Agenda

Agenda

Rappel historique sur l'Internet

- 1983 : Réseau de recherche d'enviton 100 super-ordinateurs
- 1992 : Démarrage de l'activité économique en ligne
  - Multiplication du nombre de serveurs
  - Croissance exponentielle des connexions
- 1993 : Epuisement des adresses IPv4 de classe B
  - Allocation systématique d'adresses de classe C
  - Augmentation des entrées dans les tables de routage
- Prédiction d'un effondrement du réseau pour 1998!
  - 1999 : Bob Metcalfe a dû manger son article Infoworld 1995 où il fit cette prédiction


Mesures d'urgence : Gestion des adresses

RFC 1517 - RFC 1520 (Septembre 1993)

- Retour demandé des préfixes alloués (RFC 1917)
- Ré-utilisation des adresses de class C
- CIDR (Classless Internet Domain Routing)
  - Allocation de taille variable ⇒moins de gaspillage d'adresses
  - Notation préfixe / taille préfixe
  - Aggrégation ⇒réduction de la table de routage
- Introduction du concept d'adresses privées (RFC 1918)

Mesures d'urgence : Adresses privées (RFC 1918 BCP : Best Current Practice)

- Plusieurs plages d'adresses privée définies
- Adresses pour usage interne, non routables sur Internet
- Passerelle nécessaire pour l'accès à l'Internet
  - Mécanisme NAT : Network Address Translation
  - RFC 1631, RFC 2663 and RFC 2993
- Pas de bout-en-bout entre Internet et réseau privé
  - Passerelle NAT en coupure
  - Connexions initiées en interne seules autorisées
  - Meilleure sécurité ?

Impact du NAT RFC 2993

1ère conséquence :

Le NAT casse le modèle bout-en-bout de l'Internet:

- Il est difficile de contacter directement un équipement derrière un NAT
- L'application derrière un NAT ne connait pas son adresse IP publique

2nde conséquence :

Le NAT contraint le fonctionnement des protocoles réseaux

- Le NAT modifie le contenu des paquets
- Il n'y a pas de comportement standardisé du NAT

3ième conséquence :

Le NAT crée un point de faiblesse (SPOF) dans le réseau :

- Passage à l'échelle couteux
- Redondance compliquée à mettre en œuvre

Taux d'utilisation des adresses IPv4

Mars 2019

- L'espace IPv4 est constitué de 256 blocs de 16.78 millions d'adresses[1]

Delegated to/status	Blocks	remaining
AfriNIC	4	0.3646
APNIC	50	0.2145
ARIN	39	0
LACNIC	10	0.0722
RIPE NCC	38	0.3008
LEGACY	80	
UNALLOCATED	35	

Source : http://www.potaroo.net/tools/ipv4/

- Nombre d'adresses IPv4 utilisables : 3706.65 M (221 /8 blocks)
- Nombre d'adresses IPv4 disponibles : 16 M (0.95 /8 blocks)

Agenda

IPv6 : une évolution d'IPv4

- Premier standard définissant IPv6 en 1995 (RFC 1883)
- Extension de la taille des adresses à 128 bits (ou 16 octets)
- Conservation des caractéristiques d'IP :
  - Adresses de taille fixe
  - Routage par paquet basé sur l'information de son en-tête
  - Communications de bout-en-bout
- En-tête simplifiée
  - En-tête de taille fixe
  - Pas de checksum
  - Fonctionnalités extensibles
- Correction de dysfonctionnements d'IPv4
  - Pas de fragmentation par les routeurs intermédiaires

⇒En terme de fonctionnalité, IPv4 et IPv6 sont au même niveau

Périmètre de la migration vers IPv6

Rappel: une adresse IP est utilisée

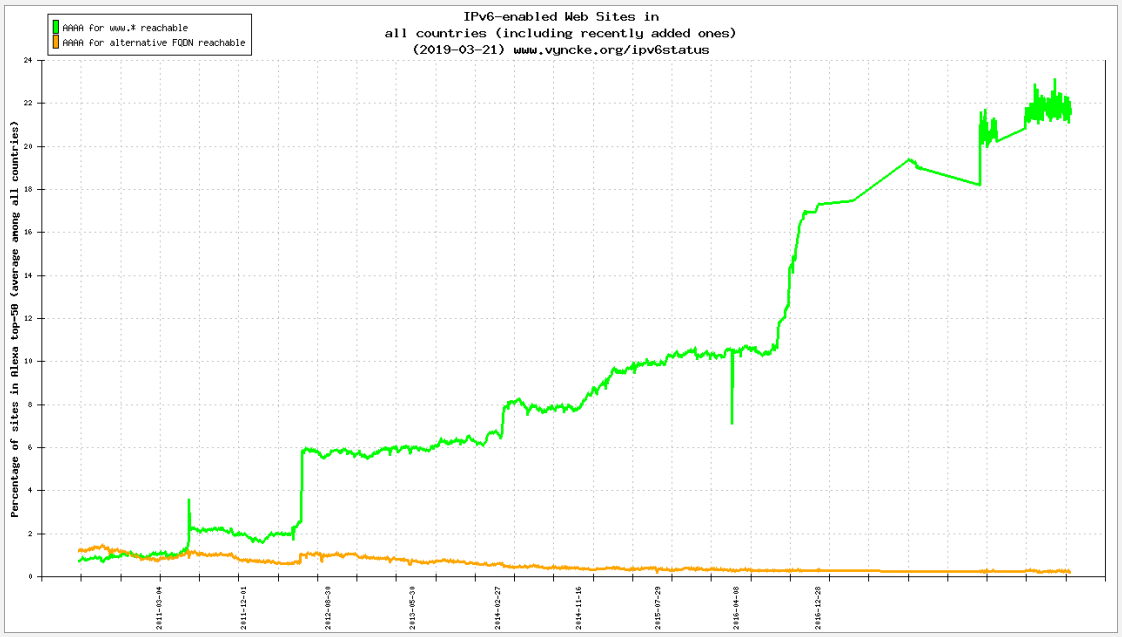
- par le réseau pour localiser un équipement
- par l'application pour identifier un pair

Une communication IPv6 de bout-en-bout nécessite :

- Le cœur du réseau Internet doit faire transiter IPv6
- Les réseaux d'accès doivent fournir une connectivité IPv6
- Les équipements terminaux doivent comporter une pile IPv6
- Les applications doivent être compatibles IPv6

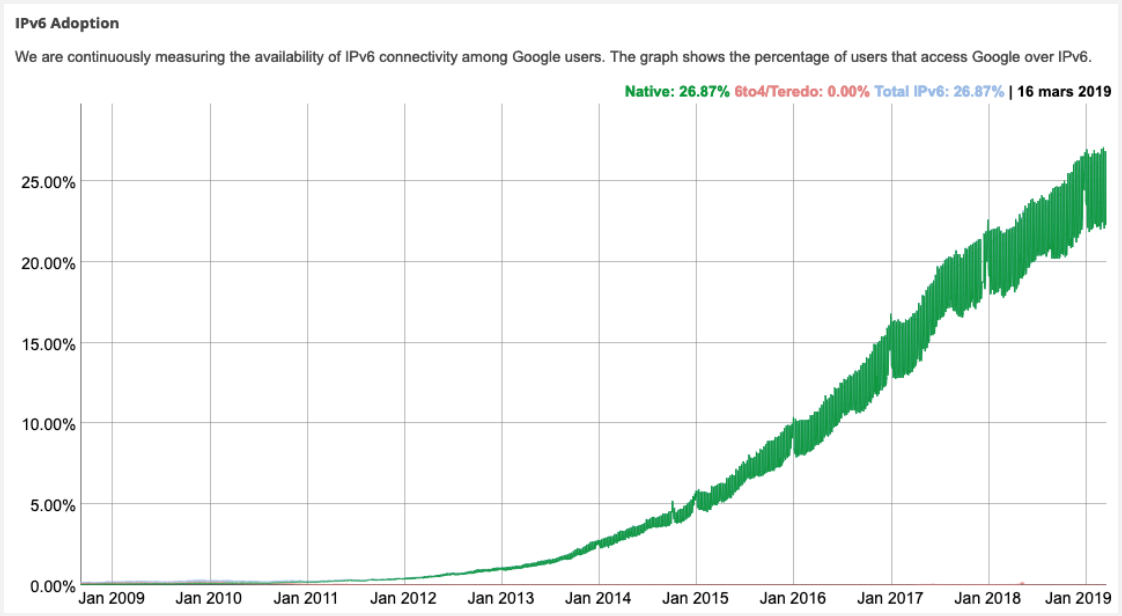
⇒La migration s'impose donc à toutes les niveaux !

Progression de la migration vers IPv6 (services)



Source :https://www.vyncke.org/ipv6status/

Progression de la migration vers IPv6 (utilisateurs)



Source :http://www.google.com/ipv6/statistics.html

Agenda

Défi : Soutenir le développement de l'Internet

L'espace d'adressage est suffisament large pour le futur du réseau

- 2^{128} est un TRÈS grand nombre
- Evolution de l'adressage envisageable (renumérotation, nouveaux schémas d'allocation d'adresses, ...)

Les nouveaux réseaux seront IPv6-only

- Les adressses IPv4 résiduelles sont insuffisantes et coûteuses
- IPv6 doit être déployé et testé à plusieurs niveaux

IPv6 est une mise à jour majeure de l'Internet !

Défi : Catalyser les nouveaux usages du réseau

IPv6 rétablit le principe du bout en bout

- Le réseau doit rester transparent pour les applications
- *Smart ends and dump pipes*

IPv6 optimise l'accès au réseau

- En-tête simple, orientée performance
- configuration réseau automatique Plug-and-play

IPv6 va accélérer l'adoption de nouveaux usages

- Internet of Things, Smart Cities, Ambient Computing,
- Network-as-a-Service, Devops,
- et bien d'autres à venir ...

Défi : Sensibiliser à la nécessité d'IPv6

IPv6 est une mise à jour globale (full-stack) :

- Réseaux
- Équipements
- Systèmes d'exploitations
- Services / Applications
- Humains ! (utilisateurs, services support, ...)

Convaincre de la nécessité du changement

- Combattre le syndrôme "Trop occupé pour évoluer"
- Identifier les freins au changement

Sensibiliser sur la conduite de la migration vers IPv6

- Définir un projet d'évolution de l'infrastructure
- Elargir le périmètre du projet à l'ensemble du système d'information
- Obtenir le soutien hierarchique pour ce projet

Challenge : Préparer les générations futures

Le déploiement d'IPv6 sera une tâche pour plusieurs années . . .
. . . les nouveaux diplômés doivent y être préparés dès maintenant !

- L'enseignement IT doit aborder IPv6
- Chaque université scientifique doit offrir un lab IPv6
- IPv6 doit être enseigné comme la norme du réseau, IPv4 comme un héritage

L'enseignement d'IPv6 est un facteur clé du succès du déploiement

Agenda

Association G6

Association à but non lucratif, crée en 1995
Missions :

- Sensibiliser à IPv6 dans l'espace francophone
- Inciter au partage d'expérience entre acteurs du déploiement
- Former à IPv6
  - Tutoriaux et formations (Opérateurs, administrateurs, ...)
  - Livre en ligne "IPv6, Théorie et pratique" : `http://livre.g6.asso.fr/`

Le premier MOOC sur IPv6 !

Le G6 et l'Institut Mines-Telecom ont ouvert le premier M(assive) O(pen) O(line) C(ourse) sur la technologie IPv6.

- Disponible sur France Université Numérique (`http://fun-mooc.fr`)
- Cours basé sur des vidéos, un support écrit, des exercices en ligne
- Forum de discussion entre les enseignants et les apprenants

4 sessions depuis 2015

- 20k inscrits
- 1000 fils de discussion dans les forums
- 1200 attestations délivrées

Nouvelle version en 2019 !

IPv6 Forum Certification

This course is certified by the IPv6 Forum with Gold Level
`http://www.ipv6forum.com/ipv6_education/`

Programme de la formation

5 jours, chacun sur un sujet précis :

- Jour 1: L'adressage en IPv6
- Jour 2: Le protocole IPv6
- Jour 3: La gestion d'un réseau IPv6
- Jour 4: L'intégration d'IPv6 dans l'Internet
- Jour 5: Les applications compatibles IPv6

Des travaux pratiques virtuels (jours 2,3,4)

- Réseau IPv6 émulé sous GNS3
- Manipulation et Configuration proche du réel

Un quiz de validation des connaissances en fin de formation

Agenda

Agenda

1. Introduction

2. Cours 1 : Adressage IPv6
   - Qu'est ce qu'une adresse IP ?
   - Syntaxe d'une adresse IPv6
   - Adresses IPv6 unicast
   - Adresses IPv6 multicast
   - Mise en œuvre des adresses IPv6

3. Cours 2 : Protocole IPv6

4. Cours 3 : Gestion d'un réseau IPv6

5. Cours 4 : Interopérabilité IPv4/IPv6

6. Cours 5 : Applications IPv6

Fonctions d'une adresse IP

Localiser un équipement dans le réseau

- Une adresse est un nombre appartenant à un espace d'adressage
- L'espace d'adressage est structuré selon la topologie du réseau
- Cette structuration sert au routage des paquets
- Un équipement change d'adresse avec sa localisation dans le réseau

Identifier un équipement lors d'une communication

- Les adresses IP source et destination identifient une communication
- Les adresses IP sont utilisés dans les politiques de sécurité réseau
- Des adresses IP peuvent aussi identifier des utilisateurs ou des services

Comparaison adresses IPv4 et IPv6

Points communs

- Adresses de taille fixe
- Délégation selon le principe CIDR
- Notation `prefixe / longueur`

Différences

- Taille (IPv4: 32bits, IPv6: 128bits)
- Notation (IPv4: décimale, IPv6: hexa)
- Nb d'adresses par interface (IPv4: 1, IPv6: plusieurs)
- Structuration de l'espace d'adressage

IPv6 suffisant pour le futur ?

Taille de l'espace d'adressage IPv6 :

- 2^{128} = Environ $3.4x10^{38}$ adresses
- $6.67x10^{23}$ adresses par m^2 terrestre
- $5x10^{28}$ adresses par habitant sur Terre

Rappel pour IPv4 :

- $4.3x10^{9}$ adresses (8 adresses par Km^2)
- 6 adresses par résident US, 1 pour l'Europe, 0.01 pour la Chine et 0.001 pour l'Inde

Date prévue pour l'épuisement des adresses IPv6 : an 9 000 000[2]

[2] https://samsclass.info/ipv6/exhaustion-2016.htm

Agenda

La notation des adresses IPv6 est déroutante (au début)

Par exemple, voici plusieurs notations d'adresses valides :

```
F2C:544:9E::2:EF8D:6B7 F692::  A:1455::A:6E0 D:63:D::4:3A:55F B33:C::F2 7:5059:3D:C0::
9D::9BAC:B8CA:893F:80 1E:DE2:4C83::4E:39:F35:C875 2::  A:FDE3:76:B4F:D9D::  D6::  369F:9:F8:DBF::2
DD4:B45:1:C42F:BE6:75::  9D7B:7184:EF::3FB:BF1A:D80 FE9::B:3 EC:DB4:B:F:F11::E9:090 83:B9:08:B5:F:3F:AF:B84
E::35B:8572:7A3:FB2 99:F:9:8B76::BC9 D64:07:F394::BDB:DF40:08EE:A79E AC:23:5D:78::233:84:8
F0D:F::F4EB:0F:5C7 E71:F577:ED:E:9DE8::  B::3 1D3F:A0AA::  70:8EA1::8:D5:81:2:F302 26::8880:7 93::  F::9:0
E:2:0:266B::  763E:C:2E:1EB:F6:F4:14:16 E6:6:F4:B6:A888:979E:D78:09 9:754:5:90:0A78:A1A3:1:7 2:8::
97B:C4::C36 A40:7:5:7E8F:0:32EC:9A:D0 8A52::575 D::4CB4:E:2BF:5485:8CE 07:5::41 6B::A9:C
94FF:7B8::D9:51:26F 2::E:AE:ED:81 8241::  5F97::  AD5B:259C:7DB8:24:58:552A::  94:4:9FD:4:87E5::
5A8:2FF:1::CC EA:8904:7C::  7C::D6B7:A7:B0:8B DC:6C::34:89 6C:1::5 7B3:6780:4:B1::E586
412:2:5E1:6DE5:5E3A:553:3::  7F0::  B39::1:B77:DB 9D3:1F1:4B:3:B4E6:7681:09:D4A8 61:520::E0
1:28E9:0:095:DF:F2::  1B61:4::1DE:50A 34BC:99::E9:9EFB E:EF::  BDC:672A:F4C8:A1::4:7:9CB7
C697:56AD:40:8:0::62
```

Ne vous inquiétez pas

Les adresses ne sont pas des nombres aléatoires, elles sont souvent faciles à mémoriser, du moins en partie

Notation des adresses IPv6 RFC 5952

Règles fondamentales d'écriture :

- 8 mots de 16 bits separés par ":"
- chacun des mots est représenté en écriture héxadécimale
- le symbole "::" représente plusieurs mots de 16 bits de valeur 0

Une même adresse peut être notée de plusieurs manières :
`2001:0db8:0000:009f:0000:0000:0000:000a`
`2001:db8:0:9f:0:0:0:a`
`2001:DB8:0:9F:0:0:0:A`
`2001:db8:0:9f::a`

La **notation canonique** (RFC 5952) définit la bonne pratique de notation.
Cependant : *Be strict with what you give, be permissive with what you accept*

Forme canonique IPv6 RFC 5952

2001:0DB8:000:009F:0000:0000:0000:000A

Règles de notation :

1. Ecrire les lettres en minuscules.
2. Enlever les 0 non-significatifs à gauche de chaque mot.
3. Une série de plusieurs mots de valeur 0 est abrégé par "::"
4. Le symbole "::" ne doit être utilisé qu'une seule fois
5. La plus longue séquence de mots à 0 est abrégée

Forme canonique IPv6 RFC 5952

2001:0db8:0000:009F:0000:0000:0000:000a

Règles de notation :

1. Ecrire les lettres en minuscules.
2. Enlever les 0 non-significatifs à gauche de chaque mot.
3. Une série de plusieurs mots de valeur 0 est abrégé par "::"
4. Le symbole "::" ne doit être utilisé qu'une seule fois
5. La plus longue séquence de mots à 0 est abrégée

Forme canonique IPv6 RFC 5952

2001:db8:0:9f:0:0:0:a

Règles de notation :

1. Ecrire les lettres en minuscules.
2. Enlever les 0 non-significatifs à gauche de chaque mot.
3. Une série de plusieurs mots de valeur 0 est abrégé par "::"
4. Le symbole "::" ne doit être utilisé qu'une seule fois
5. La plus longue séquence de mots à 0 est abrégée

Forme canonique IPv6 RFC 5952

2001:db8:0:9f::a

Règles de notation :

1. Ecrire les lettres en minuscules.
2. Enlever les 0 non-significatifs à gauche de chaque mot.
3. Une série de plusieurs mots de valeur 0 est abrégé par `"::"`
4. Le symbole `"::"` ne doit être utilisé qu'une seule fois
5. La plus longue séquence de mots à 0 est abrégée

Préfixe d'adresse

- Identification d'adresses contiguës (équivalent au *netmask* IPv4)
- Notation CIDR : combinaison de l'adresse avec la longueur du préfixe

ipv6-address/prefix-length

- *prefix-length*
  - Nombre de bits à gauche communs à toutes les adresses contiguës
- Par exemple, considérons 60 bits communs `2001:0db8:0000:d0d0`:
  - Notation du préfixe : `2001:db8:0:d0d0::/60`
  - Première adresse dans le préfixe : `2001:db8:0:d0d0::0`
  - Dernière adresse : `2001:db8:0:d0df:ffff:ffff:ffff:ffff`
  - Une adresse sur une interface : `2001:db8:0:d0d0:1e1a:deca:dead:face/60`

Attention:

`2001:db8:3::/40` correspond à `2001:db8:0003::/40` et non `2001:db8:0300::/40`

Préfixe d'adresse

- Identification d'adresses contiguës (équivalent au *netmask* IPv4)
- Notation CIDR : combinaison de l'adresse avec la longueur du préfixe

ipv6-address/prefix-length

- *prefix-length*
  - Nombre de bits à gauche communs à toutes les adresses contiguës
- Par exemple, considérons 60 bits communs `2001:0db8:0000:d0d0`:
  - Notation du préfixe : `2001:db8:0:d0d0::/60`
  - Première adresse dans le préfixe : `2001:db8:0:d0d0::0`
  - Dernière adresse : `2001:db8:0:d0df:ffff:ffff:ffff:ffff`
  - Une adresse sur une interface : `2001:db8:0:d0d0:1e1a:deca:dead:face/60`

Attention:

`2001:db8:3::/40` correspond à `2001:db8:0003::/40` et non `2001:db8:0300::/40`

Schéma d'adressage — RFC 4291

- Il définit les différents types d'adresses IPv6 :
  - loopback (`::1`)
  - lien-local (`fe80::/10`)
  - global unicast (`2000::/3`)
  - local unicast (`fd00::/8`)
  - multicast (`ff00::/8`)
- Une interface peut avoir plusieurs adresses IPv6
  - au moins une adresse lien-local
  - une adresse globale unicast si connectée à un réseau IPv6

Note:

Il n'y a pas de diffusion globale (*broadcast*) en IPv6, ces fonctions ont été remplacées par le multicast

Autres adresses spéciales

- loopback
  0:0:0:0:0:0:0:1⇒::1
- unspecified
  0:0:0:0:0:0:0:0 ⇒::
  - Indique une adresse vide
  - Utilisée pour noter la route par défaut (::/0)
  - Ne doit jamais être utilisée comme adresse destination

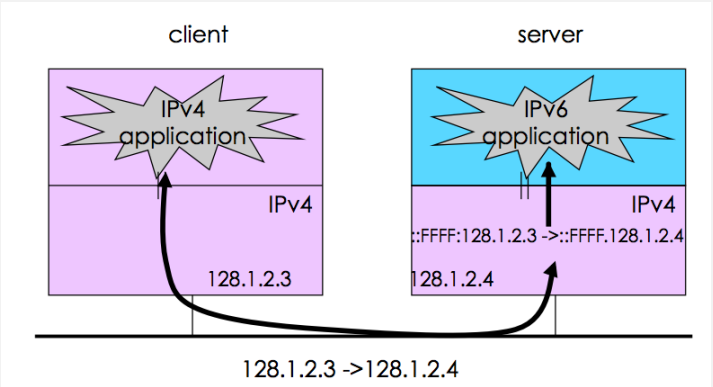
Pour plus de détails

"Overview of IPv6" - Cisco:
`http://www.cisco.com/en/US/docs/interfaces_modules/services_modules/ace/vA5_1_0/configuration/rtg_brdg/guide/ipv6.html`

Adresses IPv6 IPv4-mappée RFC 4038

```
|               80 bits               | 16 |      32 bits        |
+-------------------------------------+----+---------------------+
|0000.............................0000|FFFF|    IPv4 address     |
+-------------------------------------+----+---------------------+
```

- Notation : ::ffff:192.0.2.1
- Utilisées pour représenter une adresse IPv4 dans l'espace IPv6
- Permet d'assurer une compatibilité ascendante pour les applications
  - Connexions IPv4 vues comme des connexions IPv6
  - Manipulations des adresses indépendantes du protocole



Utilisation de l'espace d'adressage IPv6

```
0000::/8 Reserved by IETF [RFC4291]
0100::/8 Reserved by IETF [RFC4291]
0200::/7 Reserved by IETF [RFC4048]
0400::/6 Reserved by IETF [RFC4291]
0800::/5 Reserved by IETF [RFC4291]
1000::/4 Reserved by IETF [RFC4291]
2000::/3 Global Unicast [RFC4291]
4000::/3 Reserved by IETF [RFC4291]
6000::/3 Reserved by IETF [RFC4291]
8000::/3 Reserved by IETF [RFC4291]
a000::/3 Reserved by IETF [RFC4291]
c000::/3 Reserved by IETF [RFC4291]
e000::/4 Reserved by IETF [RFC4291]
f000::/5 Reserved by IETF [RFC4291]
F800::/6 Reserved by IETF [RFC4291]
fc00::/7 Unique Local Unicast [RFC4193]
fe00::/9 Reserved by IETF [RFC4291]
fe80::/10 Link Local Unicast [RFC4291]
fec0::/10 Reserved by IETF [RFC3879]
ff00::/8 Multicast [RFC4291]
```

`http://www.iana.org/assignments/ipv6-address-space`

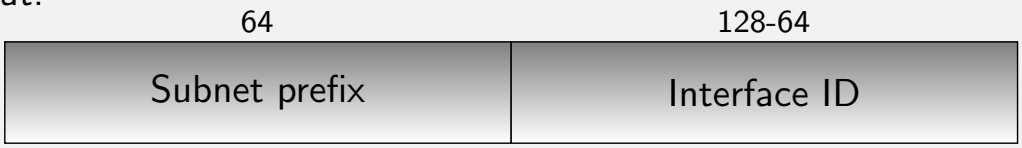
Agenda

Adresses Unicast

Une adresse unicast désigne une et une seule interface connectée au réseau

- Structurée en 2 parties de 64 bits chacune :
  - Identifiant de réseau
  - Identifiant de l'interface

Format:



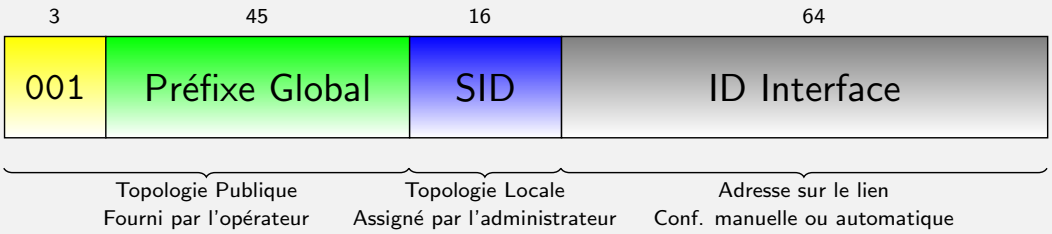
Plusieurs types d'adresses unicast :

- lien-local : désigne une interface sur le même lien
- local unicast : désigne une interface dans un même réseau *privé*
- global unicast : désigne une interface dans l'Internet IPv6 global

Adresses Globales Unicast RFC 3587

Préfixe commun à l'ensemble des adresses globales : `2000::/3`

Format:



- Préfixe Global : structuré selon la hiérarchie RIR/LIR
- SID: Subnet ID : définie par le plan d'adressage global
- ID Interface : doit être unique sur le lien

