Administration réseau
Introduction

A. Guermouche

Plan

1. Introduction
   - Organisation
   - Contenu

2. Quelques Rappels : Internet et le modèle TCP/IP
   - Visage de l'Internet
   - Le modèle TCP/IP

Plan

1. Introduction
   - Organisation
   - Contenu

2. Quelques Rappels : Internet et le modèle TCP/IP
   - Visage de l'Internet
   - Le modèle TCP/IP

Objectifs

Objectif du module : former des administrateurs réseaux

- ⋆ connaître le modèle Client/Serveur (90% des applications de l'Internet)
- ⋆ avoir des notions de conception d'applications Client/Serveur
- ⋆ connaître les protocoles applicatifs de l'Internet et savoir mettre en place les services associés sous Linux et sous Windows

$\Rightarrow$ Manipulation des notions/outils nécessaire à un administrateur réseaux.

Organisation du module

26h de cours + 28h de travaux pratiques
TP par groupe :

1. en salle avec droits d'administration
   - accès restreint à l'extérieur
   - possibilité de câblage
2. en salle standard en utilisant des machines virtuelles *User-mode Linux*.

Dans les deux cas root sur les machines (réelles pour 1 et virtuelles pour 2) ☺

Contrôle continu : un ou plusieurs TPs notés + partiel sur machine.
Examen final : examen de 1h30 concernant le cours.

Administration réseau

Le rôle d'un administrateur réseau consiste (entre autre) à :

- Mettre en place et maintenir l'infrastructure du réseau (organisation, . . .).
- Installer et maintenir les services nécessaires au fonctionnement du réseau.
- Assurer la sécurité des données internes au réseau (particulièrement face aux attaques extérieures).
- S'assurer que les utilisateurs "n'outrepassent" pas leurs droits.
- Gérer les "*logins*" (i.e. noms d'utilisateurs, mot de passe, droits d'accès, permissions particulières, . . .).
- Gérer les systèmes de fichiers partagés et les maintenir.

L'administrateur réseau est responsable de ce qui peut se passer à partir du réseau administré.

Administration réseau

Le rôle d'un administrateur réseau consiste (entre autre) à :

- Mettre en place et maintenir l'infrastructure du réseau (organisation, . . .).
- Installer et maintenir les services nécessaires au fonctionnement du réseau.
- Assurer la sécurité des données internes au réseau (particulièrement face aux attaques extérieures).
- S'assurer que les utilisateurs "n'outrepassent" pas leurs droits.
- Gérer les "*logins*" (i.e. noms d'utilisateurs, mot de passe, droits d'accès, permissions particulières, . . .).
- Gérer les systèmes de fichiers partagés et les maintenir.

L'administrateur réseau est responsable de ce qui peut se passer à partir du réseau administré.

Administration réseau

Le rôle d'un administrateur réseau consiste (entre autre) à :

- Mettre en place et maintenir l'infrastructure du réseau (organisation, . . .).
- Installer et maintenir les services nécessaires au fonctionnement du réseau.
- Assurer la sécurité des données internes au réseau (particulièrement face aux attaques extérieures).
- S'assurer que les utilisateurs "n'outrepassent" pas leurs droits.
- Gérer les "*logins*" (i.e. noms d'utilisateurs, mot de passe, droits d'accès, permissions particulières, . . .).
- Gérer les systèmes de fichiers partagés et les maintenir.

L'administrateur réseau est responsable de ce qui peut se passer à partir du réseau administré.

Administration réseau

Le rôle d'un administrateur réseau consiste (entre autre) à :

- Mettre en place et maintenir l'infrastructure du réseau (organisation, . . .).
- Installer et maintenir les services nécessaires au fonctionnement du réseau.
- Assurer la sécurité des données internes au réseau (particulièrement face aux attaques extérieures).
- S'assurer que les utilisateurs "n'outrepassent" pas leurs droits.
- Gérer les "*logins*" (i.e. noms d'utilisateurs, mot de passe, droits d'accès, permissions particulières, . . .).
- Gérer les systèmes de fichiers partagés et les maintenir.

L'administrateur réseau est responsable de ce qui peut se passer à partir du réseau administré.

Administration réseau

Le rôle d'un administrateur réseau consiste (entre autre) à :

- Mettre en place et maintenir l'infrastructure du réseau (organisation, ...).
- Installer et maintenir les services nécessaires au fonctionnement du réseau.
- Assurer la sécurité des données internes au réseau (particulièrement face aux attaques extérieures).
- S'assurer que les utilisateurs "n'outrepassent" pas leurs droits.
- Gérer les "*logins*" (i.e. noms d'utilisateurs, mot de passe, droits d'accès, permissions particulières, ...).
- Gérer les systèmes de fichiers partagés et les maintenir.

L'administrateur réseau est responsable de ce qui peut se passer à partir du réseau administré.

Administration réseau

Le rôle d'un administrateur réseau consiste (entre autre) à :

- Mettre en place et maintenir l'infrastructure du réseau (organisation, ...).
- Installer et maintenir les services nécessaires au fonctionnement du réseau.
- Assurer la sécurité des données internes au réseau (particulièrement face aux attaques extérieures).
- S'assurer que les utilisateurs "n'outrepassent" pas leurs droits.
- Gérer les "*logins*" (i.e. noms d'utilisateurs, mot de passe, droits d'accès, permissions particulières, ...).
- Gérer les systèmes de fichiers partagés et les maintenir.

L'administrateur réseau est responsable de ce qui peut se passer à partir du réseau administré.

Administration réseau

Le rôle d'un administrateur réseau consiste (entre autre) à :

- Mettre en place et maintenir l'infrastructure du réseau (organisation, ...).
- Installer et maintenir les services nécessaires au fonctionnement du réseau.
- Assurer la sécurité des données internes au réseau (particulièrement face aux attaques extérieures).
- S'assurer que les utilisateurs "n'outrepassent" pas leurs droits.
- Gérer les "*logins*" (i.e. noms d'utilisateurs, mot de passe, droits d'accès, permissions particulières, ...).
- Gérer les systèmes de fichiers partagés et les maintenir.

L'administrateur réseau est responsable de ce qui peut se passer à partir du réseau administré.

Contenu

- Routage et passerelle :
  - Configuration d'une passerelle.
  - Configuration d'un réseau privé : NAT (**N**etwork **A**ddress **T**ranslation), IP masquerading . . .
- Sécurité dans les réseaux:
  - Configuration de pare-feu (*firewall*):
    - Manipulation des tables *iptables*.
    - Règles de filtrage.
    - . . .
  - Outils de diagnostic :
    - *nmap*
    - . . .
- Configuration et manipulation de services spécifiques :
  - Gestion d'utilisateurs distants (NIS)
  - Un annuaire fédérateur (LDAP)
  - Transfert de fichiers et autres (FTP, TFTP, NFS, SMB)
  - Connexions à distance (telnet, rlogin, ssh, X11, ...)
  - Les serveurs de noms (DNS)

Plan

1. Introduction
   - Organisation
   - Contenu

2. Quelques Rappels : Internet et le modèle TCP/IP
   - Visage de l'Internet
   - Le modèle TCP/IP

Le visage de l'Internet (1)

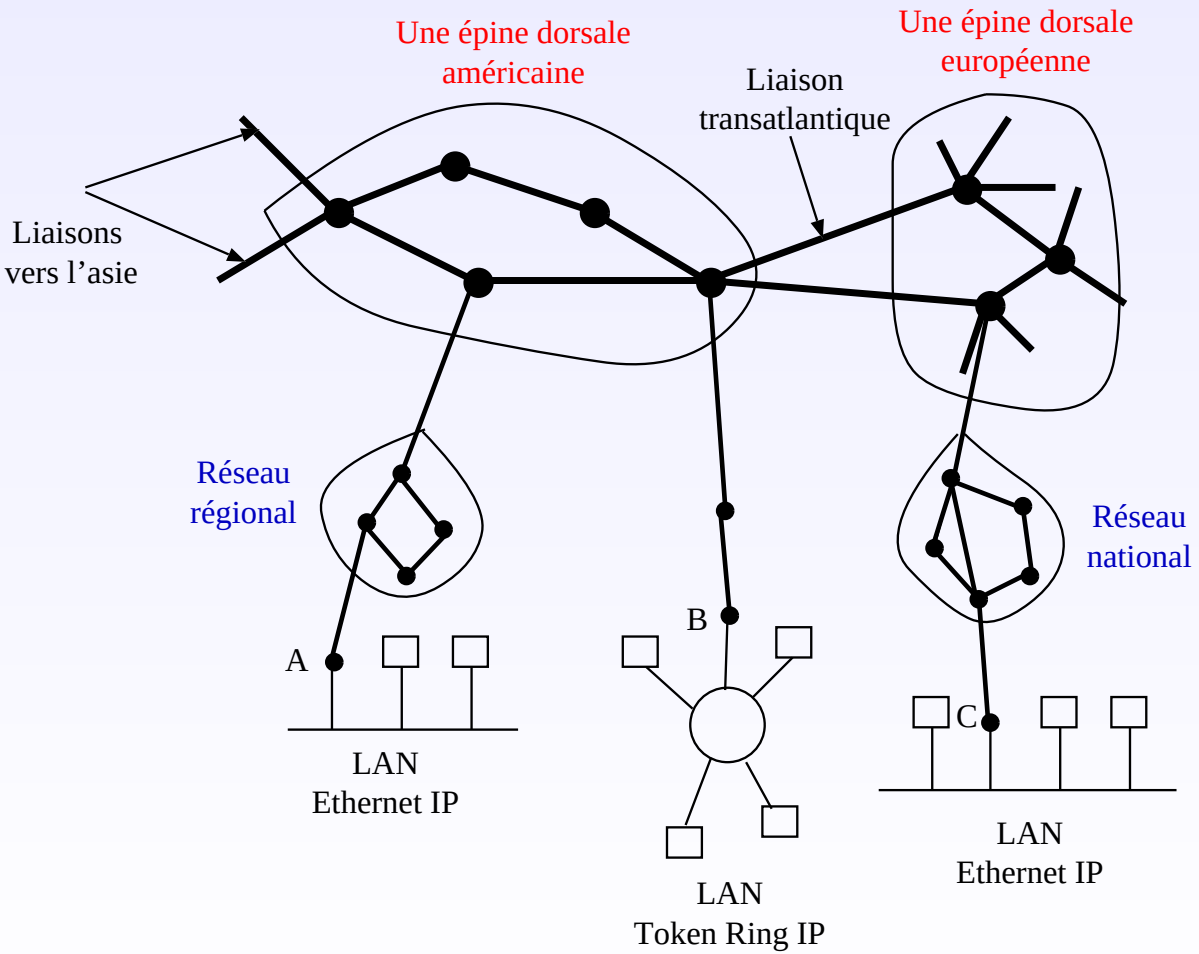
- Un réseau de réseaux
- Un ensemble de logiciels et de protocoles
- Basé sur l'architecture TCP/IP
- Fonctionne en mode Client/Serveur
- Offre un ensemble de services (e-mail, transfert de fichiers, connexion à distance, WWW, . . .)
- Une somme « d'inventions » qui s'accumulent
  - mécanismes réseau de base (TCP/IP)
  - gestion des noms et des adresses
  - des outils et des protocoles spécialisés
  - le langage HTML

Le visage de l'Internet (2)

- Une construction à partir du « bas »
  - réseau local (laboratoire, département)
  - réseau local (campus, entreprise)
  - réseau régional
  - réseau national
  - réseau mondial
- 3 niveaux d'interconnexion
  - postes de travail (ordinateur, terminal...)
  - liaisons physiques (câble, fibre, RTC...)
  - routeurs (équipement spécialisé, ordinateur...)

Le visage de l'Internet (3)

Un ensemble de sous-réseaux indépendants (Autonomous System) et hétérogènes qui sont interconnectés (organisation hiérarchique)



L'architecture de TCP/ IP (1)

Une version simplifiée du modèle OSI

Application FTP, WWW, telnet, SMTP, ...

Transport TCP, UDP (entre 2 processus aux extrémités)

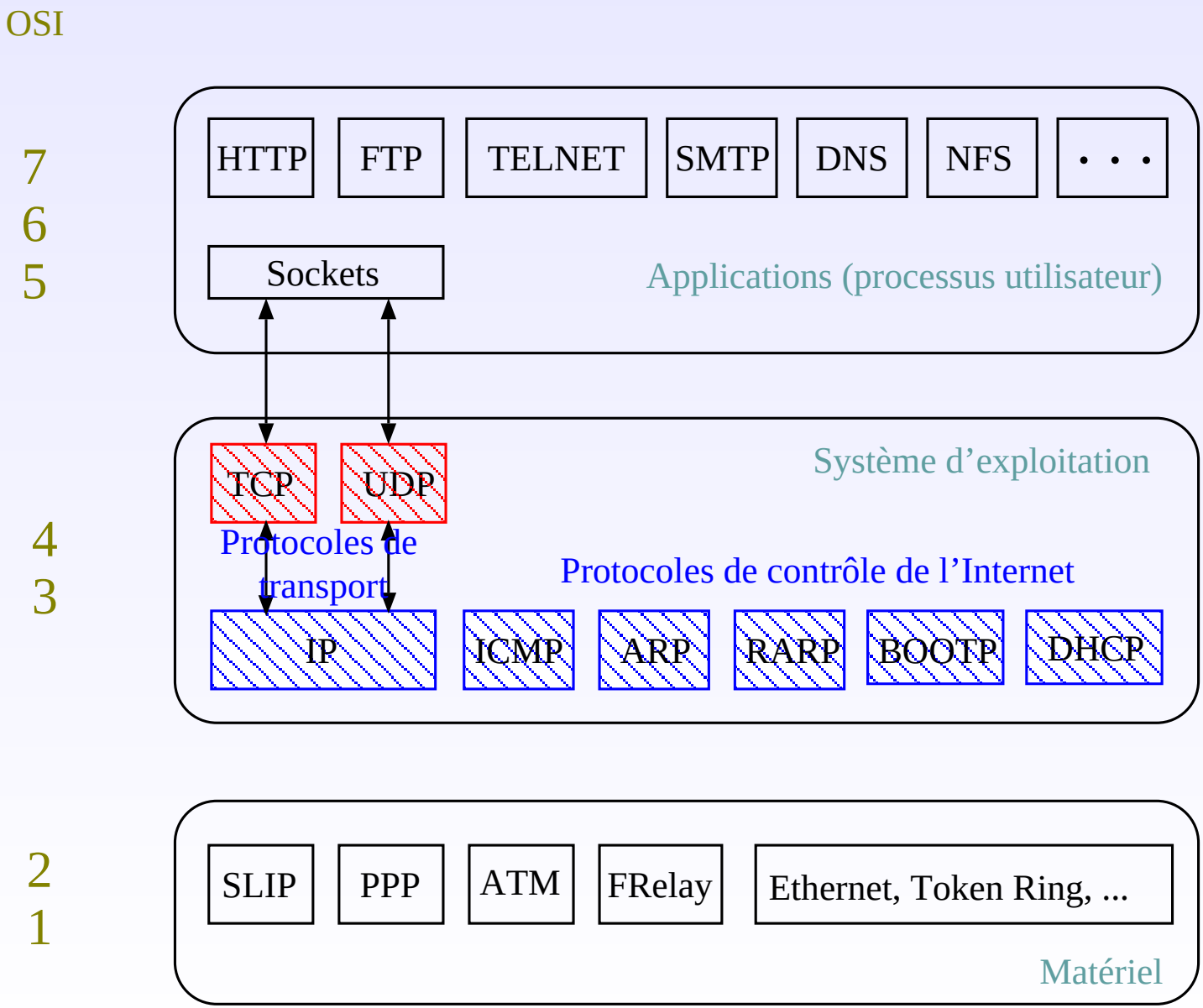
- ⋆ TCP : transfert fiable de données en mode connecté
- ⋆ UDP : transfert non garanti de données en mode non connecté

Réseau IP (routage)

Physique transmission entre 2 sites

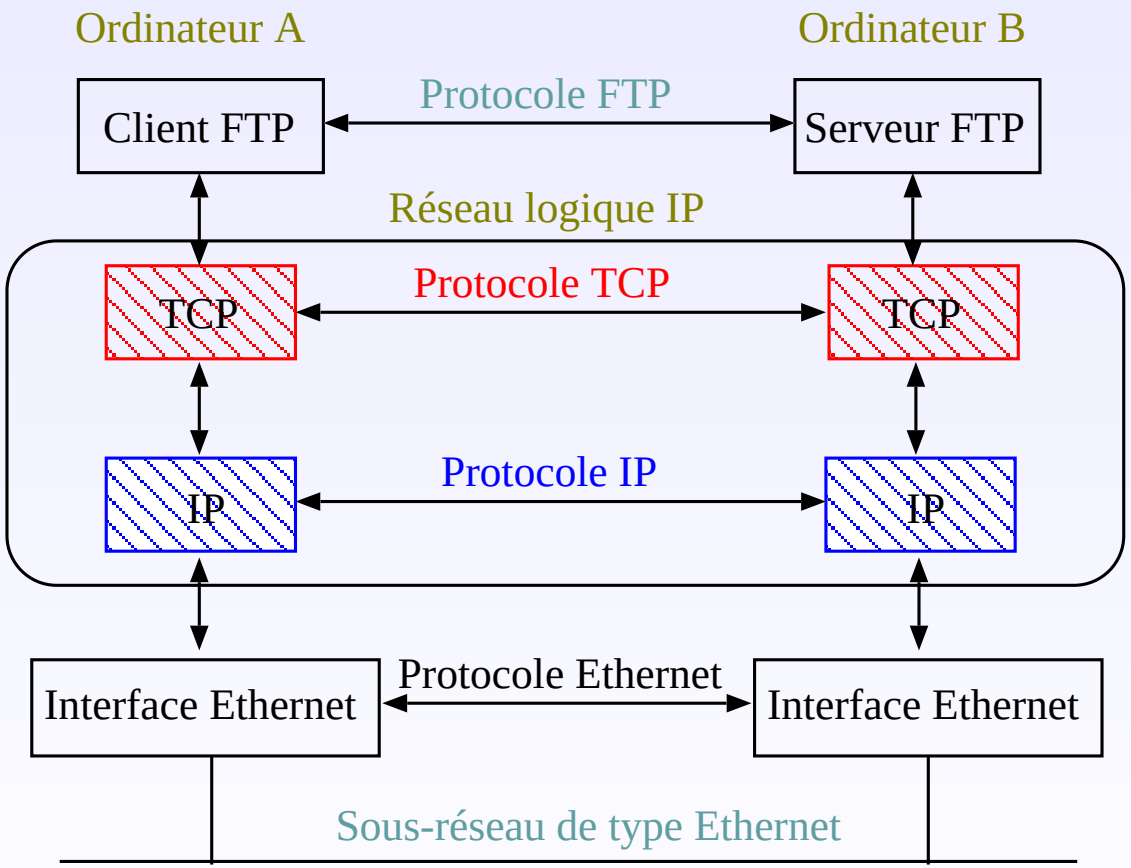
TCP → Transport Control Protocol
UDP → User Datagram Protocol
IP → Internet Protocol

L'architecture de TCP/ IP (2)



