Licence ASRALL

2021-2022

Adaptation réseaux

Étienne André

IUT Charlemagne, Université de Lorraine
Etienne.Andre@univ-lorraine.fr
www.loria.science/andre/enseignement/Reseaux/

Version: 5 octobre 2021

Adaptation - Réseaux

- Volume horaire
  - 5 séances de 2 heures
  - Un examen (sur papier)

- Contenu : des rappels (en théorie)
  - Généralités
  - Adressage
  - Routage
  - Interactions adressage et routage $\rightsquigarrow$ ARP
  - Configuration réseau sous Linux

`loria.science/andre/enseignement/Reseaux/`

Les réseaux aujourd'hui

- Haut débit pour tout le monde
  - 100 Mb/s ou 1 Gb/s à la maison
  - backbone Tb/s
- Transport des données multimédia
  - Téléphone, télévision, jeux, ...
- Terminaux connectés en permanence
  - WiFi, 4G/5G
- Information partout, n'importe quand

Architecture de communication

- Grand nombre de machines à connecter
- Interconnexion de systèmes (ou de réseaux) différents
  - ordinateur, téléphone portable
  - réseaux filaires, sans-fil
- Connexions non fiables
- Multiplexage de communications
  - Congestion ?
- ...

Comment faire fonctionner tout ça en pratique ?

- Protocoles
- Architectures en couches

Protocoles (1/2)

Définition (Protocole)

Règles de communication entre 2 parties (*peers*) de même niveau
Les protocoles définissent le format, l'ordre des messages envoyés et reçus, et les actions possibles lors de la transmission et de la réception.

Protocoles humains

- Exemples :
  - « Quelle heure est-il ? »
  - Introductions
    - Bonjour, ça va ?
    - Allô ?

Protocoles réseaux

- Machines à la place des humains

Toutes les communications sur Internet sont dirigées par des protocoles

Protocoles (2/2)

Protocoles d'Internet :

- Décrits par des RFC (*Request For Comment*)
  - Plus de 6000 RFCs
  - Mais certains protocoles ne sont pas standardisés
- organisme de standardisation :
  IETF (*Internet Engineering Task Force*)

Architecture en couche

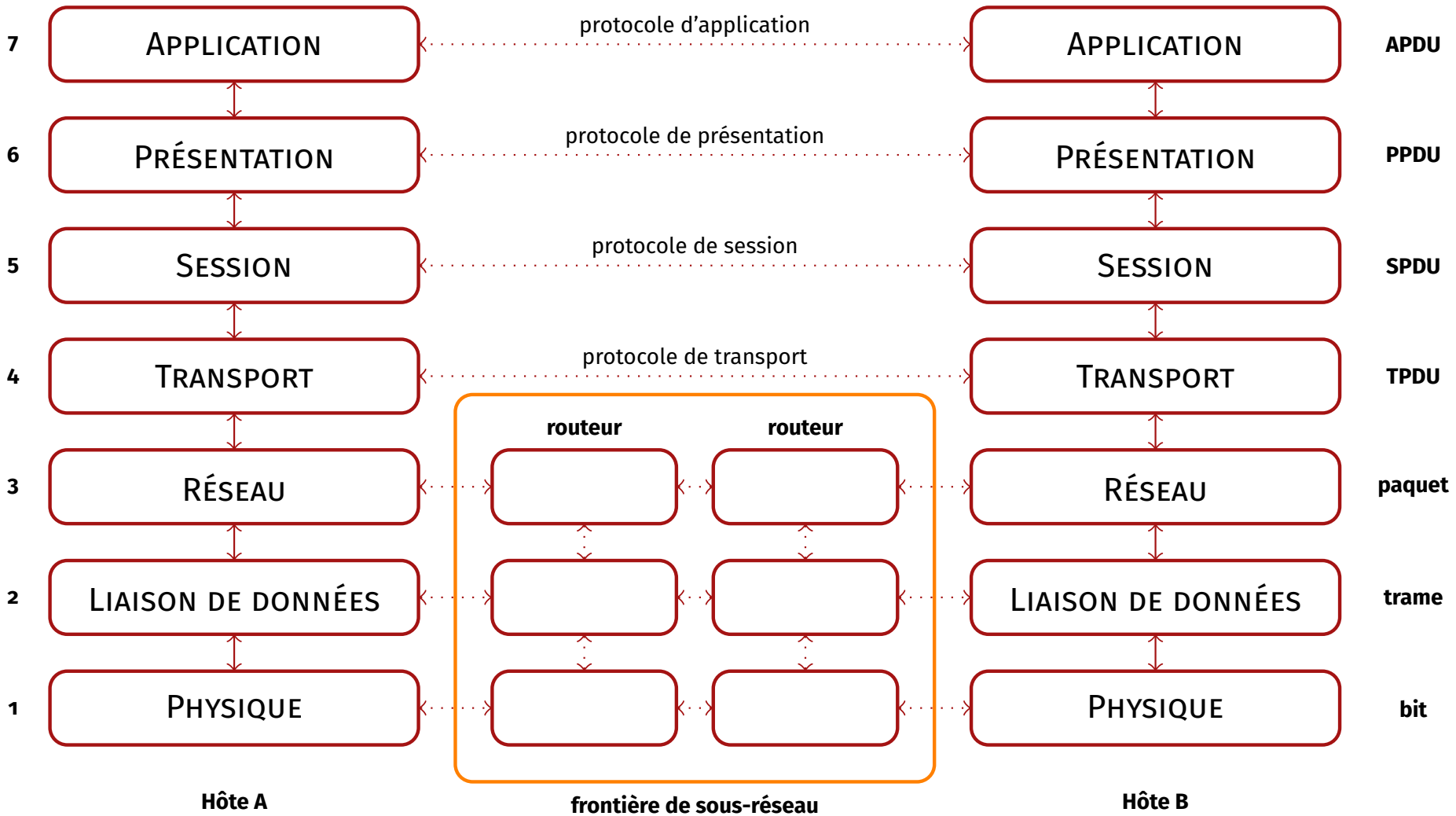
- « *chaque couche fait une chose, mais la fait bien, et fournit une interface plus simple à la couche au-dessus* »
  - Fournit des services à la couche supérieure (via un ensemble de primitives)
  - Utilise les services de la couche inférieure
  - Ajoute éventuellement un en-tête (*header*) et/ou un postambule (*trailer*) aux données, avant de transmettre les données à la couche inférieure
- Permet de séparer la spécification et l'implémentation

Communications :

- horizontale : entre parties (peers), qui comprennent et parlent le même protocole
- verticale : entre couches, qui utilisent un protocole d'interface (pas un protocole réseau)

Modèle OSI (1/2)

OSI : *Open Systems Interconnection*



Modèle OSI (2/2)

- **Application** : programmes d'application
- **Présentation** : interprétation des données (encodage)
- **Session** : notions de connexions, multiplexage, ...
- **Transport** : gère le transfert de bout en bout
- **Réseau** : gère le transfert entre les extrémités (routage)
- **Liaison** : contrôle d'erreur, contrôle de flux
- **Physique** : interface avec le support physique

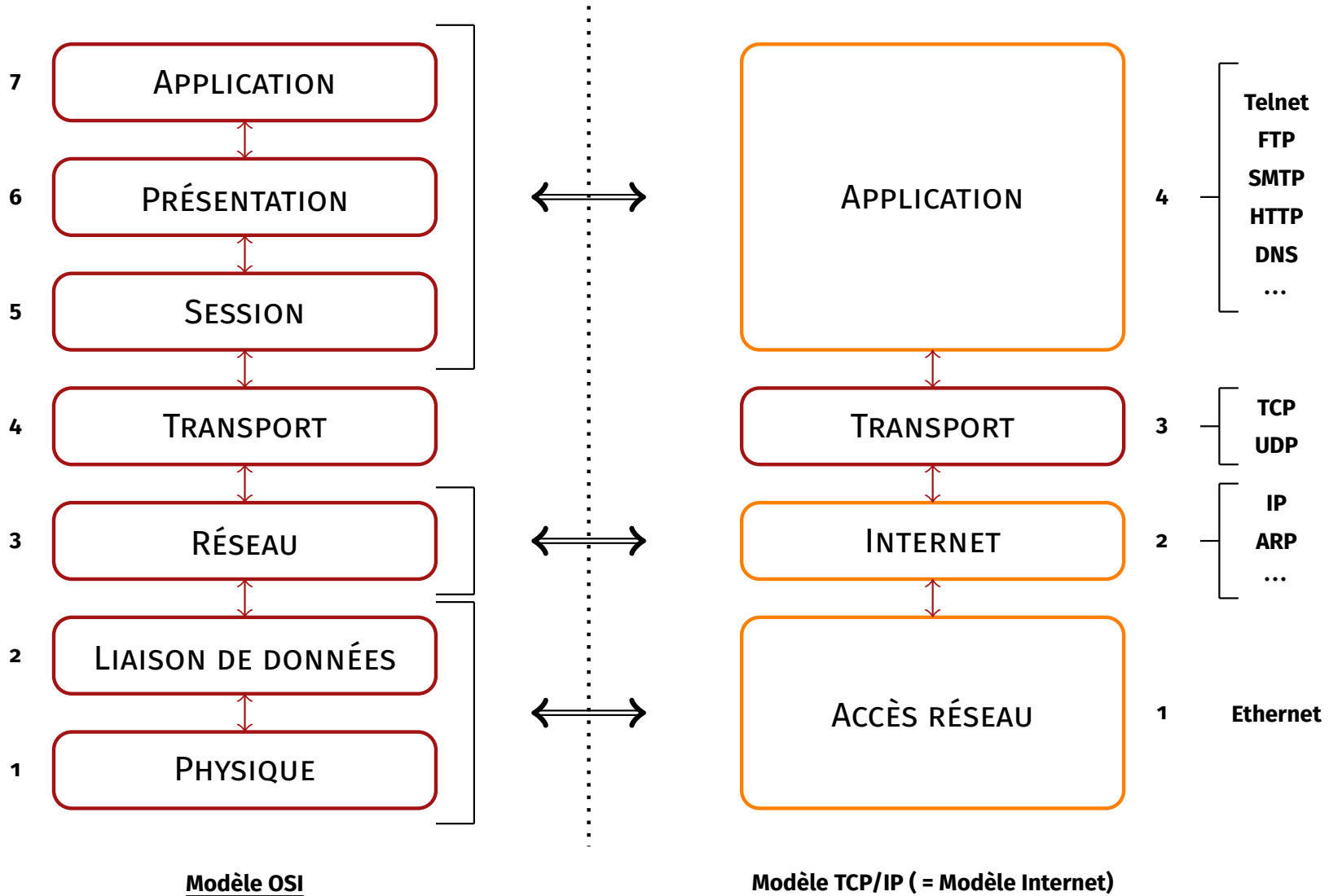
TCP/IP

Ensemble des protocoles utilisés sur Internet

Organisés en couche, mais certaines couches du modèle OSI sont fusionnées :

- **Applications** : HTTP, DNS, SMTP, peer-to-peer, ...
- **Transport** : TCP, UDP
- **Réseau** : IP
- **Transmission** : Ethernet, réseaux sans fil

Modèle OSI (1984) vs. Modèle TCP/IP (1976)



Exercice : philosophes

Deux philosophes ne parlant pas la même langue et ne pouvant pas se déplacer souhaitent converser ensemble. L'un est burkinabè, l'autre est taïwanaise. La seule possibilité pour eux est de faire appel à des traducteurs anglais et de télégraphier les courriers respectifs.

Modélisez cet exemple sous la forme d'un schéma s'inspirant du modèle OSI en donnant le rôle de chaque couche.

Rappels sur les unités (1/2)

Temps : seconde **s**

- milliseconde ms ($0.001s$) : $10^{-3}s$
- microseconde μs ($0.000001s$) : $10^{-6}s$
- nanoseconde ns ($0.000000001s$) : $10^{-9}s$

Volume de données (émises, reçues) : bit : **b**, ou octet **o**.

- Kilobit **Kb** (1 000) 10^3b
- Megabit **Mb** (1 000 000) 10^6b
- Gigabit **Gb** (1 000 000 000) 10^9b
- Terabit **Tb** (1 000 000 000 000) $10^{12}b$
- 1 **o** (octet) = 8 **b** (bits) = 1 **B** (Byte)

Rappels sur les unités (2/2)

Attention

- Réseaux : 1 K = 1000 (pour les débits)
- Mémoire : 1 K = 1024
- Volume de données (stockage) : ça dépend !