Schéma d'assignation des préfixes IPv6 globaux

Schéma CIDR défini pour IPv4 et repris pour IPv6

- IANA : délègue des blocs d'IP aux RIRs
- 5 RIRs : délèguent des blocs d'IP aux LIRs
  - APNIC (Asia Pacific Network Information Centre)
  - ARIN (American Registry for Internet Numbers)
  - LACNIC (Regional Latin-American and Caribbean IP Address Registry)
  - RIPE NCC (Réseaux IP Européens - Network Coordination Center)
  - AfriNIC (Africa)
- LIRs (ISPs, hébergeurs) : assignent adresses ou préfixes à leurs clients

Exemple d'assignation de préfixes pour Renater

RIPE-NCC

2001:660::/32

POP

2001:660:7300::/40

Site

2001:660:7301::/48

Images/G6.png

2001:660:7300::/40

Images/renater.png

Valeurs pour le SID : Subnet IDentifier

Le SID correspond à la partie du préfixe laissée pour la topologie locale du site

- Généralement 16 bit = 65 535 sous-réseaux
  - Suffisant pour la plupart des organisations
- Pour les clients domestiques, l'ISP peut déléguer un /56 ou /60 (256 à 16 sous-réseaux)

Les valeurs du SID sont définies par le plan d'adressage local :

- de manière séquentielle : 1, 2, ...
- selon le numéro de VLAN
- selon des plans plus complexes pour permettre l'agrégation

Exemple du plan d'adressage pour une université

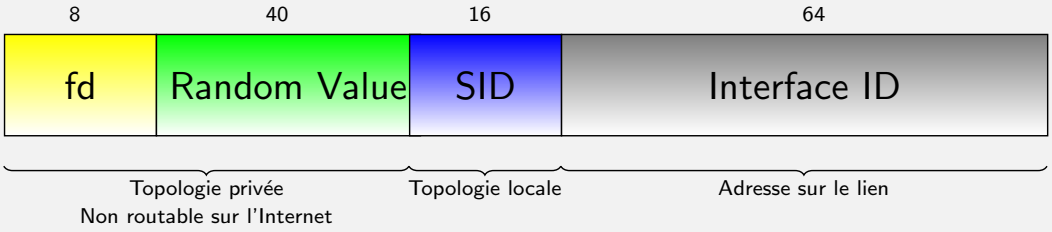
4bits : Communités	**8bits**	**4bits**
0 : Infrastructure	*Adresses spécifiques*	
1 : Tests	*Adresses spécifiques*	
6 : Point6	*Gérer par Point6*	
8 : Wifi invités	*Adresses spécifiques*	
A : Employés	Entités géographiques	sous réseaux
E : Students	Entités géographiques	sous réseaux
F : Autre (Start up, etc.)	*Adresses spécifiques*	

- Les règles de filtrage utilisent les préfixes avec les 4 premiers bits en commun
- Les règles de routage se basent sur les préfixes géographiques
- Compromis :
  - Une règle de filtrage pour une même communité sur différents sites
  - Plusieurs règles de routage pour un même site (une par communauté)

Adresses Unicast-Locales (ULA) RFC 4193

- Equivalentes aux adresses privées en IPv4
- Préfixe : `fd::/8` + 40 bits privés + SID
- Préfixes privés non-routable sur l'Internet
- 40 bits privés à définir aléatoirement:
  - pour éviter un problème si 2 réseaux ULA doivent fusionner
  - ou d'autres ambiguités si utilisation de VPN

Structure d'une adresse ULA :

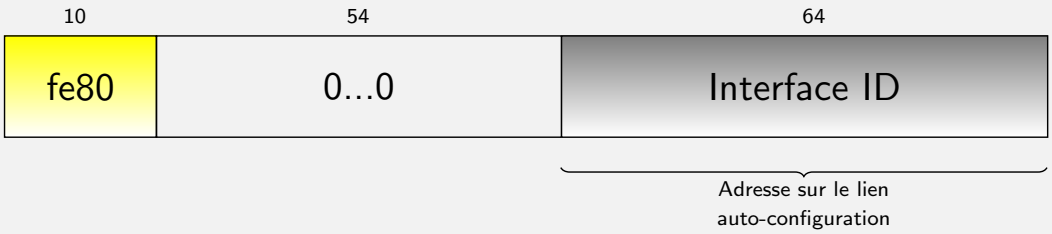


Pour créer votre préfixe ULA : `http://www.sixxs.net/tools/grh/ula/`. Ce service est arrêté depuis le 6 Juin 2017, la base de données existante est proposée en lecture seule.

Adresses Lien-Locales

- Générées et assignées automatiquement au démarrage d'une interface
- Adresses valables uniquement sur le lien et non routable
- Préfixe commun `fe80::/64`

Format:



- Si adresse LL destinatrice, l'interface de sortie n'est pas définie
- Elle est spécifiée en suffixant à l'adresse la notation `%iface`
- Exemple dans une table de routage :

```
Destination                    Gateway                 Flags      Netif
default                fe80::213:c4ff:fe69:5f49%en0     UGSc       en0
```

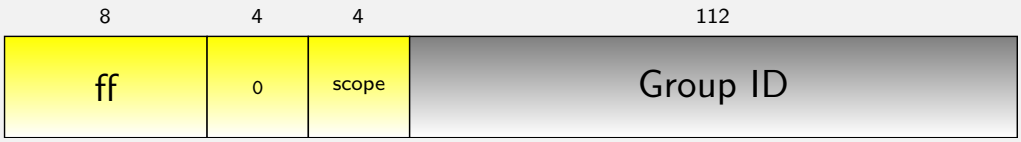
Agenda

Adresses IPv6 Multicast

Une adresse multicast désigne un groupe d'interfaces
Format:

- T (Transient) 0: addresse standardisée - 1: addresse temporaire
- P (Prefix) 1 : définie depuis un préfixe réseau (T=1) RFC 3306
- R : Contient un point de rendez-vous (RP) RFC 3956
- Scope :
  - 1 - interface-local
  - 2 - link-local
  - 3 - reserved
  - 4 - admin-local
  - 5 - site-local
  - 8 - organisation-local
  - e - global
  - f - reserved

Adresses multicast standardisées




```
ff02:0:0:0:0:0:0:1 All Nodes Address (link-local scope)
ff02:0:0:0:0:0:0:2 All Routers Address
ff02:0:0:0:0:0:0:5 OSPFIGP
ff02:0:0:0:0:0:0:6 OSPFIGP Designated Routers
ff02:0:0:0:0:0:0:9 RIP Routers
ff02:0:0:0:0:0:0:fb mDNSv6
ff02:0:0:0:0:0:1:2 All-dhcp-agents
ff02:0:0:0:0:1:ffxx:xxxx Solicited-Node Address
ff05:0:0:0:0:0:1:3 All-dhcp-servers (site-local scope)
```

http://www.iana.org/assignments/ipv6-multicast-addresses

Agenda

Identifiant d'interface (IID)

Doit être unique sur le lien

L'identifiant d'interface peut être choisi de différentes manières :

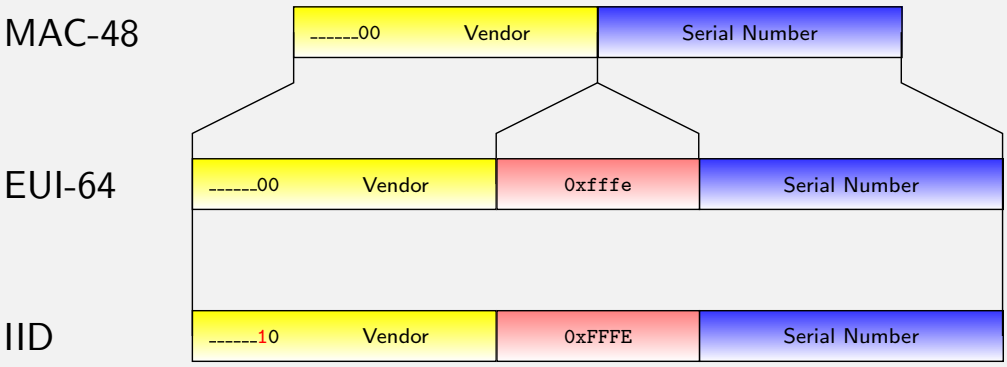
- Dérivé de l'adresse de niveau 2 (i.e. MAC address) :
  - pour les adresses Lien-Local
  - pour les adresses Unicast Global auto-configurées
- Assigné manuellement :
  - pour garder une adresse stable (serveur)
  - pour obtenir une adresse facilement mémorisable
- Valeur aléatoire :
  - Valeur changée régulièrement (chaque heure, à chaque reboot...)
  - Préserve l'anonymat des connexions (Voir RFC 4941)
  - Aujourd'hui solution préférée aux IID dérivés des adresses MAC
- Allouée par DHCPv6
  - Politique définie par l'administrateur

Exemple sur un système Unix

```
%ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
        inet 127.0.0.1 netmask 0xff000000
en1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        inet6 fe80::216:cbff:febe:16b3%en1 prefixlen 64 scopeid 0x5
        inet 192.168.2.5 netmask 0xffffff00 broadcast 192.168.2.255
        inet6 2001:660:7307:6031:216:cbff:febe:16b3 prefixlen 64
        autoconf

        ether 00:16:cb:be:16:b3
        media: autoselect status: active
        supported media: autoselect
```

Construction d'un IID depuis l'adresse MAC RFC 2464

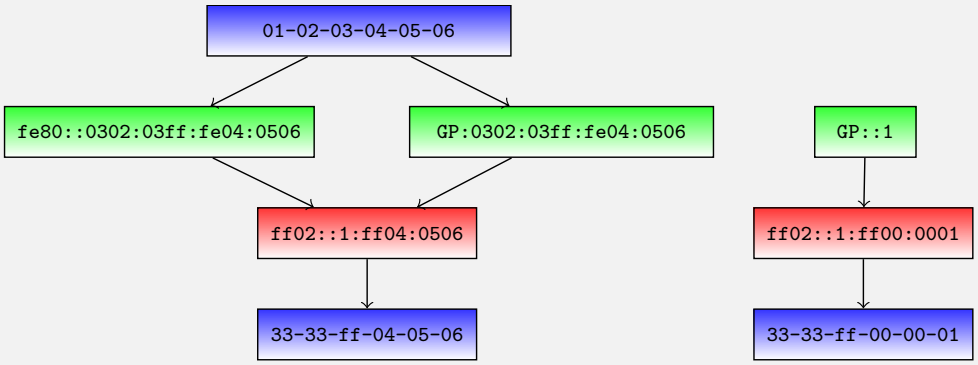
- 64 bits compatible avec EUI-64 (i.e. IEEE 1394 FireWire, ...)
- IEEE propose une méthode pour transformer un MAC-48 en EUI-64
- Bit Universal/Local modifié pour la numérotation



- Permet d'éviter les conflit avec les IID assigné manuellement

Adresses Multicast Sollicité

- Adresse IPv6 Multicast dérivées d'une adresse Unicast
  - Utilisée dans le protocole de découverte des voisins
  - Permet de s'affranchir du *broadcast* sur le réseau local



Example

1. IPv6 addr: `2001:0660:010a:4002:4421:21FF:FE24:87c1`
2. Sol. Mcast addr: `FF02:0000:0000:0000:0000:0001:FF24:87c1`
3. Ethernet: `33:33:FF:24:87:c1`

Exemple

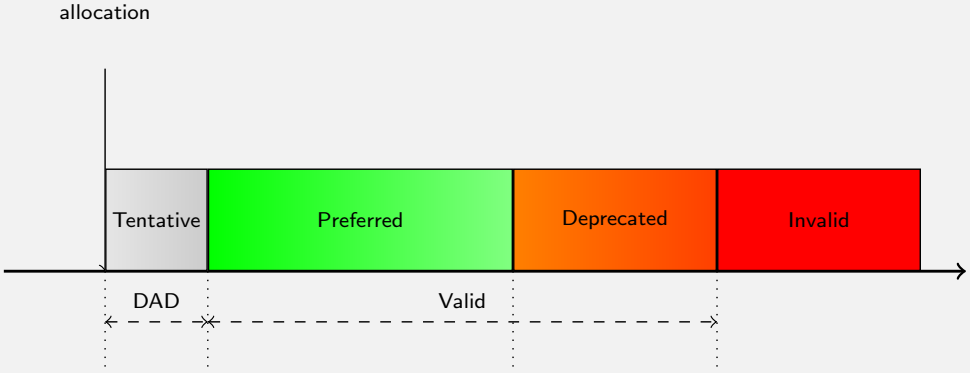
```
Vlan5 is up, line protocol is up
IPv6 is enabled, link-local address is fe80::203:fdff:fed6:d400
Description: reseau C5
Global unicast address(es):
  2001:660:7301:1:203:fdff:fed6:d400, subnet is 2001:660:7301:1::/64
Joined group address(es):
  ff02::1  <- All nodes
  ff02::2  <- All routers
  ff02::9  <- RIP
  ff02::1:ffd6:d400  <- Solicited Multicast
```

Etats d'une adresse IPv6

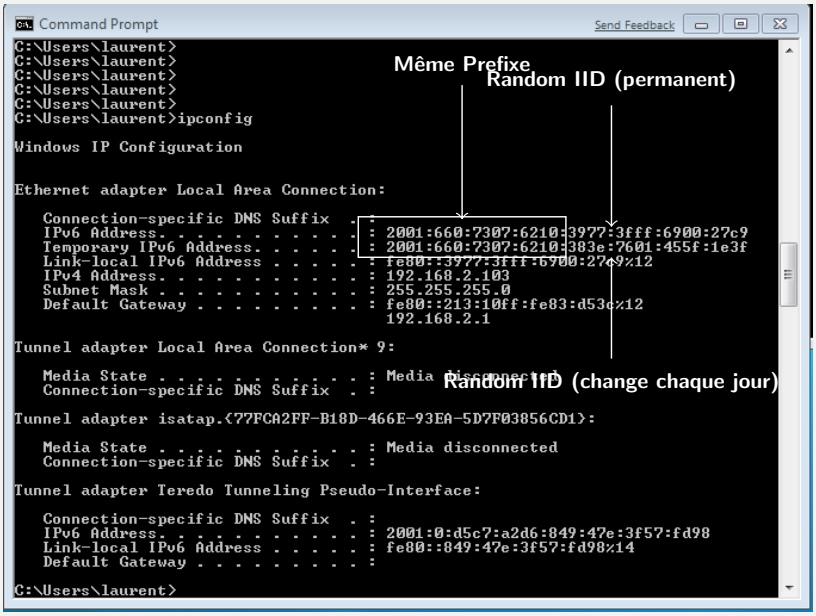
Une adresse IPv6 unicast locale ou globale possède un état pouvant évoluer au cours du temps

- état "DAD" : en cours de vérification de l'unicité
- état "préférée" : état opérationnel pour une communication
- état "dépréciée" : adresse en cours d'obsolescence
- état "invalide" : adresse interdite à l'utilisation

Des compteurs gèrent le passage d'un état à un autre



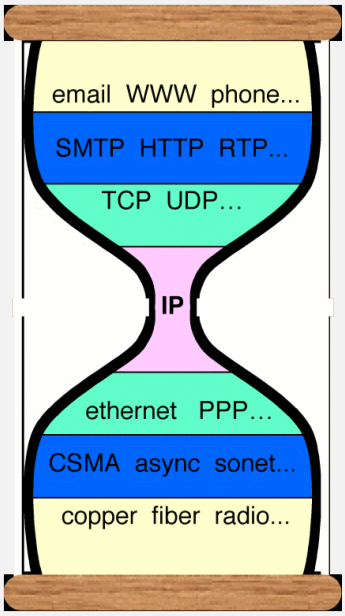
Windows 7



Agenda

Agenda

Couche Niveau 3 : IP



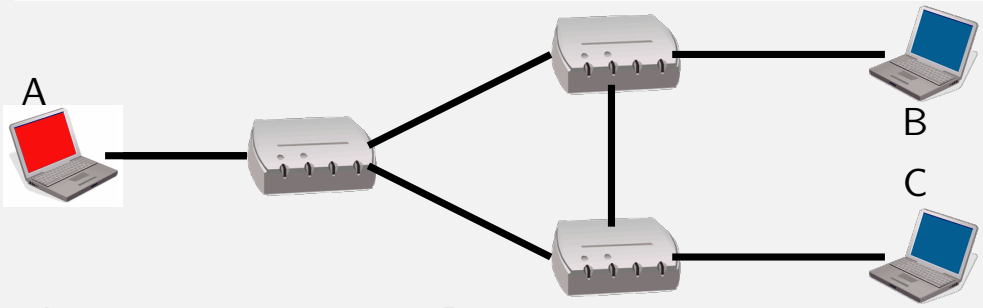
- IP est un protocole simple
  - Fonction : transmettre un paquet vers une destination
- IP universalise les supports de transmission
  - IP a été adapté sur tous les protocole de niveau 2
- IP est le support des protocoles de haut niveau
  - Les applications réseaux utilisent TCP/IP
- IP facilite l'interconnexion par le réseau
  - Identification assurée par l'unicité des adresses

`http://www.ietf.org/proceedings/01aug/slides/plenary-1/index.html` Steve deering, Watching the Waist of the Protocol Hourglass, IETF 51, London

Qu'est-ce qu'un paquet ?

Definition

1. Un paquet est l'élément unitaire de donnée transmis par IP
2. Chaque paquet est traité indépendamment
3. L'adresse destination doit être répétée dans chaque paquet
4. L'ensemble des équipements doivent convenir d'un même format d'en-tête du paquet



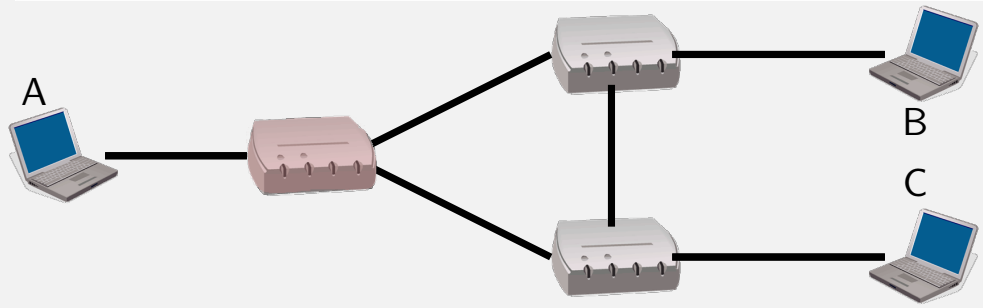
A envoie un paquet vers B

Cours 2 : Protocole IPv6 | Concept d'un protocole de couche 3

Qu'est-ce qu'un paquet ?

Definition

1. Un paquet est l'élément unitaire de donnée transmis par IP
2. Chaque paquet est traité indépendamment
3. L'adresse destination doit être répétée dans chaque paquet
4. L'ensemble des équipements doivent convenir d'un même format d'en-tête du paquet

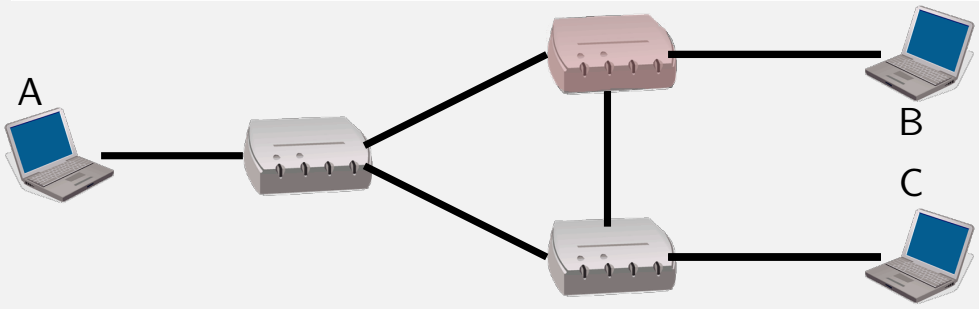


Le premier routeur examine l'entête pour déterminer l'interface de sortie

Qu'est-ce qu'un paquet ?

Definition

1. Un paquet est l'élément unitaire de donnée transmis par IP
2. Chaque paquet est traité indépendamment
3. L'adresse destination doit être répétée dans chaque paquet
4. L'ensemble des équipements doivent convenir d'un même format d'en-tête du paquet

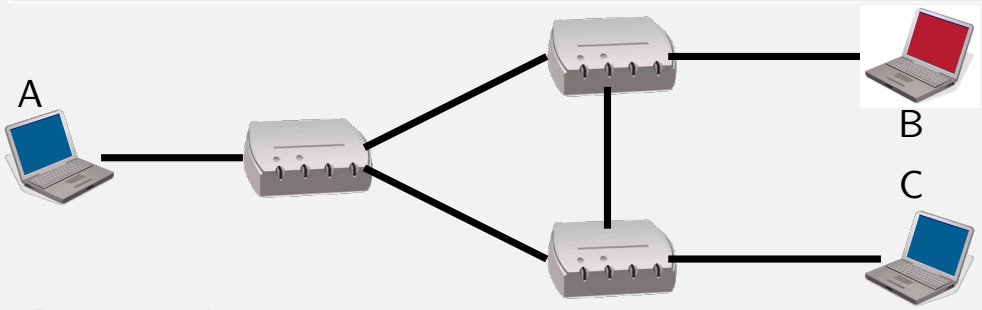


Le second routeur examine l'entête pour déterminer l'interface de sortie

Qu'est-ce qu'un paquet ?

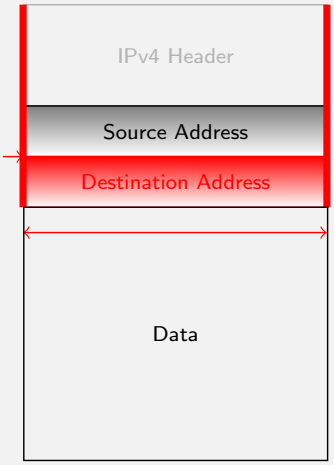
Definition

1. Un paquet est l'élément unitaire de donnée transmis par IP
2. Chaque paquet est traité indépendamment
3. L'adresse destination doit être répétée dans chaque paquet
4. L'ensemble des équipements doivent convenir d'un même format d'en-tête du paquet



B accepte le paquet

Traitement de l'Adresse Destination



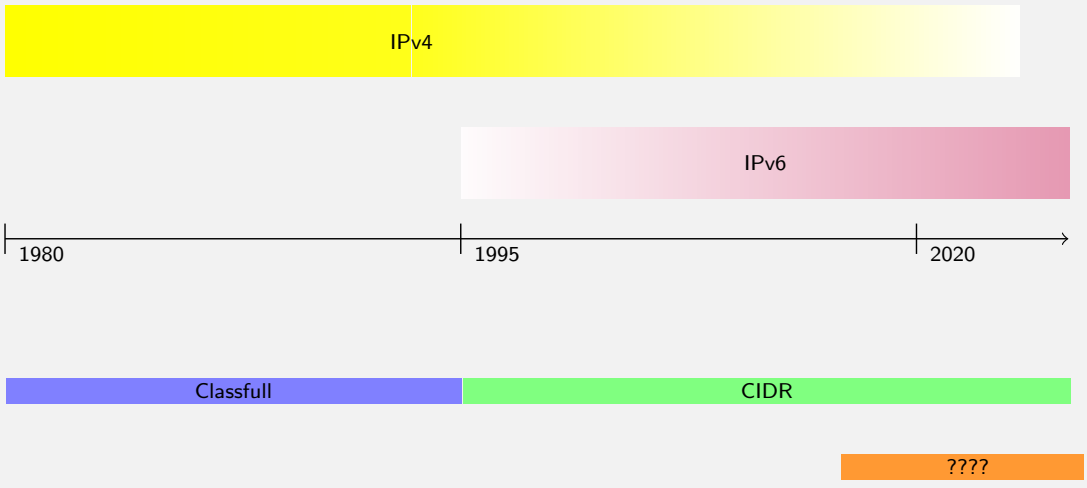
Accès optimisé à l'adresse destination :

- Emplacement fixe
- Taille fixe
- Alignée en mémoire

IPv4 (RFC 791, 1981)

`Les adresses ont une taille fixe de 4 octets (32 bits)`

Evolution des adresses et du format de paquet



Motivations pour IPv6

Répondre au problème de croissance

- Espace d'adressage plus vaste
- Meilleure gestion des adresses
- Restauration du principe de bout-en-bout

Renforcer les points fort d'IP

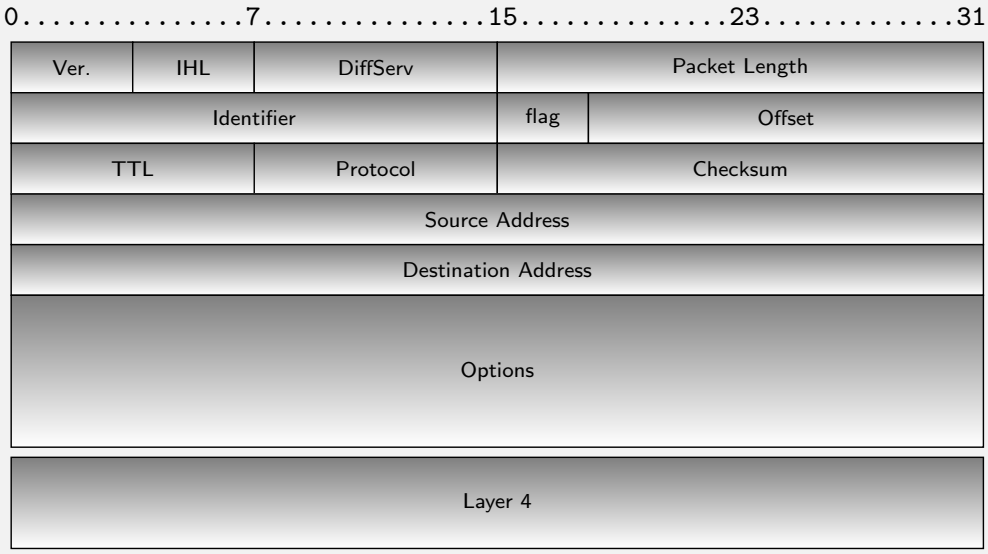
- Optimisation du transfert sur le réseau
- Traitement réduit dans les routeurs
- Fonctionnalités déportées sur les extrémités de la communication

Agenda

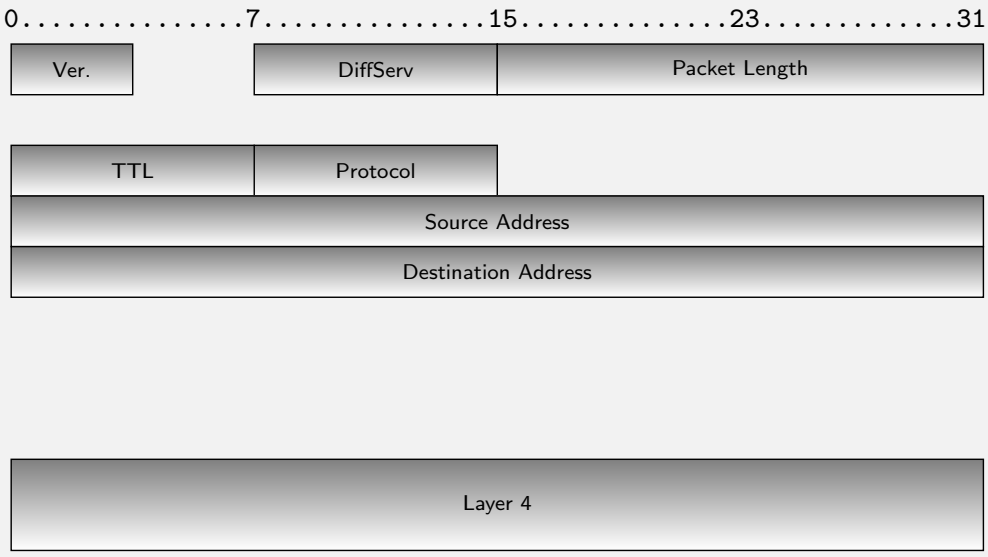
Evolutions depuis IPv4 vers IPv6

- L'en-tête IPv6 suit les principes d'IP :
  - Adresse de taille fixe . . . mais 4 fois plus longue
  - Alignement sur 64 bits
- Taille maximum de paquet (MTU) 1280 octets minimum
  - Si un niveau 2 a une MTU<1280 ⇒couche d'adaptation
- Refonte de certaines fonctionnalités inefficaces en IPv4
  - Mécanisme d'options : remplacé par les extensions d'en-tête
  - Checksum : supprimé de l'en-tête IP, obligatoire niveau 2 & 4
  - Fragmentation: seulement depuis les extrémitéss

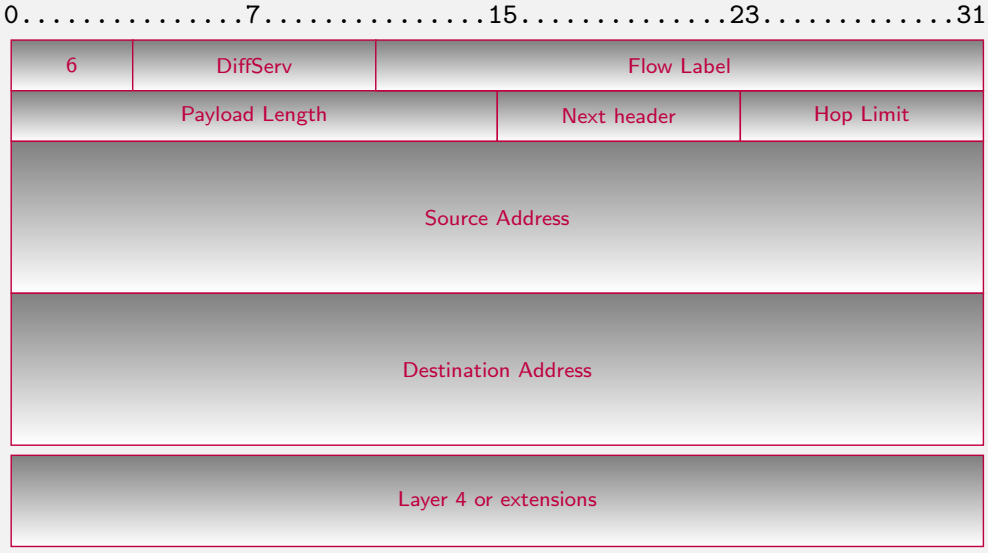
En-tête IPv6 RFC 8200



En-tête IPv6 RFC 8200



En-tête IPv6 RFC 8200



Est-ce suffisant pour le futur ?