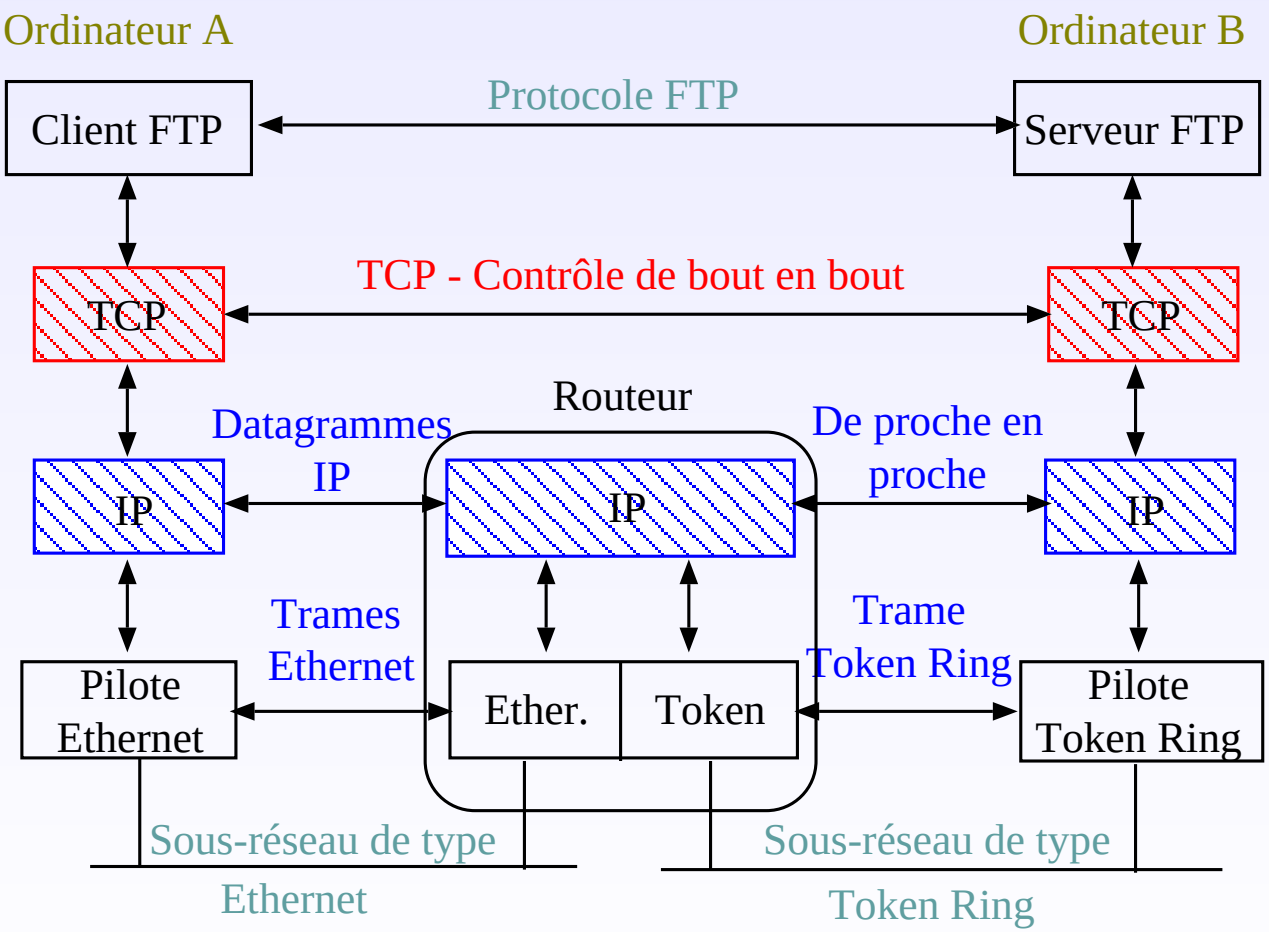
L'architecture de TCP/ IP (3)

Deux machines sur un même sous-réseau IP



L'architecture de TCP/ IP (4)

Prise en compte de l'hétérogénéité



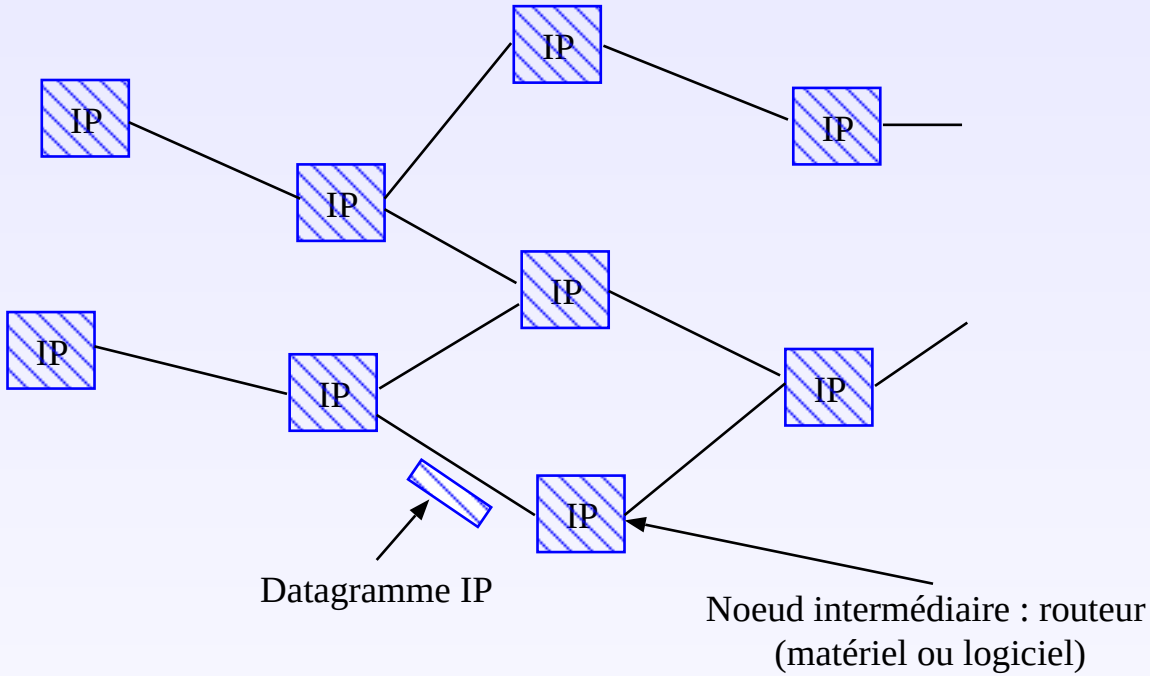
Application

Système d'exploitation

Carte réseau

L'architecture de TCP/ IP (5)

Couche réseau : communications entre machines

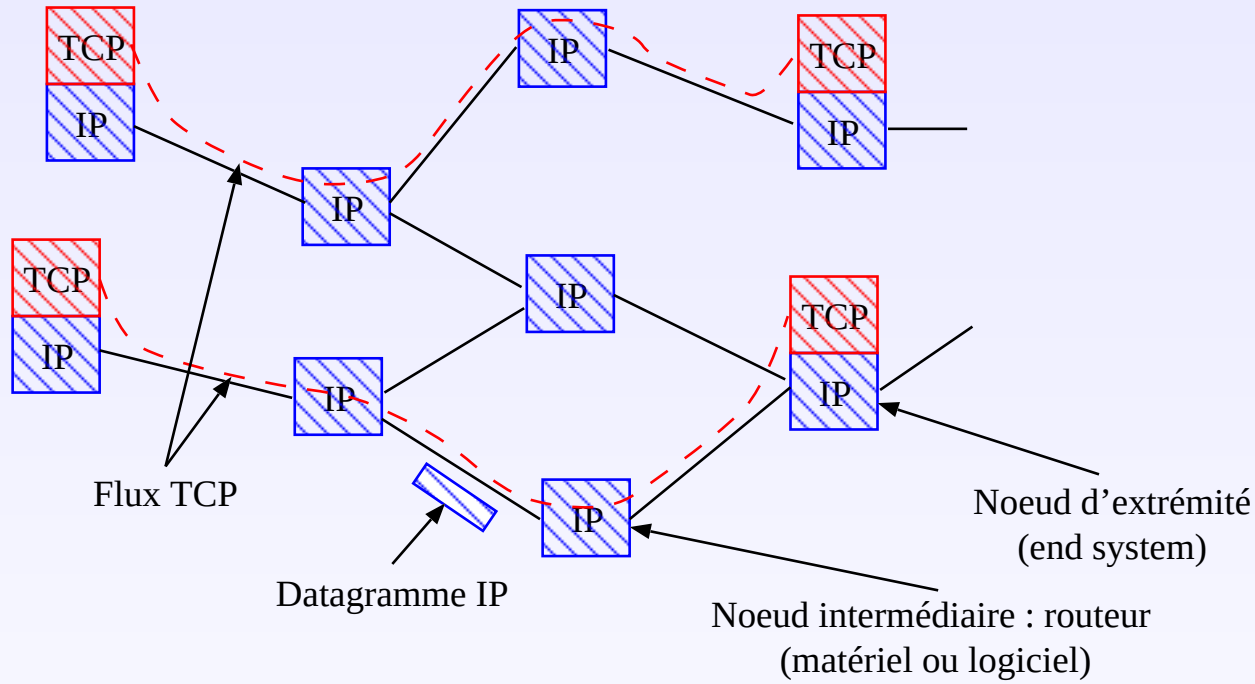


IP - protocole d'interconnexion, best-effort

- ★ acheminement de datagrammes (mode non connecté)
- ★ peu de fonctionnalités,
- ★ pas de garanties simple mais robuste (défaillance d'un noeud intermédiaire)

L'architecture de TCP/ IP (6)

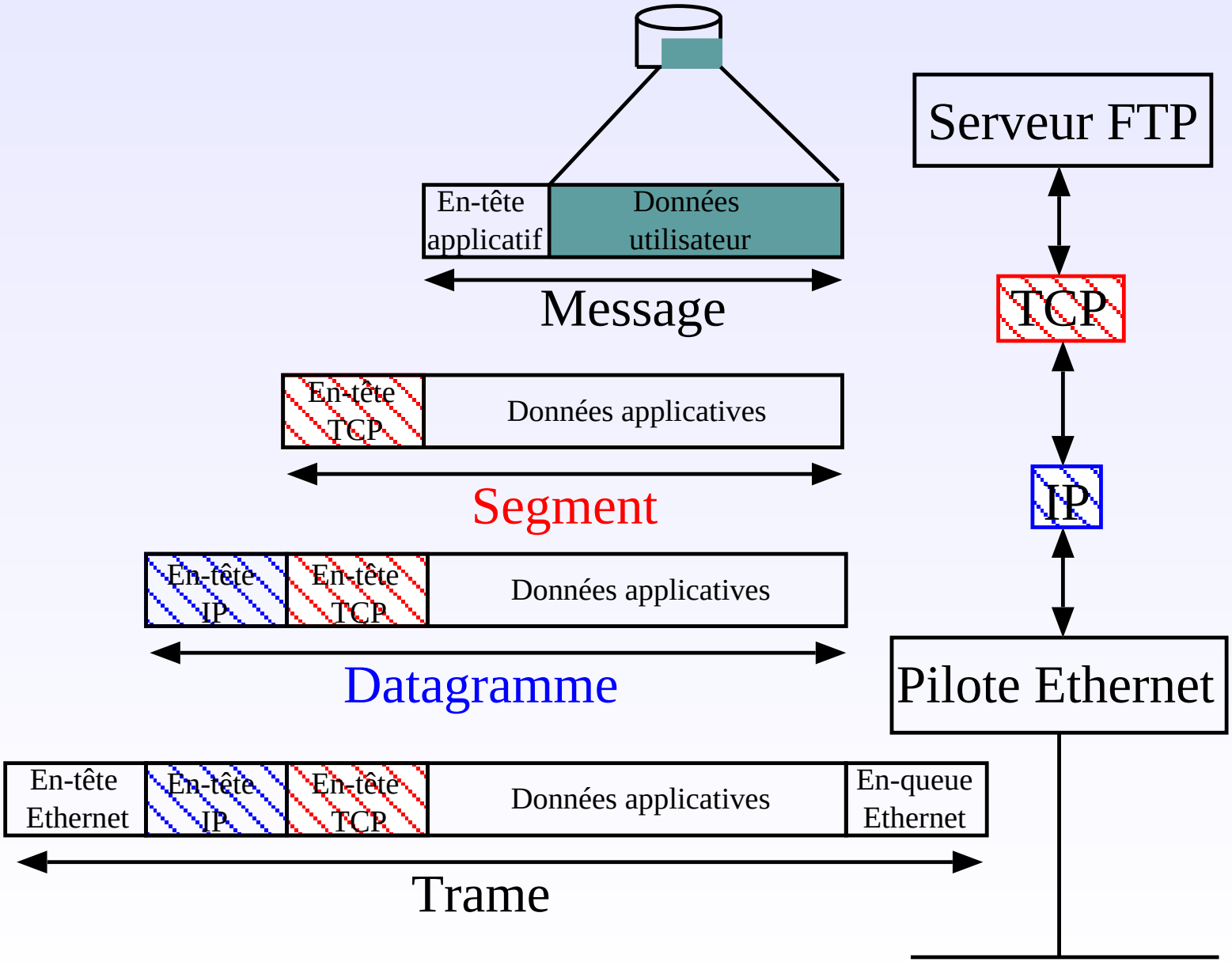
Couche transport : communications entre applications



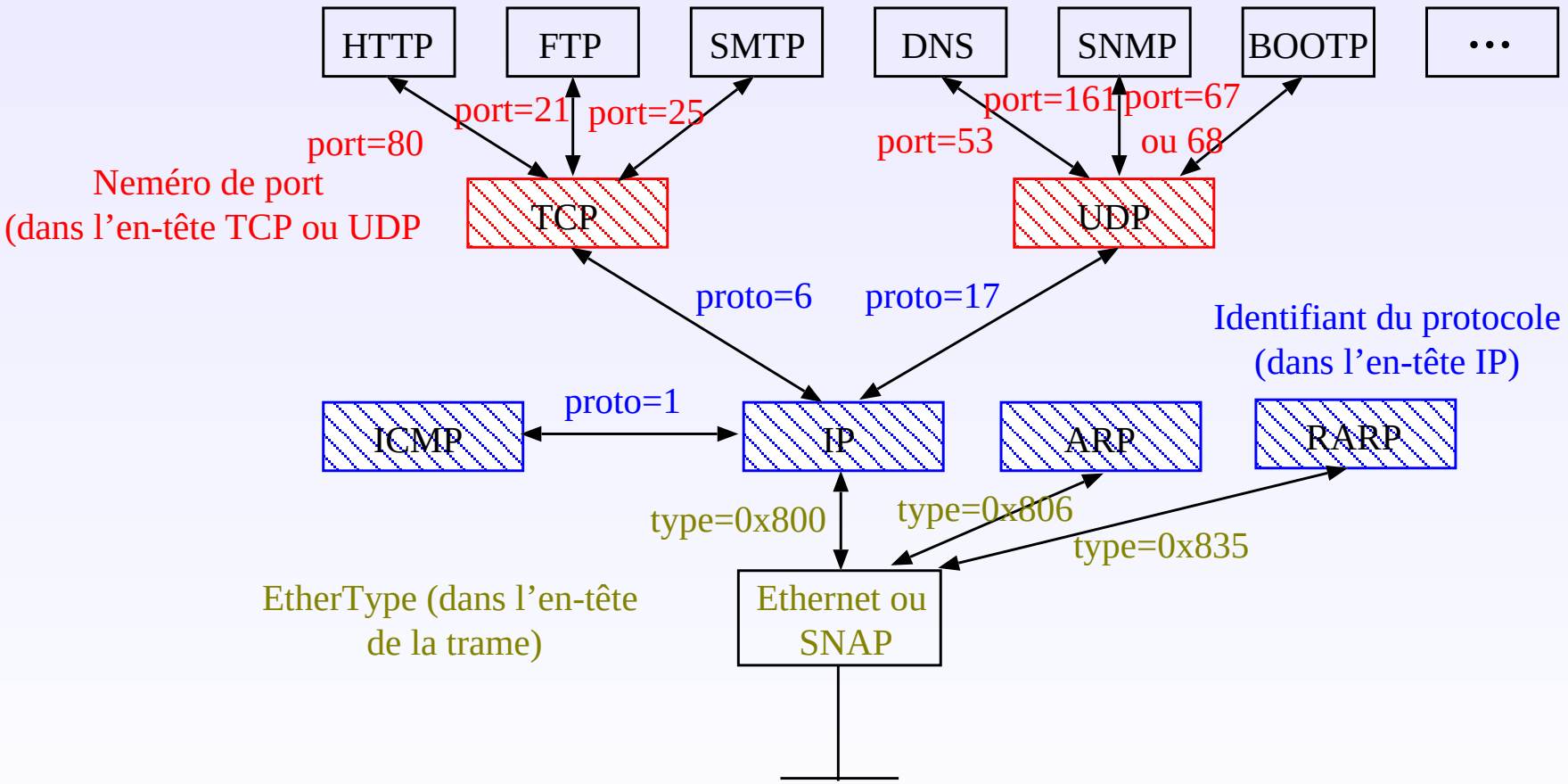
TCP - protocole de transport de bout en bout

- ★ uniquement présent aux extrémités
- ★ transport fiable de segments (mode connecté)
- ★ protocole complexe (retransmission, gestion des erreurs, séquencement, . . .)

L'architecture de TCP/ IP (7)



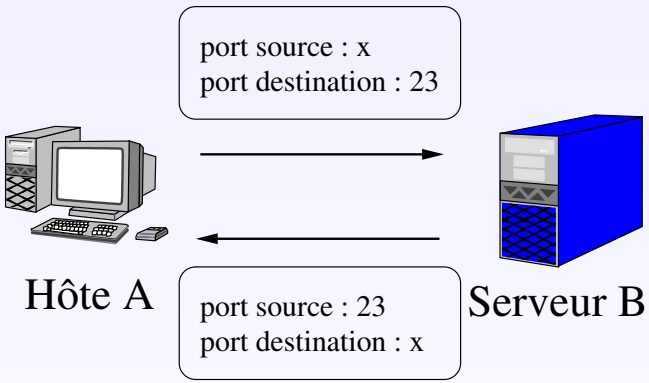
Identification des protocoles (1)



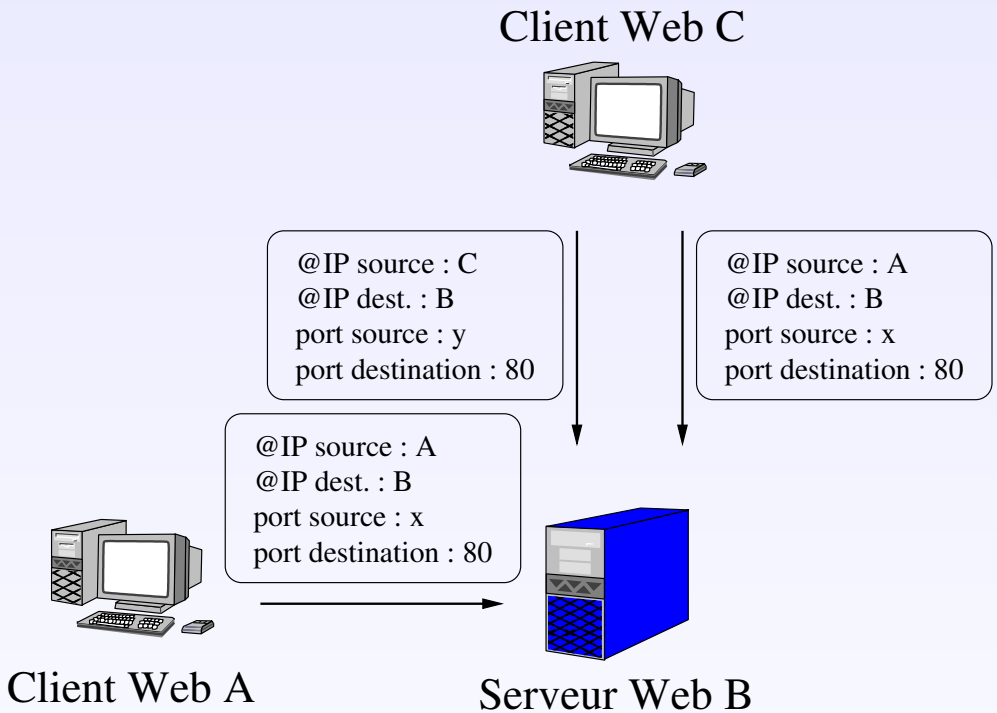
Identification des protocoles (2)

- Une adresse de transport = une adresse IP + un numéro de port (16 bits) $\rightarrow$ adresse de socket
- Une connexion s'établit entre une socket source et une socket destinataire $\rightarrow$ une connexion = un quintuplé (proto, src, port src, dest, port dest)
- Deux connexions peuvent aboutir à la même socket
- Les ports permettent un multiplexage ou démultiplexage de connexions au niveau transport
- Les ports inférieurs à 1024 sont appelés ports réservés

Identification des protocoles (3)



Telnet Simple



Serveur Web

Le protocole UDP

UDP (RFC 768) - User Datagram Protocol

- protocole de transport le plus simple
- service de type best-effort (comme IP)
  - les segments UDP peuvent être perdus
  - les segments UDP peuvent arriver dans le désordre
- mode non connecté : chaque segment UDP est traité indépendamment des autres

Pourquoi un service non fiable sans connexion ?