Pour lever l'ambiguïté : préfixes binaires (peu utilisés)

- Kibi, Mebi, Gibi, ...
- $1Kio = 2^{10}o = 1024o$

Débit

Débit (D, en b/s) :

- Nombre de bits par unité de temps
- On parle parfois de bande passante (*bandwidth*)

Exemples :

- USB 2.0 : 480 Mb/s (ou Mbps)
- Wifi : 11 Mbps (802.11b) ou 54 Mbps (802.11g) ou plus

Temps d'émission, de propagation, de transfert

Temps d'émission (ou de transmission) **Te** : dépend du débit

Temps de propagation **Tp** : dépend de la vitesse de propagation **Vp** du signal (fonction du support), et de la distance

- Câble : $V_p = 2 \times 10^8 m/s$
- Fibre optique : $V_p = 3 \times 10^8 m/s$

Temps de transfert **Tt** : temps de transfert de bout en bout

$$T_e = \frac{N}{D}$$

$$T_p = \frac{L}{V_p}$$

$$T_t = T_e + T_p$$

(N : nb de bits ; D : débit ; L : distance)

Exercice : transfert de données

On souhaite transférer 100 Go de données entre l'IUT et le LORIA, distants de $2,5$ km.

1. La première solution est de réaliser le transfert à l'aide d'un disque dur et d'un vélo (à la vitesse moyenne de 18 km/h, soit 5 m/s). Calculez les temps d'émission et/ou de propagation (ou donnez-en une estimation, en justifiant). Calculez ensuite le débit moyen de cette solution.

2. La deuxième solution est de transférer les données en utilisant un réseau en fibre optique, d'un débit de 1 Gbps (on rappelle la vitesse de propagation du signal dans la fibre optique : $3 \times 10^8 m/s$). Calculez les temps d'émission et/ou de propagation (ou donnez-en une estimation, en justifiant). Calculez ensuite le débit moyen de cette solution.

3. Quelle est la meilleure solution ?

4. Sachant que la vitesse d'écriture sur un disque dur récent est d'environ 100 Mo/s, que pouvez-vous dire de ce calcul ?

Exercice : transfert de données (solution)

1. $T_e = 0$ **car on suppose les données déjà écrites sur le disque**

   $T_p = \frac{L}{V} = \frac{2500m}{5m/s} = 500\,s$ **(8mn20)**

   Donc $T_t = 500\,s$

2. **Cette fois, on doit compter le temps d'émission.**

   $T_e = \frac{N}{D} = \frac{100\,Go}{1Gb/s} = \frac{800\,Gb}{1Gb/s} = 800\,s = 13\,mn\,20\,s$

   A priori le temps de propagation est négligeable. Calculons-le néanmoins :

   $T_p = \frac{L}{V} = \frac{2500m}{3\times10^8\,m/s} = \frac{2500m}{300\,000\,000\,m/s} = 0,000008333\,s$ **(ordre de la** μs**)**

   Donc $T_t = 800\,s(+\epsilon)$

3. **Le vélo !**

4. **Temps de copie des données sur le disque dur :**

   $T_e = \frac{N}{D} = \frac{100\,Go}{100Mo/s} = \frac{100\,000\,Mo}{100Mo/s} = 1000\,s$

   S'il faut copier les données sur le disque du vélo, la 2e solution (fibre optique) redevient plus intéressante.

Niveau 2 vs niveau 3

- Niveau 2 (par exemple Ethernet) :
  - Réseau « local »
  - Équipements : hubs, switchs (commutation de trames)
  - Adresses MAC
- Niveau 3 (IP) :
  - Interconnexion de réseaux locaux
  - Équipements : routeurs
  - Adresses IP, tables de routage
- Protocole ARP : résolution de l'adresse MAC correspondant à une adresse IP

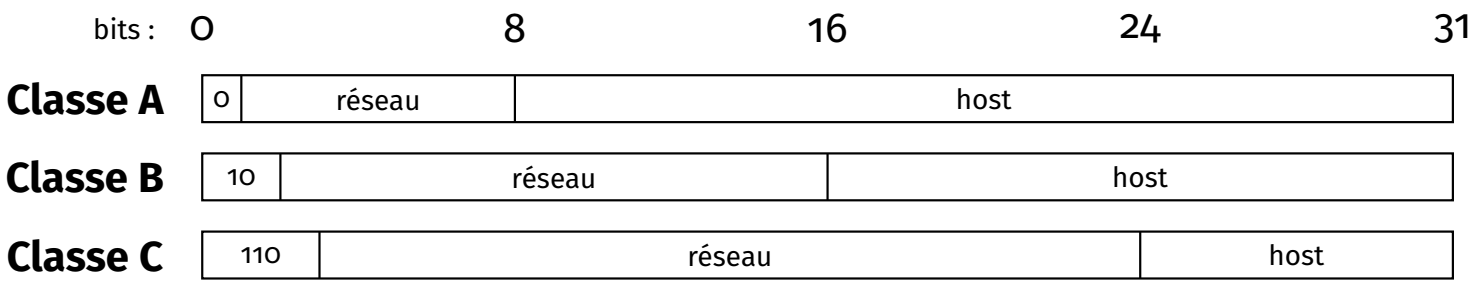
Adressage IP

- Adressage hiérarchique
  - Adresse de réseau IP
  - Adresse de chaque machine dans le réseau
- Réseau IP : suite d'adresses contiguës
- Format d'adresse IPv4 :
  - 32 bits : adresse réseau (n bits) + adresse machine (m bits)
  - Un réseau IP possède 2^m valeurs réparties entre :
    - l'adresse IP du réseau : les m bits sont à 0
    - les adresses des machines (max $2^m - 2$ machines)
    - une adresse de diffusion (*broadcast*) les m bits sont à 1

Classes de réseaux IPv4

- Définit le nombre d'octets pour l'adresse réseau
  - classe A $\longrightarrow$ 1 octet
  - classe B $\longrightarrow$ 2 octets
  - classe C $\longrightarrow$ 3 octets

- Historiquement :
  - classe A : 1er bit de l'adresse à 0
  - classe B : 2 premiers bits de l'adresse : 10
  - classe C : 2 premiers bits de l'adresse : 11



- Actuellement, on parle de classe A, B, C sans prendre en compte les premiers bits de l'adresse (uniquement le nombre d'octets)

Notation CIDR

- Classes de réseaux : grain trop gros
- Besoin d'attribuer des plages d'adresses de manière plus fine
- CIDR : *Classless Interdomain Routing*
  - Indication du nombre de bits de l'adresse réseau
  - Exemple : `charlemagne.iutnc.univ-lorraine.fr` : `194.214.170.56/22` ou `194.214.170.56/255.255.252.0`

Masque de sous-réseau

Notation utilisée pour identifier le nombre de bits du réseau (n)

Exemple : soit la machine `194.214.170.61` sur un réseau `/22`.

Quel est le réseau ?

Adresse IP machine	`194.214.170.61`	`11000010.11010110.10101010.00111000`
Masque	`255.255.252.0`	`11111111.11111111.11111100.00000000`
Adresse IP réseau	`194.214.168.0`	`11000010.11010110.10101000.00000000`

Adresses privées IPv4 (RFC 1918)

- Réservées à un usage local (NAT, etc.)
- `10.0.0.0` - `10.255.255.255` (`10/8`)
- `172.16.0.0` - `172.31.255.255` (`172.16/12`)
- `192.168.0.0` - `192.168.255.255` (`192.168/16`)
- Autre adresse spéciale :
  - `127.0.0.1` - boucle locale

Exercices 1/5

- Combien d'adresses utilisables par des machines comporte un réseau :
  1. de masque `255.255.240.0` ?
     On passe en binaire :
     `11111111.11111111.11110000.00000000`
     On compte 12 bits à 0, donc 12 bits pour les machines.
     Donc $2^{12} - 2 = 4094$ machines différentes
  2. de masque `255.255.255.192` ?
     On passe en binaire :
     `11111111.11111111.11111111.11000000`
     On compte 6 bits à 0, donc 6 bits pour les machines.
     Donc $2^6 - 2 = 62$ machines différentes
  3. `/12` ?
     Correction : 12 bits pour le nom du réseau, donc $32 - 12 = 20$ bits pour les machines. Donc $2^{20} - 2 = 1\,048\,574$ machines différentes

Exercices 2/5

- Une machine d'adresse 129.88.61.10 appartient à un réseau /22. Donnez :
  1. le masque de sous-réseau
     /22 se convertit en 11111111.11111111.11111100.00000000
     Donc masque (toujours en décimal) : 255.255.252.0
  2. l'adresse du réseau
     On convertit en binaire : 10000001.01011000.00111101.00001010
     On garde les 22 1ers bits 10000001.01011000.00111100.00000000
     On reconvertit en décimal : 129.88.60.0
  3. l'adresse de diffusion (*broadcast*)
     On convertit en binaire : 10000001.01011000.00111101.00001010
     On garde les 22 1ers bits et on met le reste à 1
     10000001.01011000.00111111.11111111
     On reconvertit en décimal : 129.88.63.255
  4. les adresses des première et dernière machines du réseau
     Astuce : 1re machine = nom du réseau $+1$ 129.88.60.1
     Astuce : dernière machine = broadcast -1 129.88.63.254
  5. le nombre d'adresses utilisables par des machines
     Correction : 22 **bits pour le nom du réseau, donc** $32 - 22 = 10$ **bits pour les machines. Donc** $2^{10} - 2 = 1\,022$ **machines différentes**

Exercices 3/5

- Mêmes questions pour 152.81.15.82/20.
  1. le masque de sous-réseau
     /20 **se convertit en** 11111111.11111111.11110000.00000000
     Donc masque (toujours en décimal) : 255.255.240.0
  2. l'adresse du réseau
     On convertit en binaire : 10011000.01010001.00001111.01010010
     On garde les 20 1ers bits 10011000.01010001.00000000.00000000
     On reconvertit en décimal : 152.81.0.0
  3. l'adresse de diffusion (*broadcast*)
     On convertit en binaire : 10011000.01010001.00001111.01010010
     On garde les 20 1ers bits et on met le reste à 1
     10011000.01010001.00001111.11111111
     On reconvertit en décimal : 152.81.15.255
  4. les adresses des première et dernière machines du réseau
     Astuce : 1re machine = nom du réseau $+1$ 152.81.0.1
     Astuce : dernière machine = broadcast -1 152.81.15.254
  5. le nombre d'adresses utilisables par des machines
     Correction : 20 **bits pour le nom du réseau, donc** $32 - 20 = 12$ **bits pour les machines. Donc** $2^{12} - 2 = 4\,094$ **machines différentes**

Exercices 4/5

- Mêmes questions pour 100.64.85.152/28.
  1. le masque de sous-réseau
     /28 se convertit en 11111111.11111111.11111111.11110000
     Donc masque (toujours en décimal) : 255.255.255.240
  2. l'adresse du réseau
     On convertit en binaire : 01100100.01000000.01010101.10011000
     On garde les 28 1ers bits 01100100.01000000.01010101.10010000
     On reconvertit en décimal : 100.64.85.144
  3. l'adresse de diffusion (*broadcast*)
     On convertit en binaire : 01100100.01000000.01010101.10011000
     On garde les 28 1ers bits et on met le reste à 1
     01100100.01000000.01010101.10011111
     On reconvertit en décimal : 100.64.85.159
  4. les adresses des première et dernière machines du réseau
     Astuce : 1re machine = nom du réseau $+1$ 100.64.85.145
     Astuce : dernière machine = broadcast -1 100.64.85.158
  5. le nombre d'adresses utilisables par des machines
     Correction : 28 **bits pour le nom du réseau, donc** $32 - 28 = 4$ **bits pour les machines. Donc** $2^4 - 2 = 14$ **machines différentes**

Exercices (5/5)

- La machine A a pour adresse $173.129.174.14$ et pour masque $255.255.255.240$. La machine B a pour adresse $173.129.174.17$ et pour masque $255.255.255.240$.
  Est-il possible de joindre B depuis A sans passer par un routeur ? Expliquer.