- Hop Limit :
  - Compte le nombre de saut maximal jusqu'à la destination
  - 1 octet = 256 sauts maximum
  - Pas limitant car l'Internet croit en largeur, peu en profondeur
- Payload Length :
  - Taille maximale encodée : 64 Ko
  - Ethernet utilise généralement 1.5 Ko jusqu'à 9 Ko
  - Pour des cas spécifiques, l'extension Jumbogram autorise 4 Go

Flow Label (RFC 6437)

- Un flux est une séquence de paquets participant à la communication entre 2 applications à travers le réseau
  - Identifiable par le 5-tuple : adresses/ports source/destination et protocole transport
  - Ingénérie de trafic : spécifier les règles de traitement du trafic (QoS) par flux
- Le Flow Label field a été conçu pour faciliter la classification des packets en flux
  - Sans Flow Label, un classifieur doit analyser le 5-tuple
    - ★ Peu performant (analyse à chaque routeur/classifieur)
    - ★ Parfois impossible (fragmentation ou chiffrement du contenu)
- Flow Label : identifiant de flux fixé par la source
  - Flow label + adresse source = id unique
  - Le traitement différencié peut s'effectuer sur la base du Flow Label
  - Exemple : Load-balancing (RFC 7098)

Agenda

Impact sur les protocoles de niveau 2

IPv6 est supporté sur la plupart des moyens de transmission :

- IEEE 802.3 (LAN), 802.11 (WLAN), 802.15 (WPAN)
- 3GPP (UMTS, LTE)
- ATM, PPP

MTU minimale en IPv6 est de 1280 octets ⇒ Couche d'adaptation si le niveau 2 ne le permet pas :

- AAL5 pour ATM (format de cellule = 5 octets d'entête et 48 octets de données)
- 6LowPAN pour 802.15.4 (trame de 127 octets, dont 44 octets d'entête, 81 octets de données et 2 octets de FCS)

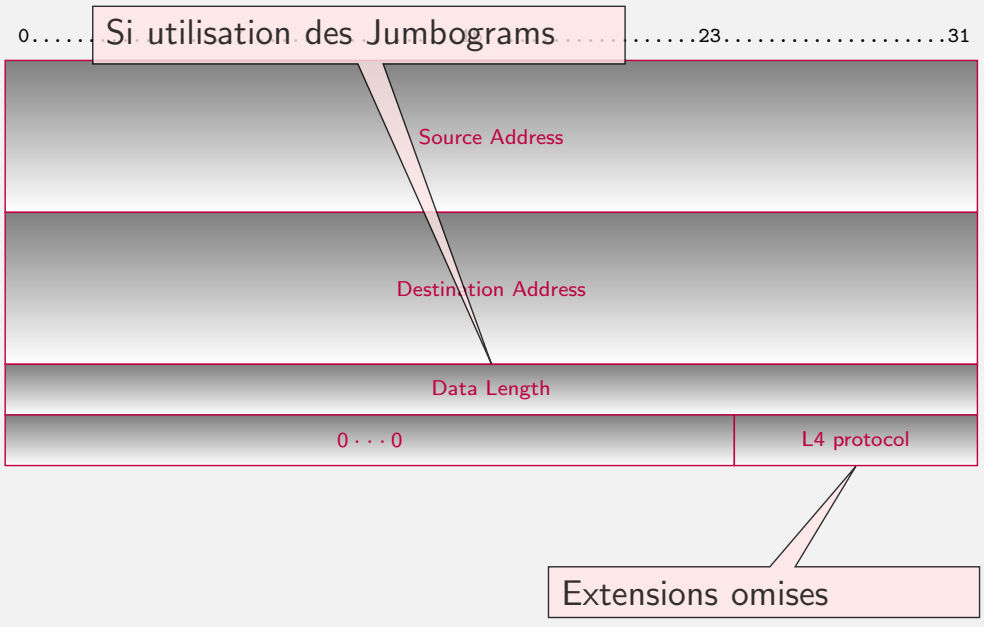
Impact sur les protocoles de niveau 4 (1/2)

Le Checksum est maintenant obligatoire pour les protocoles transportés sur IPv6 :

- TCP: pas de problème , le checksum est déjà présent
- UDP: le checksum est omis pour les packets multimédia ⇒UDP-lite (RFC 3828) spécifie la gestion du checksum avec IPv6
- ICMP version 6 inclut désormais un Checksum

Afin de protéger certaines parties de l'en-tête IPv6, le Checksum au niveau 4 inclue des valeurs de cette en-tête.

IPv6 Pseudo-Header (pour le calcul du Checksum)



Impact sur les protocoles de niveau 4 (2/2)

IPv6 est quasi transparent pour les protocoles de niveau 4, excepté:

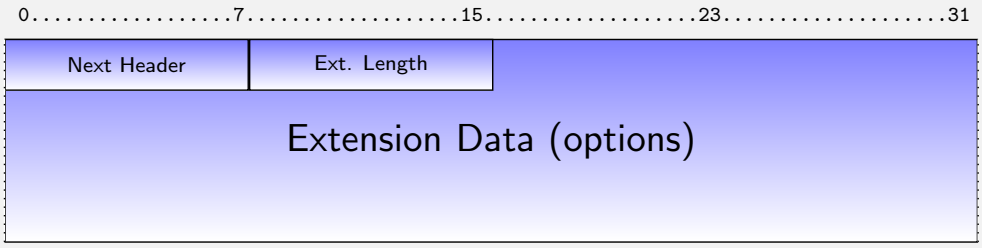
- si utilisation des Jumbograms :
  - UDP: Si *length* $>$ 65535 $\Rightarrow$ UDP.length $=$ 0, considerer la taille dans l'extension
  - TCP: Si *Length* $>$ 65535, considérer la valeur de PMTU
- SCTP (Stream Control Transmission Protocol): durant l'ouverture de sessions, des adresses IPv4 et IPv6 peuvent être échangées.

Agenda

Extensions de l'en-tête IPv6 RFC 8200

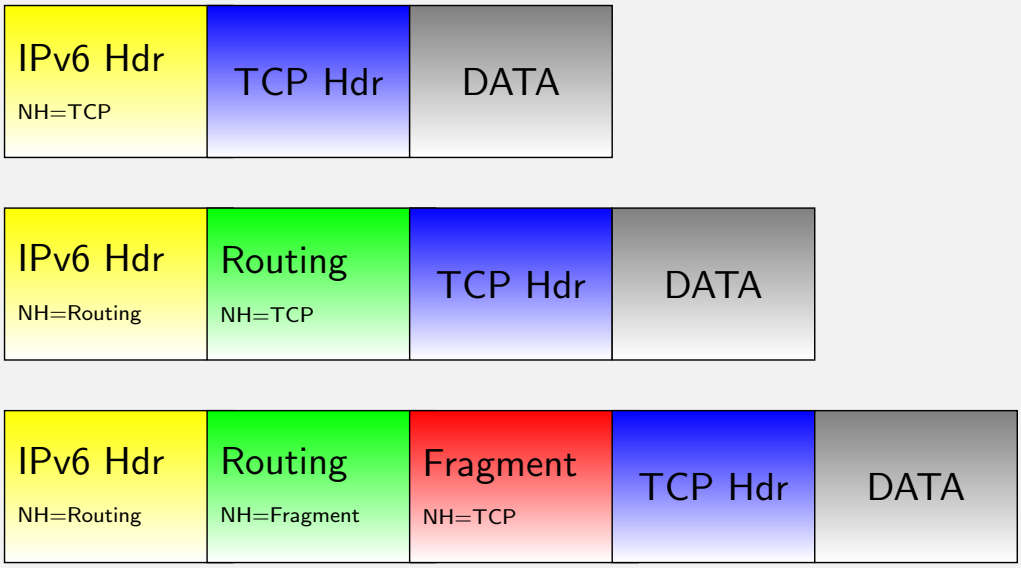
- Extensions des fonctionnalités du protocole IPv6
- Equivalentes aux options d'IPv4 (mais en plus performant)
- Insérées par la source, traitées par la destination
  - Exceptée Hop-by-Hop traitées par chaque routeur
- Pas de limitation de taille
- Vues comme des en-têtes de niveau 4
- Chainage possible pour cumuler des fonctions étendues
- Types différents selon la fonctionnalité :
  - Destination (mobilité IP)
  - Proche-en-Proche
  - Routing (routage par la source, mobilité IP)
  - Fragmentation
  - Authentication (IPsec AH)
  - Security (Chiffrement IPsec)

Format Générique des Extensions

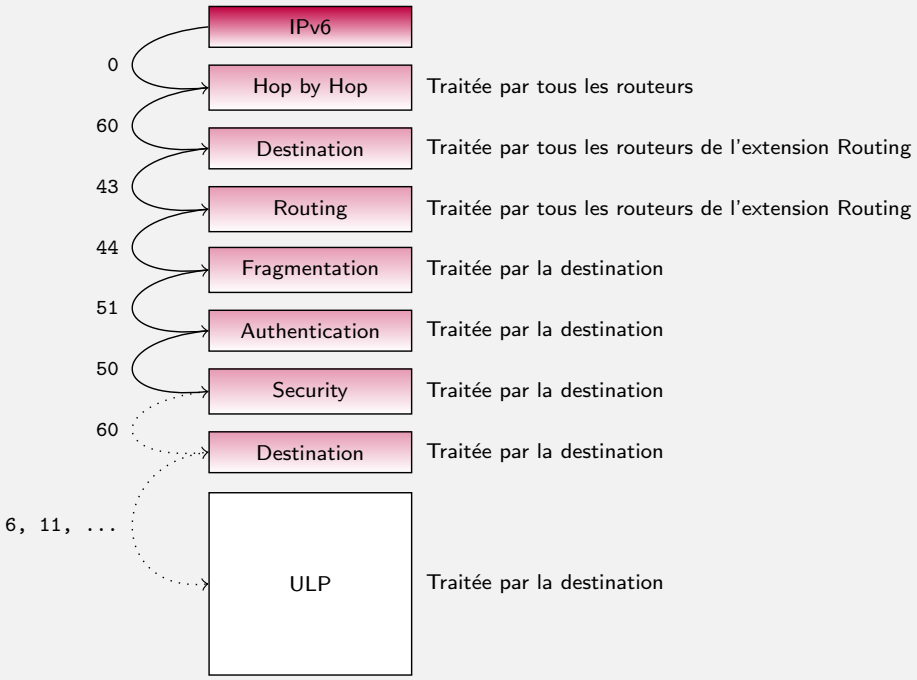


- Next Header : en-tête suivant (protocole niveau 4 ou extension)
- Length : nombre de mots de 64-bits pour les extensions de taille variable (0 si taille fixe)
- Data : données spécifique à l'extension

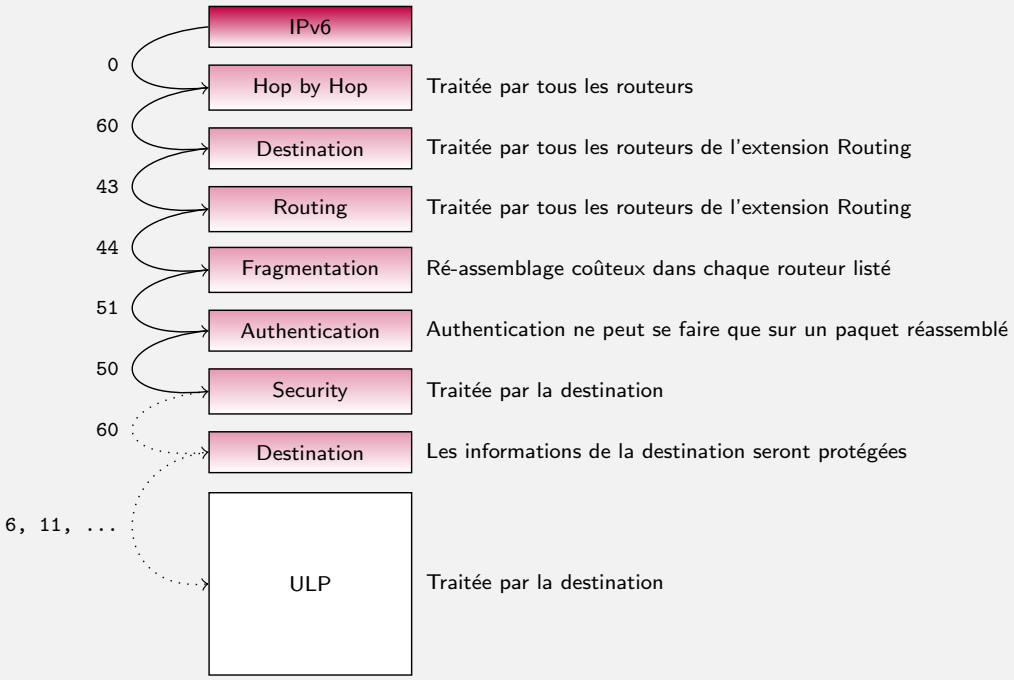
Encapsulation des extensions



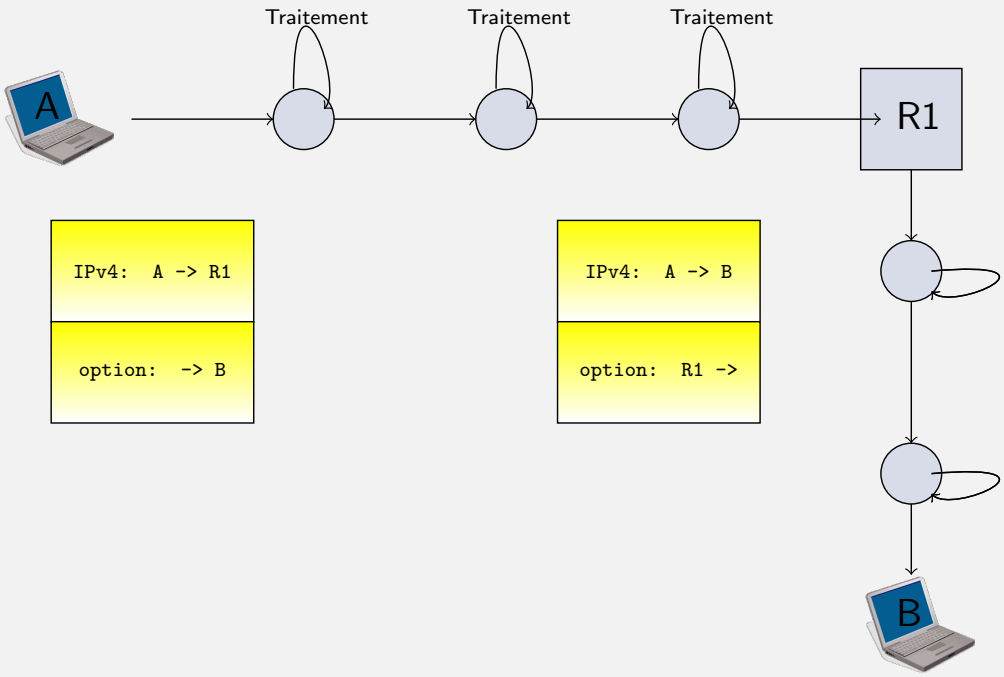
L'ordre des extensions est important



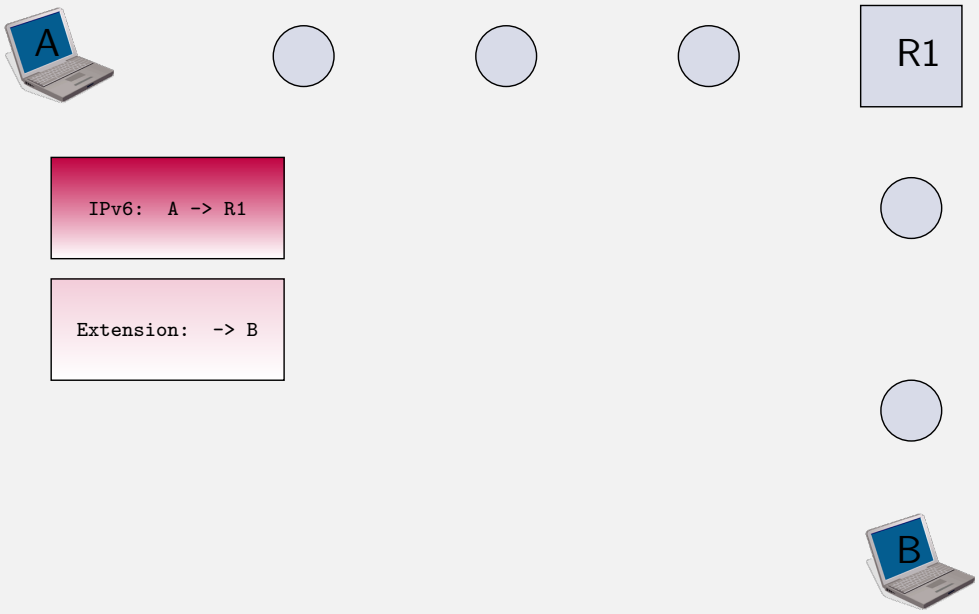
L'ordre des extensions est important



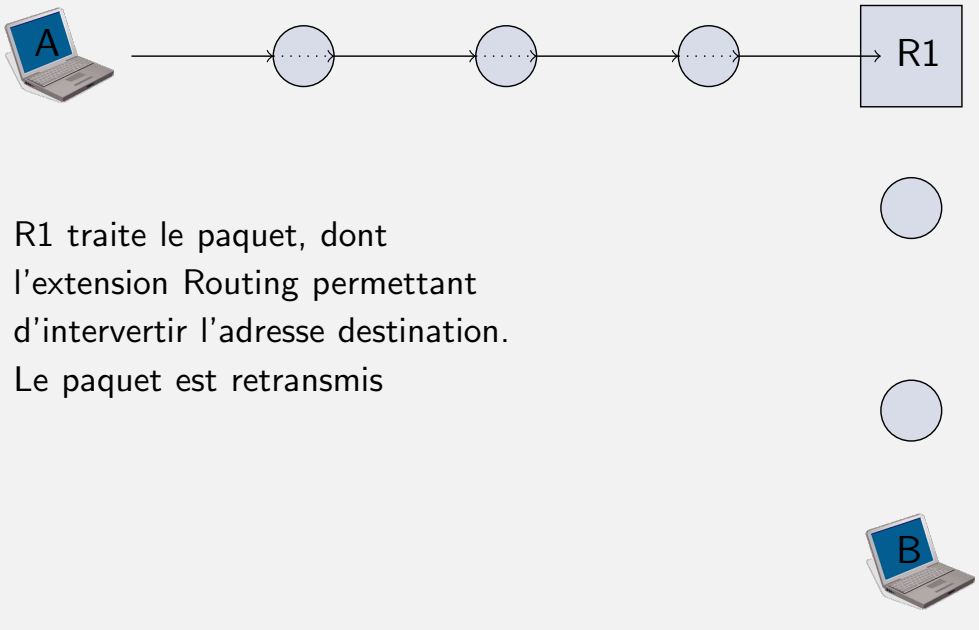
Superiorité des extensions



Superiorité des extensions

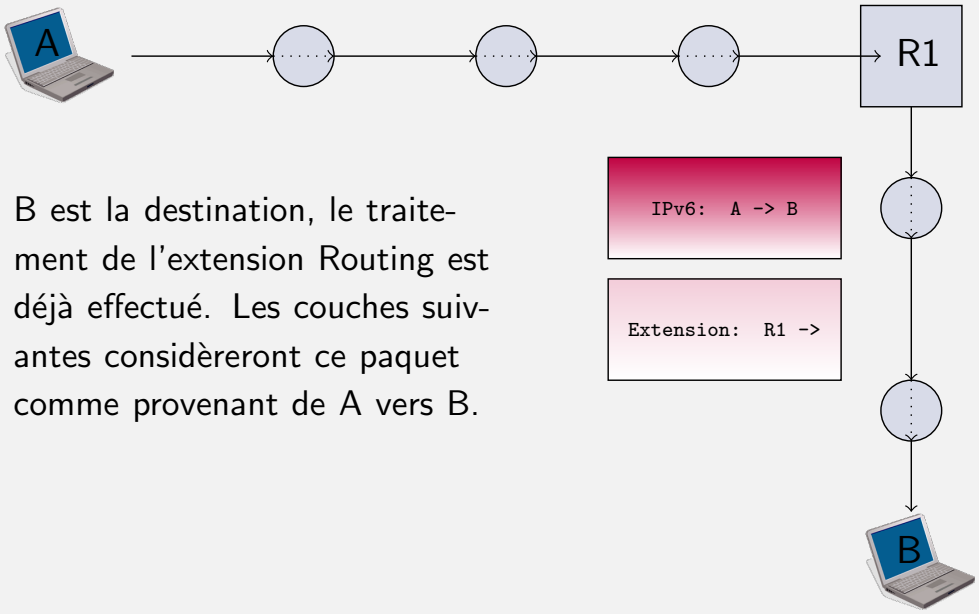


Superiorité des extensions



R1 traite le paquet, dont
l'extension Routing permettant
d'intervertir l'adresse destination.
Le paquet est retransmis

Superiorité des extensions



B est la destination, le traitement de l'extension Routing est déjà effectué. Les couches suivantes considèreront ce paquet comme provenant de A vers B.

Agenda

Règles pour la taille des paquets IPv6

Un paquet devant être envoyé sur le réseau doit avoir une taille inférieure à la taille maximale autorisée par le réseau sous-jacent

⇒ *MTU: Maximum Transmission Unit* : taille maximale autorisée sur un réseau

Un paquet pourra atteindre sa destination directement si sa taille est inférieure à la plus basse MTU autorisée sur l'ensemble du chemin.

⇒ *PMTU: Path Maximum Transmission Unit* : MTU minimale sur le chemin

Règle 1

La MTU pour IPv6 (ainsi que la PMTU) doit être supérieure ou égale à 1280 octets.

Règles pour la taille des paquets IPv6

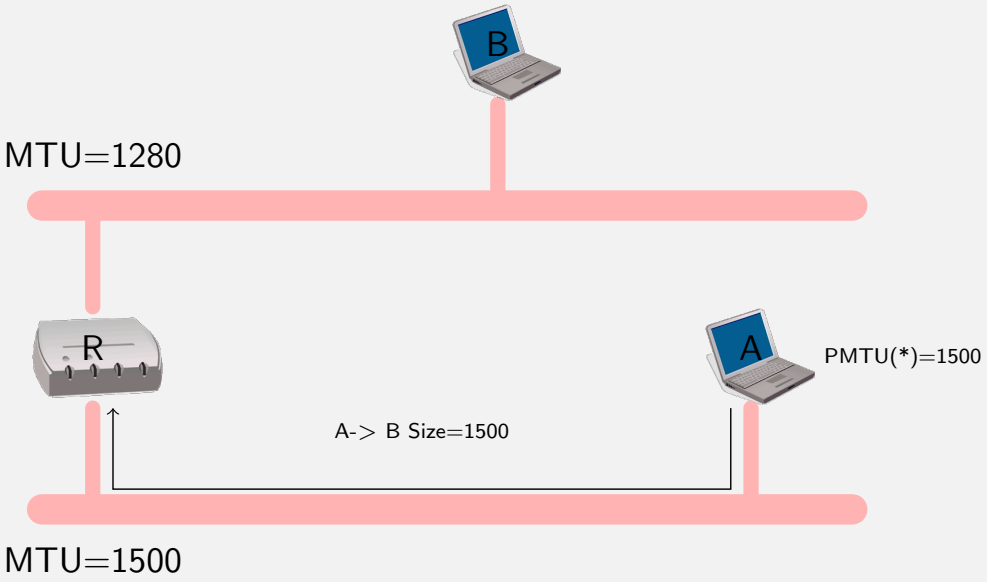
Règle 2

La fragmentation d'un paquet ne peut intervenir qu'à l'émetteur.