- simple donc rapide (pas de délai de connexion, pas d'état entre émetteur/récepteur)
- petit en-tête donc économie de bande passante
- sans contrôle de congestion donc UDP peut émettre aussi rapidement qu'il le souhaite

Les utilisations d'UDP

- Performance sans garantie de délivrance
- Souvent utilisé pour les applications multimédias
  - tolérantes aux pertes
  - sensibles au débit
- Autres utilisations d'UDP
  - applications qui envoient peu de données et qui ne nécessitent pas un service fiable
  - exemples : DNS, SNMP, BOOTP/DHCP

Transfert fiable sur UDP

- ajouter des mécanismes de compensation de pertes (reprise sur erreur) au niveau applicatif
- mécanismes adaptés à l'application

Le protocole TCP

Transport Control Protocol (RFC 793, 1122, 1323, 2018, 2581)

Attention: les RFCs ne spécifient pas tout - beaucoup de choses dépendent de l'implémentation

Transport fiable en mode connecté

- point à point, bidirectionnel : entre deux adresses de transport (@IP src, port src) $\rightarrow$ (@IP dest, port dest)
- transporte un flot d'octets (ou flux)
  - l'application lit/écrit des octets dans un tampon
- assure la délivrance des données en séquence
- contrôle la validité des données reçues
- organise les reprises sur erreur ou sur temporisation
- réalise le contrôle de flux et le contrôle de congestion (à l'aide d'une fenêtre d'émission)

Exemples de protocole applicatif

HTTP - HyperText Transport Protocol

- protocole du web
- échange de requête/réponse entre un client et u serveur web

FTP - File Transfer Protocol

- protocole de manipulation de fichiers distants
- transfert, suppression, création, ...

TELNET - TELetypewriter Network Protocol

- système de terminal virtuel
- permet l'ouverture d'une session distante

DNS - Domain Name System

- assure la correspondance entre un nom symbolique et une adresse Internet (adresse IP)
- bases de données réparties sur le globe

Administration réseau
Routage et passerelle

A. Guermouche

Plan

1. Introduction

2. Routage dans IP
   - Principes de base
   - Manipulation des tables de routage

3. Mise en place d'un réseau
   - comment connecter des machines entre elles?
   - Routage avec Linux

4. Réseaux privé

Plan

1. Introduction

2. Routage dans IP
   - Principes de base
   - Manipulation des tables de routage

3. Mise en place d'un réseau
   - comment connecter des machines entre elles?
   - Routage avec Linux

4. Réseaux privé

Sous-réseaux

Définition :

- Un sous-réseau est un sous-ensemble d'un réseau de classe

Intérêts :

- diviser un réseau de grande taille en plusieurs réseaux physiques connectés par des routeurs (locaux ou distants)
- possibilité de faire coexister des technologies de réseaux différentes
- diminution de la congestion du réseau par redirection du trafic et réduction des diffusions

Comment ?

- ID sous-réseau en séparant les bits d'ID d'hôtes en plusieurs sections

Linux : Positionner/Modifier une adresse IP

La manipulation des adresses IP se fait à l'aide de l'utilitaire *ifconfig*.

Syntaxe :

`ifconfig` *`interface`* `@IP netmask` *`masque`* `...`

Exemples :

- `ifconfig eth0` (consulter la configuration de l'interface eth0)
- `ifconfig eth0 192.168.0.1 netmask 255.255.255.0` (configurer l'interface eth0)
- `ifconfig eth0 down` (Suppression de la configuration de l'interface eth0)

Plan

1. Introduction

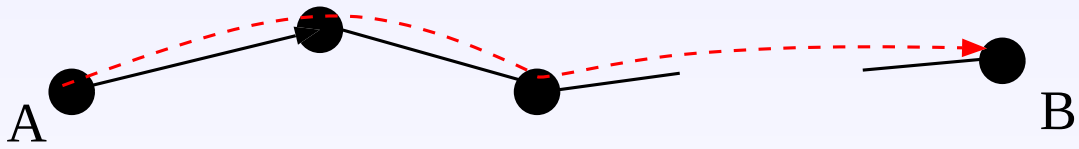
2. Routage dans IP
   - Principes de base
   - Manipulation des tables de routage

3. Mise en place d'un réseau
   - comment connecter des machines entre elles?
   - Routage avec Linux

4. Réseaux privé

Problématique du routage

Objectif : Acheminer des datagrammes IP d'une machine source A vers une machine destination B.
Problématique : Comment atteindre la machine B en connaissant son adresse IP?



→ Nécessité d'identifier toutes les machines intermédiaires.

Routage IP : principe de base

Définition :

- Processus de choix des chemins par lesquels les paquets sont transmis à la machine destinataire
- Processus basé sur une table de routage *IP routing table* contenant les informations relatives aux différentes destination possibles et à la façon de les atteindre
- Exemple : netstat -r (sous UNIX)

Principe de base :

- L'émetteur ne connaît pas la route complète mais l'adresse du prochain site IP qui le rapprochera de la destination (prochain saut)
- Simplicité des tables de routage
- Changements dynamiques possibles (en cas de pannes par exemple)

Routage IP : algorithme de base (1/2)

- Extraire du datagramme l'adresse IP de destination (*IPDest*)
- Calculer l'adresse du réseau de destination (*IPRes*)
- Si cette adresse *IPRes* correspond à l'adresse réseau du réseau local :
  - *IPdest* est directement accessible sur le réseau élémentaire commun
  - La couche IP locale tente la translation adresse logique *IPdest* en une adresse physique à travers la table maintenue en cache
  - Si le réseau est de type Ethernet (Tokenring, ...), le protocole ARP est utilisé pour construire les éventuelles entrées manquantes dans la table et émettra le datagramme
  - Si le réseau est d'un autre type (Transpac, ...), les adresses physiques destinataires X21 auront dû être configurées à la main au préalable

Routage IP : algorithme de base (2/2)

- Sinon (ce n'est pas une adresse accessible, il faut alors consulter la table de routage IP locale)
  - Si *IPres* est dans la table alors :
    - Router le datagramme selon les indications de la table (vers un autre nœud du réseau local, avec résolution adresse IP $\rightarrow$ adresse physique, ou vers un autre coupleur connecté à un réseau externe)
  - Sinon *IPres* n'est pas dans la table alors
    - Prendre la route par défaut indiquée dans la table
    - Router le datagramme selon les indications de l'entrée par défaut de la table (vers un autre nœud du réseau local, avec résolution adresse IP $\rightarrow$ adresse physique, ou vers un autre coupleur connecté à un réseau externe)

Tables de routage IP dans Linux

La consultation/modification de la table de routage peut être faite avec la commande route.

Exemple :

```
Destination     Passerelle      Genmask         Indic  Metric  Ref  Use  Iface
147.210.20.0    *               255.255.255.0   U      0       0    0    eth0
default         vlan2.labri.fr  0.0.0.0         UG     0       0    0    eth0
```

Cette table de routage montre que :

- Notre hôte peut dialoguer directement avec les machines faisant partie du réseau 147.210.20.0/24
- La route par défaut le fait passer par la passerelle `vlan2.labri.fr`

Tables de routage IP dans Linux

La consultation/modification de la table de routage peut être faite avec la commande route.

Exemple :

`route add default gw @passerelle` (ajouter une route par défaut)

`route add -host @hôte gw @passerelle dev` *`iface`* (ajouter une route utilisant l'interface réseau *iface* vers un hôte particulier)

`route add -net @réseau netmask` *`masque`* `dev` *`iface`* `gw @passerelle` (ajouter une route utilisant l'interface *iface* vers un réseau particulier)

Pour les suppressions de route, il suffit de remplacer l'opération `add` par `del`.

Analyse de la route

traceroute. montre le chemin vers des machines distantes en indiquant chaque 'hop' (saut) que fait un paquet sur la route vers la destination.

Exemple :

```
traceroute to vivaldi.emi.u-bordeaux.fr
(147.210.13.225), 30 hops max, 40 byte
packets
          1 vlan2.labri.fr (147.210.20.254)
          5.579 ms 1.697 ms 6.420 ms
          2 b3a1.labri.fr (147.210.9.254)
          1.840 ms 4.924 ms 2.615 ms
          3 labri-reaumur.u-bordeaux.fr
          (147.210.246.190) 4.527 ms 5.561
          ms 2.050 ms
          4 vivaldi.emi.u-bordeaux1.fr
          (147.210.13.225) 8.491 ms 2.448
          ms
```

pathchar, pchar, bing, . . . Outils de mesure de bande passante.

Plan

1. Introduction

2. Routage dans IP
   - Principes de base
   - Manipulation des tables de routage

3. Mise en place d'un réseau
   - comment connecter des machines entre elles?
   - Routage avec Linux

4. Réseaux privé

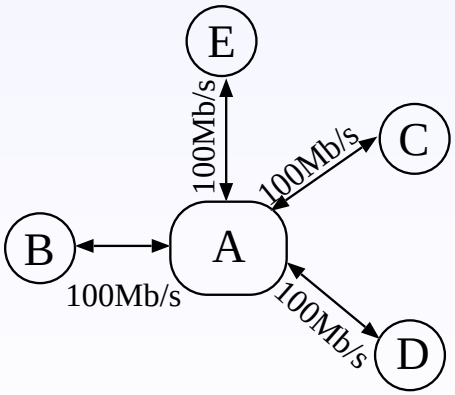
Comment connecter des machines entre elles?

Concentrateur (*hub*)
- ⋆ Partage de bande passante entre les hôtes raccordés.

Commutateur (*switch*)
- ⋆ Pas d'interférences entre connexions simultanées.

Routeur
- ⋆ Pas d'interférences entre des connexions simultanées.
- ⋆ Possibilité de communication entre 2 réseaux logiques différents.

Exemple :

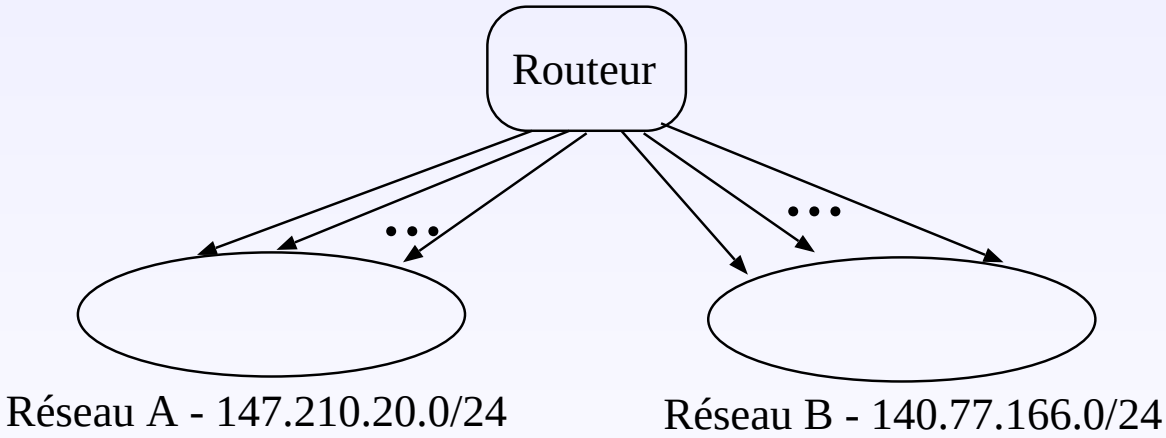


	Type de A		
	Hub	Switch	Routeur
Débit (C→D) lorsque (B→E)	50 Mb/s	100 Mb/s	100 Mb/s
Comm. entre B et C si B et C sont dans deux réseaux différents	impossible	impossible	possible

Différences entre routeurs et passerelle

Passerelle. Élément faisant relais entre deux réseaux physiques utilisant des technologies différentes.

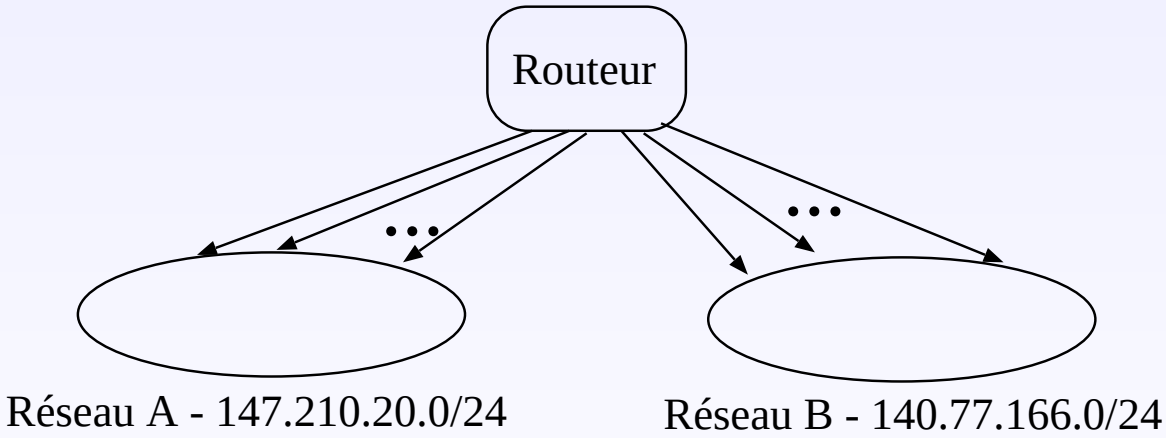
Routeur. Élément reliant deux réseaux différents (et les faisant communiquer).



Différences entre routeurs et passerelle

Passerelle. Élément faisant relais entre deux réseaux physiques utilisant des technologies différentes.

Routeur. Élément reliant deux réseaux différents (et les faisant communiquer).

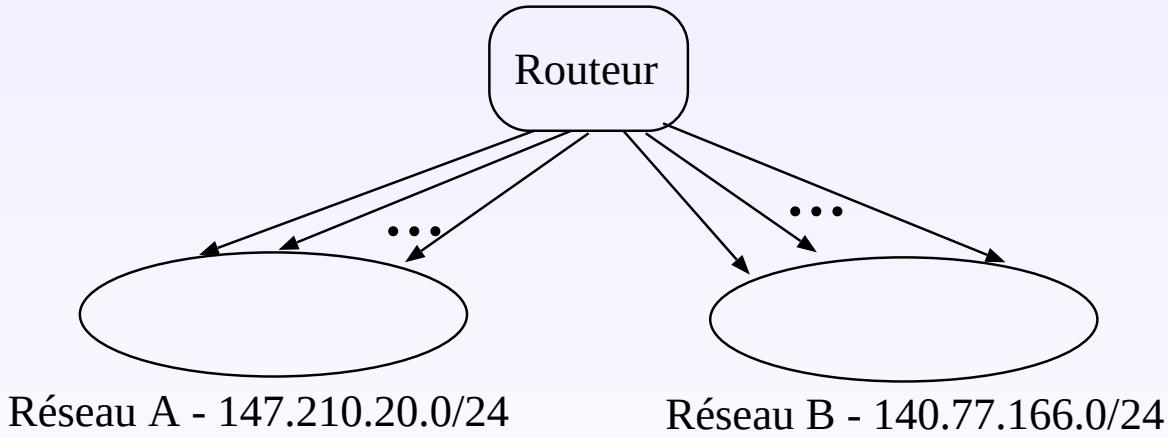


→ La différence est difficile à mettre en évidence de nos jours.

Différences entre routeurs et passerelle

Passerelle. Élément faisant relais entre deux réseaux physiques utilisant des technologies différentes.

Routeur. Élément reliant deux réseaux différents (et les faisant communiquer).



→ La différence est difficile à mettre en évidence de nos jours.
→ Matériels ayant "généralement" plusieurs cartes réseau.

Linux et routage

- Support pour le routage disponible dans le noyau Linux.
- Nécessité d'activer la fonctionnalité de "relais" dans le noyau. Deux méthodes peuvent être utilisée :
  - Modifier "à chaud" le paramètre qui contrôle la fonctionnalité : `echo 1 > /proc/sys/net/ipv4/ip_forward`
  - Configuration automatique à chaque démarrage : Ajout de `net.ipv4.ip_forward=1` au fichier `/etc/sysctl.conf`

→ La machine en question fera alors office de relais entre ses différentes interfaces réseau.

Exercice

Étant données 4 machines, A,B,C et D.
Nous voulons connecter ces machines de telles sortes que A, B et C soient sur 3 sous-réseaux différents. D, quant à elle jouera le rôle de routeur.

Remarque :

Nous disposons d'un *switch* à au moins 4 ports.

Nous considérerons les cas où D a 1 et 3 cartes réseau.

1. Déterminer les tables de routages pour les machines A, B, C et D. Montrer la séquence des commandes à lancer pour positionner les tables de routage à la bonne valeur.
2. Donner l'ensemble des étapes à effectuer au niveau de la machine D pour la configurer en routeur.

Plan

1. Introduction

2. Routage dans IP
   - Principes de base
   - Manipulation des tables de routage

3. Mise en place d'un réseau
   - comment connecter des machines entre elles?
   - Routage avec Linux

4. Réseaux privé

Pourquoi avoir des adresses privées?

- ⋆ Gérer la pénurie d'adresses au sein d'un réseau
- ⋆ Masquer l'intérieur du réseau par rapport à l'extérieur (le réseau peut être vu comme une seule et même machine)
- ⋆ Améliorer la sécurité pour le réseau interne
- ⋆ Assouplir la gestion des adresses du réseau interne
- ⋆ Faciliter la modification de l'architecture du réseau interne

$\longrightarrow$ Mécanisme de translation d'adresses (NAT - Network Address Translation)
Deux types de NAT :

statique. association entre *n* adresses publiques et *n* adresses privées.

dynamique. association entre 1 adresse publique et *n* adresses privées.

Administration réseau
Réseaux privés

A. Guermouche

Plan

1. Introduction

2. NAT statique

3. NAT dynamique : Masquerading

4. Proxy

Plan

1. Introduction

2. NAT statique

3. NAT dynamique : Masquerading

4. Proxy

Pourquoi avoir des adresses privées?

- ⋆ Gérer la pénurie d'adresses au sein d'un réseau
- ⋆ Masquer l'intérieur du réseau par rapport à l'extérieur (le réseau peut être vu comme une seule et même machine)
- ⋆ Améliorer la sécurité pour le réseau interne
- ⋆ Assouplir la gestion des adresses du réseau interne
- ⋆ Faciliter la modification de l'architecture du réseau interne

$\longrightarrow$ Mécanisme de translation d'adresses (NAT - Network Address Translation)
Deux types de NAT :

statique. association entre *n* adresses publiques et *n* adresses privées.

dynamique. association entre 1 adresse publique et *n* adresses privées.