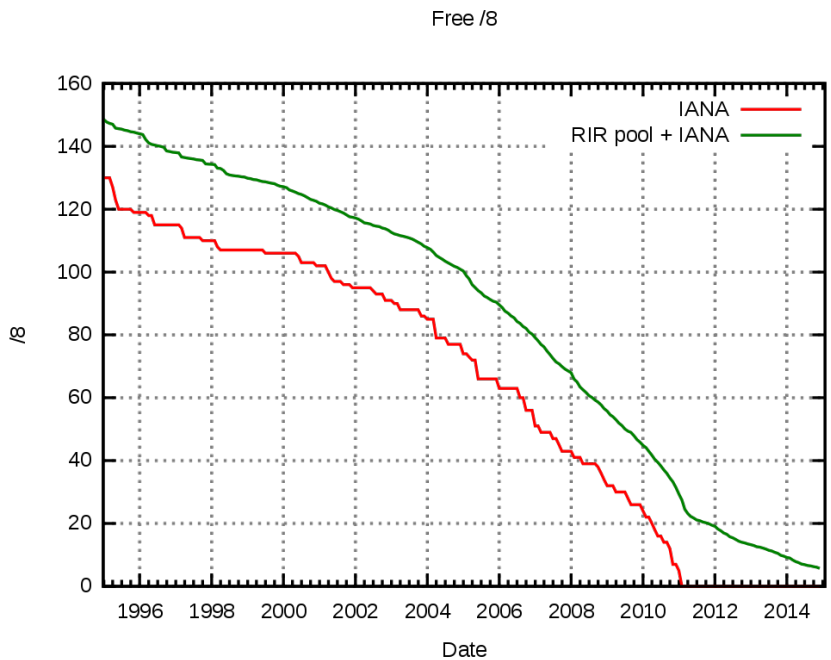
On passe le masque en binaire : $11111111.11111111.11111111.11110000$
(Donc masque $/28$**.)**
On convertit l'adresse de A **en binaire :**
$10101101.10000001.10101110.00001110$
Nom du réseau de A **(28 1ers bits de** A**) :**
$10101101.10000001.10101110.00000000$ **soit** $173.129.174.0$
On convertit l'adresse de B **en binaire :**
$10101101.10000001.10101110.00010001$
Nom du réseau de B **(28 1ers bits de** B**) :**
$10101101.10000001.10101110.00010000$ **soit** $173.129.174.16$
Or $173.129.174.0 \neq 173.129.174.16$ **donc les machines *ne sont pas* sur le même réseau. Donc impossible d'aller de** A **à** B **sans passer par un routeur.**

Attribution des adresses IP v4

- Une machine hôte obtient son adresse du bloc IP de son organisation
- Une organisation obtient son bloc IP à partir du bloc d'adresses de son ISP (*Internet service provider*, fournisseur d'accès à Internet ou FAI)
- Un FAI obtient son bloc d'adresses de son propre fournisseur ou de l'un des 5 RIR (*Regional Internet Registries*, registre Internet régional)
- Les RIR coopèrent avec l'ICANN (*Internet Cooperation for Assigned Names and Numbers*)

Épuisement des adresses IPv4

- Seulement (en théorie) 2^{32} adresses $\rightarrow 4,3$ milliards !
- Beaucoup de besoins : terminaux mobiles, modems connectés en permanence, adressage inefficace (grandes entreprises, universités), virtualisation



Vers IPv6

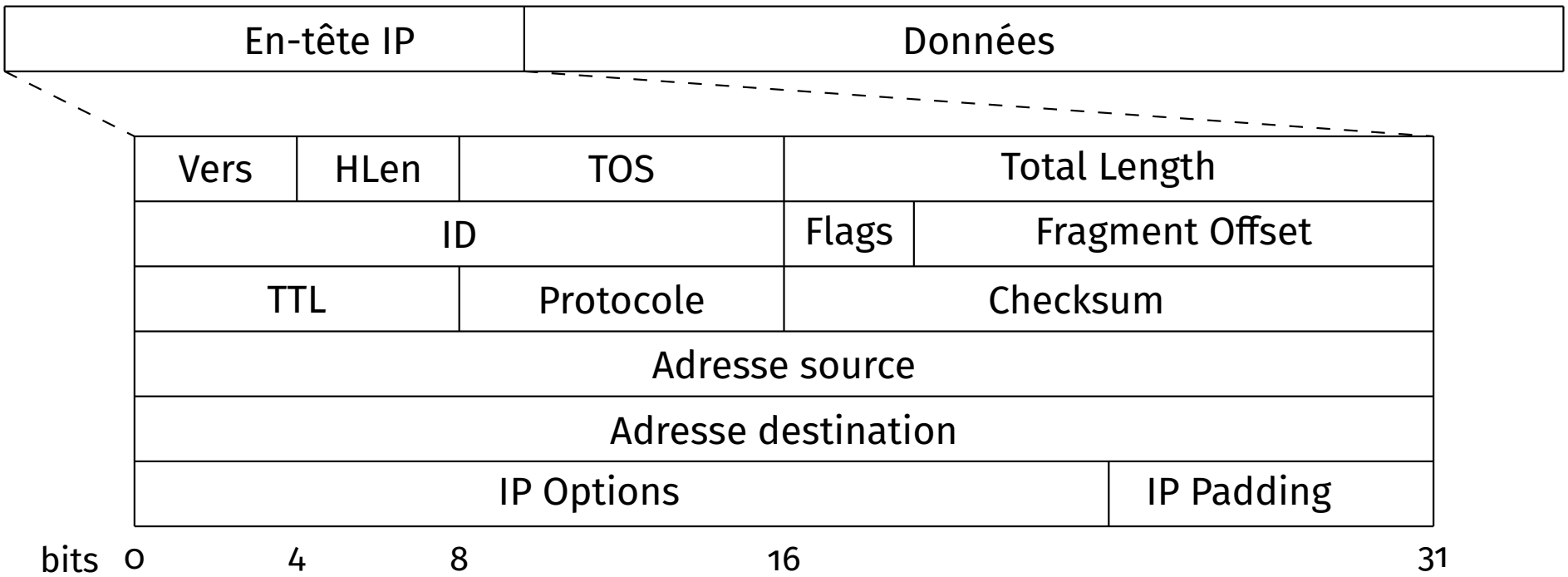
Solution à l'épuisement des adresses IPv4 : IPv6 ?

- Adresses sur 128 bits
- Combien d'adresses IPv6 peut-on attribuer ?
  $2^{128} \approx 3,4 \times 10^{38}$ **(≈ 340 milliards de milliards de milliards de milliards)**
- Exemple : `framasoft.org` a pour IPv6 `2a01:4f8:141:3421::212`

```
nslookup framasoft.org
Server:    127.0.0.53
Address:   127.0.0.53#53

Non-authoritative answer:
Name: framasoft.org
Address: 144.76.131.212
Name: framasoft.org
Address: 2a01:4f8:141:3421::212
```

Paquet IPv4 (1/3)



Paquet IPv4 (2/3)

- VERS : Version du protocole
  - 4 : IPv4
  - 5 : version expérimentale
  - 6 : IPv6
- HLEN : Longueur du header en unités de 32 bits (sans data)
- TOS : Qualité de service (*QoS*), etc…
- Total Length : Longueur totale (en octets), header et données
- ID : identifiant unique pour le réassemblage de paquets
- Flag : bits de contrôle
  - bit 1 : 0 (réservé)
  - bit 2, DF : 0 signifie fragmentation autorisée, 1 non
  - bit 3, MF : 0 signifie dernier fragment, 1 signifie d'autres fragments suivent

Paquet IPv4 (3/3)

- Fragment Offset : nombre de segments de 64 bits (sans header) déjà transmis dans des fragments précédents
- TTL (*Time To Live*) : nombre de routeurs que le paquet peut traverser
- Protocole :
  - 0 : réservé
  - 1 : Internet Control Message Protocol (ICMP)
  - 4 : IP (encapsulation IP)
  - 6 : TCP
  - 17 : UDP
  - ...
- Somme de contrôle (*checksum*)

Exemple 1

```
45000047
d62c4000
401170c2
c0a83602
d41b28f1
b6c40100
...
```

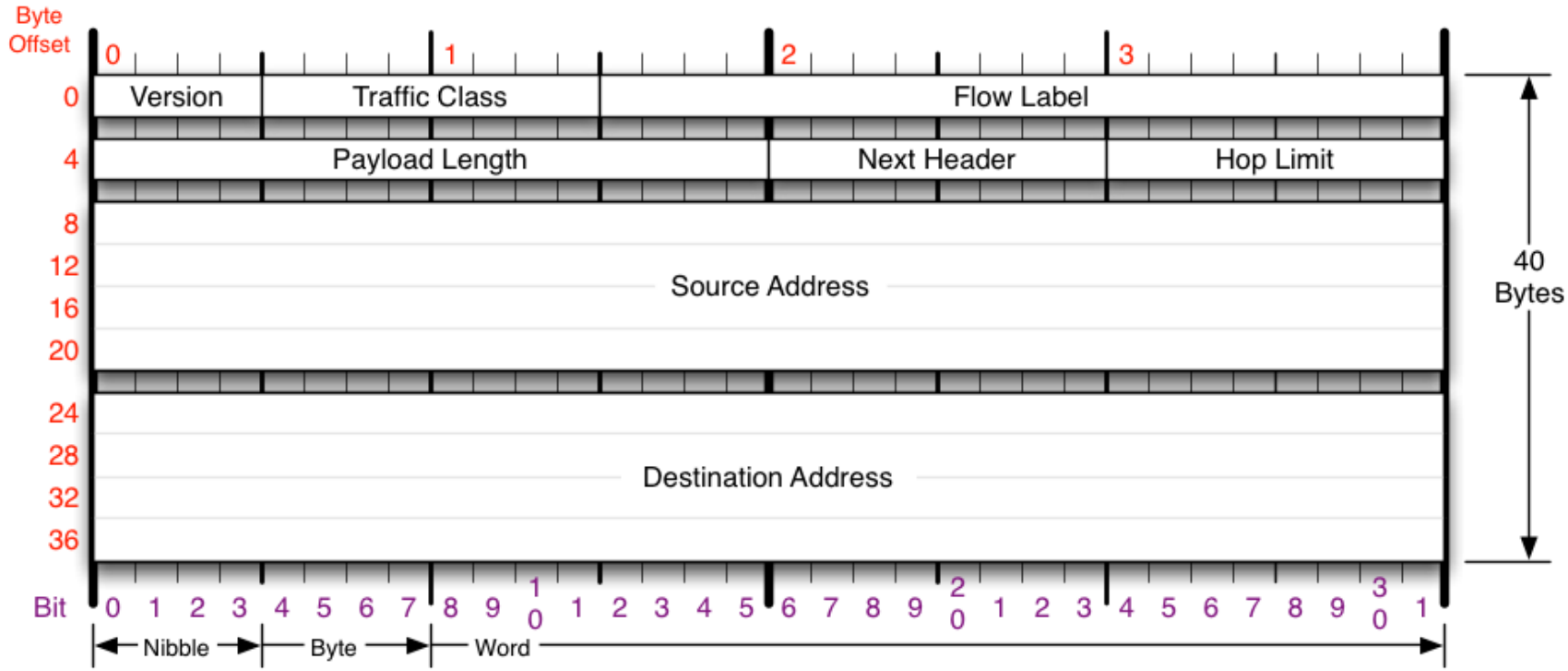
Exemple 2

```
450005dc
b0462000
4001b689
c0a836fe
c0a83602
00004205
...
```

Fragmentation IP

- Utilisée si le protocole de niveau 2 (par exemple Ethernet) ne permet pas de transmettre des trames de taille suffisante
- **Flags :** indique si la fragmentation est autorisée, et si il y a d'autres fragments pour le paquet courant
- **Identification :** indique le numéro du paquet IP fragmenté
- **Fragment Offset :** indique la position du fragment (en octets depuis le début des informations)

IPv6 Header



Version

Version of IP Protocol. 4 and 6 are valid. This diagram represents version 6 structure only.

Traffic Class

8 bit traffic class field.

Flow Label

20 bit flow label.

Payload Length

16-bit unsigned integer. Length of the IPv6 payload, i.e., the rest of the packet following this IPv6 header, in octets. Any extension headers are considered part of the payload.

Source Address

128-bit address of the originator of the packet.

Next Header

8-bit selector. Identifies the type of header immediately following the IPv6 header. Uses the same values as the IPv4 Protocol field.

Destination Address

128-bit address of the intended recipient of the packet (possibly not the ultimate recipient, if a Routing header is present).

Hop Limit

8-bit unsigned integer. Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.

RFC 2460

Please refer to RFC 2460 for the complete Internet Protocol version 6 (IPv6) Specification.