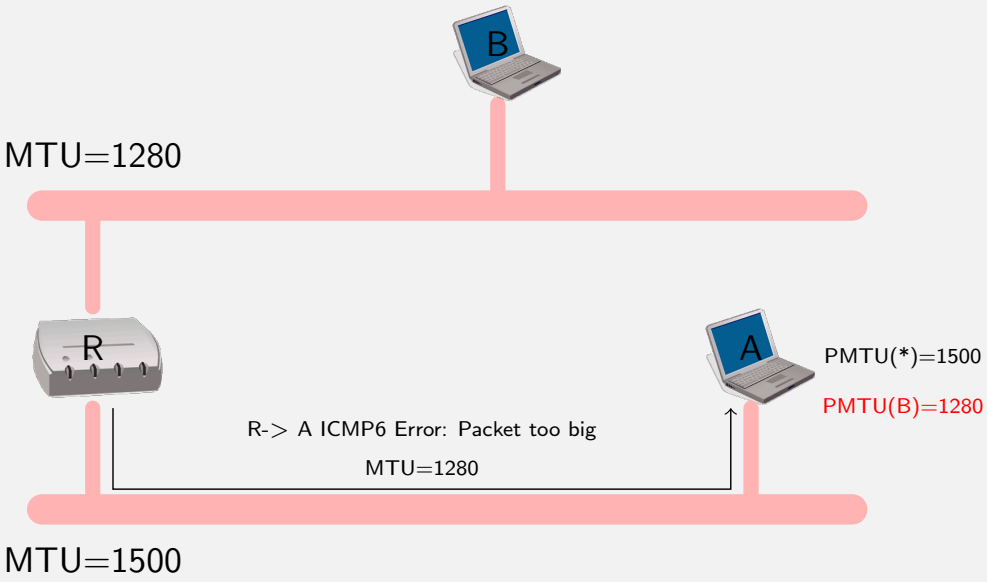
Conséquences:

- Pour éviter la fragmentation, l'émetteur doit connaitre la PMTU vers le destinataire ⇒ **PMTU discovery**
- Une fois PMTU connue, la taille des données est ajustée au niveau 4
- Si la taille ne peut pas être ajustée (UDP); la fragmentation est utilisée
  - L'émetteur envoie des fragments avec l'extension Fragmentation
  - Le ré-assemblage des paquets s'effectue à la destination

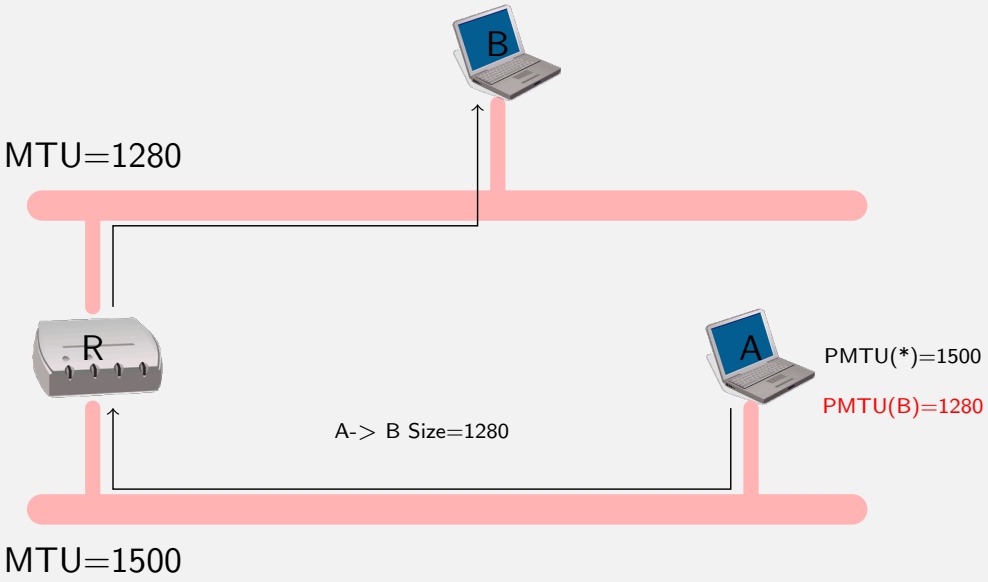
Découverte de la MTU du chemin RFC 8201



Découverte de la MTU du chemin RFC 8201



Découverte de la MTU du chemin RFC 8201

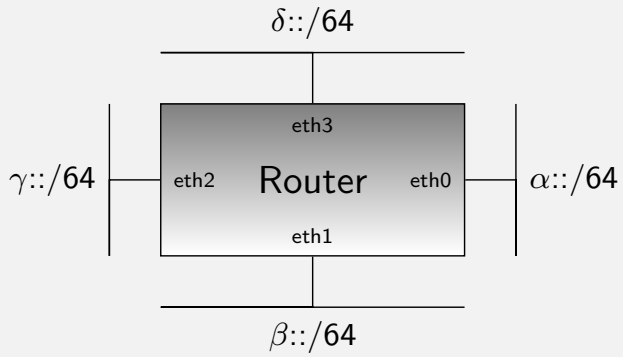


Agenda

Configuration IPv6 d'un routeur

Les interfaces d'un routeur sont configurées manuellement en IPv6

- Assignation à l'interface d'une addresse et d'une longueur de prefixe
- Le routeur installe le préfixe correspondant à chaque interface dans sa FIB



FIB

Prefix	Next Hop
α	eth0
β	eth1
γ	eth2
δ	eth3

Exemple : Routeur Cisco

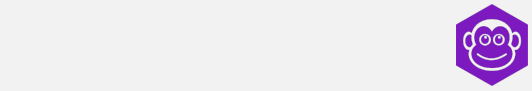
```
cisco_showroom#    show ipv6 route

IPv6 Routing Table - 7 entries

Codes: C - connected, L - Local, S - Static, R - RIP, B - BGP
U - Per-user Static route, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea,
IS - ISIS summary, O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1,
OE2 - OSPF ext 2, ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2


S ::/0 [1/0] via FE80::216:17FF:FE87:A7, Vlan338
C 2001:660:7301:3303::/64 [0/0] via ::, Vlan333
L 2001:660:7301:3303::1/128 [0/0] via ::, Vlan333
C 2001:660:7301:3308::/64 [0/0] via ::, Vlan338
L 2001:660:7301:3308:20D:29FF:FE75:43C4/128 [0/0] via ::, Vlan338
L FE80::/10 [0/0] via ::, Null0
L FF00::/8 [0/0] via ::, Null0

....
```

Exemple: Linux

Table de routage IPv6 du noyau

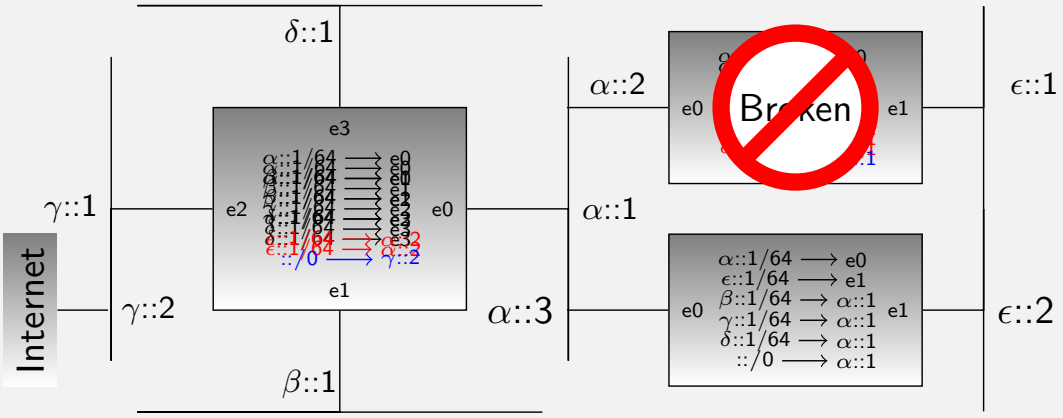
netstat -rn ip -6

Destination	Next Hop	Flags	Metric	Ref	Use	Iface
::1/128	::	U	0	6	1	lo
2001:660:7301:3302::/128	::	U	0	0	2	lo
::/0	fe80::213:c4ff:fe69:5f49	UG	1	34532	0	eth1
2001:660:7301:3303::/64	fe80::20d:29ff:fe75:43c4	UG	1024	6708480	0	eth0.338



Routage statique

- Les routes sont configurées manuellement



- Simple à configurer, mais sujet aux erreurs (boucles de routage)
- Un chemin alternatif peut être déclaré en cas de panne en spécifiant une métrique moins favorable que la route principale (*route statique flottante*)

Exemple : commandes de configuration

- BSD:
  - `route add -inet6 default fe80::216:17ff:fe87:a7%en0`
  - `route add -inet6 2001:660:7301:3303::  -prefixlen 64 fe80::20d:29ff:fe75:43c4%en0`
- Linux:
  - `route -A inet6 add default gw fe80::216:17ff:fe87:a7 dev eth0`
  - `ip -6 route add 2001:660:7301:3303::/64 via fe80::20d:29ff:fe75:43c4 dev eth0`
- Cisco:
  - `ipv6 route ::/0 vlan 338 fe80::216:17ff:fe87:a7`

Exemple : Routeur Cisco

```
cisco_showroom#    show ipv6 route
IPv6 Routing Table - 7 entries

Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, U - Per-user Static route

S ::/0 [1/0] via FE80::216:17FF:FE87:A7, Vlan338
C 2001:660:7301:3303::/64 [0/0] via ::, Vlan333
L 2001:660:7301:3303::1/128 [0/0] via ::, Vlan333
C 2001:660:7301:3308::/64 [0/0] via ::, Vlan338
L 2001:660:7301:3308:20D:29FF:FE75:43C4/128 [0/0] via ::, Vlan338
L FE80::/10 [0/0] via ::, Null0
L FF00::/8 [0/0] via ::, Null0
...
```

Protocoles de routage

Definition

Un protocole de routage permet le partage d'informations entre les routeurs afin de construire leur FIB

- Chaque protocole est instancié dans le routeur par une base de donnée alimantant la FIB
- Deux grandes familles de protocole de routage :
  - Interior Gateway Protocol
  - Exterior GAteway Protocol

IGP

Interior Gateway Protocol

- Configuration simplifiée (même si le protocole peut être complexe)
- Découverte des routeurs du réseau, puis échange d'information
- Exemples : RIPng (Distance Vector), OSPFv3 ou IS-IS (Link State)

EGP

Exterior GAteway Protocol

- Configuration exhaustive, rien n'est automatisée
- Politiques de filtrage de préfixes
- Utilisés dans les échanges entre opérateurs (*peering*)
- Un protocole existant : Border Gateway Protocol (+ extensions MP-BGP)

Protocoles de routage pour IPv6

- IGP
  - **RIPng**: RFC 2080, RFC 2081 (Extension de RIPv2 pour IPv6)
  - **OSPFv3**: RFC 5340 (OSPF pour IPv6)
    - ⋆ Voir aussi RFC 5185 (OSPF Multi-Area Adjacency) et RFC 5838 (Support of address families in OSPFv3)
  - **ISIS**: RFC 5308 (Routage d'IPv6 avec IS-IS et gestion des plans de routage)
- EGP
  - **MP-BGP (BGP4+)**: RFC 2545 (Multi-protocol extensions for IPv6 Inter-Domain Routing)

Conclusion

Pas de différences majeures avec IPv4

Exemples de configuration du routage pour IPv6

- IGP
  - **RIPng**: `https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_rip/configuration/xe-3s/asr1000/ip6-rip-xe.html` Cisco, IPv6 Routing: RIP for IPv6
  - **OSPFv3**: `https://www.cisco.com/c/en/us/support/docs/ip/ip-version-6-ipv6/112100-ospfv3-config-guide.html` Cisco, Sample Configuration for OSPFv3
  - **ISIS**: `https://www.cisco.com/c/en/us/support/docs/ip/integrated-intermediate-system-to-intermediate-system-is-is/40262-ipv6-sample-config.html` Cisco, Configuring IS-IS over IPv6:
- EGP
  - **MP-BGP (BGP4+)**: `https://www.cisco.com/c/en/us/support/docs/ip/ip-version-6-ipv6/112135-ipv6-bgp-00.html` Cisco, Multiprotocol BGP for IPv6 Configuration Example

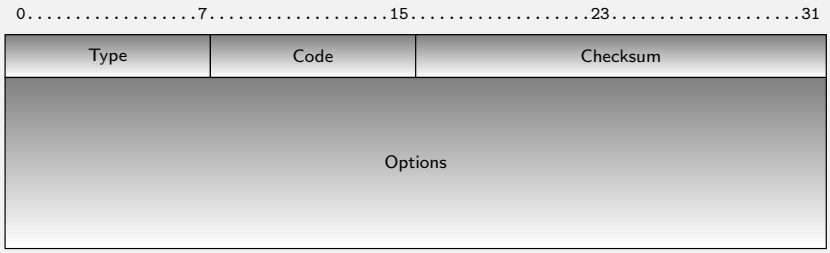
Agenda

Agenda

ICMPv6 RFC 4443

- ICMPv6 : *Internet Control Message Protocol for IPv6*
- ICMPv6 est différent d'ICMPv4
  - Valeur du champ `next header` : 58 (0x3a)
- Fonctions étendues et mieux organisées
- Filtrage d'ICMPv6 à effectuer avec précaution (RFC 4890)

Format :



Détails

`type` : Fonction du message ICMPv6
`code` : Cause du message ICMPv6 (dépend du `type`)
`checksum` : Code d'integrité du message ICMPv6 obligatoire

ICMPv6 : Deux Fonctions

- Rapports d'erreur lors de la transmission d'un paquet ($type < 128$)

1	Destination Unreachable
2	Packet Too Big
3	Time Exceeded
4	Parameter Problem

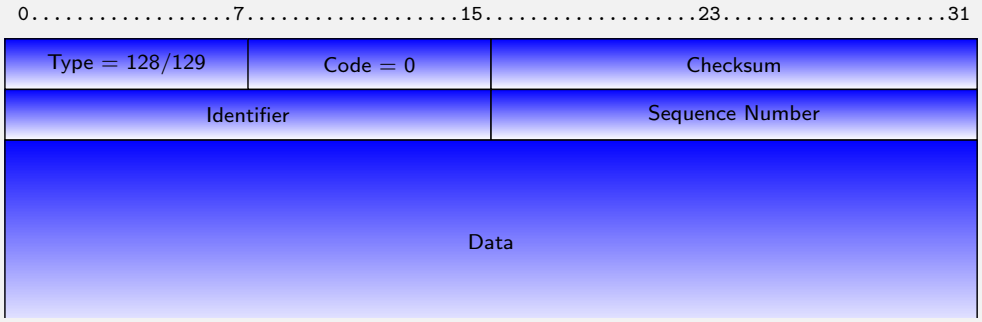
- Gestion du réseau ($type > 128$)

128	Echo Request
129	Echo Reply

130	Group Membership Query
131	Group Membership Report
132	Group Membership Reduction

133	Router Solicitation
134	Router Advertissement
135	Neighbor Solicitation
136	Neighbor Advertissement
137	Redirect

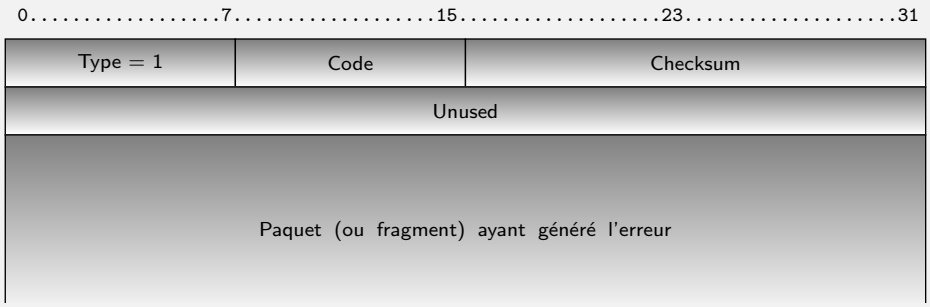
Test de connectivité (*ping*)



Type:

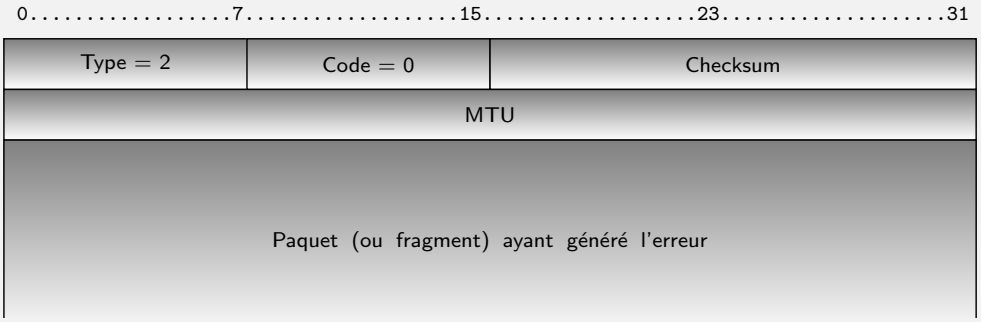
- 128 : Echo request
- 129 : Echo reply

Erreur : *Destination unreachable*



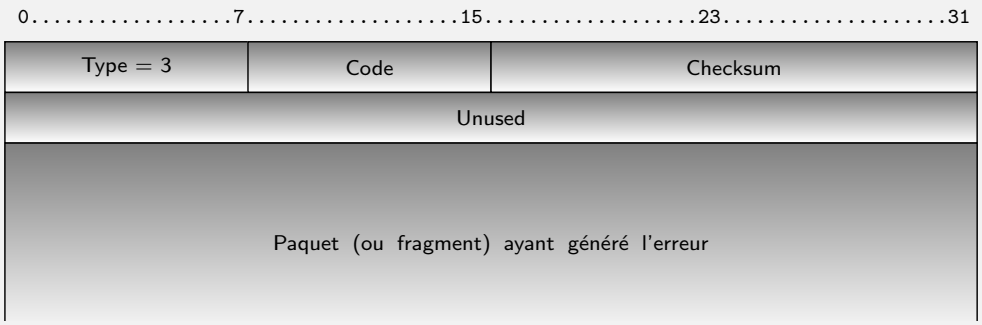
- La destination d'un paquet n'est pas joignable
- Code : (cause de l'erreur)
  - 0 - No route to destination
  - 1 - Communication with destination administratively prohibited
  - 2 - Beyond scope of source address
  - 3 - Address unreachable
  - 4 - Port unreachable
  - 5 - Source address failed ingress/egress policy
  - 6 - Reject route to destination
  - 7 - Error in Source Routing Header

Erreur : *Packet Too Big*



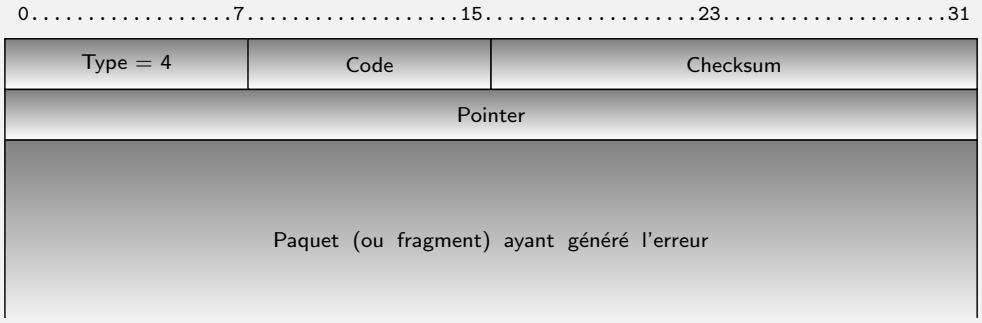
- Rapporte qu'un routeur intermédiaire (voir adresse source) n'a pas pu transmettre le paquet suite un problème de MTU
- Déclenche à la source le protocole de découverte de la PMTU

Erreur : *Time Exceeded*



- Le paquet est resté trop longtemps dans le réseau
- Code : (cause de l'erreur)
  - 0 - Hop limit exceeded in transit
  - 1 - Fragment reassembly time exceeded
- Utilisé par traceroute6 pour identifier le chemin

Erreur : *Parameter Problem*



- Une erreur est survenue dans l'interprétation du paquet
- Code : (cause de l'erreur)
  - 0 - Erroneous header field encountered
  - 1 - Unrecognized Next Header type encountered
  - 2 - Unrecognized IPv6 option encountered
  - 3 - IPv6 First Fragment has incomplete IPv6 Header Chain
- Pointer : Octet où l'erreur a été détectée

Agenda

ICMPv6 : Deux Fonctions

- Rapports d'erreur lors de la transmission d'un paquet ($type < 128$)

1	Destination Unreachable
2	Packet Too Big
3	Time Exceeded
4	Parameter Problem

- Gestion du réseau ($type > 128$)

128	Echo Request
129	Echo Reply
130	Group Membership Query
131	Group Membership Report
132	Group Membership Reduction
133	Router Solicitation
134	Router Advertissement
135	Neighbor Solicitation
136	Neighbor Advertissement
137	Redirect

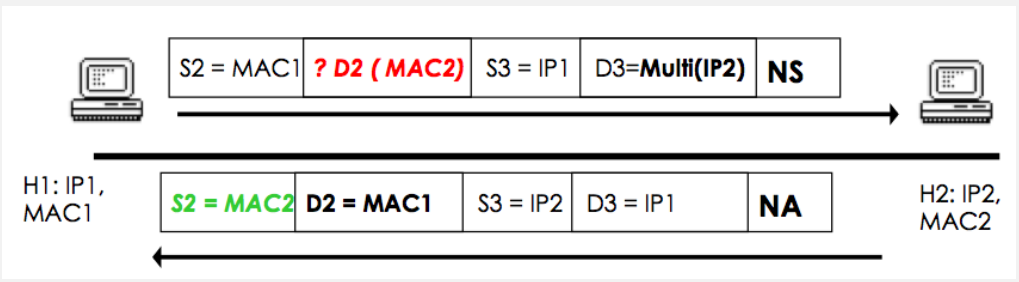
Neighbor Discovery RFC 4861

- Les équipement IPv6 partageant le même lien utilise la découverte des voisins pour :
  - Déterminer l'adresse de niveau 2 (MAC) de leurs voisins
    - ★ IPv4 : ARP
  - Configurer automatiquement leur interface
    - ★ Parametres de niveau 3 : IPv6 address, default route, MTU and Hop Limit
    - ★ Seulement pour les équipements terminaux !
    - ★ Pas d'équivalent en IPv4
  - Détecter la duplication d'adresses (DAD)
    - ★ IPv4 : gratuitous ARP
  - Maintenir l'information de présence des voisins (NUD: Neighbor Unreachability Detection Is Too Impatient, RFC 7048)
- Utilisation du multicast au niveau du lien local (*scope* = 2)
- Messages ICMPv6 utilisés :
  - Router Solicitation: 133 ; Router Advertisement: 134 ; Neighbor Solicitation: 135 ; Neighbor Advertisement: 136 ; Redirect: 137

Résolution des adresses niveau 2

Principe

- Un paquet doit être envoyé vers une machine du même réseau
- L'émetteur doit résoudre l'adresse de niveau 2 pour envoyer la trame
- Il envoie une requête ICMPv6 *Neighbor Solicitation* (NS) pour résoudre l'adresse
- Le requête est envoyée vers le groupe multicast sollicité correspondant à la destination
- Le destinataire (écoutant sur ce groupe) répond par ICMPv6 *Neighbor Advertisement* (en unicast)



Détection des adresses dupliquées (DAD) RFC 4862

Règle 1

Avant d'être valide sur une interface, une adresse IPv6 doit être vérifiée comme unique

Le mécanisme DAD utilise la découverte des voisins pour s'assurer de l'unicité d'une adresse

- L'équipement envoie un message ICMPv6 NS pour chercher à résoudre l'adresse sur le lien
- Au même moment un compteur (environ 1s) est armé
- Si aucun message NA n'est reçu à l'expiration du compteur, l'adresse est unique

Règle 2

Si un conflit d'adresse est détecté, l'administrateur se charge de sa résolution

Agenda

Configuration automatique des paramètres réseaux

Objectif : rendre la connexion réseau *plug & play*

- Faciliter l'usage du réseau : pas de configuration de l'utilisateur
- Faciliter l'administration : centralisation des paramètres

RFC 4862 : Stateless Address Auto-Configuration (SLAAC)

- Mécanisme de configuration sans état
- Routeur du réseau maintient les informations communes
- Les équipements terminaux se configurent de manière autonome

RFC 3315 : Stateful Address Auto-Configuration

- Protocole DHCPv6 (Dynamic Host Configuration Protocol for IPv6)
- Gestion administrée de la configuration automatique (pas d'autonomie)
- Utilisée seule ou en complément de la configuration sans état

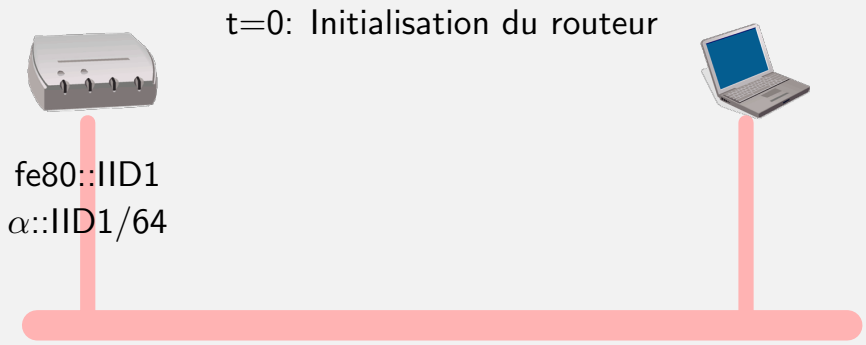
Configuration automatique sans état

- Une interface peut créer une adresse unicast globale à partir :
  - son identifiant d'interface (basé sur sa MAC, aléatoire, . . .)
  - Le préfixe global défini pour le réseau local
- Pour permettre la connectivité, l'interface nécessite aussi
  - L'adresse du routeur par défaut du réseau local
  - L'adresse du *resolver DNS*
  - La MTU à utiliser par défaut sur ce réseau
  - Les valeurs des compteurs pour la durée de vie de l'adresse