Plan

1. Introduction

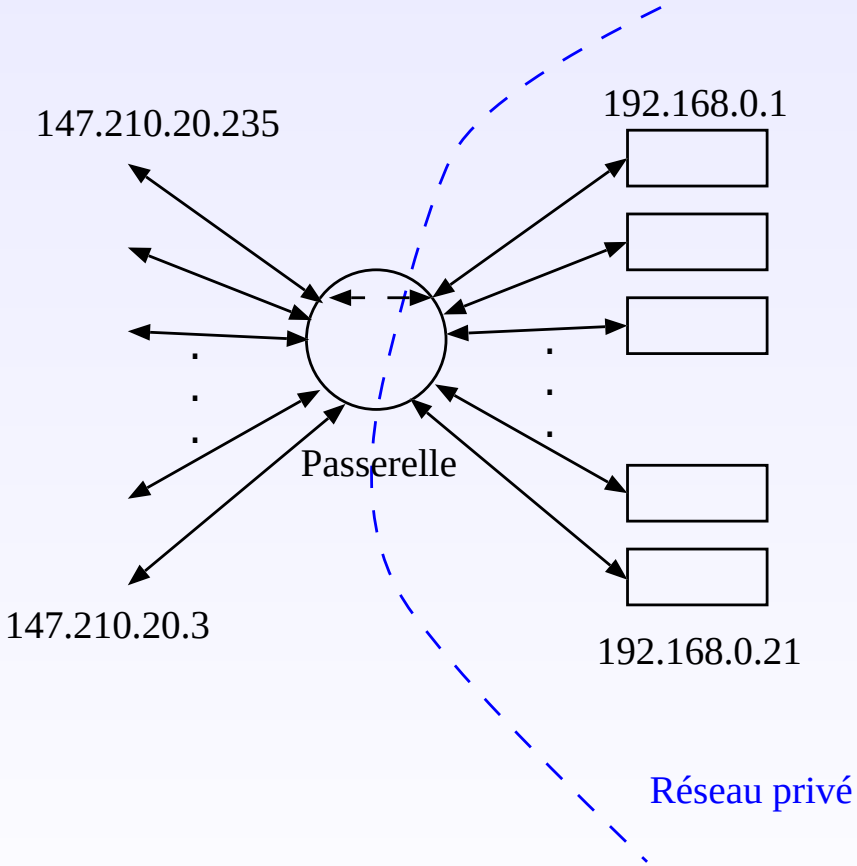
2. NAT statique

3. NAT dynamique : Masquerading

4. Proxy

NAT statique

Association entre une adresse publique et une adresse privée.



NAT statique

Association entre une adresse publique et une adresse privée.

Intérêt :

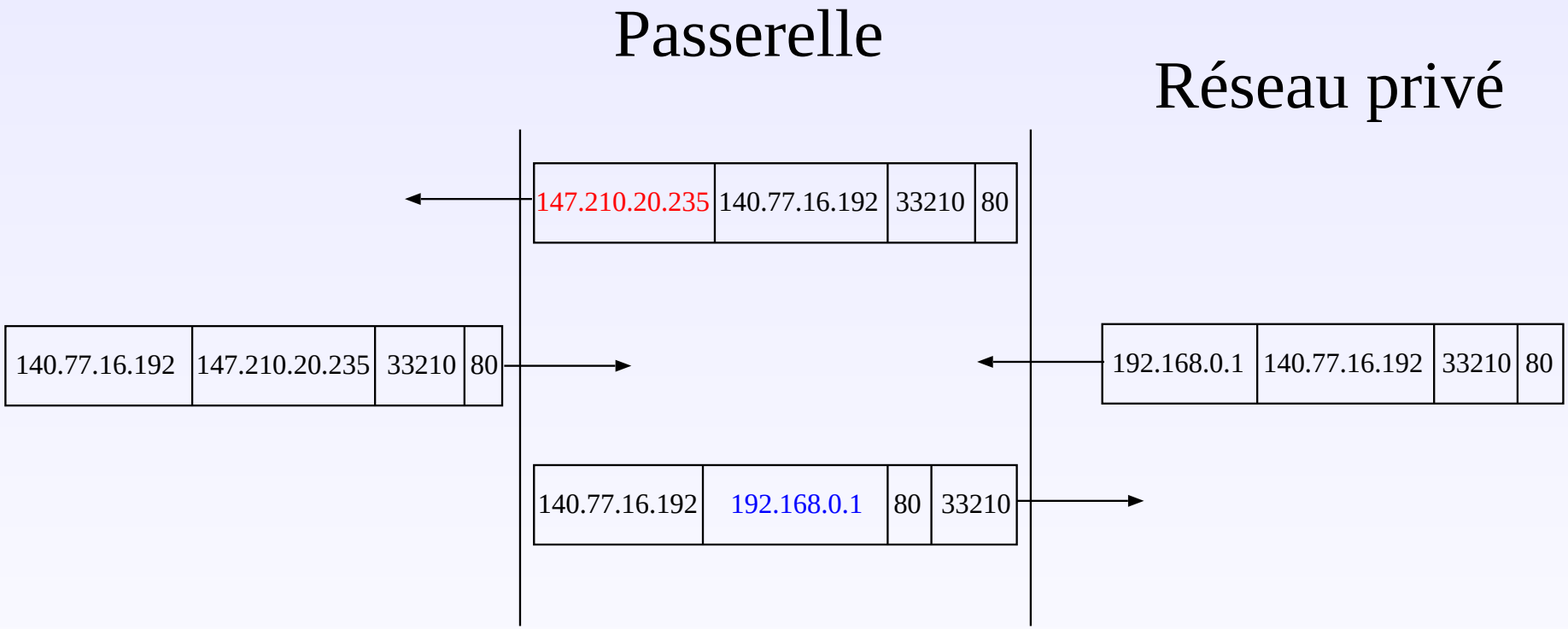
- Uniformité de l'adressage dans la partie privée du réseau (modification de la correspondance @publique/@privée facile)
- Sécurité accrue (tous les flux passent par la passerelle NAT)

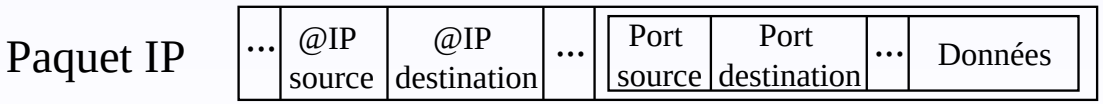
Inconvénient :

- Problème de pénurie d'adresses IP publiques non-résolu

NAT statique : Principe

Pour chaque paquet sortant (resp. entrant), la passerelle modifie l'adresse source (resp. destination).





Plan

1. Introduction

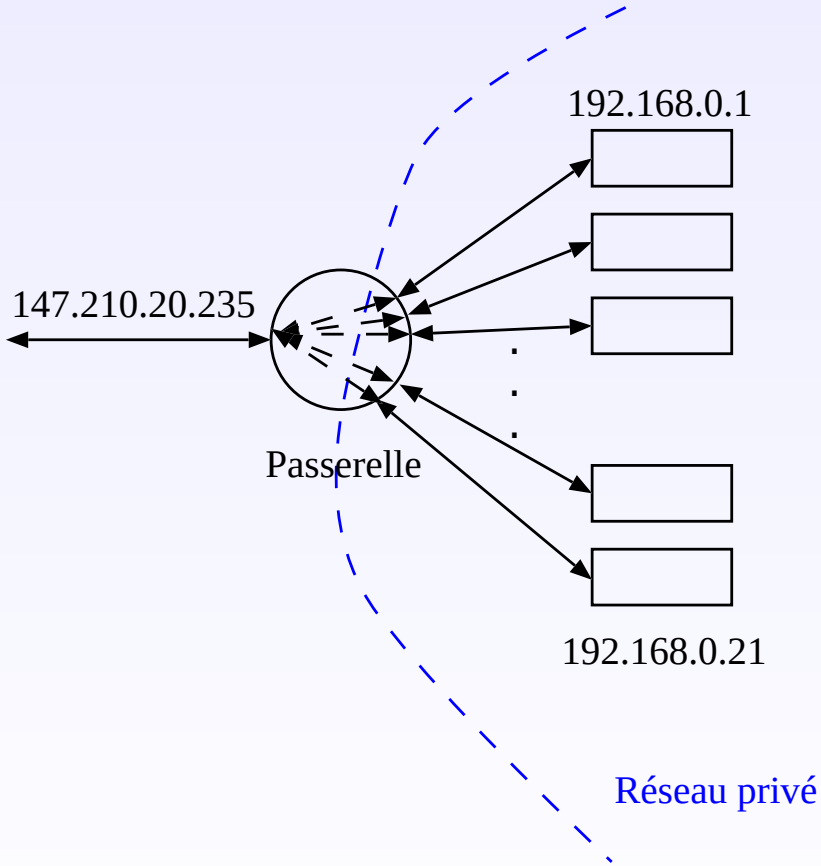
2. NAT statique

3. NAT dynamique : Masquerading

4. Proxy

NAT dynamique : Masquerading

Association entre m adresses publiques et n adresses privées ($m < n$).



NAT dynamique : Masquerading

Association entre m adresses publiques et n adresses privées ($m < n$).

Intérêt :

- Plusieurs machines utilisent la même adresse IP publique pour sortir du réseau privé
- Sécurité accrue (tous les flux passent par la passerelle NAT)

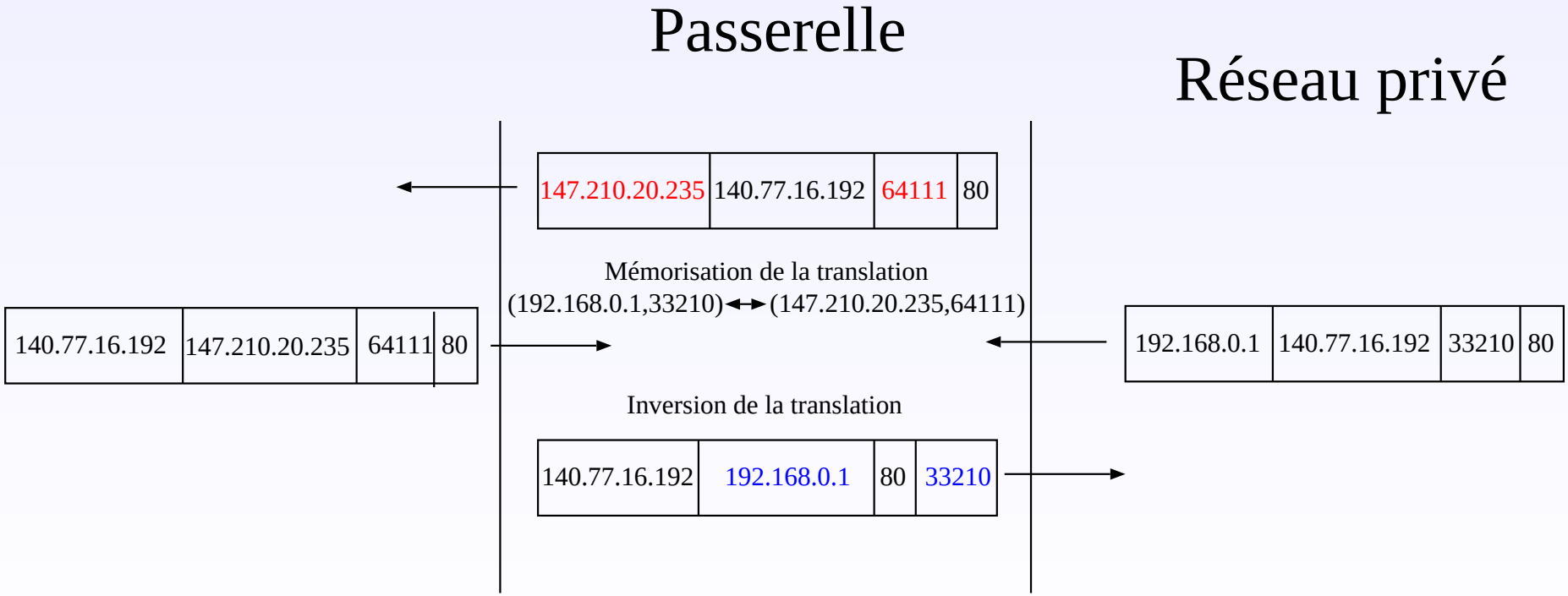
Inconvénient :

- Les machines du réseau interne ne sont pas accessibles de l'extérieur (impossibilité d'initier une connexion de l'extérieur)

NAT dynamique : Principe (1/2)

L'association de *n* adresses privées à 1 adresse publique nécessite, au niveau de la passerelle, de :

- ★ modifier l'adresse source (resp. destination) des paquets sortant (resp. entrants)
- ★ changer le numéro de port source pour les flux sortant



NAT dynamique : Principe (2/2)

Comment est ce que le routeur différencie les paquets qui lui sont destinés de ceux qu'il doit relayer?

À chaque nouvelle connexion :

1: Modifier l'adresse source et le port source : (@source_privée,port_source)→(@publique,port_source')
2: Sauvegarder l'association dans la table `NAT`

Pour chaque paquet entrant :

3: Chercher une association correspondant au couple (@destination, port_destination)
4: **Si** ∃ une association dans la table `NAT` **Alors**
5: Modifier l'adresse de destination et le port de destination
6: Relayer le paquet
7: **Sinon**
8: /* Erreur de routage */
9: **Fin du Si**

NAT dynamique : Principe (2/2)

Comment est ce que le routeur différencie les paquets qui lui sont destinés de ceux qu'il doit relayer?

À chaque nouvelle connexion :

1: Modifier l'adresse source et le port source : (@source_privée,port_source)→(@publique,port_source')
2: Sauvegarder l'association dans la table `NAT`

Pour chaque paquet entrant :

3: Chercher une association correspondant au couple (@destination, port_destination)
4: **Si** ∃ une association dans la table `NAT` **Alors**
5: Modifier l'adresse de destination et le port de destination
6: Relayer le paquet
7: **Sinon**
8: /* Erreur de routage */
9: **Fin du Si**

Le routeur gère toutes les associations

⇒ Unicité de l'association (donc du port source après translation)

Problèmes liés à NAT dynamique

Comment faire de la translation d'adresse sur des protocoles qui ne sont pas basés sur TCP ou UDP (pas de numéro de port)?

- Nécessité d'implémenter une méthode spécifique au protocole (identifiant ICMP pour ICMP par exemple).
- Dans le cas des protocoles dont les paquets contiennent des données relatives aux adresses IP, il est nécessaire de mettre en place des "proxy" (`FTP` en mode actif par exemple).

Comment rendre joignables des machines du réseau local?

- Nécessité de faire de la redirection de port (port forwarding/mapping).

  Principe. Toutes les connexions entrantes sur un port donné sont redirigée vers une machine du réseau privé sur un port (qui peut être le même ou non).

Plan

1. Introduction

2. NAT statique

3. NAT dynamique : Masquerading

4. Proxy

Proxy ou mandataire

Définition :

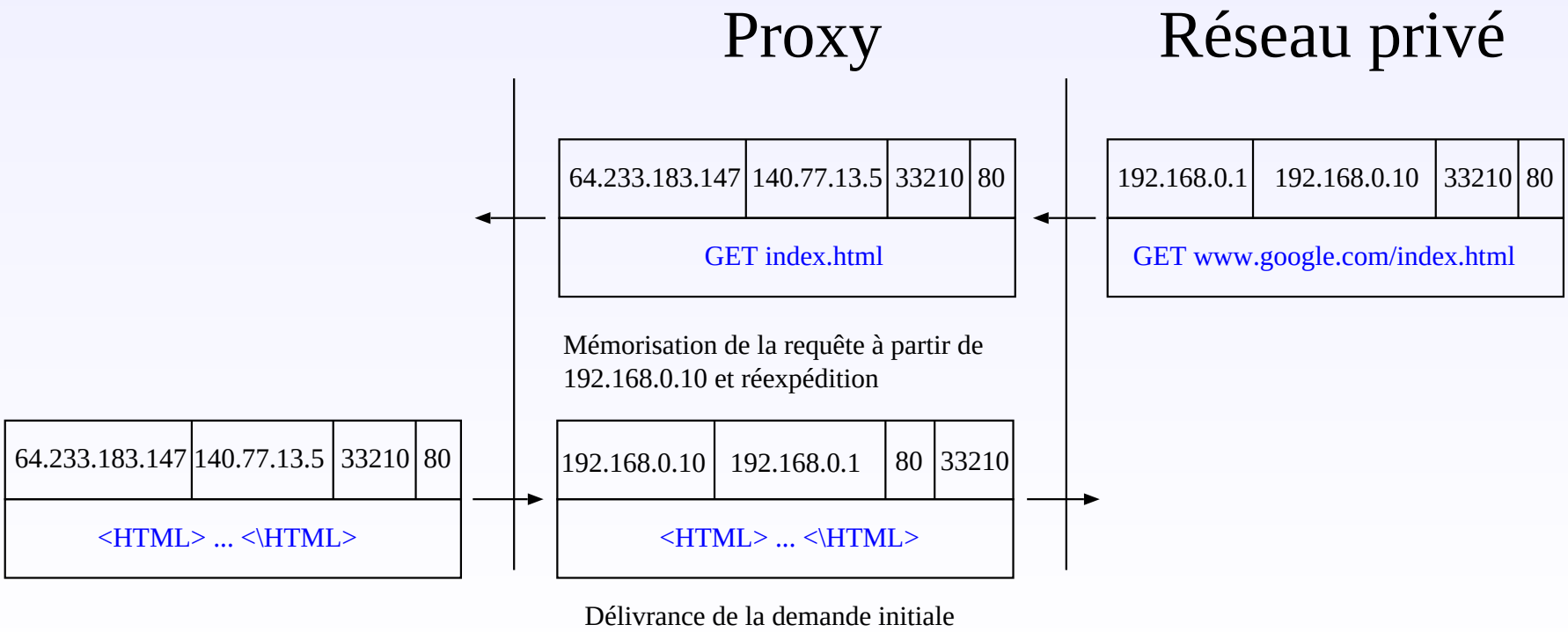
- ★ Un proxy est un intermédiaire dans une connexion entre le client et le serveur
- ★ Le client s'adresse toujours au proxy
- ★ Le proxy est spécifique à une application donnée (`HTTP`, `FTP`, ...)

→ Possibilité de modification des informations échangées entre le client et le serveur.

Proxy ou mandataire

Définition :

- Un proxy est un intermédiaire dans une connexion entre le client et le serveur
- Le client s'adresse toujours au proxy
- Le proxy est spécifique à une application donnée (`HTTP`, `FTP`, ...)



Administration réseau
Iptables et NAT

A. Guermouche

Plan

1. Logiciels de filtrage de paquets
2. Ipfwadm
3. Ipchains
4. Iptables

Plan

1. Logiciels de filtrage de paquets

2. Ipfwadm

3. Ipchains

4. Iptables

Logiciels de filtrage de paquets

- Fonctionnalités de “firewall” filtrant directement implémentée dans le noyau Linux.
- Filtrage de niveau 3 ou 4.
- 3 types de firewall filtrants :

  Ipfwadm. Jusqu’à la version 2.1.102 du noyau linux

  Ipchains. Entre les versions 2.2.0 et 2.4 du noyau linux

  Iptables. À partir des noyaux 2.4

Plan

1. Logiciels de filtrage de paquets

2. Ipfwadm

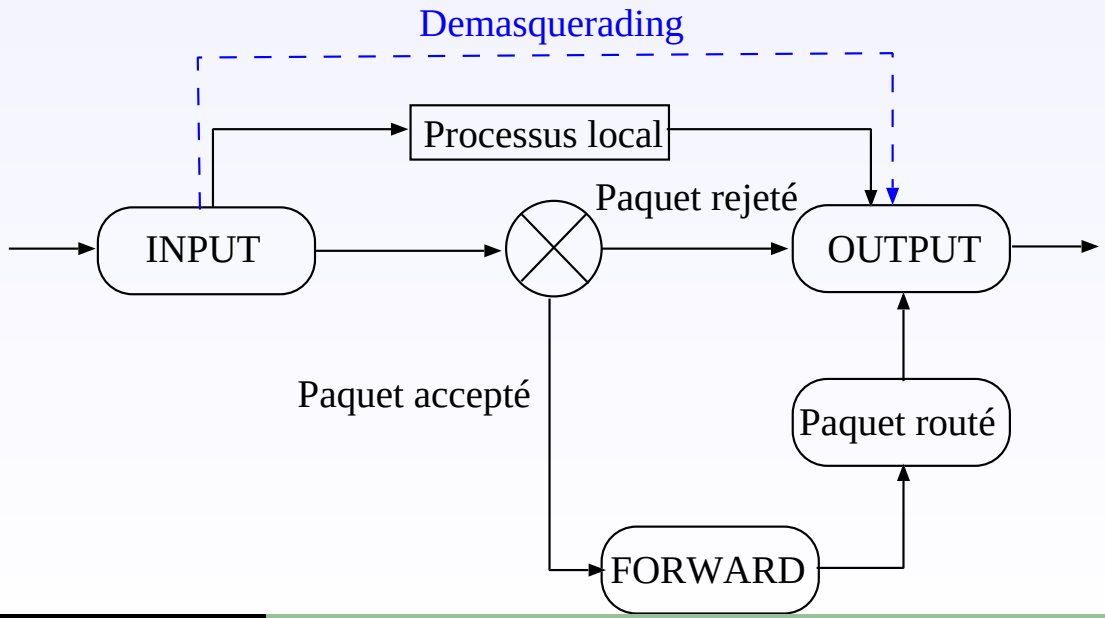
3. Ipchains

4. Iptables

Ipfwadm

- Firewall permettant la gestion des paquets TCP, UDP et ICMP.
- 3 types de règles :
  - INPUT. sont appliquées lors de l'arrivée d'un paquet.
  - FORWARD. sont appliquées lorsque la destination du paquet n'est pas le routeur.
  - OUTPUT. sont appliquées dès qu'un paquet doit sortir du routeur.