Routage IP

Routeur : nœud intermédiaire connecté à ≥ 2 réseaux

Lorsqu'un paquet arrive dans un routeur, celui-ci le retransmet :

- Soit à la machine destinataire si celle-ci est directement connectée au routeur
- Soit vers un autre routeur auquel il est directement connecté

Le choix de la *route* se fait à partir d'une **table de routage** :

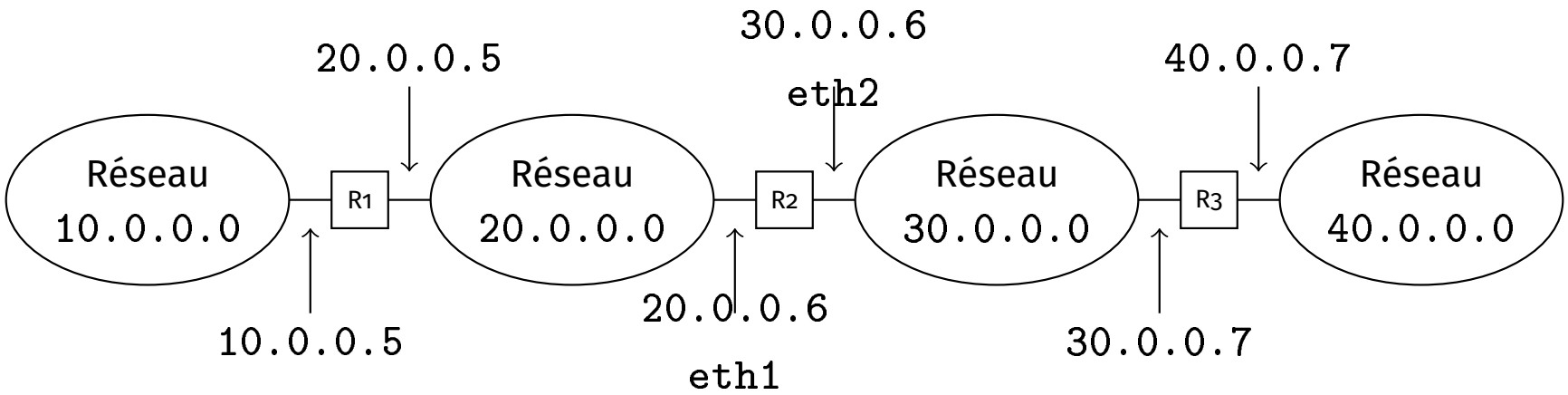
- Le routeur choisit la règle de préfixe le plus long correspondant à la destination

Table de routage

- Au moins 3 colonnes :
  - Adresse de réseau de destination
  - Passerelle à utiliser
  - Interface à utiliser pour joindre la passerelle
- Route par défaut : route utilisée lorsqu'il n'y a pas d'entrée correspondante dans la table de routage

```
$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags
    Iface
192.168.2.0     0.0.0.0         255.255.255.0   U
    eth0
172.16.16.0     0.0.0.0         255.255.255.0   U
    eth2
129.88.98.0     0.0.0.0         255.255.255.0   U
    eth1
131.254.202.0   172.16.16.254   255.255.254.0   UG
    eth2
138.96.20.0     172.16.16.254   255.255.252.0   UG
```

Table de routage : exemple



Destination	Passerelle	Interface
20.0.0.0	DIRECT	eth1
30.0.0.0	DIRECT	eth2
10.0.0.0	20.0.0.5	eth1
40.0.0.0	30.0.0.7	eth2

Table de routage de R2

Exercice : déduire un réseau à partir d'une table

Voici la table de routage d'une machine :

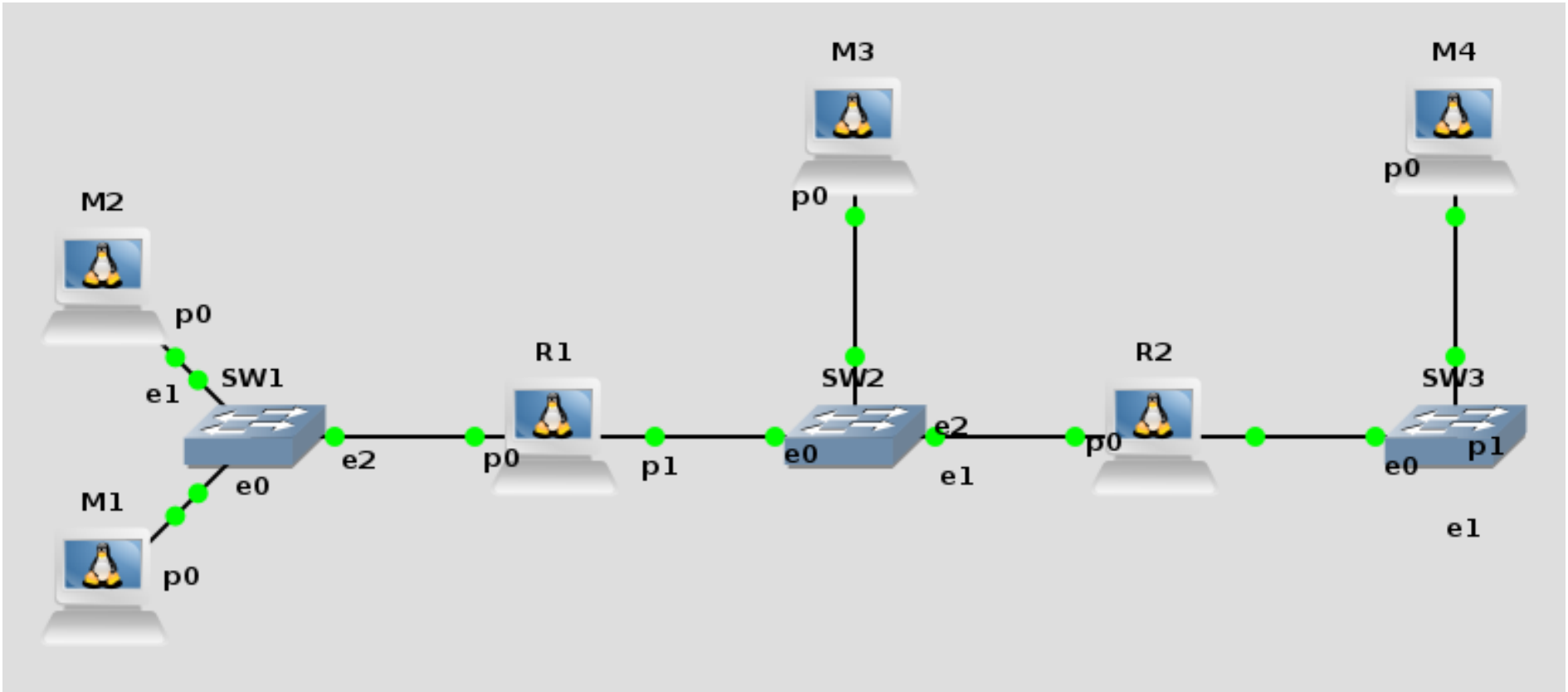
(G indique que la colonne Gateway contient une passerelle)

(H signifie « *host* » (« moi-même »))

Destination	Gateway	Interface	Flags	Netmask
127.0.0.0	127.0.0.1	lo0	UH	255.0.0.0
152.16.2.0	152.16.2.254	eth0	U	255.255.255.0
152.16.3.0	152.16.3.254	fxp0	U	255.255.255.0
152.16.1.0	152.16.1.254	fxp1	U	255.255.255.0
128.121.0.0	152.16.2.253	eth0	UG	255.255.0.0
default	152.16.3.253	fxp0	UG	0.0.0.0

1. Combien de cartes réseaux y a-t-il sur cette machine ? Combien d'adresses IP ?
2. Dessiner la carte du réseau que vous pouvez déduire de cette table de routage.

Exercice sur les tables



- Combien de réseaux de niveau 2 y a-t-il ?
- Proposer un plan d'adressage (on prendra `10.1.0.0/16` pour le réseau de gauche, puis `.2` puis...)
- Donner toutes les tables de routage (minimales)

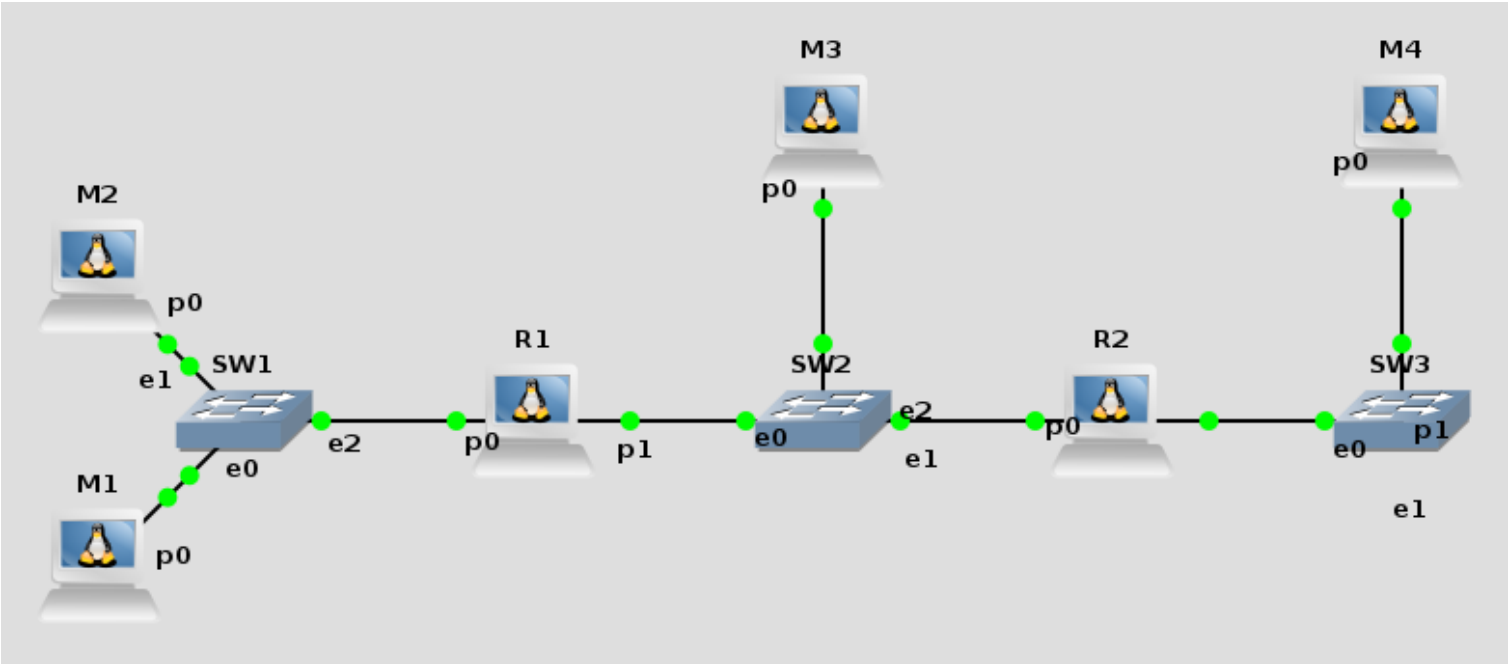
Exercice sur les tables : correction M1

Destination	Gateway	Interface	Flags	Netmask
127.0.0.0	127.0.0.1	lo0	UH	255.0.0.0
10.1.0.0	10.1.0.1	p0	U	255.255.0.0
10.2.0.0	10.1.255.254	p0	UG	255.255.0.0
10.3.0.0	10.1.255.254	p0	UG	255.255.0.0

ou bien

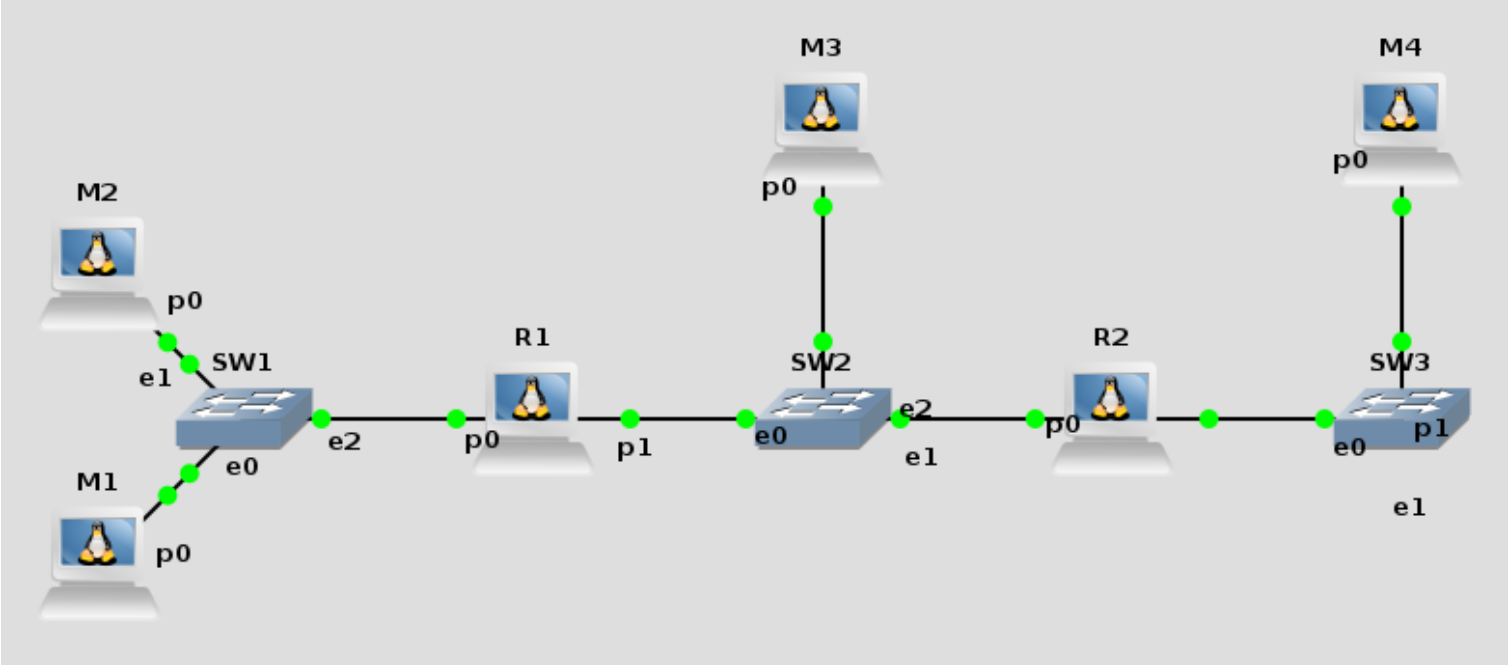
Destination	Gateway	Interface	Flags	Netmask
127.0.0.0	127.0.0.1	lo0	UH	255.0.0.0
10.1.0.0	10.1.0.1	p0	U	255.255.0.0
default	10.1.255.254	p0	UG	0.0.0.0