Paramètres communs au réseau local

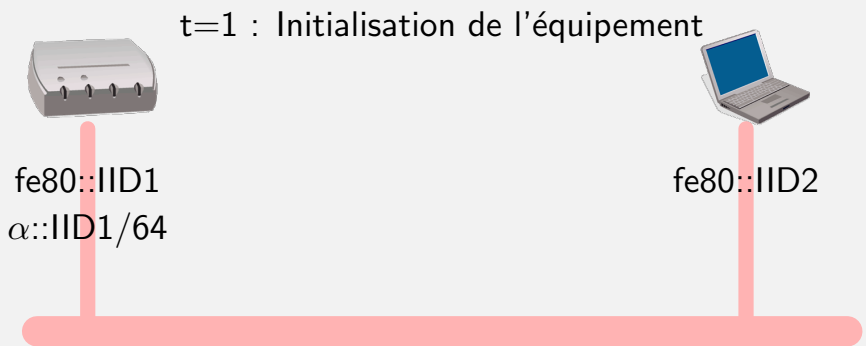
- Ces paramètres sont centralisés sur le routeur du réseau
- Les équipements terminaux découvrent et interrogent le routeur

Mécanisme de l'auto-configuration sans état



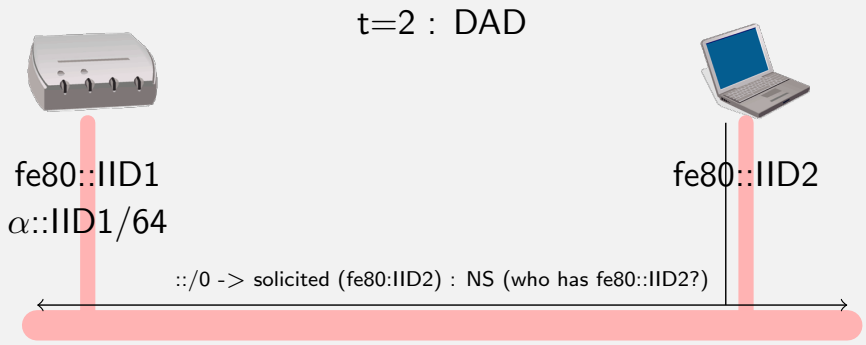
L'interface du routeur possède une adresse lien-local et est configurée avec une adresse globale (préfixe α::/64 défini par l'administrateur réseau)

Mécanisme de l'auto-configuration sans état



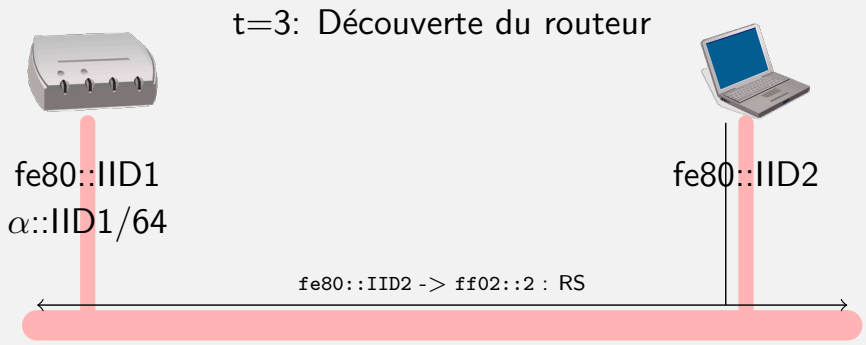
L'équipement construit son adresse lien-local (basée sur son IID : interface Identifier=adresse MAC)

Mécanisme de l'auto-configuration sans état



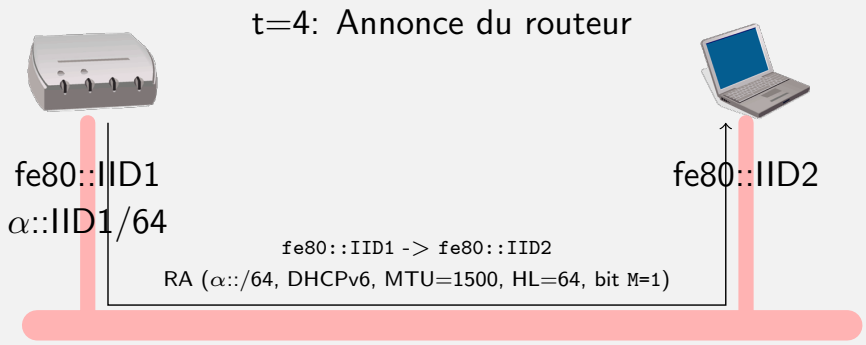
L'équipement vérifie l'unicité de son adresse lien-local par le mécanisme DAD (i.e. envoi d'un message ICMPv6 *Neighbor Solicitation*. Si aucune réponse reçue, l'adresse est unique.

Mécanisme de l'auto-configuration sans état



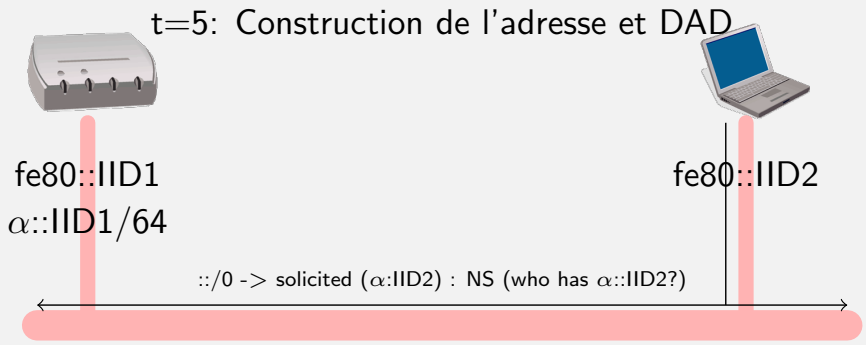
L'équipement envoie un message ICMPv6 *Router Solicitation* à destination du groupe Multicast *All-Routers*.

Mécanisme de l'auto-configuration sans état



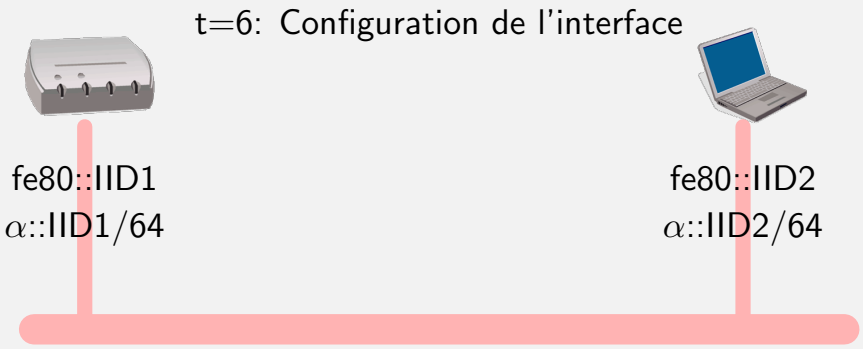
Le routeur répond à cette requête de découverte par un message ICMPv6 *Router Advertisement*. L'annonce contient les paramètres de configuration communs au réseau (préfixe du réseau, méthode de configuration de l'adresse, resolver DNS, ...).

Mécanisme de l'auto-configuration sans état



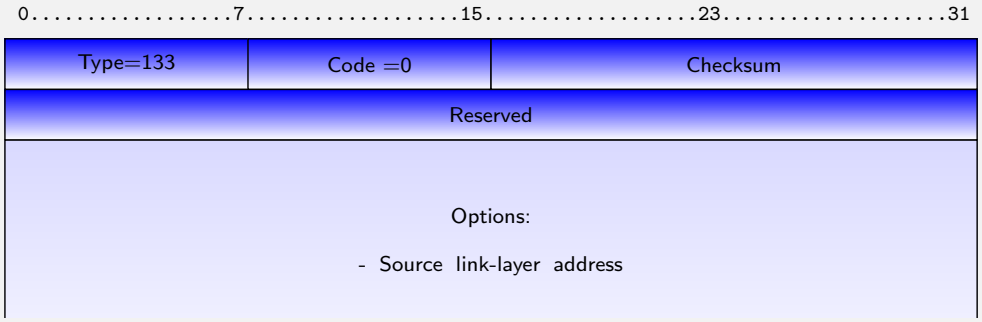
A partir du préfixe, l'équipement construit son adresse globale et vérifie son unicité.

Mécanisme de l'auto-configuration sans état



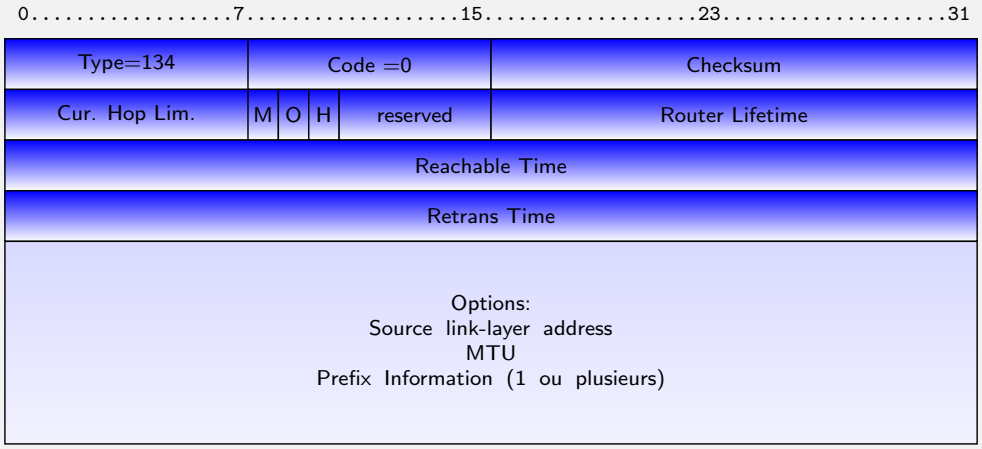
L'équipement assigne à son interface l'adresse globale construite, et ajuste les paramètres de configuration, notamment la table de routage avec l'adresse du routeur comme routeur par défaut.

Router Solicitation



- Message ICMPv6 envoyé par un équipement à la configuration d'une interface
- Adresse source: Adresse Lien Local de l'interface
- Adresse destination : `ff02::2` (Groupe multicast *All-Routers*)
- Option habituellement ajoutée :
  - *Source link-layer address*: adresse physique (MAC) de l'interface
  - permet de peupler le cache des voisins du routeur

Router Advertisement



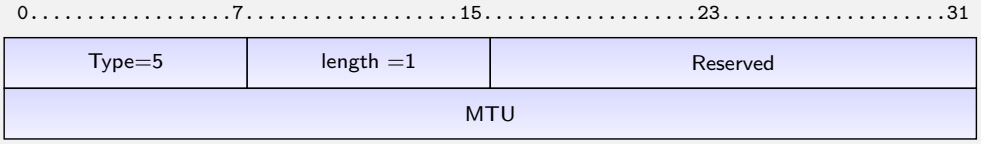
- Message ICMPv6 envoyé par un routeur en charge du réseau connecté sur son interface
  - soit en réponse à un message ICMPv6 *Router Solicitation*
  - soit périodiquement pour mettre à jour les compteurs de validité

Router Advertisement (suite)

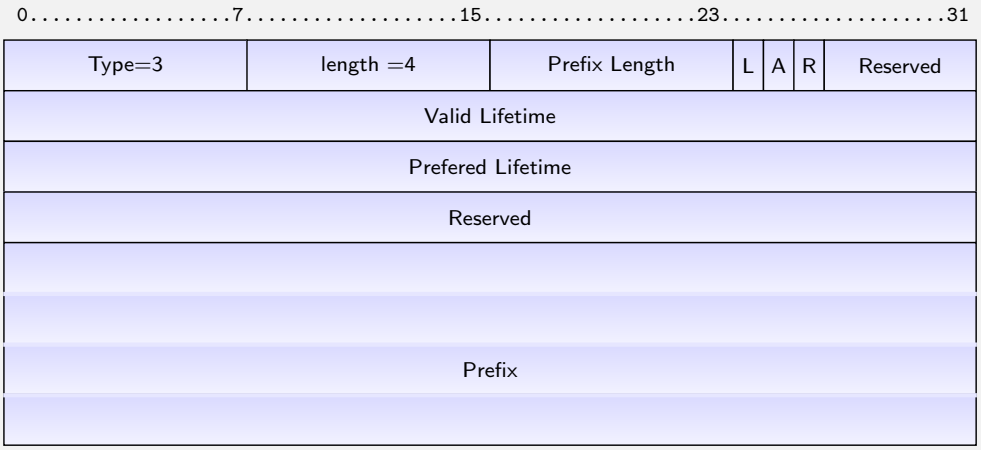
- Adresse source : Adresse lien-locale de l'interface du routeur
- Adresse destination :
  - Adresse lien-locale de l'équipement ayant solicité le routeur
  - Groupe multicast `ff02::1` pour les envois périodiques
- `Current Hop Limit`: Valeur à utiliser pour le champs `Hop Limit` d'IPv6
- `Flags`:
  - `M`: 1 = utiliser DHCPv6 pour la configuration de l'adresse
  - `O`: 1 = utiliser DHCPv6 pour d'autres paramètres (DNS)
  - `H`: 1 = ce routeur est un agent-mère (Mobilité IPv6 RFC 6275)
- `Router Lifetime`: Temps de vie du routeur
- `Reachable Time`: Temps en ms pour les entrées du cache des voisins
- `Retransmission Time`: Période in ms entre 2 messages RA

MTU, Prefix Information

MTU:



Prefix Information:

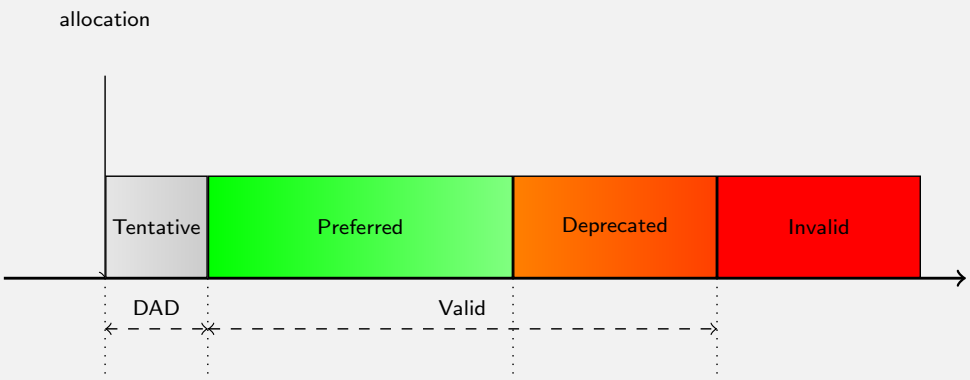


Etats d'une adresse IPv6

Une adresse IPv6 unicast locale ou globale possède un état pouvant évoluer au cours du temps

- état "DAD" : en cours de vérification de l'unicité
- état "préférée" : état opérationnel pour une communication
- état "dépréciée" : adresse en cours d'obsolescence
- état "invalide" : adresse interdite à l'utilisation

Des compteurs gèrent le passage d'un état à un autre



Exemple de configuration d'un routeur

```
interface Vlan5
 description reseau C5
 ip address 192.108.119.190 255.255.255.128
...
 ipv6 address 2001:660:7301:1::/64 eui-64
 ipv6 enable
 ipv6 nd ra-interval 10
 ipv6 nd prefix-advertisement 2001:660:7301:1::/64 2592000 604800\
 onlink autoconfig

ipv6 nd prefix-advertisement ipv6Prefix/ipv6PrefixLength validLifetime preferredLife
[ onlink ] [ autoconfig ]
```

Agenda

Auto-configuration avec état par DHCPv6

Pourquoi ?

- Contrôler des adresses configurées sur le réseau
- Fixer les adresses des équipements (pour enregistrement dans le DNS p.ex.)

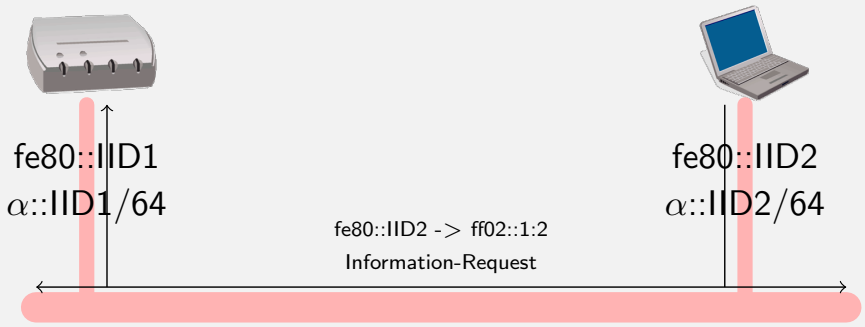
Comment ?

- Dynamic Host Configuration Protocol (DHCPv6) : évolution de DHCP pour IPv6
- Architecture Client / Serveur / Relai
- Peut être utiliser en complément de la configuration sans état

Principes

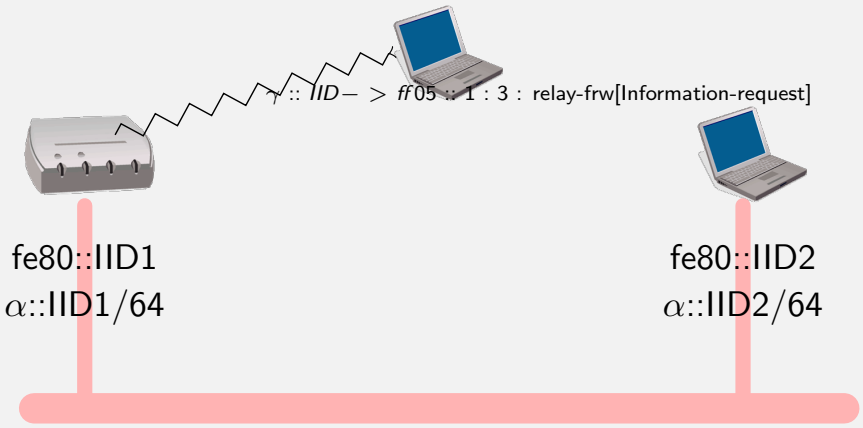
- Le serveur DHCPv6 maintient une liste d'adresses disponibles (*pool*)
- Le serveur DHCPv6 fournit à la demande adresses et paramètres de configuration
- A chaque rédémarrage, l'équipement obtient la même information.
  - Chaque client est identifié par son DUID (DHCPv6 Unique IDentifier)

DHCPv6 sans état RFC 3736



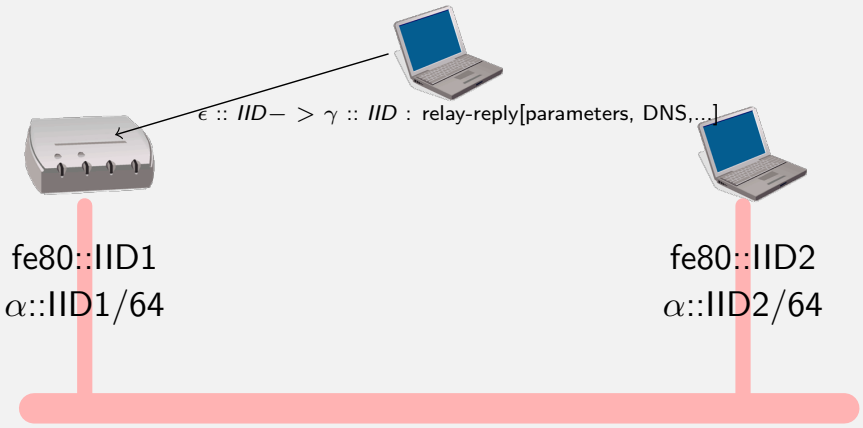
L'équipement n'a besoin que de paramêtres statiques (DNS, NTP,...). Il envoie une requête DHCPv6 *Information-Request* au groupe multicast local *All_DHCP_Agents*.

DHCPv6 sans état RFC 3736



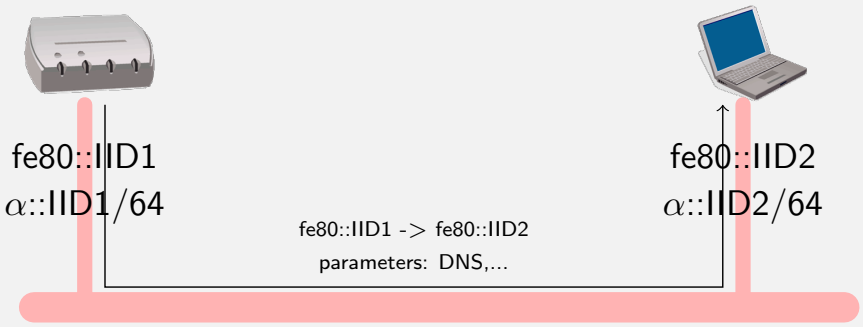
Un relai (généralement le routeur) encapsule cette requête dans un message *Relay-Forward* et l'envoie au groupe multicast site *All_DHCP_Servers*.

DHCPv6 sans état RFC 3736



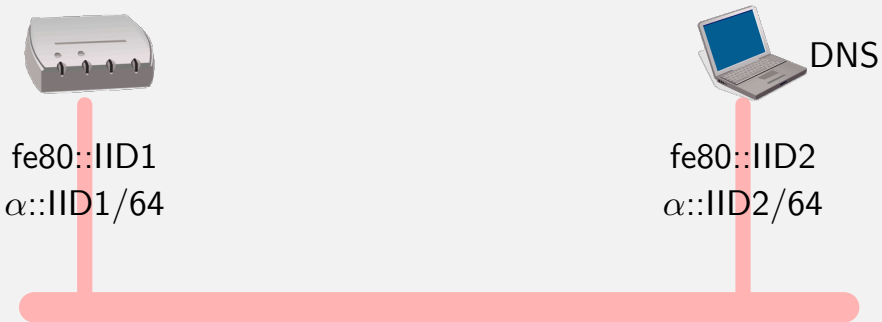
Le serveur interprête la requête encapsulée, encapsule la réponse dans un message *Relay-Reply* à destination du relai.

DHCPv6 sans état RFC 3736



Le relai extrait la réponse du serveur et la retransmet vers l'équipement.

DHCPv6 sans état RFC 3736



L'équipement est maintenant configuré.

Gestion des adresses par DHCPv6

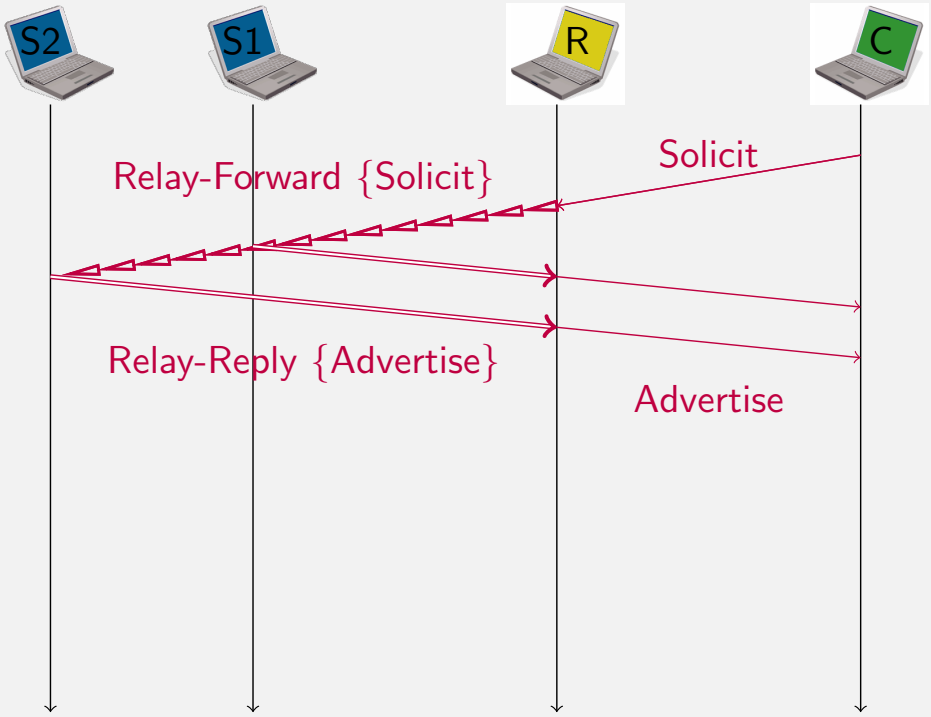
- Un échange de 4 messages permet la découverte du serveur DHCPv6 et l'allocation d'adresses :
  - Solicit : requête du client pour localiser les serveurs
  - Advertise : réponse des serveurs pour annoncer leurs ressources disponibles
  - Request : requête du client pour demander une allocation
  - Reply : réponse du serveur avec la ressource allouée
- Des messages permettent la gestion des allocations entre le client et le serveur :
  - Renew : requête du client pour renouveller l'allocation de la ressource
  - Rebind : requête du serveur pour étendre la durée d'allocation
  - Reconfigure : message du serveur signalant aux clients de nouvelles ressources disponibles
  - Release : requête du serveur pour libérer une ressource allouée
  - Decline : message d'information du client signalant que la ressource allouée n'est pas valide sur le réseau.