Fonctionnement :



Ipfwadm

- Firewall permettant la gestion des paquets TCP, UDP et ICMP.
- 3 types de règles :
  - INPUT. sont appliquées lors de l'arrivée d'un paquet.
  - FORWARD. sont appliquées lorsque la destination du paquet n'est pas le routeur.
  - OUTPUT. sont appliquées dès qu'un paquet doit sortir du routeur.

Fonctionnement :

1: lorsqu'un paquet entre, il traverse les règles de type INPUT
2: **Si** il est accepté **Alors**
3: **Si** il est destiné à une autre machine **Alors**
4: il est routé vers les règles FORWARD
5: **Sinon**
6: il est rejeté
7: le paquet est finalement émis

Dans tous les cas, le paquet traverse les règles OUTPUT

Plan

1. Logiciels de filtrage de paquets

2. Ipfwadm

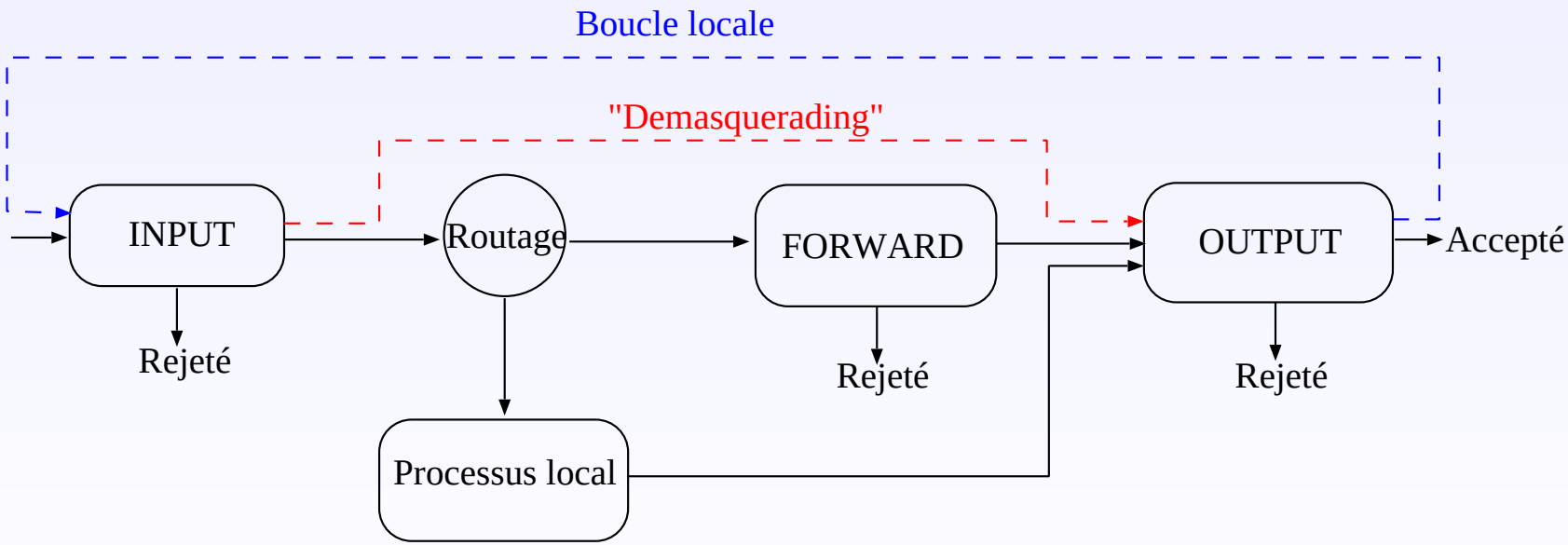
3. Ipchains

4. Iptables

Ipchains

- Module du noyau Linux réalisant le filtrage de paquets.
- Inspiré du parre-feu BSD (tout comme ipfwadm)

Fonctionnement :



Plan

1. Logiciels de filtrage de paquets

2. Ipfwadm

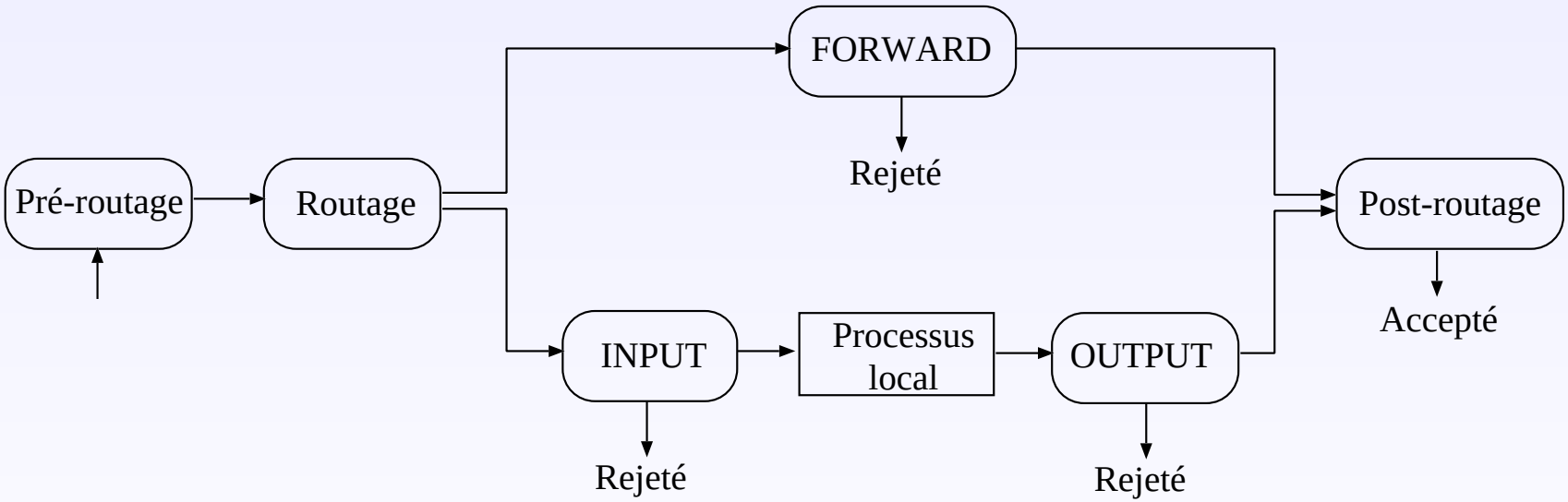
3. Ipchains

4. Iptables

Iptables (1/2)

- Module du noyau Linux réalisant le filtrage de paquets (noyaux $\geq$ 2.4).
- Améliorations en matière de filtrage et de translation d'adresses par rapport à Ipchains.

Fonctionnement :



Iptables (1/2)

- ⋆ Module du noyau Linux réalisant le filtrage de paquets (noyaux $\geq$ 2.4).
- ⋆ Améliorations en matière de filtrage et de translation d'adresses par rapport à Ipchains.

Fonctionnement :

À l'arrivée d'un paquet (après décision de routage) :

Si le paquet est destiné à l'hôte local **Alors**
    il traverse la chaîne INPUT.
    Si il n'est pas rejeté **Alors**
        il est transmis au processus impliqué.
Sinon
    Si le paquet est destiné à un hôte d'un autre réseau **Alors**
        il traverse la chaîne FORWARD
        Si il n'est pas rejeté **Alors**
            il poursuit alors sa route

Iptables (1/2)

- ⋆ Module du noyau Linux réalisant le filtrage de paquets (noyaux $\geq$ 2.4).
- ⋆ Améliorations en matière de filtrage et de translation d'adresses par rapport à Ipchains.

Fonctionnement :

À l'arrivée d'un paquet (après décision de routage) :

Si le paquet est destiné à l'hôte local **Alors**
    il traverse la chaîne INPUT.
    Si il n'est pas rejeté **Alors**
        il est transmis au processus impliqué.
Sinon
    Si le paquet est destiné à un hôte d'un autre réseau **Alors**
        il traverse la chaîne FORWARD
        Si il n'est pas rejeté **Alors**
            il poursuit alors sa route

Tous les paquets émis par des processus locaux au routeur traversent la chaîne OUTPUT.

Iptables (2/2)

Fonctionnalités :

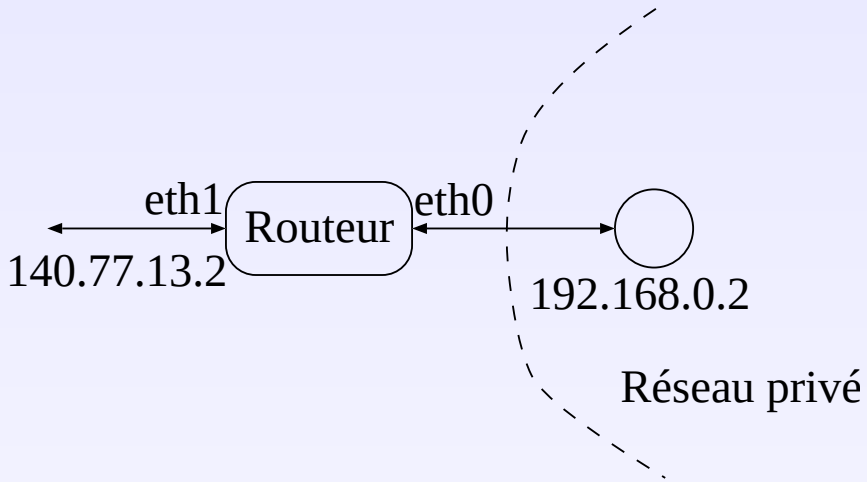
- Filtrage de paquets
- NAT
- Marquage de paquets

Architectures :Trois tables de chaînes (FILTER, NAT et MANGLE).

FILTER (filtrage des paquets)		NAT (translation d'adresses)	
INPUT	paquet entrant sur le routeur	PREROUTING	NAT de destination
OUTPUT	paquet émis par le routeur	POSTROUTING	NAT de source
FORWARD	paquet traversant le routeur	OUTPUT	NAT sur les paquets émis localement

La table MANGLE sert au marquage des paquets

Fonctionnalités NAT d'Iptables (1/2)



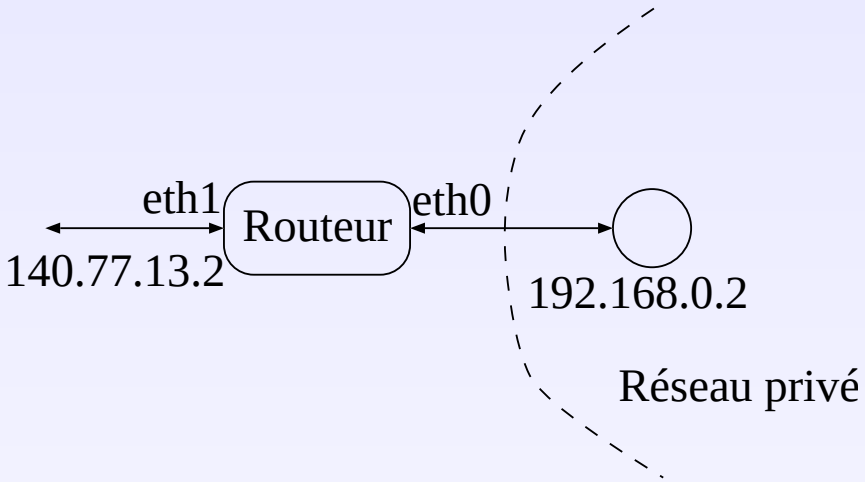
Modification de la destination du paquet avant le routage (paquet reçu de l'extérieur).

```
iptables -t nat -A PREROUTING -d 140.77.13.2 -i eth1 -j DNAT
--to-destination 192.168.0.2
```

Modification de la source du paquet après le routage (paquet émis à partir du réseau privé).

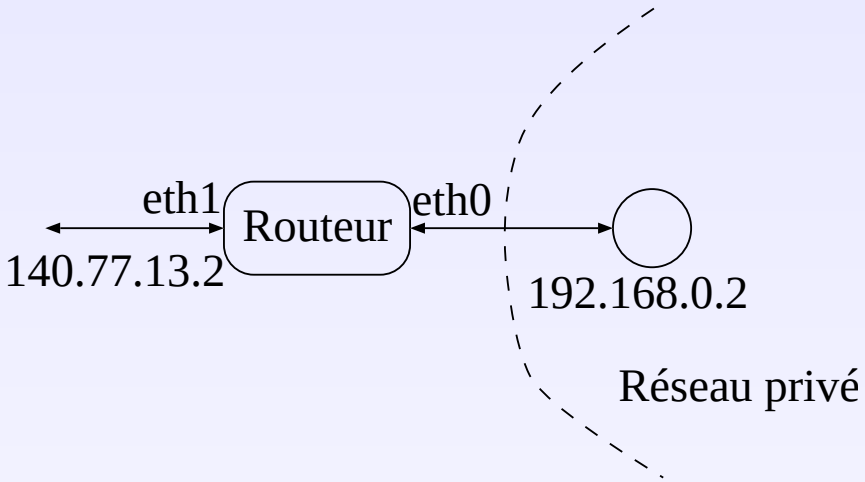
```
iptables -t nat -A POSTROUTING -s 192.168.0.2 -o eth1 -j
SNAT --to-source 140.77.13.2
```

Fonctionnalités NAT d'Iptables (1/2)



Exercice : Comment faire pour que le routeur puisse envoyer un paquet à l'adresse 140.77.13.2?

Fonctionnalités NAT d'Iptables (1/2)



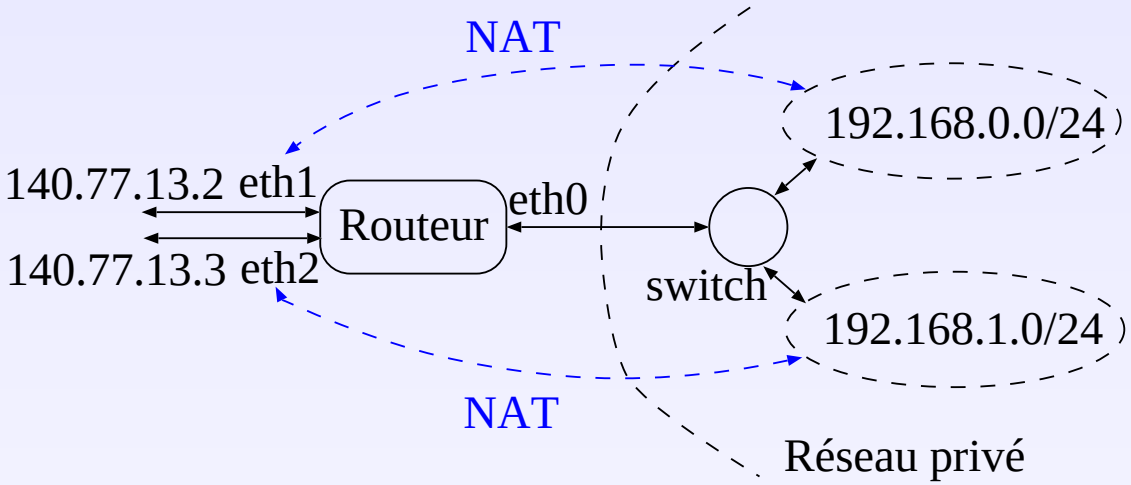
Exercice : Comment faire pour que le routeur puisse envoyer un paquet à l'adresse 140.77.13.2?

Réponse :

Il faut modifier la destination du paquet émis localement avant le routage.

```
iptables -t nat -A OUTPUT -d 140.77.13.2  -j DNAT
--to-destination 192.168.0.2
```

Fonctionnalités NAT d'Iptables (2/2)



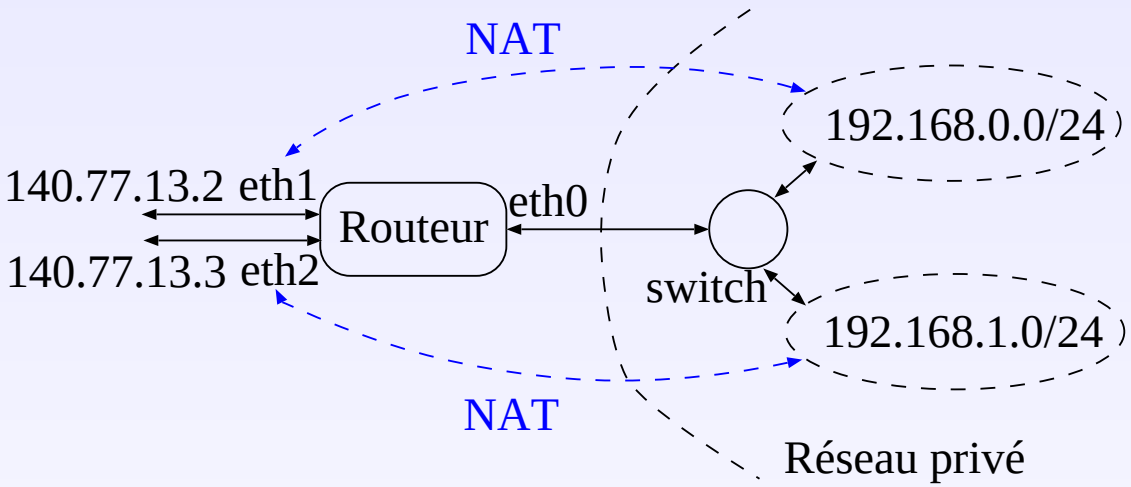
Association entre toutes les adresses privées du sous-réseau 192.168.0.0/24 avec l'interface eth1.

```
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.0.0/24 -j
MASQUERADE
```

Association entre toutes les adresses privées du sous-réseau 192.168.1.0/24 avec l'interface eth2.

```
iptables -t nat -A POSTROUTING -o eth2 -s 192.168.1.0/24 -j
MASQUERADE
```

Transfert de ports



Transférer les connexions sur le port 80 de l'adresse 140.77.13.2 sur la machine ayant l'adresse privée 192.168.0.200 sur le port 8080 :

```
iptables -t nat -A PREROUTING -p tcp -d 140.77.13.2 --dport
80 --sport 1024:65535 -j DNAT --to 192.168.0.200:8080
```

Administration réseau
Firewall

A. Guermouche

Plan

Plan

Pourquoi un firewall?

Definition

Programme, ou un matériel, chargé de vous protéger du monde extérieur en contrôlant tout ce qui passe, et surtout tout ce qui ne doit pas passer entre internet et le réseau local.

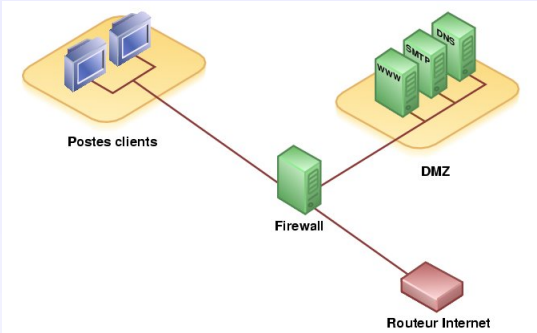
pourquoi un firewall?