Exercice sur les tables : correction M3



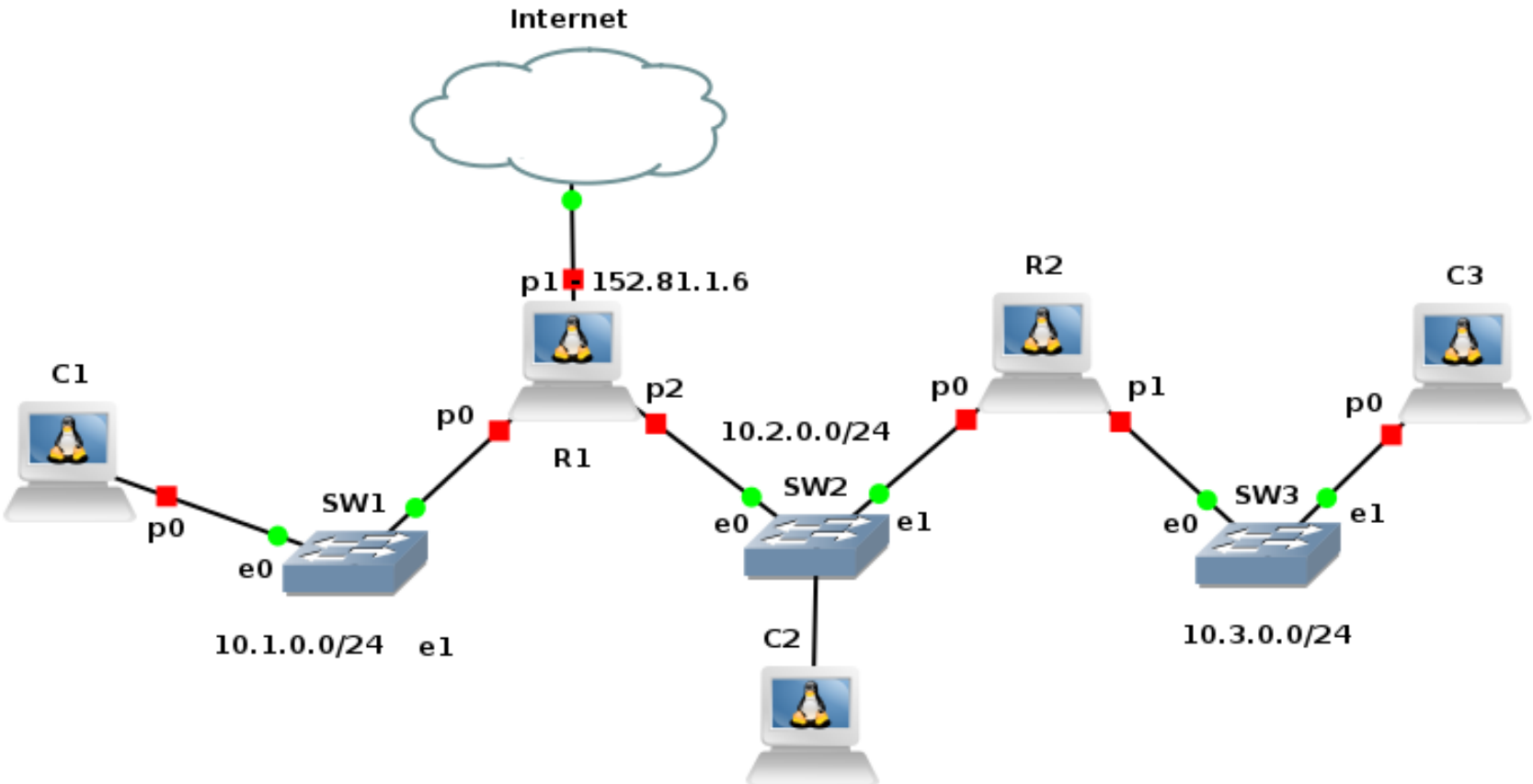
Destination	Gateway	Interface	Flags	Netmask
127.0.0.0	127.0.0.1	lo0	UH	255.0.0.0
10.2.0.0	10.2.0.1	p0	U	255.255.0.0
10.1.0.0	10.2.255.253	p0	UG	255.255.0.0
10.3.0.0	10.2.255.254	p0	UG	255.255.0.0

Exercice sur les tables : correction R1



Destination	Gateway	Interface	Flags	Netmask
127.0.0.0	127.0.0.1	lo0	UH	255.0.0.0
10.1.0.0	10.1.255.254	p0	U	255.255.0.0
10.2.0.0	10.2.255.253	p1	U	255.255.0.0
10.3.0.0	10.2.255.254	p1	UG	255.255.0.0

Exercice : routage (1/2)



Exercice : routage (2/2)

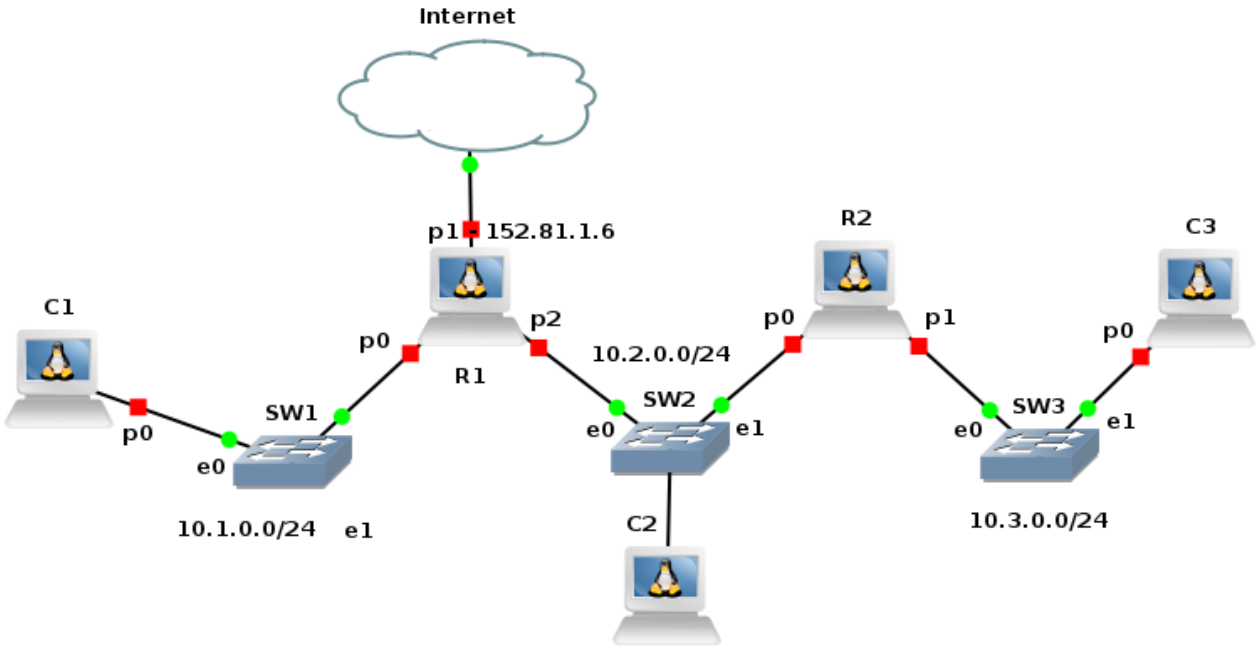
Les tables de routage fournies doivent être minimales (pas de lignes inutiles), mais permettre à chaque machine ou routeur de joindre toutes les autres machines du réseau (et d'Internet).

- Sur le schéma de la page précédente, proposer un plan d'adressage en annotant le sujet. Les machines clientes doivent être en début de plage d'adresse ; les routeurs doivent être en fin de plage.
- Donner les tables de routage de C1, C2, R2

Exercice routage : solution adressage

- **C1 :** `10.1.0.1`
- **R1 (**`p0`**) :** `10.1.0.254`
- **R1 (**`p1`**) :** `152.81.1.6`
- **R1 (**`p2`**) :** `10.2.0.254`
- **C2 :** `10.2.0.1`
- **R2 (**`p0`**) :** `10.2.0.253`
- **R2 (**`p1`**) :** `10.3.0.254`
- **C3 :** `10.3.0.1`

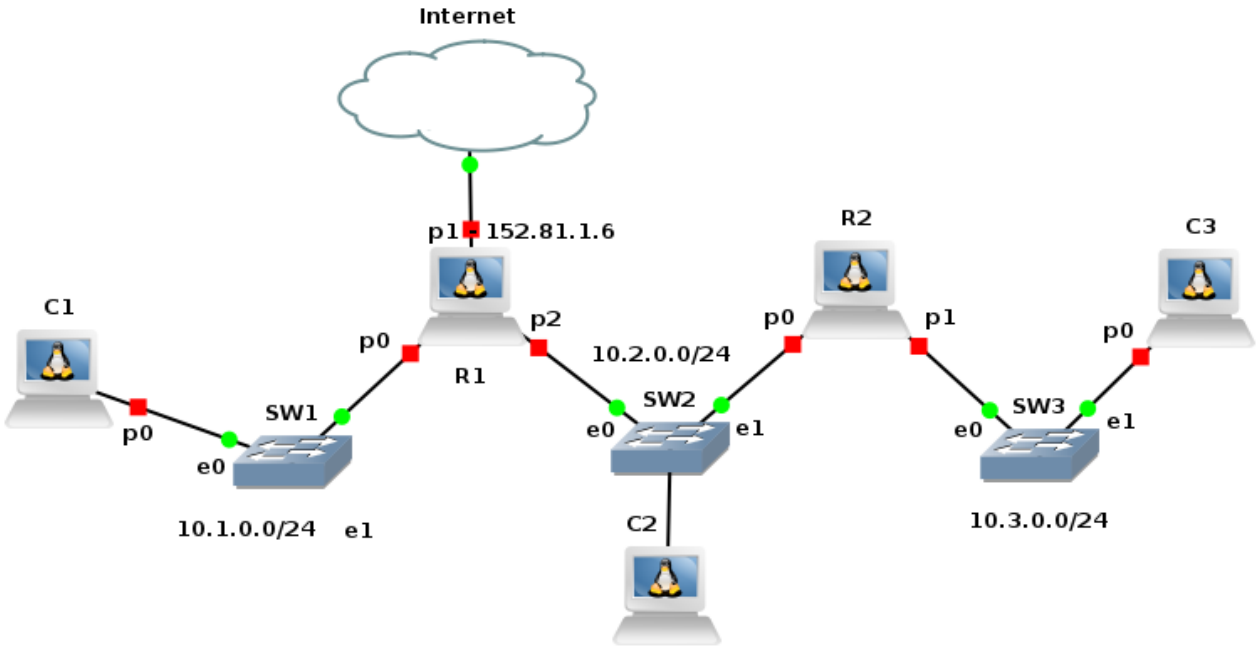
Exercice routage : correction C1



Destination	Gateway	Interface	Flags	Netmask
127.0.0.0	127.0.0.1	lo0	UH	255.0.0.0
10.1.0.0	10.1.0.1	p0	U	255.255.255.0
10.2.0.0	10.1.0.254	p0	UG	255.255.255.0
10.3.0.0	10.1.0.254	p0	UG	255.255.255.0
default	10.1.0.254	p0	UG	0.0.0.0

