par un code DSCP à EF pour limiter la surcharge et les délais.

Nous avons effectué un ping avec EF (Les bits DSCP sont à 46 en décimal.).

```
Internet Protocol Version 6, Src: 2001:db8:0:1:81ee:916c:7a21:9b4b, Dst: 2001:db8:0:1:b92b:c0d8:ec05:13ba
0110 .... = Version: 6
.... 1011 1000 .... = Traffic class: 0xb8 (DSCP: EF PHB, ECN: Not-ECT)
.... 1011 10. .... = Differentiated Services Codepoint: Expedited Forwarding (46)
.... ..00 .... = Explicit Congestion Notification: Not ECN-Capable Transport (0)
.... ..0100 0111 0110 1100 0001 = Flow label: 0x476c1
Payload length: 64
Next header: ICMPv6 (58)
Hop limit: 64
Source: 2001:db8:0:1:81ee:916c:7a21:9b4b
Destination: 2001:db8:0:1:b92b:c0d8:ec05:13ba
[Source GeoIP: Unknown]
[Destination GeoIP: Unknown]
Internet Control Message Protocol v6
Type: Echo (ping) request (128)
```

Figure 6.15 – Expedited Forwarding PHB

Pour le code DSCP : AF31, le Per-Hop Behavior est AF (Assured Forwarding). Dans ce cas, l'émetteur est assuré de la réception tant que la congestion du réseau n'est pas trop significative. Une partie de la bande passante est garantie et les classes d'AF peuvent avoir accès à un supplément si possible. Il existe plusieurs classes d'AF (6.16). Elles se distinguent par des niveaux de priorité croissant (Classe) et la probabilité de perte (Faible, moyenne, haute).

Drop	Class 1	Class 2	Class 3	Class 4
Low	001010 AF11 DSCP 10	010010 AF21 DSCP 18	011010 AF31 DSCP 26	100010 AF41 DSCP 34
Medium	001100 AF12 DSCP 12	010100 AF 22 DSCP 20	011100 AF32 DSCP 28	100100 AF42 DSCP 36
High	001110 AF13 DSCP 14	010110 AF23 DSCP 22	011110 AF33 DSCP 30	100110 AF43 DSCP 38

Figure 6.16 – Classe d'AF

Nous avons ici effectué un ping avec AF31, cela correspond donc à une classe 3 avec une faible probabilité de perte.

```
Internet Protocol Version 6, Src: 2001:db8:0:1:81ee:916c:7a21:9b4b, Dst: 2001:db8:0:1:b92b:c0d8:ec05:13ba
0110 .... = Version: 6
.... 0110 1000 .... = Traffic class: 0x68 (DSCP: AF31, ECN: Not-ECT)
.... 0110 10.... = Differentiated Services Codepoint: Assured Forwarding 31 (26)
.... ..00 .... = Explicit Congestion Notification: Not ECN-Capable Transport (0)
.... 0100 0111 0110 1100 0001 = Flow label: 0x476c1
Payload length: 64
Next header: ICMPv6 (58)
Hop limit: 64
Source: 2001:db8:0:1:81ee:916c:7a21:9b4b
Destination: 2001:db8:0:1:b92b:c0d8:ec05:13ba
[Source GeoIP: Unknown]
[Destination GeoIP: Unknown]
Internet Control Message Protocol v6
Type: Echo (ping) request (128)
```

Figure 6.17 – Assured Forwarding PHB

Pour tous les autres codes : le Per-Hop behavior est default. Le comportement est alors *"Best Effort"*.

```
Internet Protocol Version 6, Src: 2001:db8:0:1:81ee:916c:7a21:9b4b, Dst: 2001:db8:0:1:b92b:c0d8:ec05:13ba
0110 .... = Version: 6
.... 0000 0000 .... = Traffic class: 0x00 (DSCP: CS0, ECN: Not-ECT)
.... 0000 00.... = Differentiated Services Codepoint: Default (0)
.... ..00 .... = Explicit Congestion Notification: Not ECN-Capable Transport (0)
.... 0100 0111 0110 1100 0001 = Flow label: 0x476c1
Payload length: 64
Next header: ICMPv6 (58)
Hop limit: 64
Source: 2001:db8:0:1:81ee:916c:7a21:9b4b
Destination: 2001:db8:0:1:b92b:c0d8:ec05:13ba
[Source GeoIP: Unknown]
[Destination GeoIP: Unknown]
Internet Control Message Protocol v6
Type: Echo (ping) request (128)
```

Figure 6.18 – Default PHB

### 6.5.3 Test de la QoS par l'essai

Une fois la QoS fonctionnelle, nous avons recommencé l'expérience de départ : Streaming d'une vidéo 1080p sur deux ordinateurs. Lorsque le réseau était saturé, nous avons tenté un appel VOIP. Cette fois-ci, c'est la vidéo qui était ralentie. L'appel ne présentait aucune latence. L'objectif de la QoS était atteint : c'est l'appel VOIP qui était prioritaire sur le streaming vidéo.