Contrôle. Gérer les connexions sortantes a partir du réseau local.

Sécurité. Protéger le réseau interne des intrusions venant de l'extérieur.

Vigilance. Surveiller/tracer le trafic entre le réseau local et internet.

Firewall



Plusieurs types de firewalls :

- ★ Pare-feu au niveau réseau
- ★ Pare-feu au niveau applicatif
- ★ Pare-feu des applications

Différents types de firewalls

Pare-feu niveau réseau. (iptables, paquet filter, . . .)

- ⋆ Firewall fonctionnant à un niveau bas de la pile TCP/IP
- ⋆ Basé sur le filtrage des paquets
- ⋆ Possibilité (si mécanisme disponible) de filtrer les paquets suivant l'état de la connexion

Intérêt : Transparence pour les utilisateurs du réseau

Pare-feu au niveau applicatif. (inetd, xinetd, . . .)

- ⋆ Firewall fonctionnant au niveau le plus haut de la pile TCP/IP
- ⋆ Généralement basé sur des mécanisme de proxy

Intérêt : Possibilité d'interpréter le contenu du trafic

Pare-feu des applications. (/etc/ftpaccess pour ftp, . . .)

- ⋆ Restrictions au niveau des différentes applications

Plan

DMZ

Definition (DMZ)

Une zone démilitarisée (DMZ) est un sous-réseau se trouvant entre le réseau local et le réseau extérieur.

Propriétés :

- Les connexions à la DMZ sont autorisées de n'importe où.
- Les connexions à partir de la DMZ ne sont autorisées que vers l'extérieur.

Intérêt :

- Rendre des machines accessible à partir de l'extérieur (possibilité de mettre en place des serveurs (DNS, SMTP, ...).

Plan

Iptables et filtrage(1/2)

- Filtrage des paquets IP, TCP, UDP ou ICMP
- Spécification de règle pour le rejet ou l'acceptation de paquet
- Utilisation de la table FILTER et des chaînes INPUT, OUTPUT et FORWARD
- Règles traitées de manière séquentielle : Le paquet sort dès qu'il rencontre une règle qui peut lui être appliquée

Exemples :

- Accepter tous les paquets en provenance de n'importe où et destinés à l'adresse du routeur 192.168.1.1.

```
iptables -A INPUT -s 0/0 -i eth0 -d 192.168.1.1 -p TCP
-j ACCEPT
```

- Accepter de router les paquets entrant sur eth0 tels que :

@source	@dest	P-source	P-dest
0/0	192.168.1.58	1024-65535	80

```
iptables -A FORWARD -s 0/0 -i eth0 -d 192.168.1.58 -o
eth1 -p TCP --sport 1024:65535 --dport 80 -j ACCEPT
```

Iptables et filtrage(2/2)

- Accepter un paquet ICMP “echo-request” (ping) par seconde

```
iptables -A INPUT -p icmp --icmp-type echo-request -m limit --limit 1/s -i eth0 -j ACCEPT
```

- Accepter 5 segments TCP ayant le bit SYN positionné par seconde (permet d’éviter de se faire inonder)

```
iptables -A INPUT -p tcp --syn -m limit --limit 5/s -i eth0 -j ACCEPT
```

- Accepter de router les paquets entrants sur eth0 tels que :

@source	@dest	P-source	P-dest
0/0	192.168.1.58	1024-65535	80 ou 443

```
iptables -A FORWARD -s 0/0 -i eth0 -d 192.168.1.58 -o eth1 -p TCP --sport 1024:65535 -m multiport --dport 80,443 -j ACCEPT
```

Iptables et suivi des connexions

- Suivi des connexions disponible (*conntrack*)
- Quatre états possibles pour une connexion :

NEW .	Nouvelle connexion établie
ESTABLISHED .	La connexion analysée est déjà établie
RELATED .	La connexion est en relation avec une connexion déjà établie (ftp-data par exemple)
INVALID .	Le paquet reçu n'appartient à aucune des trois catégories précédentes.

Exemples :

- Autoriser tous les paquets émis par le routeur concernant des connexions déjà établies.

```
iptables -A OUTPUT -o eth0 -m state --state
ESTABLISHED,RELATED -j ACCEPT
```

Iptables et suivi des connexions

- Suivi des connexions disponible (*conntrack*)
- Quatre états possibles pour une connexion :

NEW .	Nouvelle connexion établie
ESTABLISHED .	La connexion analysée est déjà établie
RELATED .	La connexion est en relation avec une connexion déjà établie (ftp-data par exemple)
INVALID .	Le paquet reçu n'appartient à aucune des trois catégories précédentes.

Exemples :

- Autoriser le routeur à relayer tous les paquets reçus concernant de nouvelles connexions sur le port 22.

```
iptables -A FORWARD -p tcp -i eth0 --dport 22 --sport
1024:65535 -m state --state NEW -j ACCEPT
```

Outils de diagnostic

Traces iptables. Possibilité de tracer certaines actions iptables.

exemple :

1. Tracer toutes les actions iptables :

```
iptables -A OUTPUT -j LOG
iptables -A INPUT -j LOG
iptables -A FORWARD -j LOG
```

2. Rajouter une règle pour tracer les paquets rejetés

```
iptables -N LOG_DROP
iptables -A LOG_DROP -j LOG --log-prefix
'[IPTABLES DROP] : '
iptables -A LOG_DROP -j DROP
```

nmap, nessus,.... Logiciels permettant de diagnostiquer l'état d'un firewall (trouver les ports ouverts, détecter les services utilisant les ports, ...)

Administration réseau
Gestion des utilisateurs (NIS)

A. Guermouche

Plan

1. introduction

2. Fonctionnement

3. Configuration de NIS

4. Conclusion

Plan

1. introduction

2. Fonctionnement

3. Configuration de NIS

4. Conclusion

Objectifs

- Centraliser les connexions sur un réseau local
  - Se connecter à un serveur de fichier sous un compte centralisé
  - Ne pas définir de compte machine par machine
- Réseau homogène Linux
  - Connexion et authentification grâce au service NIS
  - Accès aux répertoires partagés grâce à NFS
  - Pour utiliser des stations Windows : serveur SAMBA
- Serveur NIS
  - Au moins un par réseau
  - Plusieurs : soit un par domaine NIS soit serveurs coopératifs (un maître et des esclaves)

NIS

NIS (Network Information System):

- introduit par SUN en 1985 (Yellow Pages (yp) à l'origine)
- n'est pas un standard, mais est très largement utilisé
- une base de donnée distribuée qui permet le partage d'informations système (login, mot de passe, . . .)

Objectifs:

- simplifier la gestion des comptes, des mots de passe et les tâches d'administration dans le monde UNIX
- il suffit de créer un utilisateur sur le serveur NIS pour que chaque machine client NIS ait accès aux informations de *login* de cet utilisateur

Plan

1. introduction

2. Fonctionnement

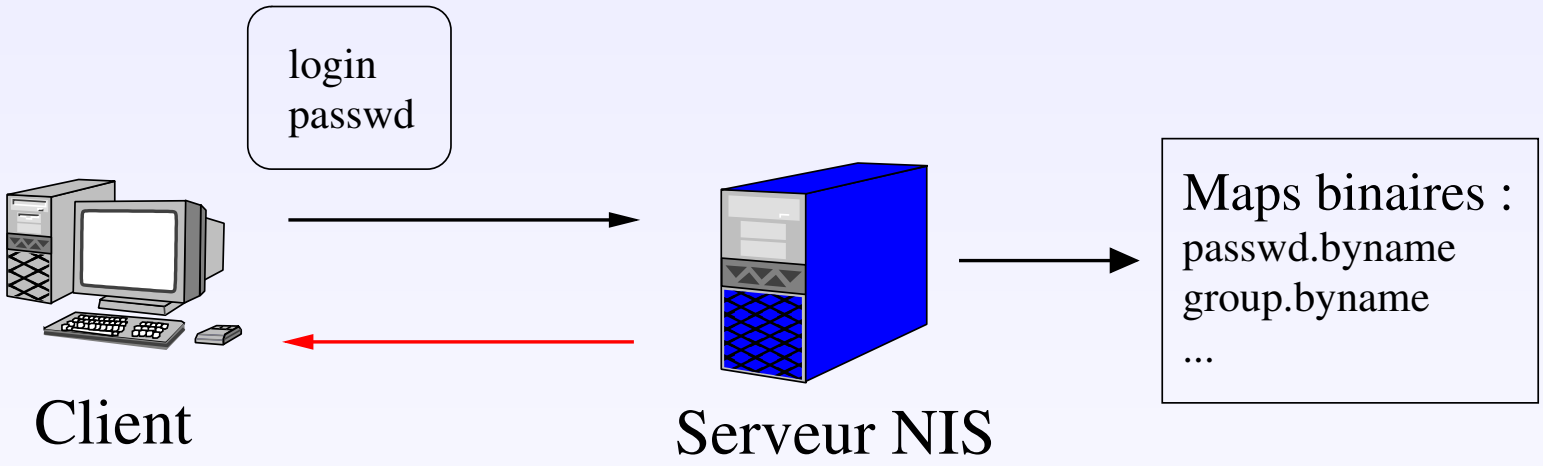
3. Configuration de NIS

4. Conclusion

Fonctionnement

NIS maintient une base de donnée au format DBM sur un domaine NIS

Exemple de fonctionnement :



- Au moins un serveur NIS par réseau
- Possibilité d'en avoir Plusieurs : soit un par domaine NIS soit serveurs coopératifs (un maître et des esclaves)

Architecture

Achitecture : Découpage en domaines.

- modèle Client/Serveur au dessus des SUN-RPC
- un domaine NIS contient :
  - un serveur NIS maître qui maintient les *“maps”* (informations contenues dans la base)
  - aucun, un ou plusieurs serveurs NIS esclaves :
    - permet de décharger le seveur NIS principal et d’être plus résistant aux pannes
    - le maître réplique ses informations vers les serveurs secondaires
    - seul le maître peut modifier une map
    - les esclaves diffusent les maps sans pouvoir les modifier (diminue les problèmes de cohérence)
- des clients NIS qui peuvent interroger les serveurs qu’ils soient maître ou esclaves.

Plan

1. introduction

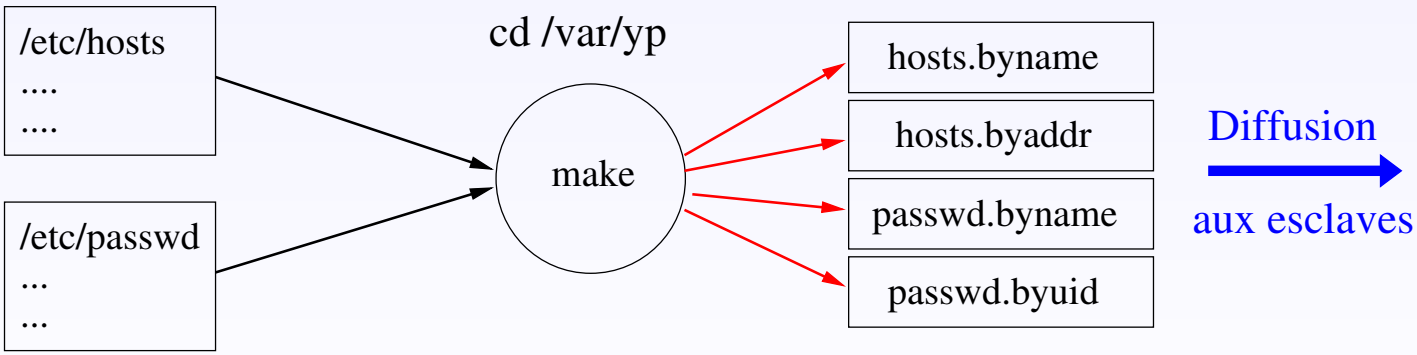
2. Fonctionnement

3. Configuration de NIS

4. Conclusion

En pratique . . .

- Les maps sont stockées dans le répertoire : `/var/yp/nom_de_domaine`
- Quand le fichier source d'une map est modifié sur le serveur (ajout d'un utilisateur, changement de mot de passe,. . .), il faut regénérer la map associée et enventuellement propager les modifications aux serveurs NIS esclaves
- Chaque map stocke des couples clé/valeur



En pratique . . .

- La commande `ypcat` permet de voir le contenu d'une map depuis n'importe quel client
- Au niveau d'Un client NIS :
  - il est nécessaire de se lier (binding) à un serveur pour pouvoir l'interroger
    - deux méthodes : diffusion (traiter la première réponse) ou désignation explicite d'un serveur
    - nom de domaine positionné à l'aide de la commande `domainname` ou dans le fichier `/etc/defaultdomain`, ...
    - Le demon `ypbind` doit tourner pour rechercher régulièrement le serveur approprié.
  - `ypwhich` permet de connaître le nom du serveur NIS
  - `ypset` permet de positionner le nom du serveur (pour désignier explicitement un serveur NIS)
  - possibilité de configuration du nom du serveur NIS correspondant à un nom de domaine dans le fichier `/etc/yp.conf` :

```
domain nom_de_domaine
ypserver nom_de_serveur
```

Configuration du client

2 méthodes :

`nsswitch.conf` :

- Détermine l'ordre de recherche pour l'authentification des utilisateurs.

  ```
  hosts:    files dns nis
  networks: nis files
  ```

  pour les entrées passwd, group et shadow : 2 solutions

  ```
  passwd:    files nis
  ```

  ou

  ```
  passwd:    compat
  ```

`/etc/passwd` :

- Ajouter +:::::: à la fin de `/etc/passwd` au niveau du client.

  Remarque :

  Il suffit de remplacer "+" par "-" pour exclure les utilisateurs NIS d'une marchine.

Configuration du serveur

- Un serveur NIS esclave doit faire tourner :
  - `ypserv` pour répondre aux requêtes de ses clients NIS
  - `ypbind` s'il est lui-même un client NIS (n'est pas obligatoire)
- Un serveur NIS maître doit faire tourner :
  - `ypserv` pour répondre aux requêtes de ses clients NIS
  - `ypbind` s'il est lui-même un client NIS (n'est pas obligatoire)
  - `ypxfrd` pour répondre aux demandes de mise à jour des maps de la part des serveurs esclaves
  - `rpc.yppasswd` pour assurer les demandes de changement de mot de passe (`passwd`)

Plan

1. introduction

2. Fonctionnement

3. Configuration de NIS

4. Conclusion

NIS: évolutions

Défauts de NIS :

- Pas d'authentification des clients NIS : il suffit de connaître le nom de domaine pour interroger le serveur et connaître le contenu des maps.
- les maps sont transmises dans leur intégralité même en cas de faible modification de leur contenu.
- pas adapté aux WAN (broadcast,...)

NIS+ : Un successeur éphémère sans succès qui a été abandonné au profit de LDAP.

NIS: évolutions

Défauts de NIS :

- Pas d'authentification des clients NIS : il suffit de connaître le nom de domaine pour interroger le serveur et connaître le contenu des maps.
- les maps sont transmises dans leur intégralité même en cas de faible modification de leur contenu.
- pas adapté aux WAN (broadcast,...)

NIS+ : Un successeur éphémère sans succès qui a été abandonné au profit de LDAP.

Cependant, NIS continue à être largement utilisé.

Administration réseau
Accès aux fichiers distants

A. Guermouche

Plan

1. Introduction

2. NFS

3. SAMBA

Plan

1. Introduction

2. NFS

3. SAMBA

Accès aux fichiers distants

Différences avec le transfert de fichier :

- ⋆ l'accès aux fichiers distants est complètement transparent pour l'utilisateur
- ⋆ tout se passe comme si le système de fichier distant était local
- ⋆ l'utilisateur peut éditer le fichier, le modifier, . . . ; les modifications seront répercutées sur le système fichier distant

Les deux principaux protocoles :

NFS. Network File System (Unix/Sun-RPC)

SMB. Sserver Message Block (issu du monde Microsoft)

Plan

1. Introduction

2. NFS

3. SAMBA

NFS : Network File System

Présenté par SUN en 1985 pour permettre à ces stations sans disque d'accéder à un système de gestion de fichiers distants (RFC 1904).
Utilise les appels de procédures distantes Sun-RPC (qui sont issues des travaux sur NFS)

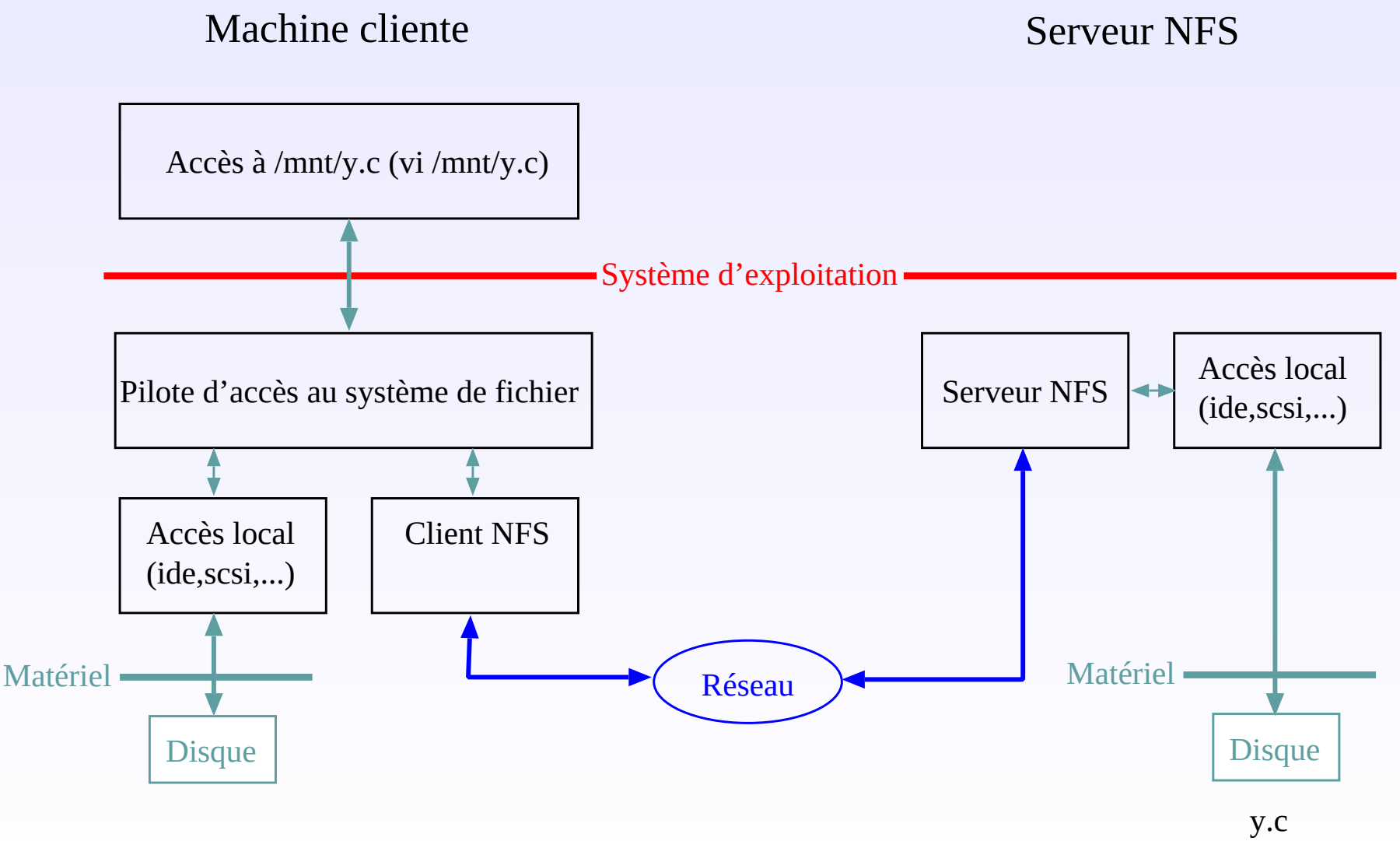
- à priori, les clients et serveur NFS devraient être des processus utilisateur s'exécutant au-dessus de RPC/XDR/UDP/IP.
- en fait, le client et le serveur NFS s'exécutent dans le noyau
  - le client pour rendre transparent l'accès à un fichier via NFS
  - le serveur pour des raisons d'efficacité

NFS : Network File System

Présenté par SUN en 1985 pour permettre à ces stations sans disque d'accéder à un système de gestion de fichiers distants (RFC 1904).
Utilise les appels de procédures distantes Sun-RPC (qui sont issues des travaux sur NFS)

- à priori, les clients et serveur NFS devraient être des processus utilisateur s'exécutant au-dessus de RPC/XDR/UDP/IP.
- en fait, le client et le serveur NFS s'exécutent dans le noyau
  - le client pour rendre transparent l'accès à un fichier via NFS
  - le serveur pour des raisons d'efficacité

Principe de fonctionnement



Les éléments d'accès aux fichiers

- Manipule des descripteurs, chemins, déplacements

Sytème d'exploitation

- Manipule des Files, Dentries, Inodes, déplacements
- Masque les différences à l'application (API uniforme, ...)