Identifiant DHCPv6

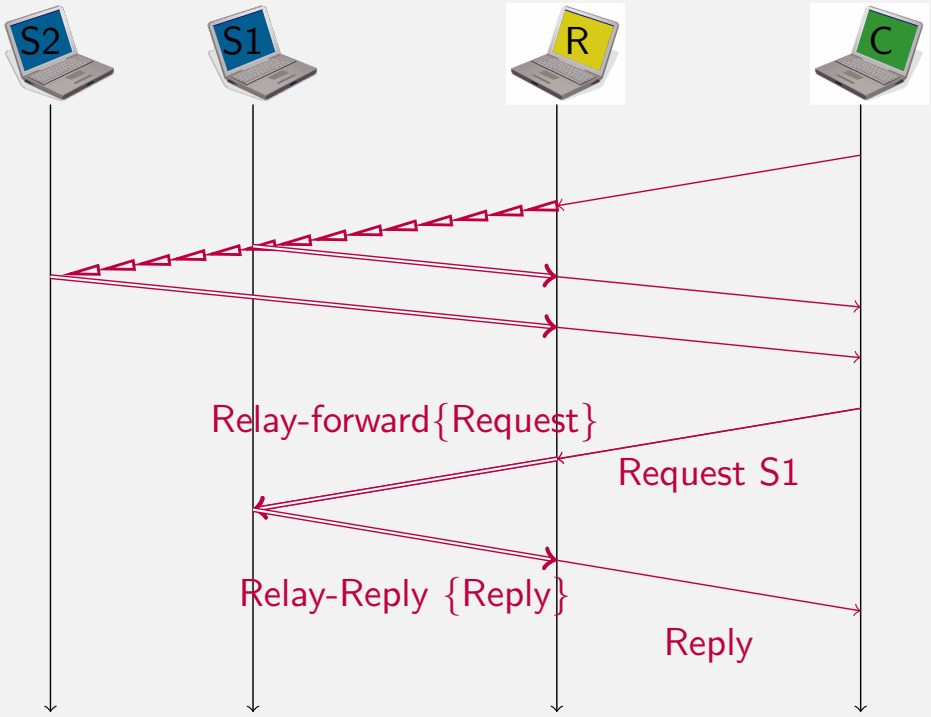
- DHCPv6 définit plusieurs types d'identifiants stables
- En utilisant le même identifiant, un équipement récupère les mêmes informations par DHCPv6
- DUID (DHCPv6 Unique IDentifier) :
  - Adresse de niveau 2
  - Adresse de niveau 2 + estampille temporelle
  - Identifiant spécifique au vendeur
  - Valeur fixée par l'administrateur
- Exemple sous Linux :

```
>od -x /var/db/dhcp6c_duid
0000000 000e 0100 0100 5d0a 5233 0400 9e76 0467
```

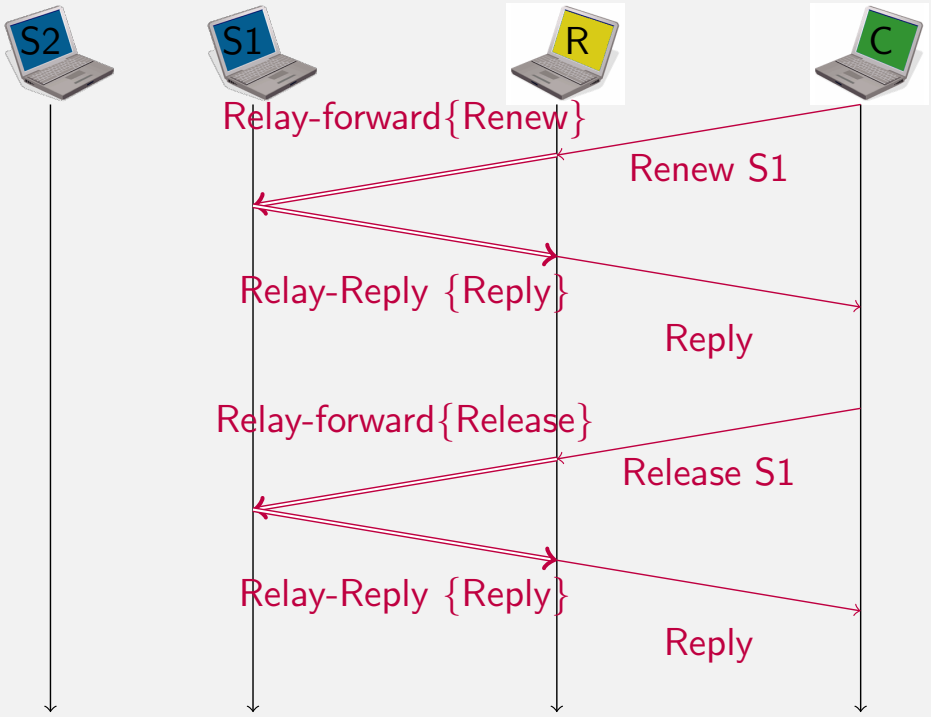
Allocation par DHCPv6



Allocation par DHCPv6



Allocation par DHCPv6



Auto-configuration: Avec ou sans état ?

Sans état (SLAAC)

Avantages :

- Configuration manuelle réduite
- Paramètres centralisés sur le routeur

Inconvénients :

- Pas de contrôle sur les adresses du réseau
- Problèmes de vulnérabilité

Avec état (DHCPv6)

Avantages :

- Contrôle des adresses sur le réseau
- Stabilité des adresses

Inconvénients :

- Nécessite une architecture supplémentaire
- Les messages RA toujours nécessaires

- sans état : terminaux utilisateurs, domestiques
- avec état : réseaux administrés, serveurs
- Si la sécurité est primordiale ⇒configuration statique !

Agenda

2 manières de considérer le DNS

Le DNS comme une application TCP/IP

- Le service DNS est accessible à travers différents transports (UDP/TCP) and différentes versions d'IP (v4/v6)
- L'intégration d'IPv6 pour l'accès au DNS est une étape cruciale du déploiement
- L'accès au service doit être uniforme en IPv4 et en IPv6 !

Le DNS comme une base de données

- Stocke différents types d'enregistrements (RR), notamment liés à des adresses IPv4 ou IPv6 : SOA, NS, A, AAAA, MX, PTR, TXT
- Les équipements et services IPv6 seront visibles lorsque leurs enregistrements seront inscrits dans le DNS
- Les données retournées par le DNS doivent être indépendantes de l'accès utilisé !

Publication de ressources IPv6 dans le DNS RFC 3596

Nouveau type d'enregistrement `AAAA` pour les adresses IPv6

Exemple pour un nom accessible en IPv4 et IPv6

```
foo.example.com.      A       192.0.2.1
                      AAAA    2001:db8:cafe:deca::1
```

Pour la résolution inverse, nouvelle racine `ip6.arpa`

Résolution inverse en IPv6

```
$ORIGIN a.c.e.d.e.f.a.c.8.b.d.0.1.0.0.2.ip6.arpa.
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0 IN PTR foo.example.com.
```

Transport du DNS sur IPv6

Le transport du DNS sur IPv6 n'est pas obligatoire pour résoudre des adresses IPv6 !
Activer l'accès IPv6 sur le serveur DNS

Exemple pour Bind :

```
listen-on-v6 { any; };
```

Support des systèmes d'exploitation :

- *BSD, MacOSX: OK
- Linux: OK
- Windows 7, 8, . . . : OK
- Android, iOS : OK

Découverte du serveur de nom récursif

Le serveur de nom récursif est en charge de traiter les requêtes de résolution pour son réseau

En IPv4 cette information est :

- Soit configurée manuellement (p.ex. dans `/etc/resolv.conf` sous Unix)
- Soit découverte par DHCPv4

En IPv6 : RFC 4339 (IPv6 Host Configuration of DNS Server Information Approaches)

- Par DHCPv6 (avec état): RFC 3315
- Par DHCPv6 (sans état): RFC 3736, *DHCPv6-light*
- Par les messages RA: RFC 8106 ("IPv6 Router Advertisement Options for DNS Configuration")
- Par configuration manuelle
- aussi par DHCPv4 !

Recommandations pour l'expoitation d'un serveur DNS en IPv6

RFC 3901: "DNS IPv6 Transport Operational Guidelines"

- Afin d'assurer la continuité de service entre IPv4 et IPv6, le service DNS doit être accessible en double-pile
- Les zones DNS doivent être servies par au moins un serveur d'autorité accessible en IPv4 → Eviter les serveurs IPv6-only

A garder à l'esprit :

- Le service DNS doit rester universel pendant la phase de transition IPv4 vers IPv6

RFC 4472 "Operational Considerations and Issues with IPv6" :

- Effets de bord entre les serveurs DNS et les Load-balancers
- Problèmes d'accessibilité avec des adresses IPv6 restreintes
- IPv6 et DNS dynamique (RFC 2136)

Agenda

Considérations sur la sécurité en IPv6

Du point de vue d'un attaquant, compromettre un réseau IPv6 est :

- **Difficile** depuis l'extérieur :
  - Scan de réseau très couteux (2^{64} adresses)
    - ⋆ difficulté accrue avec les IID aléatoires
  - pas d'adresse de diffusion
  - Les attaques de l'extérieur vont concerner les serveurs inscrits dans le DNS
- **Facile** depuis le réseau local:
  - Le protocole de découverte des voisins n'est pas sécurisé
  - Vulnérabilités similaires à celles d'ARP
  - IPv6 est récent : des failles spécifiques peuvent apparaitre

Il existe déjà des outils exploitant ces failles !

See `http://www.thc.org/thc-ipv6/`

Exemples d'attaques utilisant ND

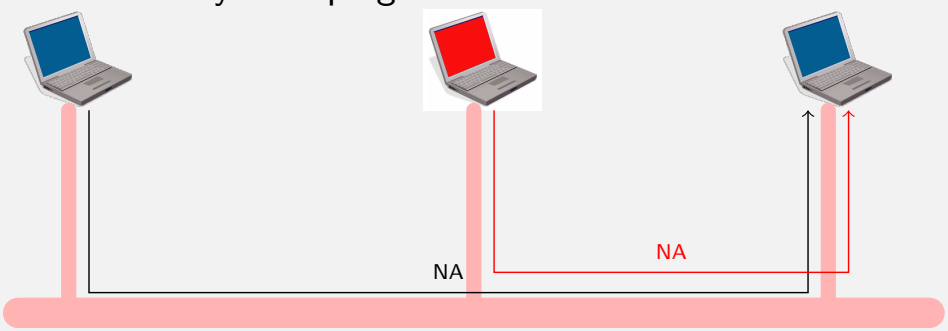
Neighbor Discovery Snooping

Utilisation de la découverte des voisins pour :

- Résoudre l'adresse de niveau 2 à partir d'une adresse IPv6
- Vérifier l'unicité d'une adresse IPv6

Exemples d'attaques utilisant ND

Neighbor Discovery Snooping

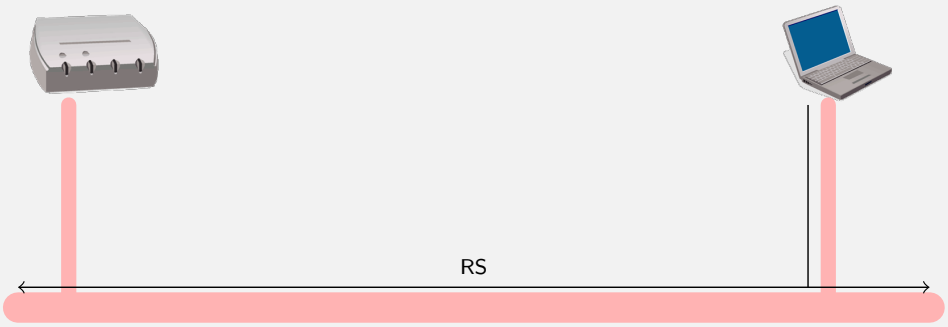


Un attaquant sur le réseau peut exploiter la découverte des voisins :

- Usurper l'adresse de niveau 2 => Man in the Middle
- Empêcher la configuration automatique => Déni de service

Exemples d'attaques utilisant ND

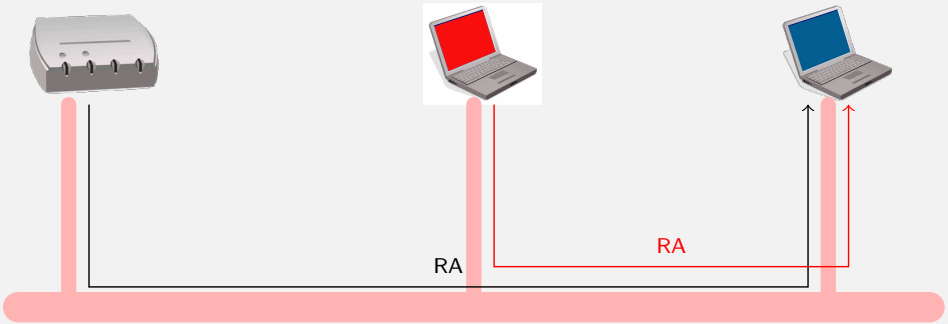
Routeur malicieux (Rogue router)



La découverte du routeur est utilisée pour obtenir l'adresse du routeur par défaut et obtenir le préfixe du sous-réseau

Exemples d'attaques utilisant ND

Routeur malicieux (Rogue router)



Un attaquant peut exploiter la découverte du routeur pour :

- Se faire passer pour le routeur du réseau => Man in the Middle
- Annoncer un préfixe différent sur le réseau => Déni de Service

Exemple : Annonces sur le réseau d'une conférence IETF

```
en3: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
   inet6 fe80::223:6cff:fe97:679c%en3 prefixlen 64 scopeid 0x6
   inet6 2002:8281:1c8c:d:223:6cff:fe97:679c prefixlen 64 autoconf
   inet6 2002:c15f:2011:d:223:6cff:fe97:679c prefixlen 64 autoconf
   inet6 fec0::d:223:6cff:fe97:679c prefixlen 64 autoconf
   inet6 2001:df8::24:223:6cff:fe97:679c prefixlen 64 autoconf
   inet 130.129.28.215 netmask 0xfffff800 broadcast 130.129.31.255
   inet6 2002:8281:1ccb:9:223:6cff:fe97:679c prefixlen 64 autoconf
   inet6 fec0::9:223:6cff:fe97:679c prefixlen 64 autoconf
   ether 00:23:6c:97:67:9c
   media: autoselect status: active
   supported media: autoselect
```

Solutions pour mitiger ou prévenir ces attaques ?

Prévention :

- SEND (Secure Neighbor Discovery)
  - Solution proposée par l'IETF: RFC 3971
  - Utilise des messages ND signés avec des certificats de confiance
  - Cependant solution complexe à déployer sur tous les sites
- Fitrage au niveau 2
  - Filtrage des messages ND au niveau des ports des switchs
  - P.ex. un seul port est autorisé à diffuser des RAs
  - Les switchs récents offrent cette fonctionnalité (*MLD snooping*: Multicast Listener Discovery)

Détection des attaques par écoute du réseau

- Similaire à ARP-watch
- Détection des usurpations et dénis de services
- ndpmon : `http://ndpmon.sf.net`

Contrôle de l'accès au réseau

Beaucoup de sites utilisent l'allocation d'adresse par DHCP pour contrôler l'accès au réseau

- Adresse MAC connue : adresse allouée, accès au réseau
- Adresse MAC inconnue : pas d'adresse allouée

Mauvaise stratégie !

- Accès niveau 2 contrôler par le niveau 3
- Utilisation de l'adresse IP comme identifiant de l'utilisateur
- Problèmes de sécurité inhérents !

Le contrôle de l'accès au réseau (niveau 2) doit être fait au niveau 2 !

- 802.1x pour les réseaux Ethernet filaires
- 802.11i pour les réseaux Ethernet sans fil

Concept de filtre pare-feu

- Pare-feu : équipement en coupure de différents réseaux
- Rôles d'un pare-feu
  - Filtrer les paquets selon les politiques de sécurité
  - Router les paquets entre les différentes zones (in/out/DMZ)
- Que change IPv6 ?
  - Nouvelles règles pour IPv6
  - Routage IPv6

Règles de filtrage sur les adresses IPv6

- Filtrage des adresses selon la portée
- Voir RFC 6890
- Adresses devant être filtrées :
  - Unicast Lien-local (`fe80::/10`)
  - Adresse de bouclage (`::1`)
  - Multicast de portée autre que global comme source ou destination
  - Multicast global comme source
  - Unicast local (en bordure de site)
  - Adresses IPv4 compatibles

Autres règles de filtrage IPv6

- ICMPv6 ne doit pas être filtré de la même manière qu'IPv4
  - RFC 4890 ("Recommendations for Filtering ICMPv6 Messages in Firewalls")
  - Un filtrage trop zélé peut perturber le fonctionnement d'IPv6
- Les extensions IPv6 doivent être prise en compte
  - Doivent être autorisées : Fragmentation, IPSec
  - A considérer avec attention : Hop-by-Hop, Destination (IPv6 Mobility), Routing
- Des règles de filtrages avec état permettent de bloquer les connexions entrantes
- Attention aux tunnels IPv6 sur IPv4 pouvant être des portes dérobées

Définition des zones de sécurité

- IPv6 introduit une seconde topologie à gérer
- Les zones de sécurité doivent être cohérentes entre IPv4 and IPv6
  - Un équipement doit être placé au même niveau de sécurité en IPv4 et en IPv6
- Les règles IPv4 et IPv6 doivent être cohérentes pour une même zone
  - Les mêmes restrictions doivent s'appliquer
  - Une différence peut compromettre les deux protocoles

Agenda

Agenda

Modèles de communication

Qui communique avec qui à travers quel réseau ? (6 possibilités)

1. Equipement IPv4 ⇔ Equipement IPv4 à travers réseau IPv4
2. Equipement IPv6 ⇔ Equipement IPv6 à travers réseau IPv6
3. Equipement IPv4 ⇔ Equipement IPv4 à travers réseau IPv6
4. Equipement IPv6 ⇔ Equipement IPv6 à travers réseau IPv4
5. Equipement IPv4 ⇔ Equipement IPv6
6. Equipement IPv6 ⇔ Equipement IPv4

Rappel:

IPv4 et IPv6 ne sont pas directement interopérables

Complexité croissante . . .

- 1) & 2) : Cas d'usage évidents
- 3) & 4) : Cas moins évidents, mais les solutions existent
- 5) & 6) : Cas complexes, les solutions impactent les communications

Classification des mécanismes de transition/intégration

- Communications IPv6-IPv6 ou IPv4-IPv4
  - Double pile: v4 and v6 disponibles à chaque extrémité
- Tunnel
  - Communication IPv6 à travers un réseau IPv4 (et vice & versa)
  - Tunnel : lien point-à-point encapsulant les packets IPv6 dans des paquets IPv4 afin de les transporter dans le réseau IPv4
- Interopérabilité IPv4/IPv6
  - Traduction
    - ★ En-tête / Protocole / Port (v6→v4 and v4→v6)
    - ★ Sans état *vs* Avec état
  - Relais / Passerelles applicatives (ALG: Application Level Gateway)

Plus de détails

"IPv6, passeport pour l'Internet du futur" - AFNIC:
`https://www.afnic.fr/medias/afnic-dossier-ipv6-2011-05.pdf`

Scénarios de Transition/Intégration (Groupe IETF v6ops)

- Opérateurs / Fournisseurs d'accès (RFC 6036)
  - Cœur de réseau (configuration, *peering*)
  - Réseaux d'accès (connectivité & services pour les clients)
- Réseaux d'entreprise (administrés) (RFC 7381)
  - Mise en œuvre d'un réseau et d'un système d'information IPv6
  - Scénario d'intégration incrémentale (accès, réseau, services)
- Réseaux non-administrés (SOHO, réseau domestique) (RFC 7084)
  - IPv6 dans les routeurs domestiques
  - Auto-configuration des services
- Réseaux cellulaires (3GPP/LTE) (RFC 6459)

Agenda

IPv6 pour les réseaux administrés

Motivations :

- Pas nécessairement l'épuisement des adresses IPv4
- Fournir un accès à l'internet IPv6 / Offrir des services IPv6
- Montée en compétences sur les nouvelles technologies Internet
- Le déploiement d'IPv6 peut s'effectuer à l'occasion d'une refonte du réseau

Objectif :

- Déploiement double-pile : Intégration progressive d'IPv6 sans perturber IPv4
- A la fin du déploiement, même services en IPv6 qu'en IPv4

Les problèmes peuvent survenir :

- Disponibilité des ressources humaines et financières
- Montée en compétences : IPv6 est un changement majeur à plusieurs niveaux

Agenda pour l'intégration d'IPv6 dans les réseaux administrés

- Anticiper : Compatibilité IPv6 dans les appels d'offre pour les équipements et logiciels
  - Document RIPE 554 (`http://www.ripe.net/ripe/docs/current-ripe-documents/ripe-554`)
- Obtenir une connectivité IPv6 (et un préfixe global)
- Définir un plan d'adressage
- Intégrer IPv6 progressivement dans l'infrastructure, les serveurs et les postes clients
- Ajouter la compatibilité aux services (applications client/serveur)
- Sécuriser en parallèle le réseau IPv6
- Superviser l'usage d'IPv6
- Apporter des améliorations de manière continue

Définition d'un plan d'adressage

Un site obtient habituellement un préfixe /48
Comment définir les 16 bits du SID ? Plusieurs solutions . . .

- Priorité au routage
  - Agréger les préfixes utilisés sur une même site géographique
  - Agrégation utilisée dans les tables de routage
- Priorité au filtrage
  - Agréger les préfixes utilisés pour les mêmes usages
  - Agrégation utilisée dans les règles de filtrage
- Solutions mixtes
  - Une partie du plan pour des tests
  - Une autre partie pour le déploiement final
  - Voir aussi: `https://www.ripe.net/publications/ipv6-info-centre/deployment-planning/create-an-addressing-plan`

Un plan d'adressage n'est pas définitif.
Ne pas craindre la renumérotation !

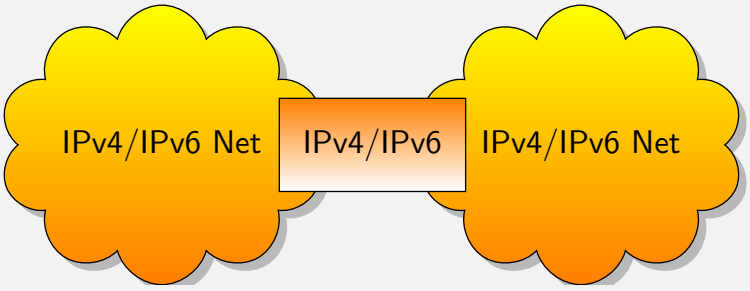
Exemple du plan d'adressage pour une université

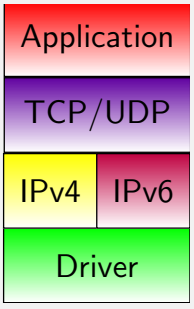
4bits : Communités	**8bits**	**4bits**
0 : Infrastructure	*Adresses spécifiques*	
1 : Tests	*Adresses spécifiques*	
6 : Point6	*Gérer par Point6*	
8 : Wifi invités	*Adresses spécifiques*	
A : Employés	Entités géographiques	sous réseaux
E : Students	Entités géographiques	sous réseaux
F : Autre (Start up, etc.)	*Adresses spécifiques*	

- Les règles de filtrage utilisent les préfixes avec les 4 premiers bits en commun
- Les règles de routage se basent sur les préfixes géographiques
- Compromis :
  - Une règle de filtrage pour une même communité sur différents sites
  - Plusieurs règles de routage pour un même site (une par communauté)

Approche Double pile RFC 4213

- IPv4 et IPv6 déployés sur le même équipement
- Idéal pour le déploiement d'IPv6 sur les réseaux existants
  - Le déploiement d'IPv6 en parallèle ne perturbe pas le fonctionnement d'IPv4
  - Les communications migrent vers IPv6 lorsque clients et serveurs deviennent compatibles





- Cependant...
  - Au moins une adresse IPv4 est toujours nécessaire pour chaque équipement
  - $\Rightarrow$ Cette solution ne permet pas de résoudre le problème d'épuisement
  - $\Rightarrow$ Cette solution nécessite une double supervision des 2 protocoles

Support d'IPv6 dans les services

Pour être fonctionnel en IPv6, un service nécessite le support à plusieurs niveaux :

- Réseau d'accès
- Système d'exploitation
- Application (côtés serveur et client)
- DNS (publication des adresses IPv6)
- Supervision

Les applications nécessitent un support explicite d'IPv6

- Les librairies réseau doivent intégrer les nouvelles API IPv6
- L'approche double-pile peut impacter la représentation de certaines données

Déployer IPv6 sur l'infrastructure réseau

Sur les différents supports

- Filaire (Ethernet, VLANs) : pas de problème
  - Les switches doivent accepter le protocole Ethernet 0x86DD
  - Certains anciens switches peuvent poser problème avec le multicast Ethernet
- Sans-fils
  - 802.11: pas de problème
  - UMTS/LTE: Intégré dans les nouvelles versions 3GPP

Sur les équipements réseaux (routeurs)

- La grande majorité est aujourd'hui compatible IPv6
- Si cas bloquant, utilisation de tunnels pour les traverser
- *Dual stack where you can ; tunnel where you must.*

Support d'IPv6 dans les systèmes d'exploitation

Pas de problème à signaler
Microsoft Windows:

- <= XP: Forget it...
- Windows 7, 8, ...: OK

Unixes:

- *BSD, MacOSX: OK
- Linux: OK

Mainframe OSes: HPUX, AIX OK
OSes embarqués : Android, iOS OK

Support d'IPv6 dans les applications

Web

- Serveur: Apache, IIS (MS)
- Clients: Firefox, Edge, Safari, Opera, Chrome

Messagerie

- MTA: Sendmail, Postfix, Exim
- MUA: Thunderbird, Mail.app, Outlook ...

Bases de données

- MySQL, PostgreSQL, Oracle (11g, R2)

Voix sur IP

- Asterisk

Application Inventory (work in progress)

`http://www.ipv6-to-standard.org/`

Sécuriser le déploiement d'IPv6

Qu'est ce qui ne change pas :

- Pare-feu sans état
- Pare-feu avec état, pour filtrer les connexions entrantes

Qu'est ce qui change:

- Filtrage d'ICMPv6 : attention à ne pas perturber IPv6
- Extensions de l'en-tête IPv6

Support d'IPv6 dans les pare-feu

- Cisco : PIX OS7, IOS 12.4 AdvancedIP (extended ACL)
- BSD Packet Filter
- Linux Netfilter

Superviser l'usage d'IPv6

La supervision d'IPv6 est importante pour :

- Voir l'impact du déploiement d'IPv6
- Vérifier la qualité de service en IPv4 et IPv6

Outils

- Traffic: MRTG/Cacti, Netflow v9...
- Services: Nagios, Zabbix...

Double pile implique double supervision !

L'accessibilité des services doit être vérifiée en IPv4 et IPv6

En résumé...

- L'approche double pile est plus l'intégration d'IPv6 qu'une migration
- L'intégration est progressive dans la chaine de communication du système d'information
- La double pile ne résoud pas le problème d'épuisement
  - Des réseaux IPv6-only vont apparaitre
- La double pile peut avoir un impact sur les performances
  - Supervision cruciale des services

Agenda

Opérateurs de cœur de réseau

- Transporter IPv6 aussi vite qu'IPv4
- Problème : des routeurs ne traitent pas IPv6 de manière optimale
  - Performances dégradées
- Les tunnels ne sont pas une solution satisfaisante
  - Encapsulation réduit les performances
- MPLS offre des solutions de transition
  - L2VPN
  - 6PE
  - 6VPN
- Quelques opérateurs ont le problème inverse :
  - Transporter IPv4 sur un cœur de réseau IPv6
  - Maillage avec des liens logiques

Fournisseur d'accès

Assignation et gestion de préfixes IPv6

- Délégation de préfixes IPv6 à ses clients
- La gestion des adresses IPv4 doit évoluer vers la gestion de préfixes IPv6

Transport d'IPv6 : différentes solutions:

- Transport natif
- Tunnels: 6to4, 6rd, Tunnel Broker, Softwire...