Les 3e et 4e lignes sont facultatives, car subsumées par la route par défaut

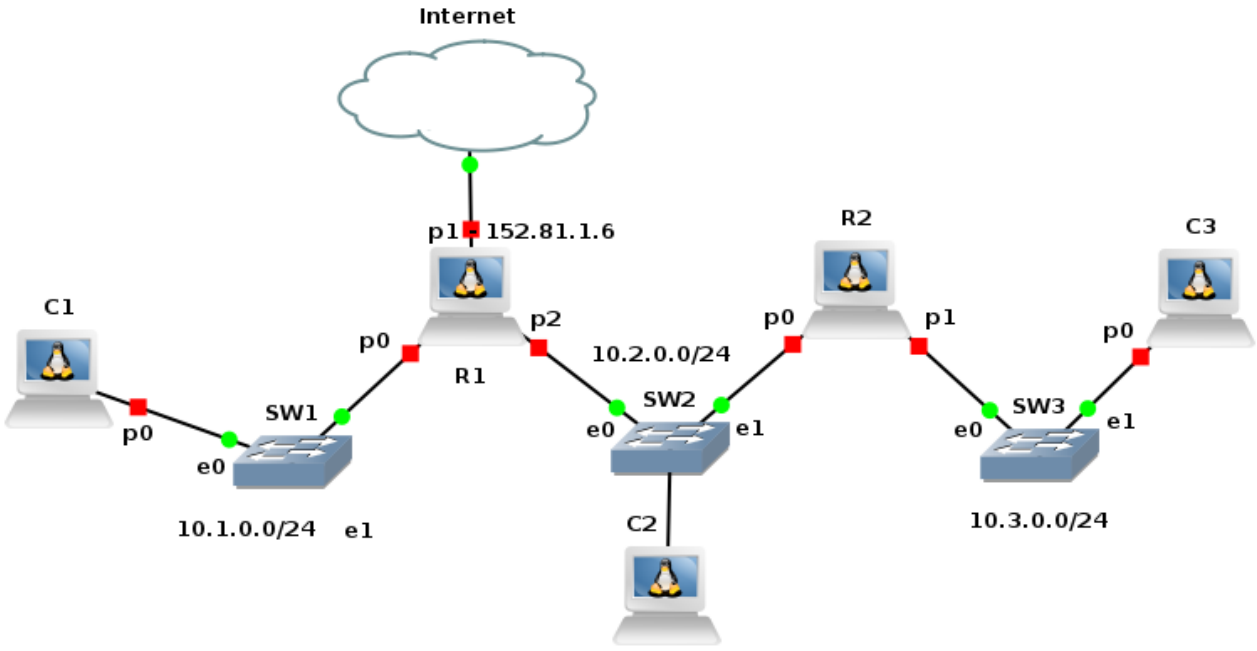
Exercice routage : correction C2



Destination	Gateway	Interface	Flags	Netmask
127.0.0.0	127.0.0.1	lo0	UH	255.0.0.0
10.2.0.0	10.2.0.1	p0	U	255.255.255.0
10.1.0.0	10.2.0.254	p0	UG	255.255.255.0
10.3.0.0	10.2.0.253	p0	UG	255.255.255.0
default	10.2.0.254	p0	UG	0.0.0.0

La 3e ligne est facultative, car subsumée par la route par défaut

Exercice routage : correction R2



Destination	Gateway	Interface	Flags	Netmask
127.0.0.0	127.0.0.1	lo0	UH	255.0.0.0
10.2.0.0	10.2.0.253	p0	U	255.255.255.0
10.3.0.0	10.3.0.254	p1	U	255.255.255.0
10.1.0.0	10.2.0.254	p0	UG	255.255.255.0
default	10.2.0.254	p0	UG	0.0.0.0

La 4e ligne est facultative, car subsumée par la route par défaut

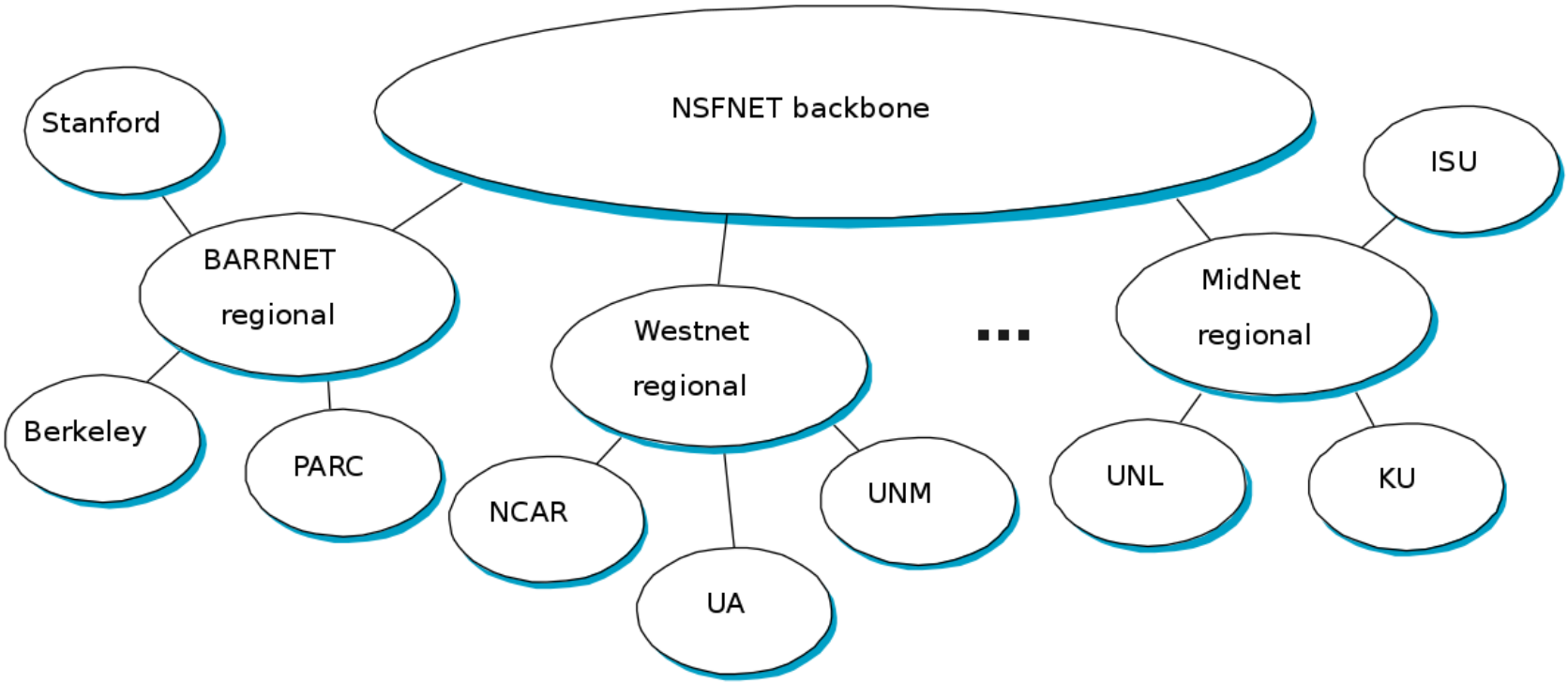
Remplissage de la table de routage

- Routage statique : table remplie manuellement
  - Ne permet pas de détecter les changements de topologie (pannes, liens engorgés)
- Routage dynamique :
  - Apprentissage automatique des différents réseaux accessibles
  - Protocole de routage :
    - À l'intérieur d'un réseau d'opérateur :
      RIP, OSPF, ISIS
    - Entre des opérateurs : BGP

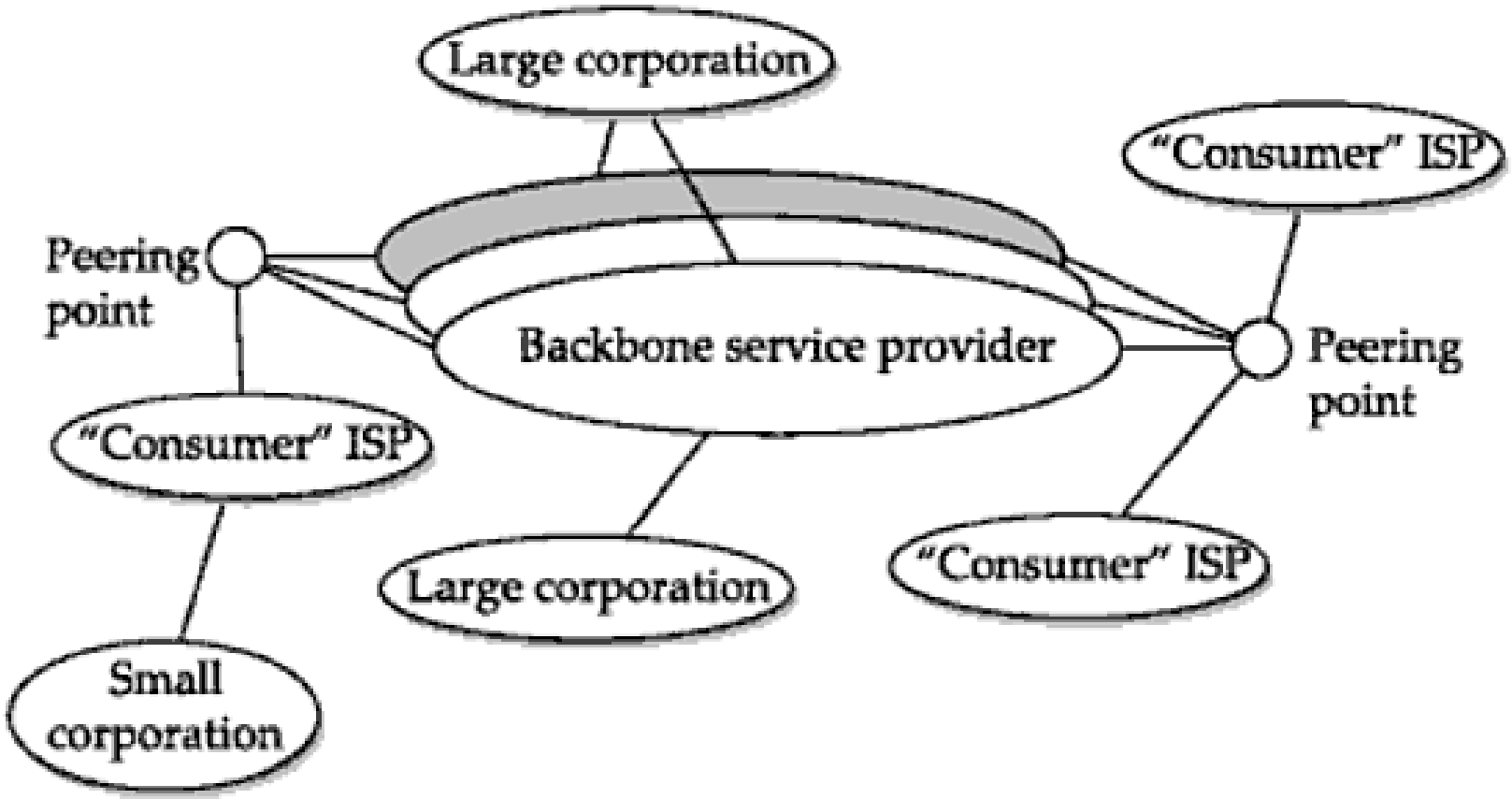
Pour éviter les boucles infinies dans le routage :