Matériel

- Manipule des blocs
- Matériel-dépendant

Le choix entre NFS, EXT3,... se fait lors de l'ouverture du fichier.

NFS et les RPCs

NFS repose sur les RPC (Remote Procedure Calls)

- ★ Utilisation du portmapper (programme portmap de Linux)
- ★ Portmapper = conversion des n° de prog RPC en numéro de port

Déroulement d'une RPC :

- ★ Serveur RPC :
  - → Indique à portmap le port qu'il utilise et les numéros de programme RPC qu'il gère
- ★ Envoi d'une requête RPC par un client :
  - → Il contacte portmap du serveur pour connaître le numéro de port du programme souhaité
  - → Il envoie les données au port correspondant

Un serveur sans état

Dans un accès à un système de fichier local :

- Les accès reposent sur un pointeur de fichier maintenu au niveau du système d'exploitation

NFS est basé sur une connexion réseau :

$\Rightarrow$ Probabilité d'une panne importante

Solution :

- Le serveur NFS ne conserve aucune information sur les accès/opérations effectuées (fichiers ouverts, accès précédents au fichier,...)
- Le système d'exploitation du client NFS se charge de maintenir les informations concernant les fichiers

Intérêts :

- Simplifie le redémarrage du serveur en cas de crash.

Sécurité

Principe d'authentification :

- $(\texttt{uid}, \texttt{gid})_{\texttt{local}}$ "mappé" sur $(\texttt{uid}, \texttt{gid})_{\texttt{distant}}$
  $\rightarrow$ équivalence entre les droits locaux et les droits distants
- Problème pour `root` :
  - Quels droits possède le *root* d'une machine cliente sur les fichiers exportés par un serveur NFS?
  - par défaut `root` (coté client) correspond a l'utilisateur `nobody` (coté serveur) pour des raisons de sécurité (sinon il faut mettre l'option `no_root_squash` dans `/etc/exports`)

Règle de non transitivité :

- Si A exporte `/home` à B; Si B monte `A:/home` dans `/home2` et exporte `/home2` à C alors C n'aura pas accès au `/home` de A

Liens symboliques :

- Les liens symboliques relatifs sont interprétés par rapport au système de fichier du client.

NFS en pratique (1/2)

Démons importants utilisés par NFS :

- **portmap.** Gestion des connexions des applications utilisant le mécanisme de RPC.
- **nfsd.** Authentification + Création, recherche, lecture et écriture de fichiers
- **mountd.** Montage des systèmes exportés (mount et umount)
- **statd.** Surveillance des nœuds du réseau (redémarrages. . .)
- **lockd.** Section critique (lock les fichiers utilisés)

NFS en pratique (2/2)

coté client. le fichier `/etc/fstab` doit contenir le chemin vers le point de montage et le chemin sur le serveur NFS.

```
192.168.0.1:/home /nfs nfs defaults,noauto
0 0
```

coté serveur. le fichier `/etc/exports` contient le chemin vers les dossiers à exporter ainsi
que la liste les machines autorisées à y accéder. `/home 192.168.1.0/255.255.255.0(rw,no_root_squash)`

Après chaque modification de `/etc/exports` il est nécessaire :

- soit d'exécuter `exportfs` pour transmettre les modifications au serveur nfs
- soit de relancer le serveur NFS

Plan

1. Introduction

2. NFS

3. SAMBA

SMB : Server Message Block

- Protocole de Microsoft et Intel permettant le partage de ressources (disques, imprimantes, . . .) à travers un réseau (1987)
- SMB est prévu pour être utilisé au dessus de l'interface NetBIOS
  - Utilisation des noms NetBIOS (15 caractères + 1 pour le type)
  - Utilisation du mécanisme de datagramme de NetBIOS par *broadcast* comme service de nommage (nom → MAC, pas d'adresse de niveau 3)

<table>
<tr><td colspan="3">Application</td></tr>
<tr><td colspan="3">SMB</td></tr>
<tr><td colspan="3">NetBIOS</td></tr>
<tr><td>TCP/IP</td><td>NetBEUI</td><td>IPX/SPX</td></tr>
<tr><td>802.x</td><td>PPP</td><td>...</td></tr>
</table>

SMB (1/2)

- Chaque machine client ou serveur possède un nom sur 15 caractères
- SMB ajoute un 16ème caractère pour distinguer les serveurs de fichiers, les clients, les imprimantes, ...
- Notion de domaine
  - un ensemble d'utilisateurs (avec nom et mot de passe) et de serveurs (avec des droits d'accès)
  - un *primary domain server* contient la base de données des utilisateurs et de leur mot de passe
- Un serveur une ou plusieurs ressources
  - fichiers, imprimantes, ...
  - à chaque triplet (domaine, serveur, ressource) correspond un nom unique : `\\serveur\ressource`

SMB (2/2)

Deux niveaux de protection :

- au niveau de chaque utilisateur : basé sur le nom des utilisateurs, permet de gérer l'accès aux ressources voire aux éléments d'une ressource
- au niveau de chaque ressource : un mot de passe commun à tous les utilisateurs est associé à une ressource pour y autoriser l'accès

Résolution de noms : 4 méthodes utilisées

broadcast. résolution par diffusion d'une requête dans le réseau

lmhost. résolution en utilisant des associations prédéfinies entre noms NetBIOS et IP

host. utilisation de DNS

wins. utilisation d'un serveur WINS (*Windows Internet Name Server*). À chaque machine est associé un serveur WINS à qui elle envoie ses requêtes et auprès duquel elle s'enregistre.

SAMBA (1/2)

Samba : Implémentation de SMB sous UNIX qui permet le partage de ressources entre les mondes UNIX et Windows
Samba permet de :

- partager un disque UNIX pour des machines Windows
- accéder à un disque Windows depuis une machine UNIX
- partager une imprimante UNIX pour des machines Windows
- utiliser une imprimante Windows à partir d'un hôte Linux.

Le serveur Samba sur la machine Unix émule un domaine SMB

SAMBA (2/2)

Serveur Samba :

- ★ configuration via le fichier `/etc/smb.conf`
- ★ travail partagé par deux démons :
  - `smbd`. pour le service “serveur”
  - `nmbd`. pour le service résolution des noms NetBIOS

Client :

`smbpasswd`. permet de changer le mot de passe d’un utilisateur SMB

`smbclient`. permet d’interroger un serveur Samba depuis UNIX `smbclient //host/ressource` permet l’accès à la ressource

Possibilité de monter une partition Windows distante à l’aide de Samba → utiliser le système de fichier `smbfs`

Exemple : (extrait du fichier `/etc/fstab`)

```
//serveur/ressource /commun smbfs defaults 0 0
```

Administration réseau
Annuaire LDAP

A. Guermouche

Plan

1. Introduction

2. Annuaire LDAP

3. Protocole LDAP

4. LDAP en pratique

Plan

1. Introduction

2. Annuaire LDAP

3. Protocole LDAP

4. LDAP en pratique

Objectifs

Permet de fusionner plusieurs bases de données en un unique annuaire informatique

- base Microsoft Excel du personnel administratif
- base Microsoft Access du personnel enseignant
- base `/etc/passwd` des comptes unix
- base `/etc/aliases` (ou Sympa) des listes de diffusion
- base Samba des utilisateurs windows
- autres bases, MySQL, ...
- ...

Exemple:

Comment envoyer un e-mail a l'ensemble du personnel administratif en sachant que l'administrateur recevra uniquement une liste de noms/prénoms?

Le concept d'annuaire

Un annuaire est comme une base de données. . .

→ on peut y mettre des information et les consulter

Cependant un annuaire est spécialisé :

→ Dédié à la lecture plus qu'à l'écriture

→ L'accès aux données se fait par des recherches multi-critères.

Son objectif est de maintenir de façon cohérente et contrôlée une grande quantité de données.

Exemples d'annuaire :

- ★ carnet d'adresses
- ★ annuaire téléphonique
- ★ répertoire des rues
- ★ . . .

Le concept d'annuaire

Un annuaire est comme une base de données. . .

→ on peut y mettre des information et les consulter

Cependant un annuaire est spécialisé :

→ Dédié à la lecture plus qu'à l'écriture

→ L'accès aux données se fait par des recherches multi-critères.

Son objectif est de maintenir de façon cohérente et contrôlée une grande quantité de données.

Différences annuaires/SGBD : Dans un annuaire :

- ⋆ pas de dépendances entre les objets stockés
- ⋆ les objets peuvent être distribués sur plusieurs annuaires pour assurer une meilleure disponibilité
- ⋆ les applications de l'annuaire n'ont pas besoin de connaître la structure interne des données stockées

Plan

1. Introduction

2. Annuaire LDAP

3. Protocole LDAP

4. LDAP en pratique

L'annuaire LDAP

LDAP → **L**ightweight **D**irectory **A**ccess **P**rotocol
Héritier de l'annuaire X500 (proposé par l'ISO)

- ⋆ standard conçu par les opérateurs télécom pour interconnecter leurs annuaires téléphoniques
- ⋆ X500 adapté à l'internet → LDAP (même modèle de schéma, ...)

LDAP a été proposé en 1995 :

- ⋆ Standard d'annuaire au dessus de TCP/IP
  - ▸ Le protocole ne concerne pas le contrôle d'accès aux données de l'annuaire
- ⋆ version 3 actuellement (RFC 2251)
- ⋆ aussi RFC 2252 à 2256, RFC 2829 à 2830, RFC 2849

Objectifs

- fournir aux utilisateurs des informations fiables, facilement accessibles
- permettre aux utilisateurs de mettre à jour eux-même leurs informations personnelles
- rendre les informations accessibles de façon contrôlée
- éviter la redondance d'informations : un seul annuaire pour l'ensemble des services
- faciliter la gestion (administration) des postes de travail, des équipements réseau

Tout ceci est fait sans remettre en cause les applications existantes

Concepts

LDAP définit :

un protocole. accéder à l'information contenue dans l'annuaire,

un modèle d'information. le type des informations contenues dans l'annuaire,

un modèle de nommage. comment l'information est organisée et référencée,

un modèle fonctionnel. comment accéder à l'information (syntaxe des requêtes,etc...),

un modèle de sécurité. comment données et accès sont protégés,

un modèle de duplication. comment la base est répartie entre serveurs,

des API. pour développer des applications clientes,

LDIF. un format d'échange de données.

Protocole LDAP

Le protocole définit :

- Comment s'établit la communication client-serveur :
  - → commandes pour se connecter ou se déconnecter, pour rechercher, comparer, créer, modifier ou effacer des entrées.
- Comment s'établit la communication serveur-serveur :
  - → échanger leur contenu et le synchroniser (réplication service)
  - → créer des liens permettant de relier des annuaires les uns aux autres (referral service).
- Le format de transport de données :
  - → pas l'ASCII (comme pour HTTP, SMTP...) mais le Basic Encoding Rules (BER), sous une forme allégée (appelée LBER : Lightweight BER)

Protocole LDAP

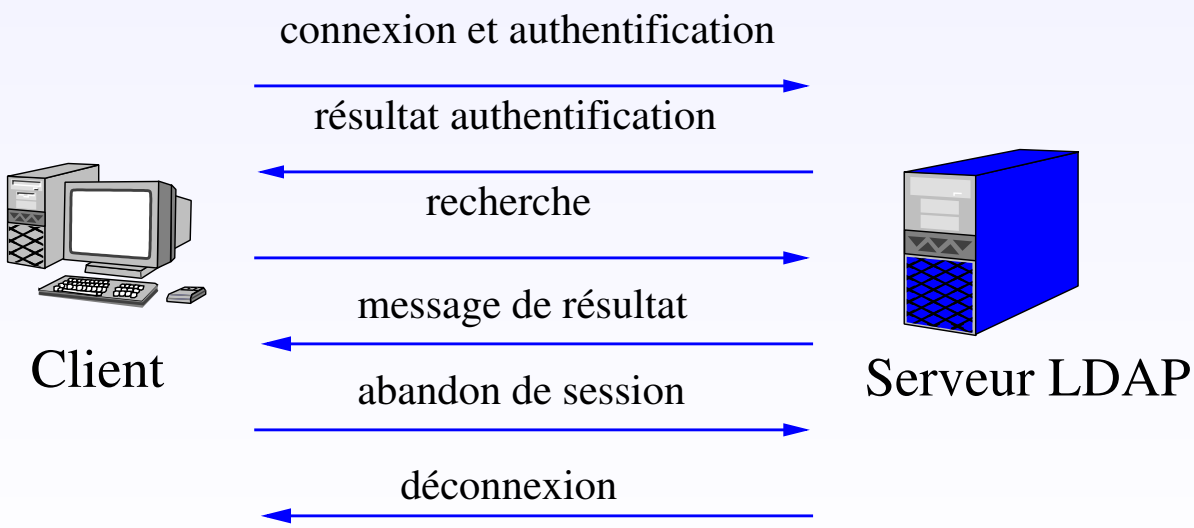
Le protocole définit (suite) :

- Les mécanismes de sécurité :
  - → méthodes de chiffrement et d'authentification
  - → mécanismes de règles d'accès aux données.
- Les opérations de base :

  interrogation : search, compare

  mise à jour : add, delete, modify, rename

  connexion au service : bind, unbind, abandon



Plan

1. Introduction

2. Annuaire LDAP

3. Protocole LDAP

4. LDAP en pratique

Le modèle d'information

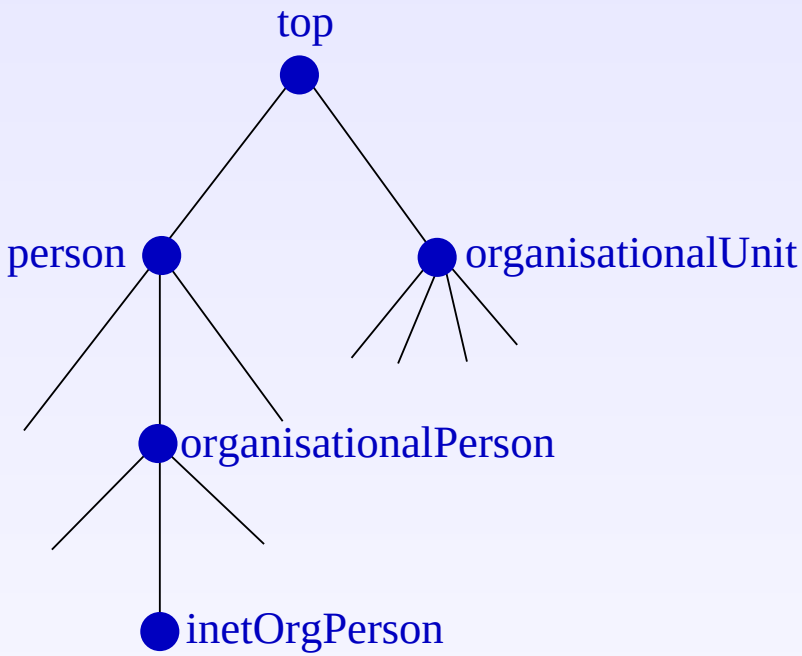
Le modèle d'information définit le type des données pouvant être stockées dans l'annuaire

- L'entrée (Entry) = élément de base de l'annuaire. Elle contient les informations sur un objet de l'annuaire.
- Ces informations sont représentées sous la forme d'attributs décrivant les caractéristiques de l'objet.
- Toute sorte de classe d'objet (réel ou abstrait) peut être représentée.
- Le *schéma* de l'annuaire définit la liste des classes d'objets qu'il connaît. Le *Directory schema* est la « charte » qui donne, pour le serveur, l'ensemble des définitions relatives aux objets qu'il sait gérer.
  - Le schéma décrit les classes d'objets, leurs types d'attribut et leur syntaxe.
  - Chaque entrée de l'annuaire fait obligatoirement référence à une *classe d'objet* du *schéma* et ne doit contenir que des attributs qui sont rattachés au type d'objet en question.

Le modèle d'information

- Un attribut est défini par :
  - un nom, un identifiant unique (OID), mono/multi valué, une syntaxe et des règles de comparaison (*matching rules*), une valeur (format+taille limite), modifiable ou non
- Les classes d'objet modélisent
  - des objets réels : Un compte UNIX (`posixAccount`), une organisation (`o`), un département (`ou`), un personnel (`organizationPerson`), une imprimante (`device`),...
  - ou abstraits : l'objet père de tous les autres (`top`),...
- Une classe d'objet est définie par
  - Un nom, OID, des attributs obligatoires, des attributs optionnels, un type (structurel, auxiliaire ou abstrait)

Le modèle d'information



- Chaque objet hérite des propriétés (attributs) de l'objet dont il est le fils.
- On précise la classe d'objet d'une entrée à l'aide de l'attribut objectClass.
- Il faut obligatoirement indiquer la parenté de la classe d'objet en partant de l'objet top et en passant par chaque ancêtre de l'objet

Le modèle d'information (exemple)

L'objet `inetOrgPerson` à la filiation suivante :

```
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
```

L'objet `person` a comme attributs : `commonName, surname, description, seeAlso, telephoneNumber, userPassword`

L'objet fils `organizationalPerson` ajoute des attributs comme : `organizationUnitName, title, postalAddress...`

L'objet petit-fils `inetOrgPerson` lui rajoute des attributs comme : `mail, labeledURI, uid (userID), photo...`

Remarques :

- Une entrée peut appartenir à un nombre non limité de classes d'objets.
- Les attributs obligatoires sont la réunion des attributs obligatoires de chaque classe.

Le modèle de nommage

Il définit comment les entrées de l'annuaire sont organisées et comment elles sont référencées.
Structure arborescente contenant deux catégories d'objets :

les conteneurs : départ d'une nouvelle branche (nœud intermédiaire de l'arbre)

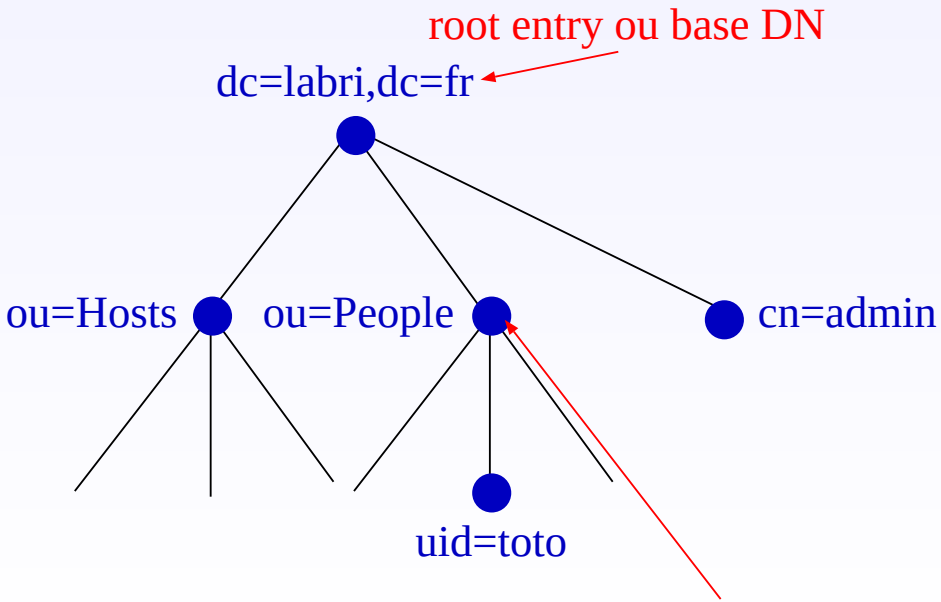
- peuvent contenir des conteneurs ou des feuilles
- généralement, une sous-organisation de l'organisation (zone géographique,...)

les feuilles : elles représentent les données (généralement les machines, les utilisateurs,...)