Les fournisseurs d'accès doivent assurer la continuité du service IPv4 et mettre au point leur stratégie de sortie

Obtenir une connectivité IPv6

Fournisseur d'accès natif

- En France : Orange, RENATER
- A Maurice : Mauritius Telecom
- A réclamer auprès de votre opérateur !

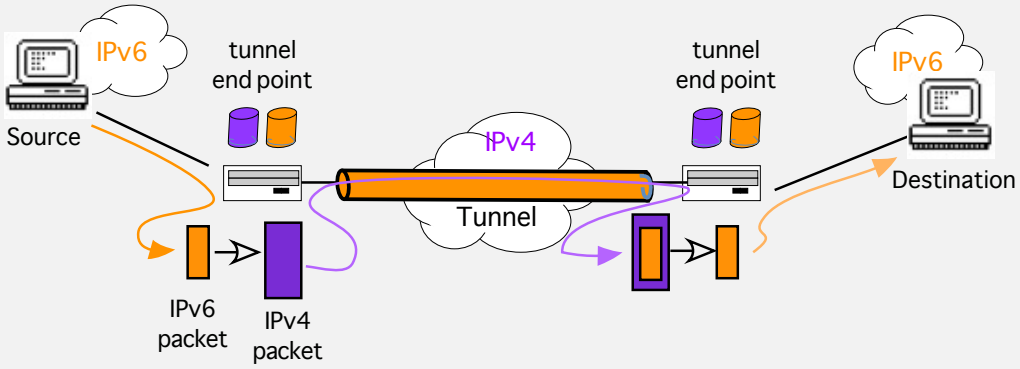
Accès par tunnel

- par l'opérateur (6rd, Softwire)
- par un opérateur tiers (Tunnel Broker)

Tunnels : Principe

Cas d'usage : Connecter 2 réseaux IPv6 à travers un réseau IPv4

- Demande des routeurs compatibles IPv6 à chaque extrémité
- Impact de l'encapsulation : MTU réduite, performances
- Rappel : le tunnel reste une solution temporaire

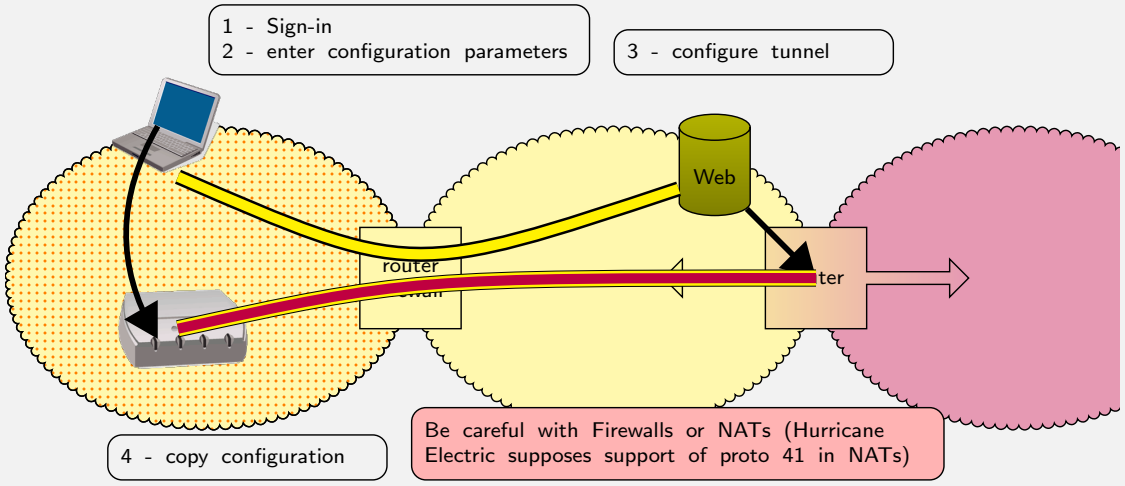


Tunnel Broker RFC 3053

Un *tunnel broker* fournit des tunnels IPv6 vers des sites ou clients finaux
Tunnel Broker = 6over4 avec authentification de l'utilisateur

- Configuration des paramètres du tunnel par Tunnel Setup Protocol (TSP)
  - TSP (RFC 5572): messages XML contenant les paramètres du tunnel (authentification, délégation de préfixe, etc.)
- Délégation et routage de préfixes à travers le tunnel
- Encapsulation IPv6/IPv4 ou IPv6/UDP/IPv4 si besoin de traverser un NAT
- Plusieurs fournisseurs de tunnels : Freenet6, HE, SixXS...
  - Liste : `https://en.wikipedia.org/wiki/List_of_IPv6_tunnel_brokers`

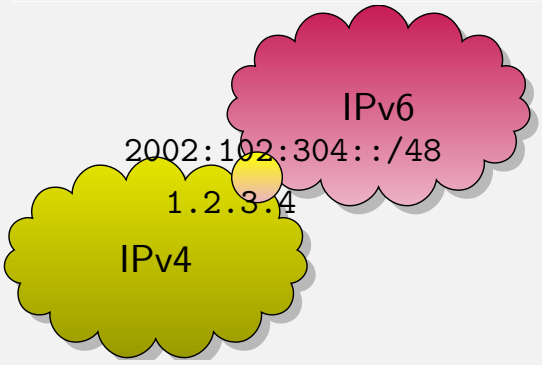
Tunnel Brokers



6to4 RFC 3056

Objectifs:

- Interconnexion de sites IPv6 à travers l'infrastructure IPv4
- Connexion de sites IPv6 à l'Internet IPv6 à travers IPv4



- Configuration automatique d'un tunnel IPv6
- Transport utilisant l'encapsulation 6over4
- Adressage IPv6 des points d'entrée dérivé des adresses IPv4

Problèmes et limitations de 6to4 RFC 3964

Problème de sécurité

- Les relais 6to4 doivent accepter tous trafics 6to4
- Les relais 6to4 sont sujets à déni de service

Performances

- Long chemin vers les relais
- 6to4 engendre du routage assymétrique

6to4 n'est pas considéré comme une solution viable globalement
IPv6 Rapid Deployment (6rd) est une adaptation de 6to4 à l'échelle d'un opérateur

6rd: 6to4 à l'échelle d'un opérateur (RFC 5569, RFC 5969)

Objectif :

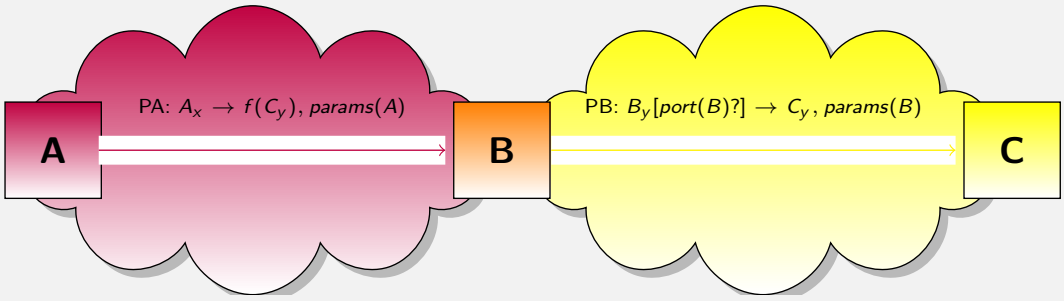
Déployer IPv6 pour un fournisseur d'accès avec un impact minimal sur l'infrastructure

6rd ré-utilise les principes de 6to4 :

- Les préfixes IPv6 des clients sont construits à partir de leur adresse IPv4
  ⇒ ne nécessite pas pour l'opérateur un plan d'adressage spécifique
- Encapsulation 6over4
  ⇒ pas de migration du réseau de collecte
- Relai 6rd vers l'infra IPv6 native de l'opérateur
  ⇒ seul équipement supplémentaire à déployer

Agenda

Approche générique pour la traduction



- $(x, y) \in \{(6, 4), (4, 6)\}$
- A est IPv_x-only, C est IPv_y-only
- A envoie un paquet PA vers C
  - Adresse source : A_x
  - Adresse destination : $C_x = f(C_y)$ (IPv_x mappée sur C_y)
- Packet PA est intercepté par B, qui assure la traduction entre IPv_x et IPv_y
- Packet PA est traduit en paquet PB retransmis vers C
  - Adresse source : B_y
  - Destination address: C_y

Traduction IPv6/IPv6 par NAT64 + DNS64

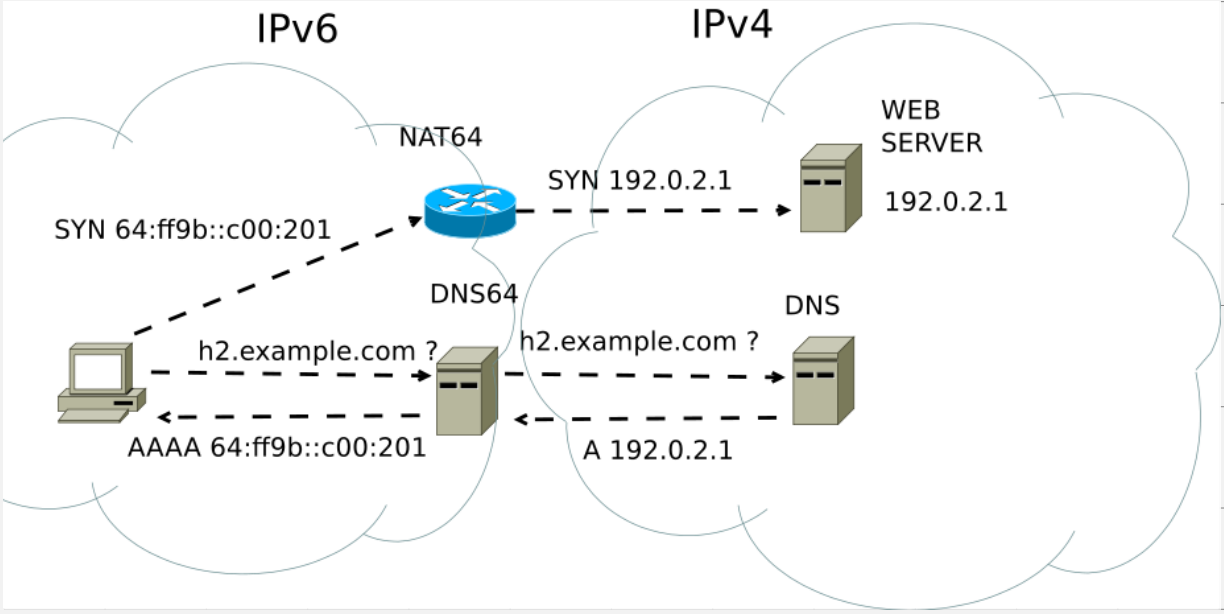
2 protocoles complémentaires :

- RFC 6146: "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers"
  Spécifie le mécanisme de traduction des en-têtes IPv6 vers IPv4 et inversement
- RFC 6147: "DNS64: DNS extensions for Network Address Translation from IPv6 Clients to IPv4 Servers"
  Spécifie le mécanisme de traduction des enregistrements DNS d'IPv4 vers IPv6

Pour plus de détails :

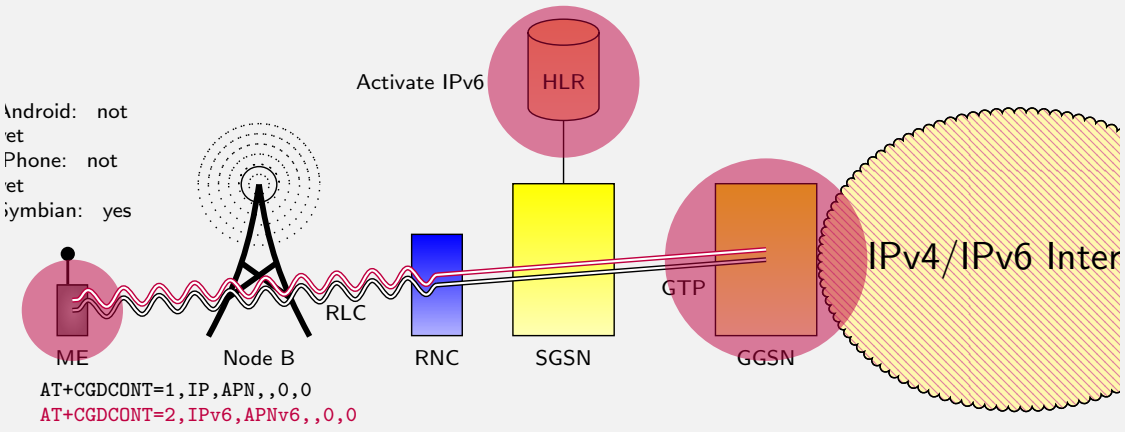
Bagnulo, M., Garcia-Martinez, A. et Van Beijnum, I.. (2012). IEEE Communications Magazine, July, 50(7).
The NAT64/DNS64 Tool Suite for IPv6 Transition.
http://dx.doi.org/10.1109/MCOM.2012.6231295

Fonctionnement NAT64 + DNS64



Source: http://commons.wikimedia.org/wiki/File:NAT64.svg

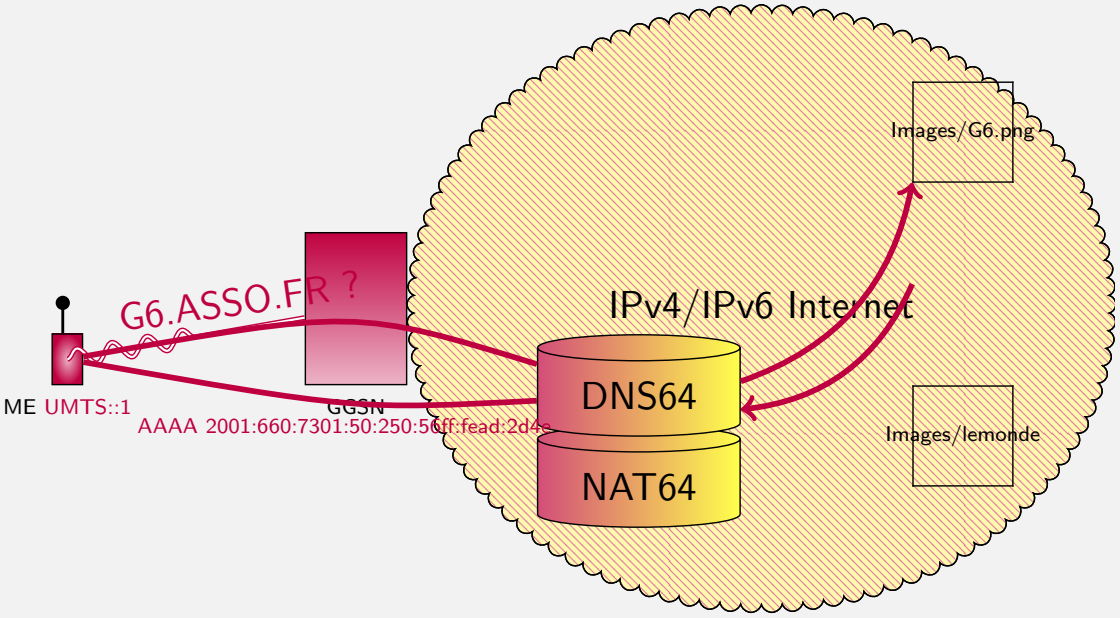
NAT64 sur les réseaux cellulaires



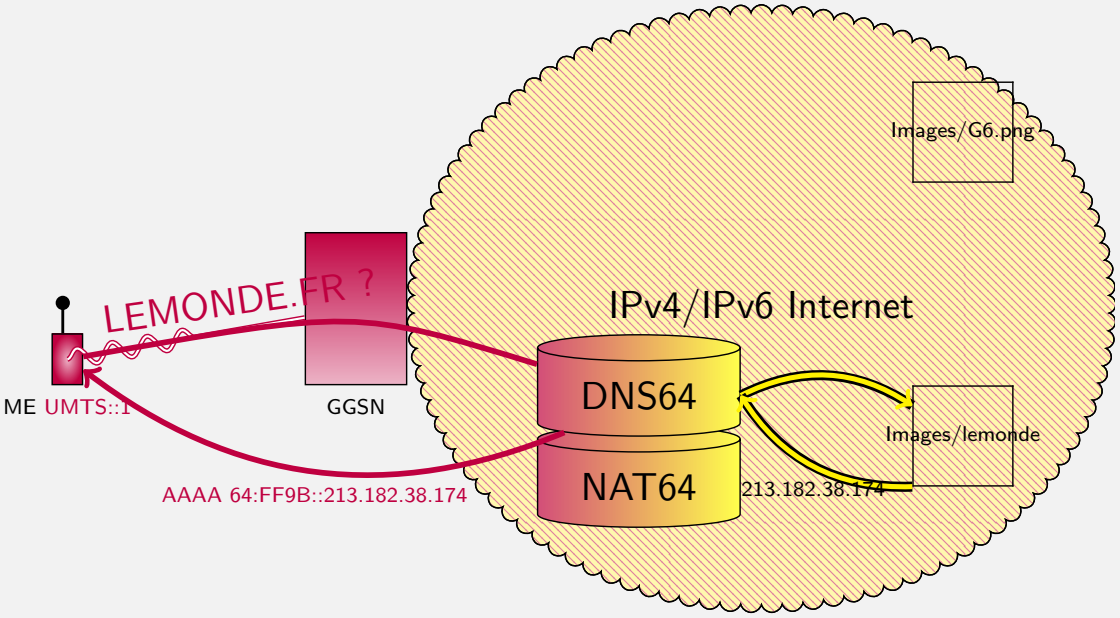
Keep only IPv6, but translate to IPv4 when needed

ME: Mobile Equipment, **RNC**: Radio Network Controller, **SGSN**: Serving GPRS Support Node, **GGSN**: Gateway GPRS Support Node, **HLR**: Home Location Register, **GTP**: GPRS Tunnelling Protocol **RLC**: Radio Link Control

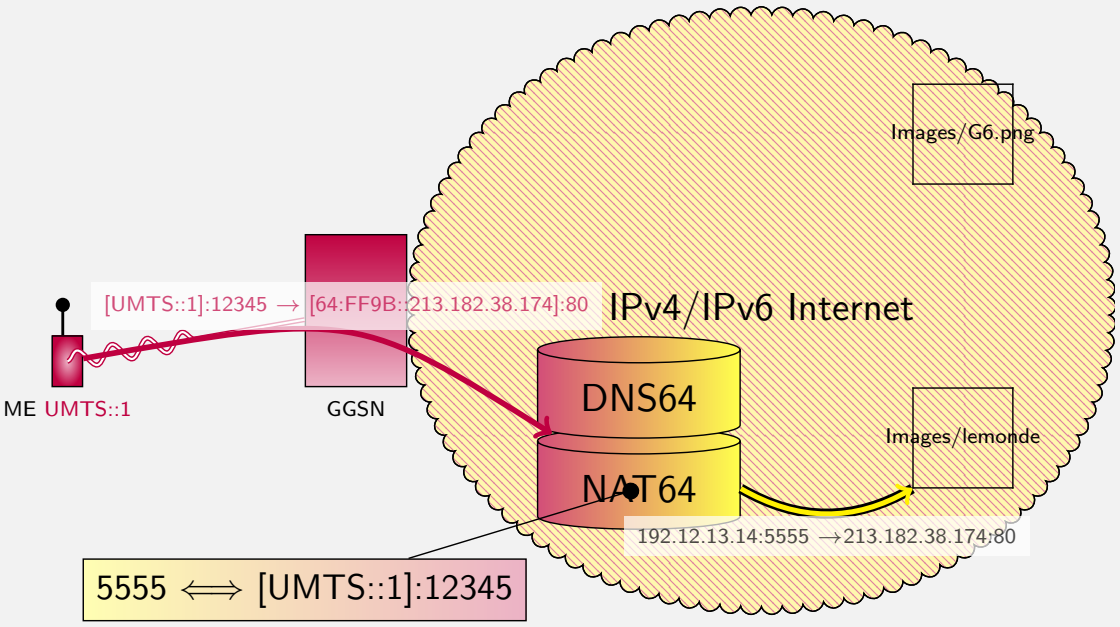
NAT64 sur les réseaux cellulaires



NAT64 sur les réseaux cellulaires

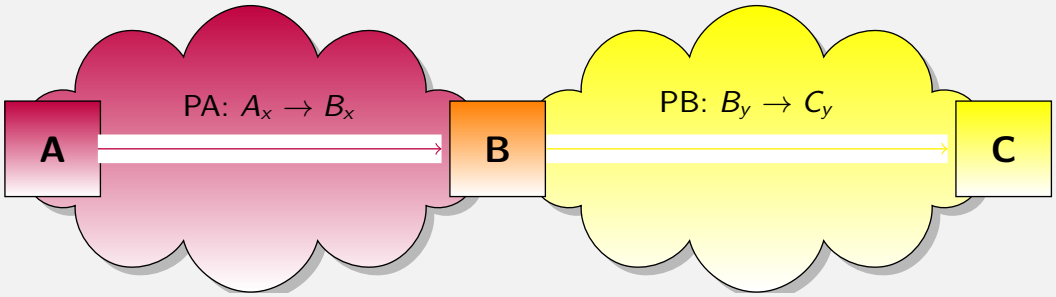


NAT64 sur les réseaux cellulaires



Agenda

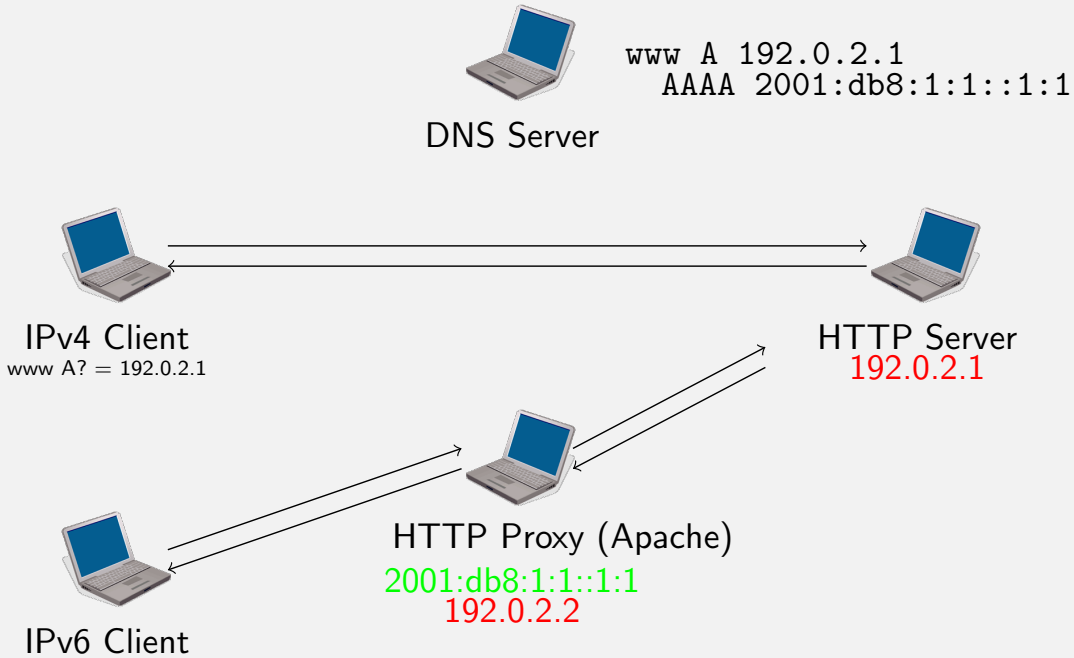
Approche générique des passerelles (ALG ou proxy)



- $(x, y) \in \{(6, 4), (4, 6)\}$
- A est un client IPv_x-only ; C est un serveur IPv_y-only
- A envoie vers B un paquet PA contenant une requête pour C
  - Adresse source : A_x
  - Adresse destination : B_x
- B est un proxy double-pile IPv_x et IPv_y
- B renvoie la requête vers C dans un **nouveau paquet** PB
  - Adresse source : B_y
  - Adresse destination : C_y
- Exemples : proxy web/ftp/DNS/mail...

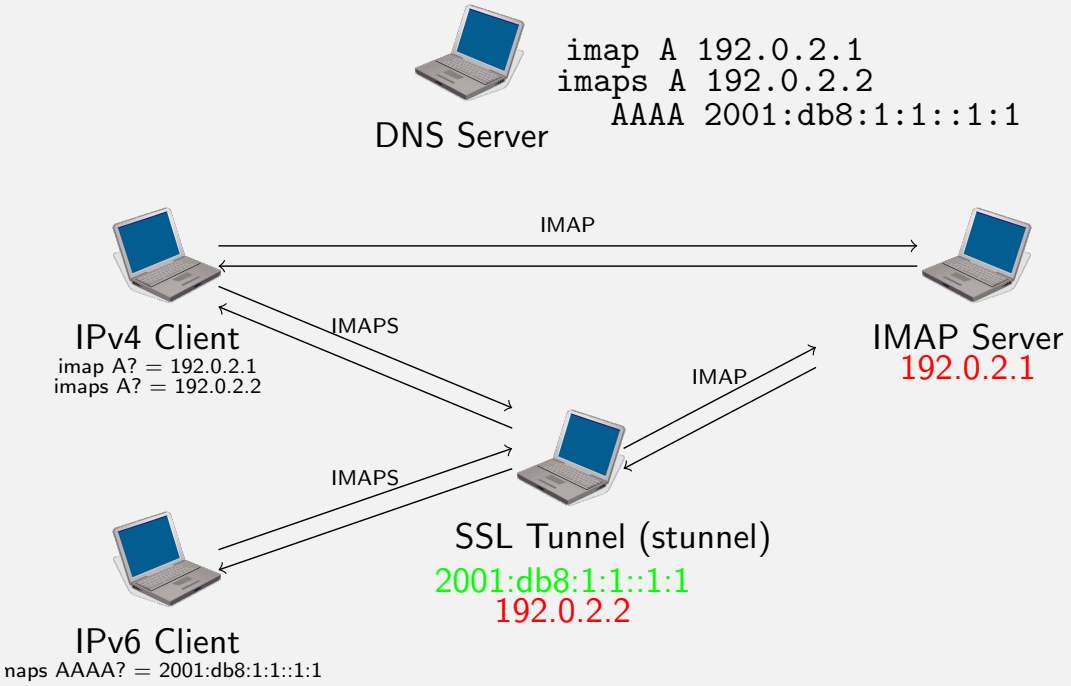
Exemple de passerelle Web

Activer l'accès IPv6 à un serveur Web en production



Exemple de passerelle SSL

Activer l'accès IPv6 et SSL à un serveur Mail en production



Agenda

Agenda

Support d'IPv6 dans les services

Pour être fonctionnel en IPv6, un service nécessite le support à plusieurs niveaux :

- Réseau d'accès
- Système d'exploitation
- Application (côtés serveur et client)
- DNS (publication des adresses IPv6)
- Supervision

Les applications nécessitent un support explicite d'IPv6

- Les librairies réseau doivent intégrer les nouvelles API IPv6
- L'approche double-pile peut impacter la représentation de certaines données

Support d'IPv6 dans les applications

Web

- Serveur: Apache, IIS (MS)
- Clients: Firefox, Edge, Safari, Opera, Chrome

Messagerie

- MTA: Sendmail, Postfix, Exim
- MUA: Thunderbird, Mail.app, Outlook ...