- Champ TTL (*Time to live*)
- Décrémenté à chaque passage par un routeur

Internet en 1990



Internet aujourd'hui

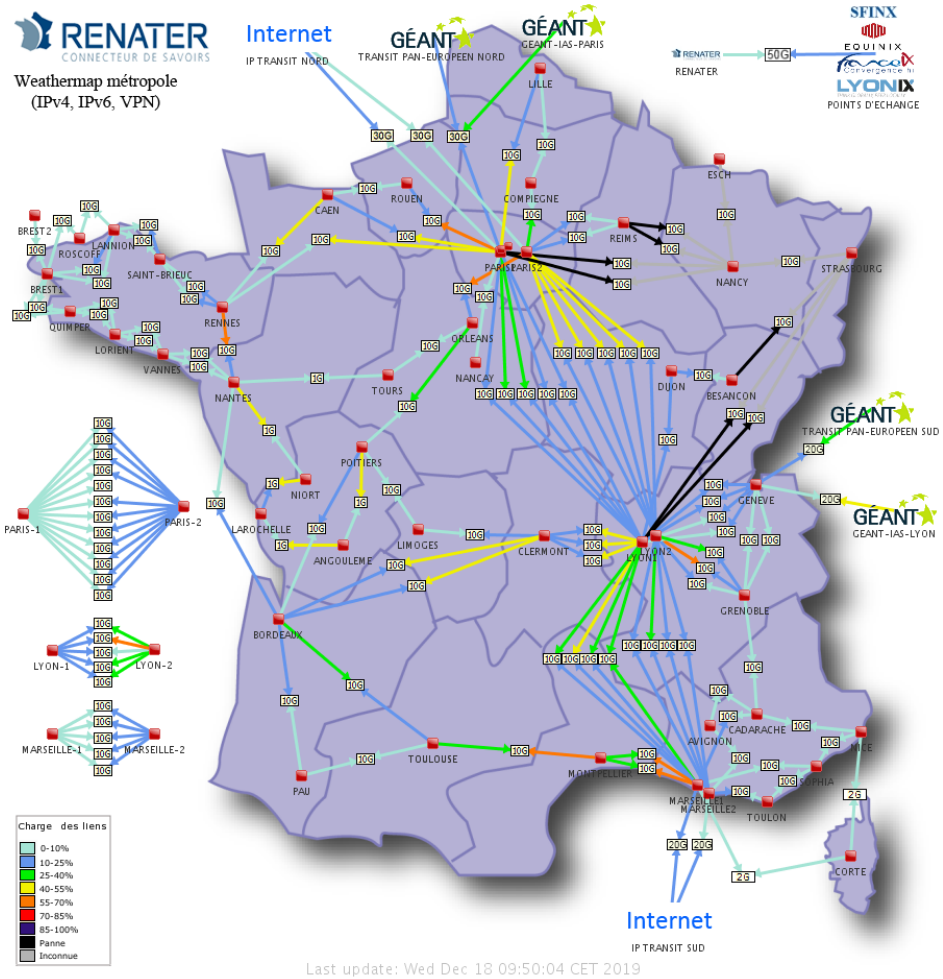


Interconnexion d'opérateurs

Quand on est opérateur, 2 solutions pour pouvoir atteindre d'autres réseaux :

- Payer un fournisseur de transit IP
- Passer des accords réciproques avec d'autres opérateurs (*peering*), souvent dans un Internet Exchange Point (IXP, GIX).
  En France : FranceIX, SFINX (Renater), etc.
- Possible source de conflits (2003 : Free vs. France Telecom ; 2008 : Sprint vs. Cogent ; 2012 : Free vs. Youtube)
  - Référence : `https://www.nextinpact.com/article/17376/93407-free-fin-annonce-ralentissements-sur-services-google-dont-youtube`

Le réseau RENATER



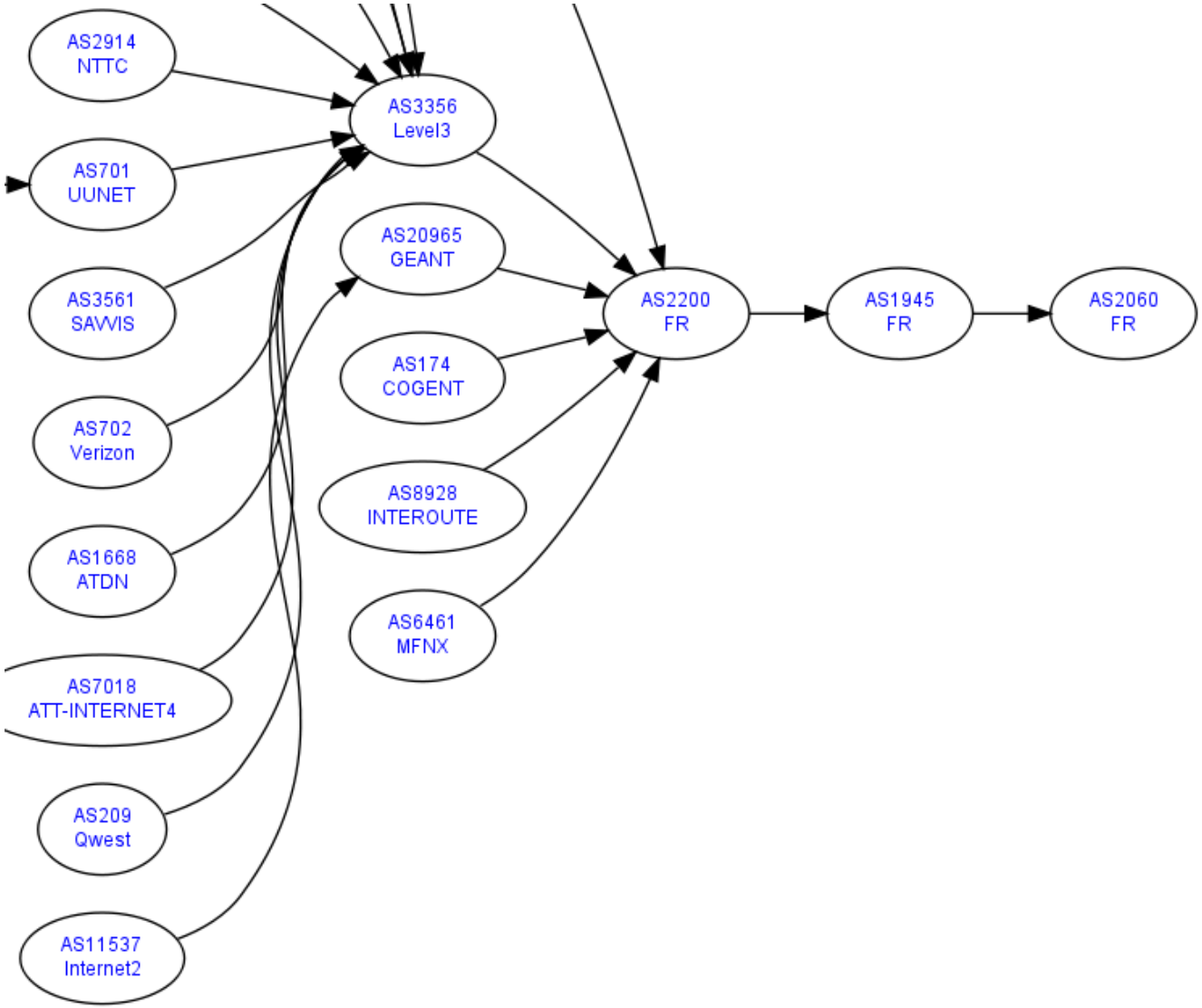
(capture du 18 décembre 2019, jour d'une panne majeure ayant isolé Nancy et Strasbourg du reste du monde)

Internet « Tiers »

- Tier 1 network : Un réseau qui peut atteindre tous les autres réseaux sur Internet sans acheter du transit IP à un fournisseur ou payer un autre opérateur.

  AT&T, Global Crossing, Level 3, NTT (Verio), Qwest, Sprint, Verizon (ex-UUNET), SAVVIS, AOL, AboveNet, Cogent, TeliaSonera, Teleglobe, XO Communications, ...
- Tier 2 network : réseau qui a passé des accords de peering avec d'autres opérateurs, mais qui achète aussi du transit IP.
- Tier 3 network : réseau qui assure sa connectivité uniquement en achetant du transit.

Exemple



http://as.robtex.com/as2200.html

DHCP

- *Dynamic Host Configuration Protocol*
- Configuration automatique des machines
- Attribution d'une adresse IP, et d'autres paramètres :
  - Masque de sous-réseau
  - Passerelle par défaut
  - Serveurs DNS
  - ...
- Protocole (utilise UDP) :
  - DHCP Discover : demande par le client (diffusion)
  - DHCP Offer : offre du serveur
  - DHCP Request : acceptation de l'offre par le client
  - DHCP ACK : notification que l'IP a été attribuée au client
- Attribution temporaire : bail (*lease*) DHCP

Couche Transport : TCP et UDP

- Permet le multiplexage de plusieurs communications (numéros de ports, sur 16 bits)
- 2 protocoles de transport : UDP et TCP
  - UDP (*User Datagram Protocol*) :
    - Service de transport minimaliste et simple
    - Mode non-connecté (pas de sessions)
    - Pas de contrôle d'erreur ni de flux
    - Échange de **datagrammes**
    - Idéal pour les protocoles simples (DNS) ou ceux où le développeur veut tout contrôler lui-même (QUIC)
  - TCP (*Transmission Control Protocol*) :
    - Protocole complexe, masquant les contraintes du réseau aux applications
    - Mode connecté (**connexion TCP**), transport fiable d'un flot d'octets
    - Échange de **segments**
    - Le protocole de transport le plus utilisé (car facile du point de vue du développeur)

Quelques numéros de ports

Port	Service
22 / TCP	SSH
23 / TCP	telnet
25 / TCP	SMTP
53 / UDP	DNS
67 / UDP	BOOTP / DHCP
68 / UDP	BOOTP / DHCP
80 / TCP	HTTP
110 / TCP	POP3
137, 138, 139	NetBIOS
143 / TCP	IMAP
443 / TCP	HTTPS

ARP

- *Address Resolution Protocol*
- À l'interface entre couches Réseau et Liaison
- Problème : comment savoir à quelle adresse MAC envoyer un paquet IP ?
- ARP = Protocole de découverte des adresses MAC
- 2 types de paquets :
  - ARP request (broadcast) :
    « Qui a l'IP `192.168.1.2` ? »
  - ARP reply (unicast) :
    « `192.168.1.2` est à `00:07:cb:c7:52:5f` »
- Utilisation d'un cache
- Exercice : sur le réseau de l'exercice précédent, C1 envoie un paquet à C2. Toutes les tables ARP sont vides. Quelles sont toutes les trames échangées ?

Configuration réseau sous Linux

- Deux jeux de commandes pour les modifications transitoires :
  - Historiques : `ifconfig`, `route`

```
ifconfig eth0 up # ou down pour arrêter
ifconfig eth0 10.0.0.1/24
route add default gw 10.0.0.254
route add -net 2.2.2.0/24 gw 1.2.3.4
netstat -rn # affiche la table de routage; -n é
    vite la résolution DNS
route # affiche aussi la table de routage
```

  - Modernes (paquet `iproute2`, conseillé) : `ip`

```
ip link show # peut être abrégé: ip l
ip addr show # => ip a
ip route show # => ip r
ip link set dev eth0 up # => ip l s eth0 up
ip addr add dev eth0 10.0.0.1/24
ip route add 192.0.2.128/25 via 192.0.2.1
```

voir `http://baturin.org/docs/iproute2/` et une cheat sheet de Red Hat

Configuration réseau persistante (1/2)

- Dans `/etc/network/interfaces` (voir `interfaces(5)`)

```
auto eth1
iface eth1 inet static
      address 192.168.1.2/24
      gateway 192.168.1.1 # va ajouter une route par défaut
      # si nécessaire, pour ajouter des routes supplémentaires
      # (ou exécuter d'autres commandes)
      post-up ip route add 10.0.0.0/8 via 192.168.1.254
      # si le paquet resolvconf est installé, on peut indiquer
      # la configuration DNS
      dns-nameserver 8.8.8.8
      dns-search foo.com

auto eth2
iface eth2 dhcp
```

- Recharger toute la configuration : `service networking restart`
- Dé-configurer une interface : `ifdown eth1`
- Configurer une interface : `ifup eth1`

Configuration réseau persistante (2/2)

- D'autres outils
  - Orientés serveurs
    - `/etc/sysconfig/` (équivalent à `/etc/network/interfaces` dans le monde RedHat)
    - `systemd-networkd` (remplaçant intégré à `systemd`)
  - Orientés poste de travail
    - Network-Manager
    - `wicd`

Activation du routage sous Linux

- Par défaut, Linux ne route pas entre ses différentes interfaces
- Pour changer ce comportement :
  - De manière transitoire (oublié après un redémarrage) :
    - ```
      echo 1 > /proc/sys/net/ipv4/ip_forward
      ```
    - ou

      ```
      sysctl -w net.ipv4.ip_forward=1
      ```
  - De manière persistante :
    - Ajouter dans `/etc/sysctl.conf` : `net.ipv4.ip_forward=1`
    - Ou plus proprement, créer un fichier `/etc/sysctl.d/01-ipforwarding.conf` et y ajouter

