Travaux Pratiques
Routage statique IPv6 et
transition IPv4 - IPv6 sous Unix avec `ipv6_care`

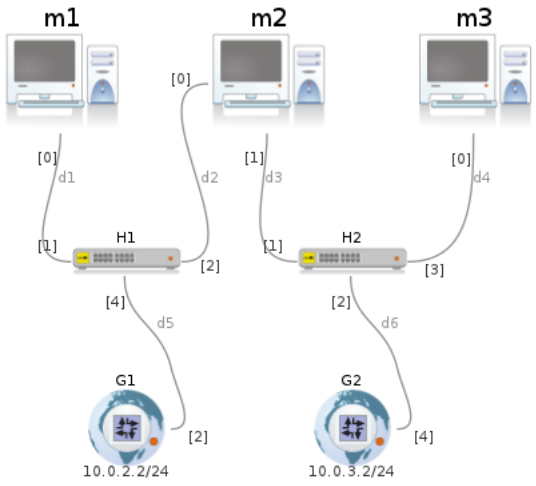
Séance de TP entièrement effectuée avec le logiciel Marionnet. Durée estimée : 1h30 avec téléchargements et compilations, 1h sinon.

Prérequis. Format des paquets IPv6, double pile TCP/IP (sockets v4/v6).

1 Câblage et configuration des réseaux locaux

On utilise 3 machines, m_1, m_2 et m_3, dont une en particulier, m_2, équipée de 2 interfaces réseau $eth0$ et $eth1$. Construisez un premier réseau local IPv4 $LAN_1 = \{m_1, m_2[0]\}$ en 10.0.2.0/24 en exploitant le service DHCP (v4) fourni par la passerelle internet G_1. Construisez de la même manière un deuxième réseau local IPv4 $LAN_2 = \{m_2[1], m_3\}$ en 10.0.3.0/24, en exploitant le service DHCP (v4) fourni par la passerelle internet G_2. Dans la fenêtre de dialogue des passerelles, vous devez cocher l'option "service DHCP" et préciser l'adresse réseau desservie.

Distributions GNU/Linux. Utilisez la distribution Mandriva pour toutes les machines si vous voulez télécharger et compiler les deux logiciels (`ipv6_care` et `nc6`). Pour un TP à réaliser en 1 heure, préférez plutôt la distribution Debian Wheezy qui contient déjà le logiciel `ipv6_care`, ce qui vous permettra d'éviter le téléchargement et la compilation de ce dernier.



Numéros IPv4 et routes. En lançant des clients dhcp sur les machines, et en définissant correctement les tables de routage des machines, chaque machine devra être en mesure à la fois (a) d'accéder à internet (en IPv4), et (b) de communiquer avec le réseau local IPv4 d'à côté.

2 Installation des logiciels IPv6

La configuration IPv4 de votre réseau étendu vous permet maintenant d'accéder à Internet. Pour la suite de cet exercice, si les adresses des logiciels indiquées en argument des commandes `wget` ne sont pas à jour, utilisez l'adresse mirroir sur le site `www.marionnet.org` :

`http://www.marionnet.org/download/contrib/mirror/`

qui contient toutes les archives nécessaires (vous pouvez retrouver les noms précis avec un butineur, en navigant sur cette page). Téléchargez, compilez et installez sur toutes les machines (en parallèle si possible) les deux logiciels utiles à la suite du TP, par la séquence de commandes suivantes :

2.1 ipv6_care

```
$ wget http://sourceforge.net/projects/ipv6-care/files/latest/download?source=files
$ tar xvzf ipv6_care-*.tar.gz
$ cd ipv6_care-*
$ ./configure && make && make check && make install && echo OK
```

2.2 nc6 (netcat en version IPv6)

```
$ wget ftp://ftp.deepspace6.net/pub/ds6/sources/nc6/nc6-1.0.tar.bz2
$ tar xvjf nc6-1.0.tar.bz2
$ cd nc6-1.0
$ ./configure && make && make install && echo OK
```

3 Configuration et routage statique IPv6

Configurez le réseau LAN_1 en `2001:db8::/32` et le réseau LAN_2 en `2001:db9::/32` en assurant le routage IPv6 sur la machine m_2. Utilisez la convention usuelle où m_i prend comme dernier octet de l'adresse la valeur i (par exemple m_3 prendra `2001:db9::3/32`). Testez la connectivité IPv6 avec `ping6` entre machines du même réseau et machines de réseaux différents (m_1 et m_3).

4 Donner une compatibilité double pile à un service simple pile sous Unix avec `ipv6_care`

4.1 IPv4 only -> Double pile

Lancer un serveur TCP/IPv4 avec `netcat` sur m_3. Testez le bon fonctionnement de la connexion en IPv4 en lançant un client `netcat` sur m_1 sur un port quelconque (par exemple 80). Testez à présent l'échec de connexion d'un client exclusivement TCP/IPv6 tournant sur m_1 :

```
m1# nc6 -6 ADRESSE_IPV6 PORT      # échec!
```

Lire à présent le manuel de `ipv6_care`, puis relancer le service TCP/IPv4 avec `netcat` en utilisant le "tuteur" `ipv6_care`. Testez à nouveau depuis m_1 :

```
m1# nc6 -6 ADRESSE_IPV6 PORT      # ça marche!
```

Vérifiez, avec une instance de `wireshark` lancée sur m_2, que la communication se fait effectivement en IPv6, grâce à `ipv6_care`, malgré le fait que le service sous tutelle soit implanté en IPv4.

4.2 IPv6 only -> Double pile

Lancer un serveur TCP/IPv6 avec `nc6` sur m_3. Testez le bon fonctionnement de la connexion en IPv6 en lançant un client `nc6 -6` sur m_1 sur un port quelconque (par exemple 80). Testez à présent l'échec de connexion d'un client exclusivement TCP/IPv4 tournant sur m_1 :

```
m1# netcat ADRESSE_IPV4 PORT      # échec!
```

Relancer alors le service TCP/IPv6 avec `nc6 -6` mais, cette fois, en utilisant le "tuteur" `ipv6_care`. Testez à nouveau depuis m_1 :

```
m1# netcat ADRESSE_IPV4 PORT      # ça marche!
```

Vérifiez à nouveau, avec une instance de `wireshark` lancée sur m_2, que la communication se fait cette fois en IPv4, malgré le fait que le service sous tutelle soit implanté en IPv6.

5 Annexe

```
$ nmap -6 -sT ::1                  # détecte les ports TCP/IPv6 ouverts sur la machine
$ route -A inet6 ip neigh show     # équivalent de arp mais pour IPv6
$ ipv6_care check netcat -l -p 80  # diagnostic ipv6_care sur un service "TCP/IPv4 only"
$ ipv6_care check nc6 -6 -l -p 80  # diagnostic ipv6_care sur un service "TCP/IPv6 only"
```