Bases de données

- MySQL, PostgreSQL, Oracle (11g, R2)

Voix sur IP

- Asterisk

Application Inventory (work in progress)

`http://www.ipv6-to-standard.org/`

Impact de la double pile sur les services

Règle :

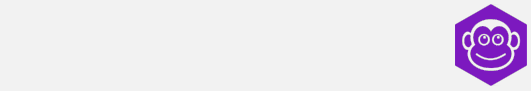
Un client double-pile se connectant à un serveur double-pile utilisera en priorité IPv6

- Dans le cas où les performances d'IPv6 sont dégradées chez les clients
- Que ce passe-t'il quand un service active son accès IPv6 ?
  - Les connexions sont lentes pour les utilisateurs IPv6
  - Les clients iront chez les concurrents IPv4-only ...
- Et si l'accès IPv6 est en panne chez le client
  - L'accès démarre en IPv6 et après un long timeout, bascule en IPv4
  - L'expérience utilisateur est fortement impactée
- *Happy Eyeballs* : Connexion IPv4 and IPv6 en parallèle

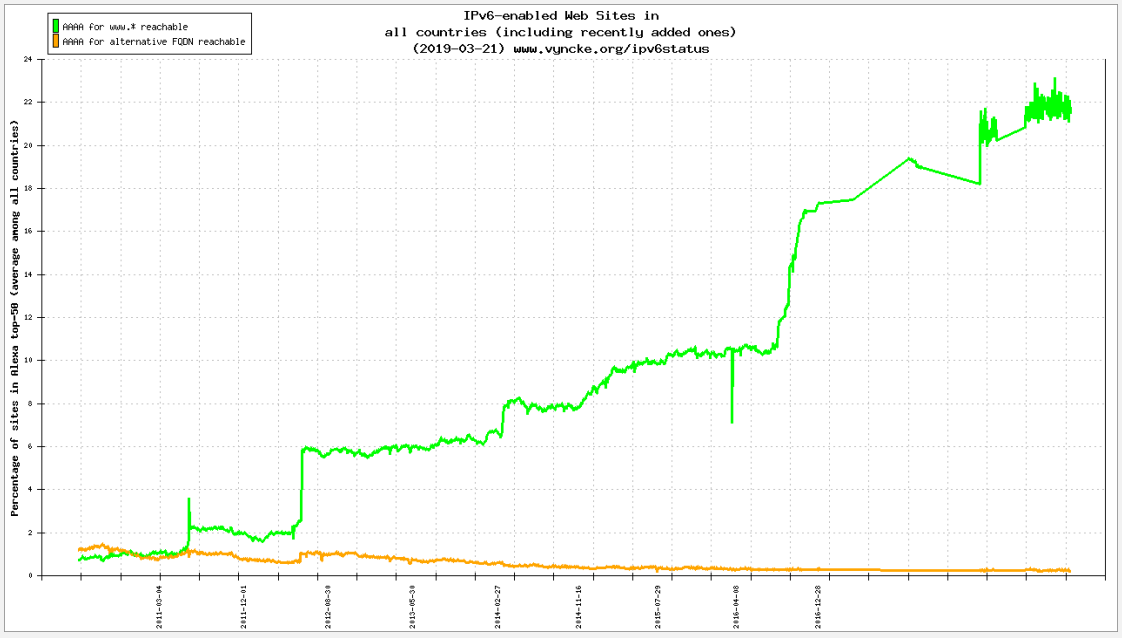
Impact de la double pile sur les services

Tests grandeur nature d'IPv6 :

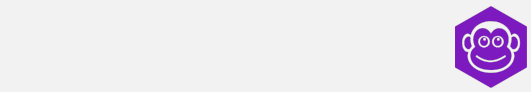
- 8 juin 2011: World IPv6 Day
  - Les sites majeurs activent IPv6 pour 1 journée
  - Conclusion : pas d'impact majeur
  - 0.3% de trafic IPv6
- Conclusion: L'activation d'IPv6 ne crée pas de nouveaux problèmes
- 6 juin 2012: World IPv6 Launch
  - IPv6 activé définitivement sur les sites majeurs (google, yahoo, facebook, akamai,...)
  - Potentiellement 50% du trafic Internet
  - Aujourd'hui 10%, car manque l'accès IPv6 des clients



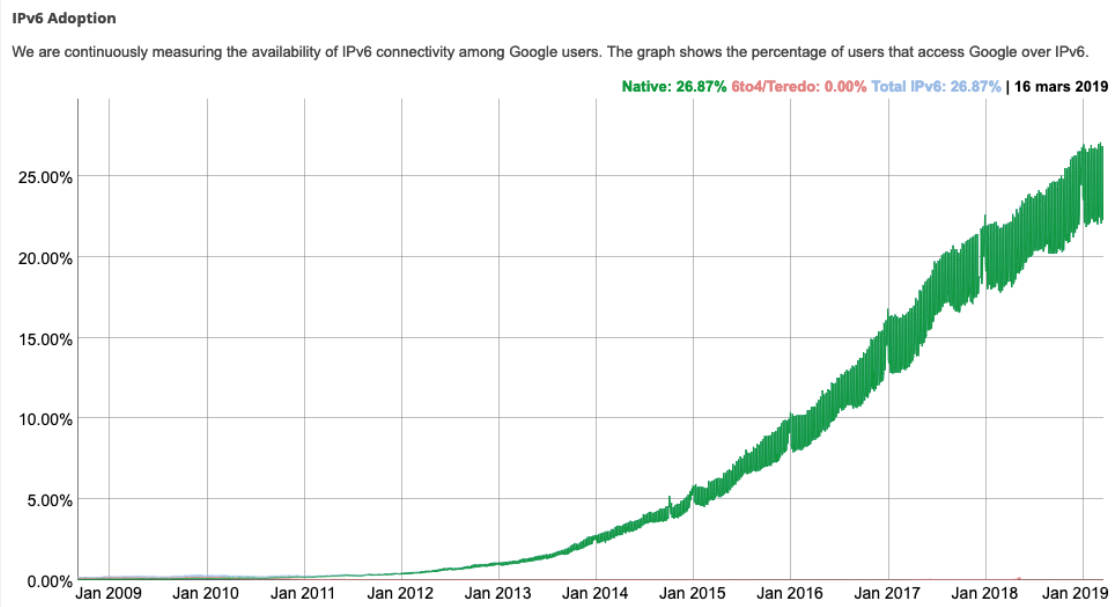
Progression de la migration vers IPv6 (services)



Source :https://www.vyncke.org/ipv6status/



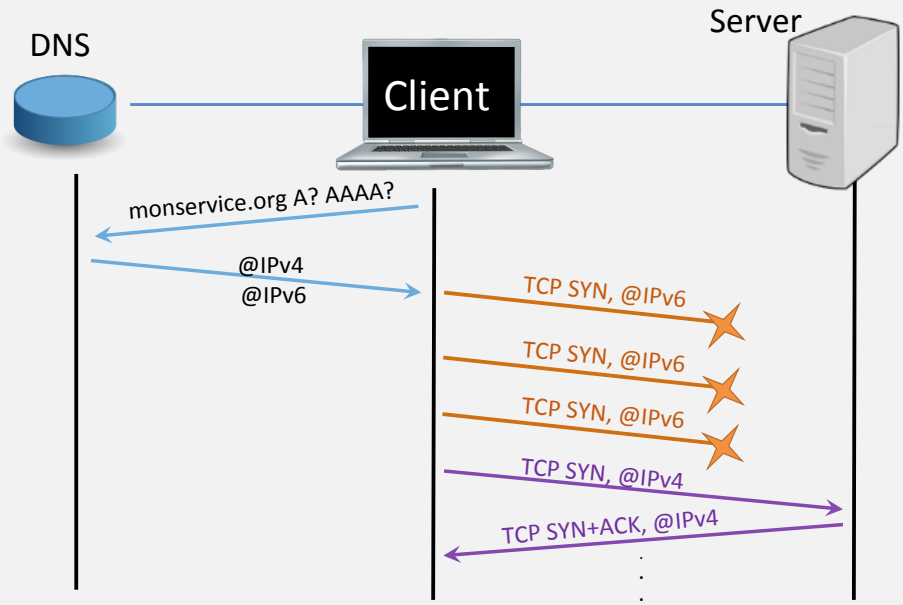
Progression de la migration vers IPv6 (utilisateurs)



Source :`http://www.google.com/ipv6/statistics.html`

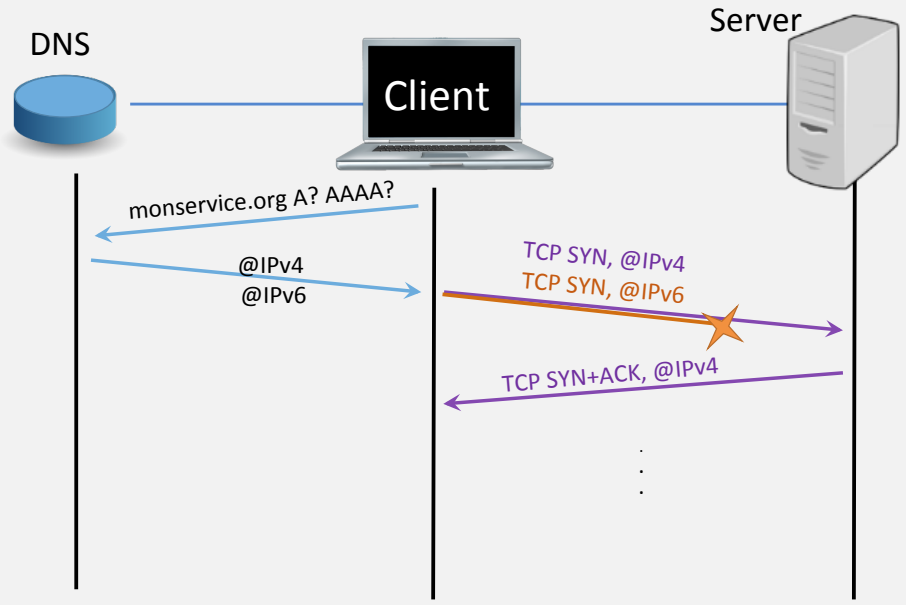
Happy Eyeballs RFC 6555

Comportement problématique :

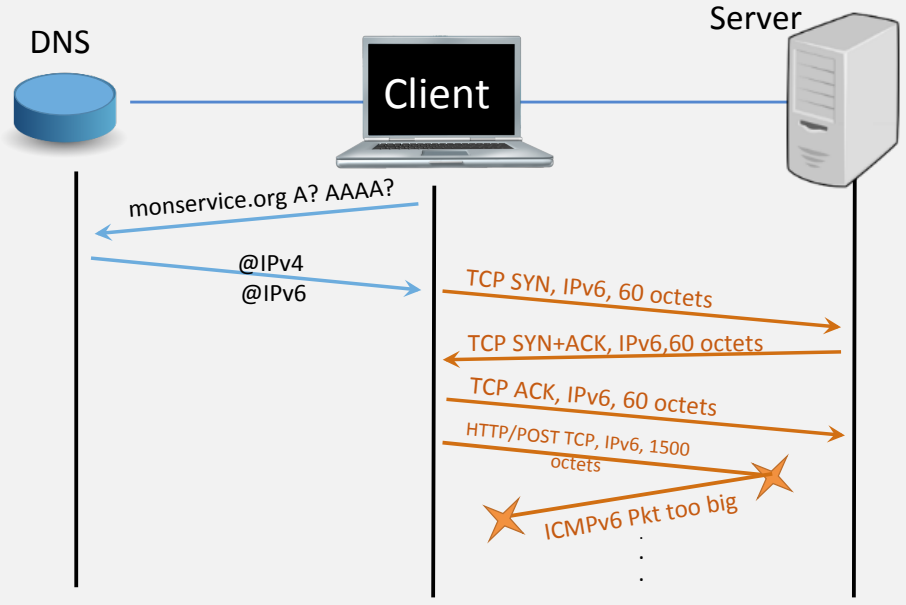


Happy Eyeballs RFC 6555

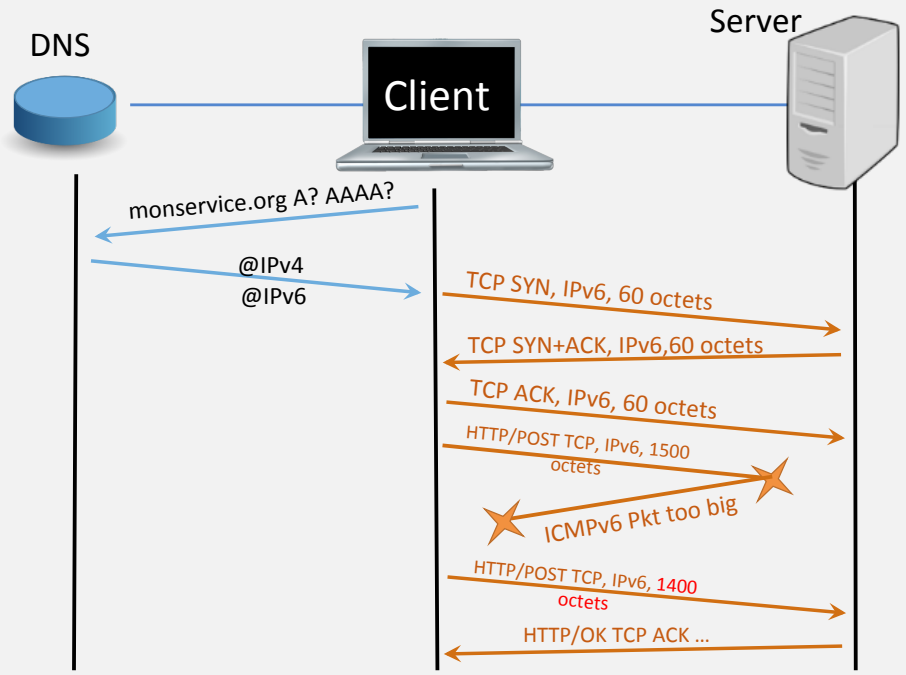
Solution :



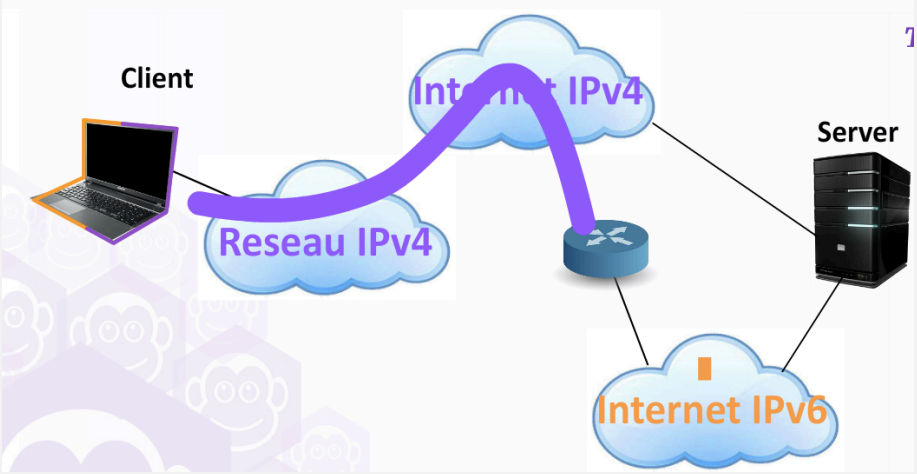
Problème de MTU



Problème de MTU



Problème de latence



Agenda

API Socket IPv6

- API Socket extendue pour IPv6
- Nouvelles familles de protocoles et d'adresse : PF_INET6 et AF_INET6
- Nouvelles structures :
  - `in6_addr`
  - `sockaddr_in6`
  - `sockaddr_storage`
- Nouvelles fonctions de conversion entre noms et addresses

Référence

RFC 3493 & Posix 1003.1g

Structure des sockets

Structure en C, C++

```
struct sockaddr_in6 {
  uint8_t          sin6_len;       /* structure length
  sa_family_t      sin6_family;    /* AF_INET6
  in_port_t        sin6_port;      /* transport layer port
  uint32_t         sin6_flowinfo;  /* IPv6 traffic class & flow info
  struct in6_addr  sin6_addr;      /* IPv6 address
  uint32_t         sin6_scope_id;  /* set of interfaces for a scope
};
```

- Similaire à `sockaddr_in` for IPv4
- Nouveaux champs pour scope et flow label

`sizeof(sockaddr_in6) > sizeof(sosckaddr_in)`

- `sockaddr_in6` ne peut être stockée dans une `struct sockaddr`
- Le code doit être modifié pour être AF-independant !

Gestion des sockets en C, C++

Gestion des sockets

- Création : identique à IPv4
  - `int s = socket(PF_INET6, SOCK_STREAM, 0);`
- Les autres fonctions ne sont pas modifiées
  - `bind, connect, listen, accept, send*, recv*, getpeername, getsockname`
- Ajout de nouvelles fonctions pour gérer les options
  - `getsockopt, setsockopt`

Sockets côté serveur

2 stratégies pour les applications serveur :

- Ouvrir une seule socket PF_INET6
  - Les connexions IPv4 seront vues comme provenant d'une adresse IPv6 IPv4-mappée
  - Problème : support de l'OS nécessaire
- Ouvrir 2 sockets : une PF_INET et une PF_INET6
  - Le client se connectera à l'une ou l'autre de ces sockets selon l'adresse du serveur
  - Le serveur doit s'attendre à des connexions sur les 2 sockets

Exemples avec `netstat -taun` (MacOSX)

Proto Rec Send Local Foreign State
tcp46 0 0 *.80 *.* LISTEN ← Serveur Apache utilise une socket
...
tcp4 0 0 *.22 *.* LISTEN ← Serveur SSH avec 2 sockets
tcp6 0 0 *.22 *.* LISTEN ←

Sockets côté client

- Le serveur peut être désigné à l'application cliente soit :
  - par une adresse ⇒choix de la socket en fonction de la famille de l'adresse
  - par un nom ⇒choix de la socket dépend de la résolution du nom
- La résolution de nom en IPv6 est possible par l'appel `getadrrdinfo`
  - `gethostbyname` ne retourne pas les enregistrements `AAAA`
  - `getarrdinfo` retourne l'ensemble des enregistrements pour le nom (IPv4+IPv6)
  - les enregistrements IPv6 sont placés en tête de liste
- Stratégie générique d'ouverture de socket côté client :
  - Résolution du nom, obtention d'une liste d'adresse
  - Tentative d'ouverture de la première adresse de la liste
  - Boucle sur la liste si l'ouverture échoue
- Attention : stratégie à adapter pour *Happy Eyeballs*

Anticiper le support d'IPv6

Structure générique pour les sockets

- Le code doit utiliser `struct sockaddr_storage` pour être AF-independant
- Cast lorsque la famille d'adresse doit être spécifiée

Socket containers

```
struct sockaddr_storage ss;
foo((struct sockaddr *)&ss);    // AF independent function

void foo(struct sockaddr *s) {
    // If we need IPv4 socket
    struct sockaddr_in *sin = (struct sockaddr_in *) s;
    // If we need IPv6 socket
    struct sockaddr_in6 *sin6 = (struct sockaddr_in6 *) s;
}
```

Résolution de nom : `getaddrinfo`

getaddrinfo() Prototype

```
int getaddrinfo(const char *nodename,
                const char *servname,
                const struct addrinfo *hints,
                struct addrinfo **res);
```

- Fonction générique, AF-independante, de résolution de nom en adresses
- Remplace `gethostbyname`
- `servname`: nom du protocole ("http") ou num. de port ("80")
- `hints`: permet de spécifier la requête (IPv4 only, IPv6 only, IPv4/IPv6)
- Retourne une liste de résultats !

Résolution inverse : getnameinfo()

getnameinfo() Prototype

```
int getnameinfo(const struct sockaddr *sa,
                socklen_t salen,
                char *host,
                socklen_t hostlen,
                char *serv, socklen_t servlen,
                int flags);
```

- Fonction générique, AF-independante, de résolution d'adresses en nom
- Remplace `gethostbyaddr`

Migration des applications existantes

Ajout de la compatibilité Ipv6 à une application

1: Remplacer les fonctions et structures IPv4-only par leur équivalent AF-independant

Generic Structure & Functions

`hostent` → `addrinfo`
`sockaddr_in` → `sockaddr_storage`
`gethostbyname` → `getaddrinfo`
`gethostbyaddr` → `getnameinfo`

2: Rechercher les usages spécifiques des structures contenant des adresses `in_addr`

- Utilisation des adresses comme identifiant, dans les logs, ...
- Ces usages doivent être AF-independants

Ajout de la compatibilité Ipv6 à une application

3: Côté serveur : choisir une stratégie d'ouverture de socket
4: Considerer qu'un même nœud peut avoir plusieurs adresses

- Dans `getaddrinfo` un même nom peut donner plusieurs adresses IPv4 et IPv6
- Attention si les adresses sont utilisées comme identifiants

5: Attention à la représentation textuelle des adresses

Notation:

http://[2001:660:7301:1::1]
scp foo.bar [2001:660:7301:1::1]:/tmp

Agenda

Support IPv6 en Java

- Java supporte IPv6 depuis JDK 1.4
- Extension à la classe InetAddress
- Héritage et polymorphisme assure la transparence de la version lors de la manipulation des adresses

Inet6Address

Nouvelle classe héritant de`InetAddress` (avec `Inet4Address`)

- Class pour instancier une adresse IPv6
- Méthode pour vérifier le scope :
  - `isIPv4CompatibleAddress` (for IPv4-mapped addresses)
  - `isLinkLocalAddress`
  - `isMulticastAddress`

InetAddress

Un objet `InetAddress` peut être une adresse IPv4 or IPv6 `InetAddress` extendue pour la résolution de noms

- Méthode `getByName` retourne seulement une adresse IPv4
- Nouvelle méthode `getAllByName` retourne les adresses IPv4 et IPv6
- Résolution inverse inchangée

Changement nécessaire pour le support IPv6 :

Remplacer les appels à `getByName` par `getAllByName` et gérer la résolution multiple

API Socket

- L'API Socket est basé sur la super-class InetAddress → pas de changement
- Choix du protocole en fonction de l'adresse fournie pour le `bind` :
  - Adresse IPv4 → Socket ecoutant en IPv4
  - Adresse IPv6 → Socket ecoutant en IPv4 et IPv6

Conséquences

- Integration d'IPv6 n'a pas d'impact sur IPv4
- IPv6 ne sera utilisé que quand le client se connectera en IPv6

Agenda

L'héritage socket

Le support réseau de Python repose sur l'API `socket`

- Héritage de l'API C
- Utilisée par Un bon nombre (toutes?) des librairies réseaux
  - socketserver
  - urllib
  - asyncio
  - TwistedMatrix
  - HTTPLib2
  - etc.

Cette API est conforme au RFC 3493:

- La socket ouverte en IPv6
- Les connexions IPv4 sont *mappée* en connexions IPv6

Un rappel sur la socket

Une socket est identifiée par 5 paramètres:

- Le protocole de transport (TCP/UDP)
- Adresse de l'hôte expéditeur
- Port de l'application origine
- Adresse de l'hôte destinataire
- Port de l'application cible

Sous Linux :

```
% netstat -taun
Connexions Internet actives (serveurs et établies)
Proto Recv-Q Send-Q Adresse locale          Adresse distante        E
tcp        0      1 10.51.0.215:43742       10.35.1.191:631         E
```

Un petit serveur IPv4

```python
# Echo server program
import socket

HOST = None               # None means all available interfaces
PORT = 50007              # Arbitrary non-privileged port
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.bind((HOST, PORT))
    s.listen(1)
    conn, addr = s.accept()
    with conn:
        print('Connected by', addr)
        while True:
            data = conn.recv(1024)
            if not data: break
            conn.sendall(data)
```

Testons !

```
% python ./echo_server_v4.py &
% nc 127.0.0.1 50007
Connected by ('127.0.0.1', 53112)
kl
kl
^Z
[2]  + 88737 suspended  nc 127.0.0.1 50007
 % netstat -tan | grep 50007
tcp4       0      0  127.0.0.1.50007        127.0.0.1.53112
tcp4       0      0  127.0.0.1.53112        127.0.0.1.50007
tcp4       0      0  *.50007                *.*
%
```

Une IPv6-fication naive

```
@@ -3,7 +3,7 @@

 HOST = None               # Symbolic name meaning all available interfaces
 PORT = 50007              # Arbitrary non-privileged port
-with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
+with socket.socket(socket.AF_INET6, socket.SOCK_STREAM) as s:
     s.bind((HOST, PORT))
     s.listen(1)
     conn, addr = s.accept()
```

Une IPv6-fication naive

```python
# Echo server program
import socket

HOST = None               # None means all available interfaces
PORT = 50007              # Arbitrary non-privileged port
with socket.socket(socket.AF_INET6, socket.SOCK_STREAM) as s:
    s.bind((HOST, PORT))
    s.listen(1)
    conn, addr = s.accept()
    with conn:
        print('Connected by', addr)
        while True:
            data = conn.recv(1024)
            if not data: break
            conn.sendall(data)
```

Testons !

```
% python ./echo_server_v6_naif.py &
% nc ::1 50007
Connected by ('::1', 53224, 0, 0)
dz
dz
^Z
[2]  + 88997 suspended  nc ::1 50007
% netstat -tan | grep 50007
tcp6       0      0  ::1.50007              ::1.53224
tcp6       0      0  ::1.53224              ::1.50007
tcp46      0      0  *.50007                *.*
% nc 127.0.0.1 50007
Connected by ('::ffff:127.0.0.1', 53242, 0, 0)
lkjlkj
lkjlkj
^Z
[2]  + 89045 suspended  nc 127.0.0.1 50007
% netstat -tan | grep 50007
tcp6       0      0  ::1.50007              ::1.53224
tcp6       0      0  ::1.53224              ::1.50007
tcp4       0      0  127.0.0.1.50007        127.0.0.1.53242
tcp4       0      0  127.0.0.1.53242        127.0.0.1.50007
tcp46      0      0  *.50007                *.*
```

Pour résumer, côté serveur

L'API socket IPv6 apporte la compatibilité ascendante avec IPv4

- Les adresses IPv4 mappée représentent les hôtes IPv4
- Les connexions IPv4 sont vues comme des connexion IPv6
- L'application n'a plus à différencier la famille d'adresse

Attention, cette glue n'est disponible que sur les OS compatibles IPv6

- Désolé pas pour MS-DOS, BeOS, etc.