      ```
      1 net.ipv4.ip_forward=1
      ```
      ```

Noms des interfaces

- Historiquement : eth0, eth1, ...
- Depuis récemment : noms prédictibles
  - Dépendent de la position physique de la carte réseau dans l'ordinateur
  - eno1 : sur la carte mère
  - ens1 : PCI slot 1
  - Voir `https://www.freedesktop.org/wiki/Software/systemd/PredictableNetworkInterfaceNames`

GNS3

- Simulateur réseau
- Par rapport à Vagrant
  - ☺ Topologies complexes, avec switches, routeurs, etc.
  - ☹ Pas adapté à des VM avec une configuration complexe de services réseaux
- Par rapport à Marionnet
  - Machines virtuelles peut-être un peu plus récentes

Installation de GNS3 (Ubuntu)

- Voir : https://doc.ubuntu-fr.org/gns3

```
sudo add-apt-repository ppa:gns3/ppa
sudo apt update
sudo apt install gns3-gui gns3-server

# Pour les consoles (xterm devrait déjà être
    présent)
sudo apt-get install telnet xterm
```

Installation de GNS3 (Debian)

- Sous Debian 9 Stretch :

```
echo 'deb [trusted=true]
    http://ppa.launchpad.net/gns3/ppa/ubuntu
    xenial main' >
    /etc/apt/sources.list.d/gns3.list
```

- Sous Debian 10 Buster : voir
  https://people.debian.org/~lucas/gns3-buster/
- Ensuite :

```
apt-get update ; apt-get -y install gns3-gui
```

Premier démarrage de GNS3

- S'ajouter aux groupes `ubridge` et `wireshark`, puis fermer/rouvrir la session Unix
- Lancer GNS3
- Sélectionner « Run appliances on my local computer »
- Laisser les options suivantes par défaut

Import des images Debian9

- Récupérer l'appliance Debian 9 :

```
wget https://people.debian.org/~lucas/gns3/debian9.gns3a

# Attention, gros fichier (~ 500 Mio, peut durer un certain
    temps)
wget
    https://people.debian.org/~lucas/gns3/debian9-0.5.qcow2.zip
```

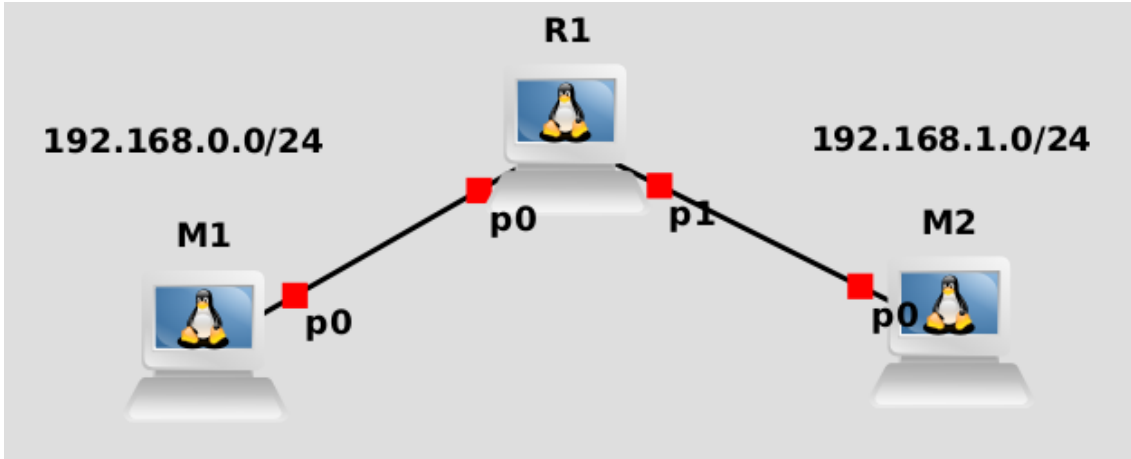
  - Décompresser le fichier .zip

- Dans GNS3, choisir « Import an appliance file (.gns3a extension) », et sélectionner le fichier .gns3a

Quelques indications

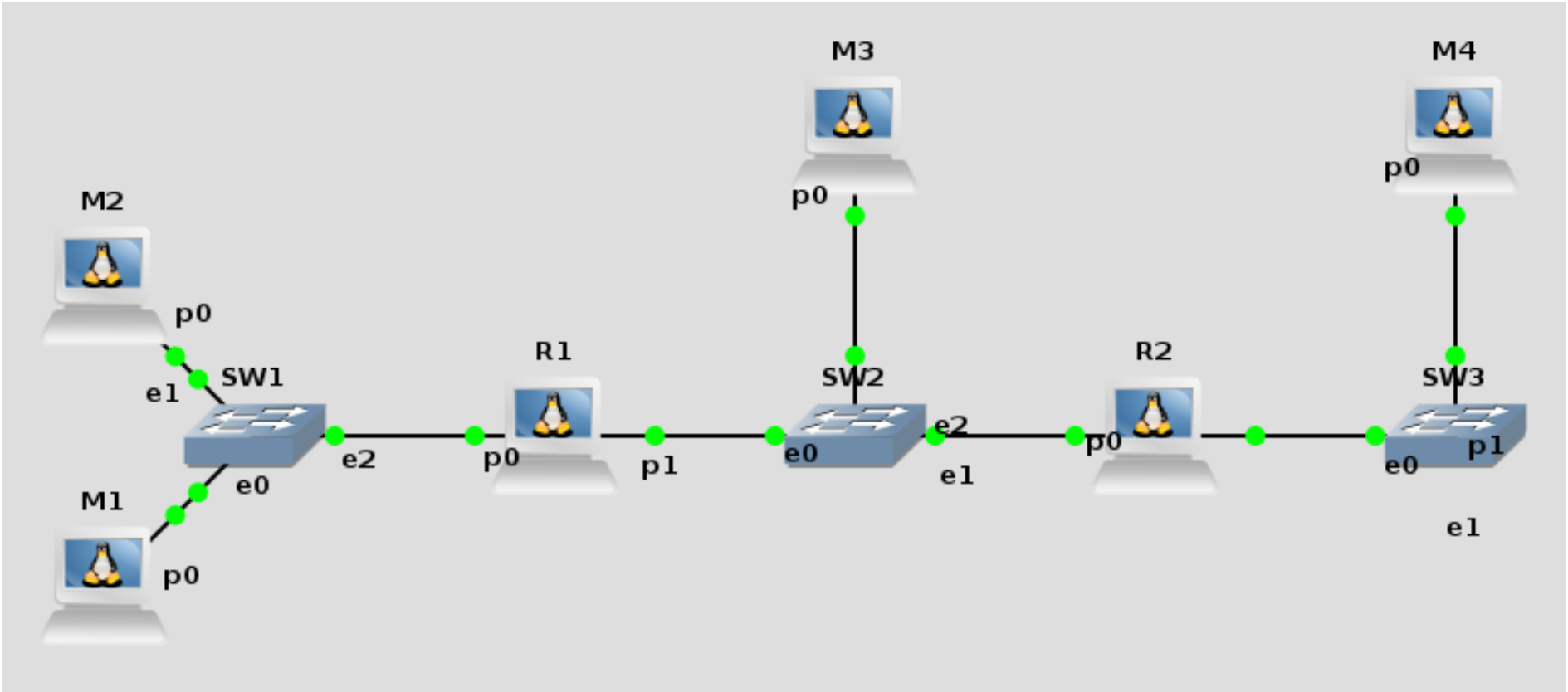
- Les machines ont pour identifiants `root` / `root`
- La machine Debian 9 importée est dans « end device »
- Un routeur est considéré comme une machine classique (Debian 9)
- Ajouter une machine se fait en glisser-déposer depuis la liste des machines
- Ajouter un câble se fait avec « Add a link » puis en cliquant et en maintenant la souris appuyée
- Le nom de l'interface `p0` est `ens3` (puis `ens4`, puis…)

Exercice : routage (1)



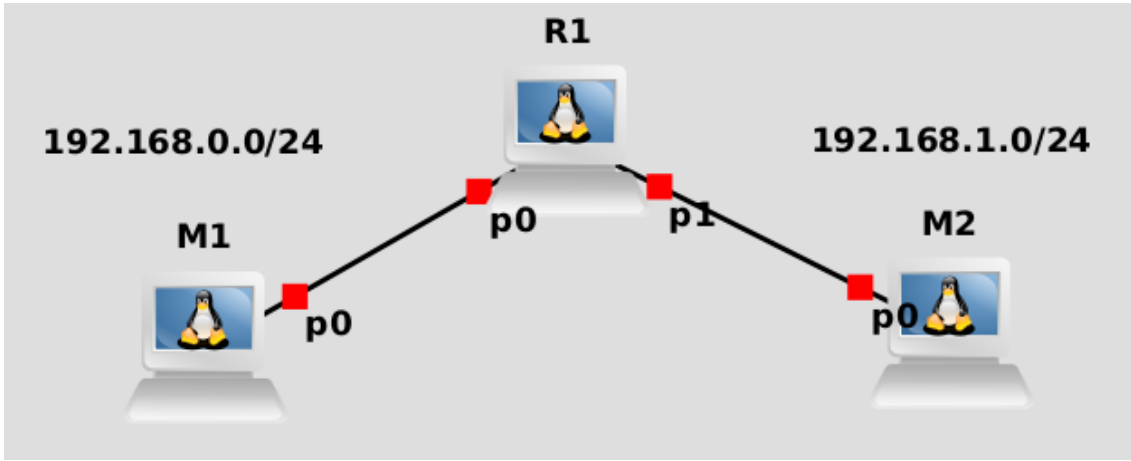
- Réaliser la topologie ci-dessus (R1 est une machine Debian 9 configurée comme un routeur), en configurant les machines d'abord de manière transitoire (avec les deux méthodes), puis de manière persistante
- Avec `ping` et `traceroute`, vérifiez que tout fonctionne bien (sinon, utilisez la fonction de capture de paquets dans GNS3 pour débugger)

Exercice : routage (2)



- Mêmes questions avec cette topologie (il est fortement conseillé de commencer par définir, sur papier, le plan d'adressage et les tables de routage à implémenter)

Exercices : firewall & ARP



- Consultez une introduction à `iptables` si vous n'êtes pas familier avec l'outil.
- Configurez des règles de filtrage sur R1, avec `iptables` :
  - SSH interdit depuis M1 vers M2
  - ICMP (`ping`) autorisé si, sur M2, on ping M1, mais pas l'inverse
- Configurez R1 pour que le réseau `192.168.0.0/24` soit NATé dessus. Utilisez Wireshark (fonction capture de paquets) pour confirmer.
- ARP : observez le contenu des tables ARP avec `ip neigh`, supprimez des entrées (`arp -d` ou `ip neigh`) et observez (avec Wireshark) les requêtes ARP nécessaires au re-peuplement de la table ARP

Exercices : firewall (2)

- Récupérez une appliance Debian 10 et importez-la comme précédemment :

```
wget https://people.debian.org/~lucas/gns3/debian10.gns3a

# Attention, gros fichier (~ 300 Mio, peut durer un certain
    temps)
wget
    https://people.debian.org/~lucas/gns3/debian10-0.1.qcow2.zip
```

- Debian 10 inclut `nftables`, qui remplace `iptables`. Refaites les questions précédentes sur le filtrage avec `nftables` au lieu de `iptables`.
- Documentation utile :
  - `https://wiki.nftables.org/`
  - `https://wiki.debian.org/nftables`

GNS3 : Notes diverses, FAQ, troubleshooting

- On peut changer le nom des machines avec :

```
hostnamectl set-hostname m1p
```

- Problèmes avec `ubridge` : l'installer par paquet, voire en compilant
- Si impossible d'ouvrir les consoles des VM : il faut installer les paquets `xterm` et `telnet`
- Erreur : failed to initialize KVM : Device or resource busy
  - Probablement due à un conflit entre KVM et une autre solution de virtualisation (Virtualbox par exemple).
  - Pour arrêter virtualbox, arrêtez toutes les machines virtuelles, puis :

```
rmmod vboxpci vboxnetflt vboxdrv
```

- Capture de paquets (clic droit sur le lien concerné)
  - Pour que ça fonctionne, il faut que votre compte utilisateur soit ajouté au groupe `wireshark`
  - Autre solution : utiliser `tcpdump` (en console)
- Si impossible d'ajouter un switch : installer le paquet `dynamips`

Crédits

- Cours basé en grande partie sur les supports de Lucas Nussbaum (IUT Charlemagne), version 2020-2021

Version : 5 octobre 2021