## Chapitre 7

# Conclusion

Ce projet expérimental dans le cadre de l'UV SR04 fut riche en apprentissage de nouvelles connaissances théoriques et pratiques pour l'ensemble de notre groupe de projet.

Nous avons pu revoir en détail la théorie du modèle OSI (Open Systems Interconnection) et le fonctionnement de sa couche réseau (couche 3). Nous avons également parlé des spécificités du routage et avons étudié en détail différents protocoles de la couche réseau : l'IPv6, l'ICMPv6. Nous avons su montrer quelles sont les nouveautés et avantages apportés par la norme IPv6 par rapport à la norme IPv4.

Nous avons ensuite présenté dans notre rapport les grands principes et mises en œuvre de la Qualité de Service (QoS) qui était le sujet central de notre étude.

Une fois l'étude théorique bien avancée, nous avons pu commencer à tester en pratique les principes étudiés précédemment en théorie. Il nous a fallu pour cela apprendre à configurer et manipuler les fonctionnalités de base d'un routeur Cisco. Puis nous avons mis en fonctionnement et testé le routage, l'adressage, l'autoconfiguration en IPv6 sur ce routeur.

Finalement, nous avons mis en place des configurations du routeur avec et sans QoS. Puis nous avons tenté de saturer le routeur en créant du trafic à l'aide de plusieurs machines et plusieurs applications (conférence audio, transfert de fichier, téléphonie, streaming vidéo) le tout de manière simultanée.

Ces tests expérimentaux, mettant en place la QoS pour les services de VoIP, nous ont permis d'apprendre à installer une QoS simple dans un réseau et à démontrer l'efficacité de la QoS dans le but de favoriser certaines applications dans une architecture réseau.

Nous tenons particulièrement à remercier M. Bouabdallah et l'équipe pédagogique de l'UV SR04 pour leur soutien, leur aide et leurs conseils apportés tout au long du projet.



# Bibliographie

- [1] Jean-Paul ARCHIER. *Ipv6 Principes et mise en oeuvre*. Editions ENI, 2012.
- [2] *Chapitre 1 : Algorithme de routage*. [http://malm.tuxfamily.org/doc/qr\\_chap1\\_algo.htm](http://malm.tuxfamily.org/doc/qr_chap1_algo.htm).
- [3] *Chapitre 3 Présentation d'IPv6*. <https://docs.oracle.com/cd/E19957-01/820-2982/6nei1phfv/index.html>.
- [4] *Config-register use - Cisco*. <http://www.cisco.com/c/en/us/support/docs/routers/10000-series-routers/50421-config-register-use.html>.
- [5] *Config-register use - Cisco*. <https://www.tucny.com/Home/dscp-tos>.
- [6] *Guide d'administration système : services IP*. <https://docs.oracle.com/cd/E19957-01/820-2982/6nei1phgi/index.html>.
- [7] Silvia HAGEN. *Ipv6 Essentials*. 3rd. ed. O'Reilly, 2014.
- [8] *Le monde des réseaux*. <http://www.gatoux.com/SECTION1/p10.php>.
- [9] *OSPF 5. Les états OSPF, la construction des adjacences*. <http://cisco.goffinet.org/s3/ospf5-states>.
- [10] *Reserved IPs*. [https://en.wikipedia.org/wiki/Reserved\\_IP\\_addresses](https://en.wikipedia.org/wiki/Reserved_IP_addresses).
- [11] *RFC4443 - ICMPv6*. <https://tools.ietf.org/html/rfc4443#section-2.2>.
- [12] *RIPng - Livre IPv6*. <http://livre.g6.asso.fr/index.php/RIPng>.
- [13] *Résumé sur le routage à vecteur de distance*. [http://cisco.goffinet.org/s2/vecteur\\_distance](http://cisco.goffinet.org/s2/vecteur_distance).
- [14] *Shichao's notes - Chapter 8*. <https://notes.shichao.io/tcpv1/ch8/#parameter-problem-icmpv4-type-12-icmpv6-type-4>.
- [15] Andrew S. TANENBAUM. *Computer Networks*. 4th. ed. Prentice Hall PTR, 2003.
- [16] *The IPv6 Portal : 6to4*. <http://www.ipv6tf.org/index.php?page=using/connectivity/6to4>.