Le modèle de nommage

Il définit comment les entrées de l'annuaire sont organisées et comment elles sont référencées.
Structure arborescente contenant deux catégories d'objets :

- ★ Structure logique hiérarchique : le DIT (Directory Information Tree)
- ★ Une entrée est identifiée par un nom unique : le DN (Distinguish Name)
- ★ RDN(Relative Distinguish Name)



Le format LDIF

LDIF → **L**DAP **I**nterchange **F**ormat

- ★ Standard de représentation des entrées sous format texte.
- ★ Permet de :
  - ▶ faire des *imports/exports* de la base ou d'une partie de la base
  - ▶ créer, ajouter, modifier, . . . un grand nombre d'entrées de manière automatisée

```
dn: uid=toto, ou=People, dc=labri, dc=fr
objectClass: top
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
uid: toto
uidNumber: 44321
gidNumber: 200
homeDirectory: /home/toto
cn: toto titi
loginShell: /bin/bash
```

Le modèle fonctionnel

Il décrit le moyen d'accéder aux données (syntaxe des requêtes) et les requêtes que l'on peut leur appliquer.
Rappel des opérations de consultation/mise-à-jour

- ⋆ opérations d'interrogation : recherche (`search`) et comparaison (`compare`) d'entrées
- ⋆ opérations de mise-à-jour des entrées de l'annuaire : `add, delete, modify, rename`

Il n'y a pas d'opération de lecture d'une entrée

→ pour connaître le contenu d'une entrée, il est nécessaire d'écrire une requête qui pointe sur cette entrée.

Le modèle de réplication

Il définit comment dupliquer l'annuaire sur plusieurs serveurs.

- ★ améliorer le temps de réponse
- ★ être tolérant aux pannes

Deux types de serveurs LDAP

supplier server: fournit les données

consumer server: reçoit les données du maître

Possibilité de partitionner l'annuaire (éclatement sur plusieurs serveurs)

- ★ liens virtuels entre les différentes partitions (*referral service*)

Le modèle de sécurité

Authentification pour se connecter au service

- Anonymous authentication, Root DN/passwd authentication (administrateur), User DN/passwd

Contrôle de l'accès aux données

- droits d'accès aux données (fonctions de l'utilisateur authentifié)
- règles définies sous forme d'ACL (*Access Control List*) au niveau du sommet d'un sous-arbre ou d'une entrée.

Chiffrement des transactions (LDAP+SSL, ...)

Plan

1. Introduction

2. Annuaire LDAP

3. Protocole LDAP

4. LDAP en pratique

Mettre en place un annuaire LDAP

Il faut bien choisir les schémas

- Quelles informations veut-on stocker dans l'annuaire?
- Quelles sont les applications qui vont utiliser l'annuaire?

Il faut réfléchir a l'organisation du DIT

- impact sur la performance, les droits d'accès, ...

Puis dans un deuxième temps

- gestion centralisée sur un seul serveur?
- nombre de serveurs redondants? Emplacement?

OpenLDAP

- Logiciel LDAP du domaine public
- le démon `slapd`
  → traite les requêtes LDAP
- le démon `slurpd`
  → permet la réplication
- des librairies LDAP
  - par exemple pour authentifier les logins via LDAP : `libpamldap, libnssldap`
- des utilitaires :
  - `ldapadd, ldapdelete, ldapmodify, ldapmodrdn, ldappasswd, ldapsearch`

Configuration du serveur ldap(1/2)

Le fichier `/etc/ldap/slapd.conf` permet de configurer le démon `sladp`

- définition des schémas utilisés
  `include inetorgperson.schema`
- définition du *backend* (type de la base de données utilisée)
  `backend bdb`
- définition de la base, de l'annuaire et de l'administrateur
  - le suffixe (racine de l'arbre)
    `suffix "dc=labri,dc=fr"`
  - l'administrateur et son mot de passe
    `rootdn "cn=Manager,dc=labri,dc=fr"`
    `rootpw MD5x0dg9sP0uUf+NRm0MIPz7Q==`
  - le répertoire où la base est stockée
    `directory "/var/lib/ldap"`

Configuration du serveur ldap(1/2)

Définition des ACLs (`man slapd.access`)

```
 # par défaut
access to attrs=userPassword
            by dn="cn=admin,dc=com" write # l'admin
            by anonymous auth # droit de lecture lors du
            bind
            by self write # le propriétaire
            by * none
access to dn.base="" by * read
# L'administrateur a un accès total en écriture, tous
# les autres utilisateurs peuvent tout lire.
access to *
            by dn="cn=admin,dc=com" write
            by * read
```

configuration du client LDAP

La configuration se fait grâce au fichier `/etc/ldap/ldap.conf`

- `man ldap.conf`
- peut aussi se faire dans `/.ldaprc`
- exemple de fichier `ldap.conf`

```
# base par défaut à contacter pour les opérations LDAP
BASE dc=labri, dc=fr
# en tant que qui le client va se connecter
# à la base
BINDDN uid=toto,ou=People,dc=labri,dc=fr
# le serveur auquel se connecter
URI ldap://147.210.20.21:389/
```

Authentification Unix via LDAP

- PAM (Pluggable Authentication Modules)
  - permet de gérer la politique d'authentification sans recompilation
  - pour authentifier via LDAP, il faut ajouter la ligne `auth sufficient pam_ldap.so` (qui signifie que l'authentification LDAP est suffisante) dans le fichier `/etc/pam.d/common-auth`. Il faut faire de même pour tous les autres fichiers `/etc/pam.d/common-*`.
  - Modifier éventuellement `/etc/pam.d/ssh`,...
- Configurer l'accès à la base dans `/etc/libnss-ldap.conf` et `/etc/pam_ldap.conf` (voir pages man)
- Indiquer dans `/etc/nsswitch.conf` l'ordre d'interrogation pour l'authentification

  → toujours laisser `files` en premier !

Administration réseau
Résolution de noms
et
attribution d'adresses IP

A. Guermouche

Plan

1. DNS
   - Introduction
   - Fonctionnement
   - DNS & Linux/UNIX
2. DHCP
   - Introduction
   - Le protocole DHCP
   - DHCP & Linux/UNIX

Plan

1. DNS
   - Introduction
   - Fonctionnement
   - DNS & Linux/UNIX

2. DHCP
   - Introduction
   - Le protocole DHCP
   - DHCP & Linux/UNIX

DNS

Comment relier les adresses IP utilisées pour acheminer les paquets aux noms utilisés par les applications?
→ **DNS** (**D**omain **N**ame **S**ervice)

- Protocole applicatif
- DNS est utilisé par d'autres protocoles applicatifs mais est rarement utilisé directement par l'application
- modèle client/serveur : un émetteur interroge un serveur de noms (serveur DNS)
- port 53/UDP
- RFC 1034, 1035, 2181, ...

Un système centralisé?

Pourquoi pas de DNS centralisé? Un seul serveur contiendrait toutes les correspondances requises par les applications de l'internet

- dimension de l'internet : trop de correspondances à gérer, nombre de requêtes au serveur trop important
- tolérance aux pannes : si le serveur DNS tombe, tout internet aussi
- trafic impossible à supporter par un seul serveur
- délai de réponse : il faut faire en sorte que la réponse soit la plus proche possible du demandeur
- problème lié à la maintenance et aux mises à jour perpétuelles de la base

Un système distribué

Aucun serveur ne connaît les les correspondances nom↔@IP

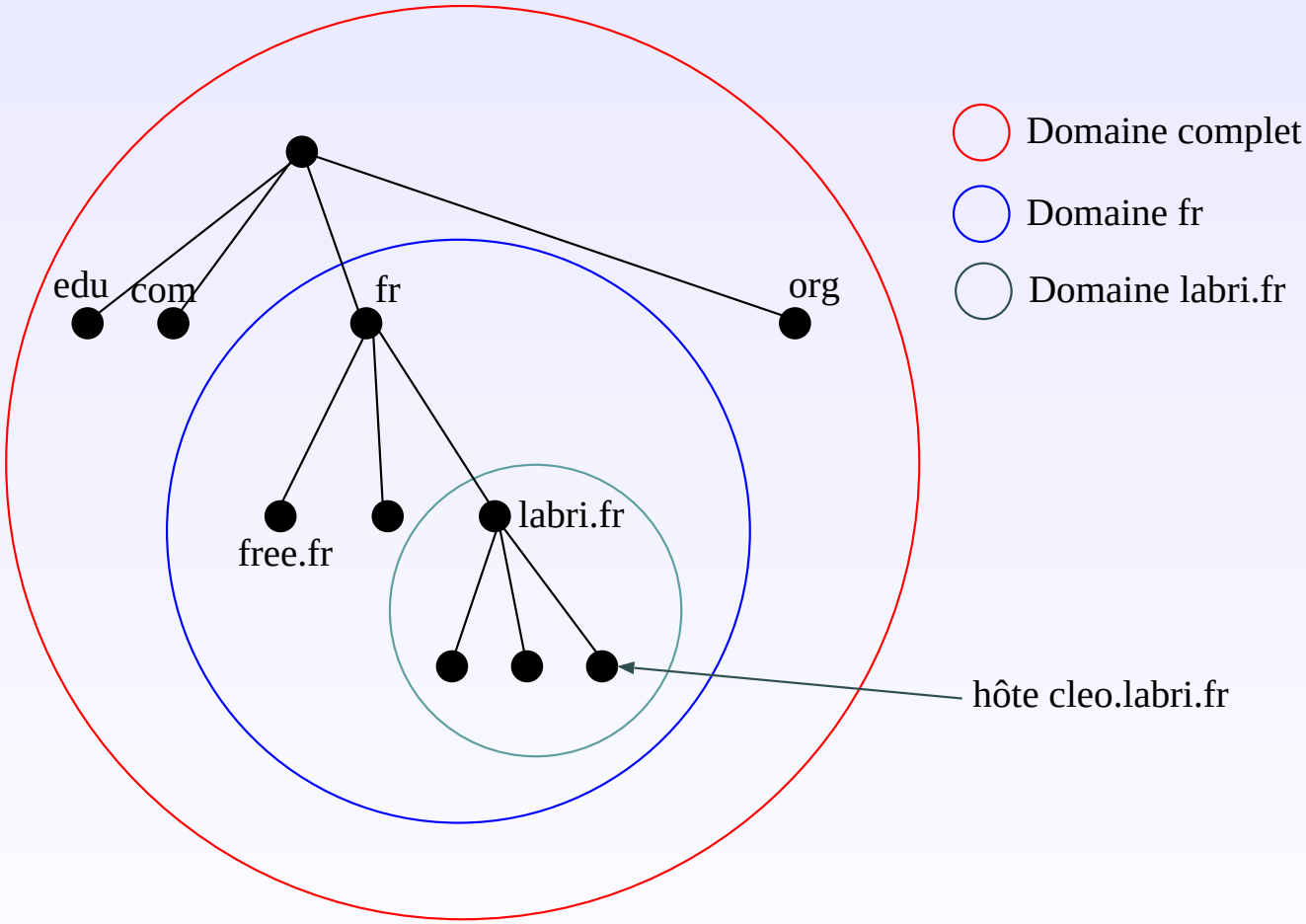
→ si un serveur ne connaît pas une correspondance, il interroge un autre serveur jusqu'à atteindre le serveur détenant l'information

Trois types de serveur DNS :

- ⋆ les serveurs de noms locaux à qui s'adressent les requêtes locales; ils sont en charge de la résolution
- ⋆ les serveurs de noms racine qui sont censés savoir comment se rapprocher de la réponse
- ⋆ les serveurs de noms de source autorisée qui contiennent les correspondances officielles

Domaine DNS (1/2)

Un domaine est un sous-arbre entier de l'espace de "nomage"



Deux nœuds différents peuvent avoir le même nom dans deux domaines différents : `cleo.labri.fr` et `cleo.free.fr`

Domaine DNS (2/2)

Le premier niveau de l'arbre

- Top Level Domain (TLD)
- géré pas l'ICANN (*Internet Corporation for Assigned Names and Numbers*)
- deux types de TLD :
  "generic TLD". `.com`, `.org`, `.gov`, `.net`,...
  "coutries TLD". `.fr`, `.de`, `.uk`,...

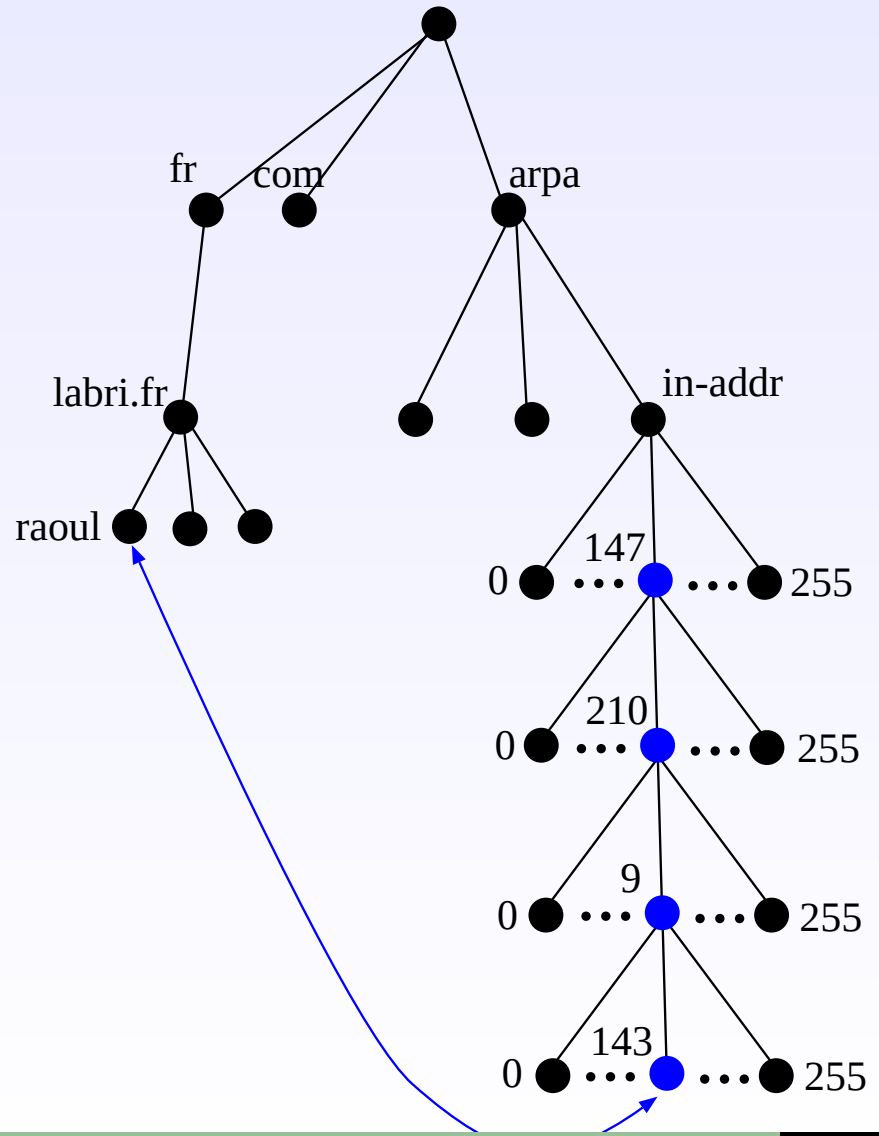
Les autres niveaux sont gérés par des entités "locales" (AFNIC pour `.fr`)

zone DNS :

- Un sous-arbre administré par un organisme qui gère la délégation des noms et sous-domaines de la zone
- Une zone = une administration centralisée avec au moins un serveur DNS (généralement un primaire et un secondaire)
- Une zone doit connaître les serveurs DNS des zones subordonnées

La résolution de noms inverse

- Retrouver le nom canonique à partir de l'adresse IP
- Le domaine `arpa` : un domaine particulier gérer par l'ICANN permettant la résolution inverse
- Résolution inverse : 143.9.210.147.in-addr.arpa ? (pour trouver le nom de la machine dont l'adresse IP est 147.210.9.143)



Les différents types de serveur DNS (1/2)

Les serveurs de noms locaux :

- chaque organisation a un serveur de nom local
  - serveur de noms par défaut de la zone
  - contient parfois les correspondances relatives à la zone de l'organisation
- toutes les requêtes en provenance de cette organisation vont vers ce serveur de noms local

Les serveurs de noms racine :

- il existe 13 serveurs racine dans internet
- chaque serveur DNS local connaît un serveur de noms racine qu'il peut interroger lorsqu'il ne connaît pas une correspondance
- un serveur de nom racine connaît au moins les serveurs de noms de source autorisée du premier niveau (`.fr`, `.com`,...)

Les différents types de serveur DNS (2/2)

Un serveur de noms racine qui ne connaît pas la réponse à une requête interroge un autre serveur de noms le rapprochant de la réponse, généralement le serveur de noms de source autorisée qui connaît la correspondance

Les serveurs de noms de source autorisée :

- chaque hôte est enregistré auprès d'au moins deux *"authoritative servers"* (le primaire et le secondaire) qui stockent son adresse IP et son nom
- un serveur de noms est dit de source autorisée pour un hôte s'il est responsable de la correspondance nom/@IP pour cet hôte (serveur primaire de la zone)
- un serveur de nom local n'est pas forcément de source autorisée de premier niveau (`.fr`,...)

Résolution de noms récursive/itérative

Résolution récursive. la machine qui demande la résolution de nom contacte un serveur DNS et attend que ce dernier lui retourne la réponse désirée.

Résolution itérative. le serveur de noms contacté fournit en réponse le nom d'un autre serveur DNS à contacter pour avancer dans la résolution.

Dans une résolution de nom, certaines requêtes peuvent être itératives, d'autres récursives.

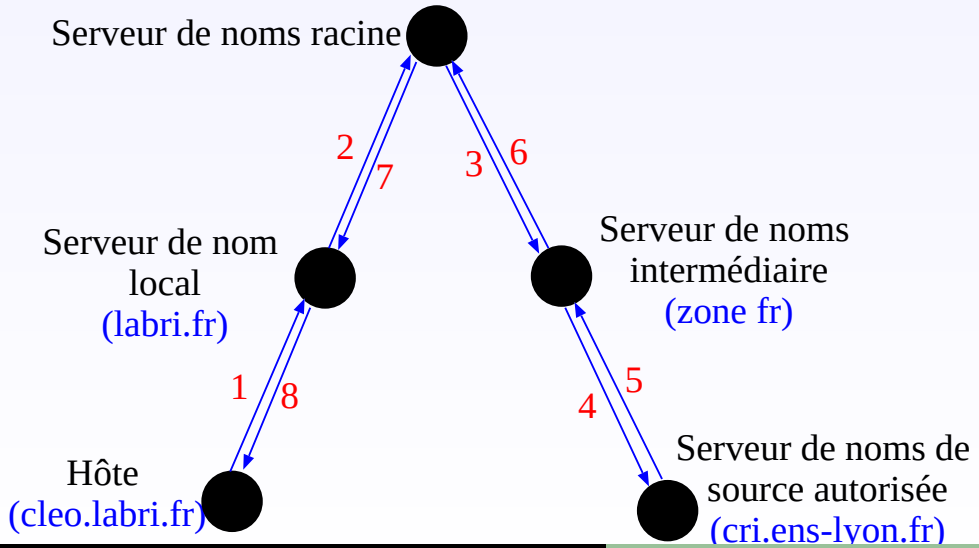
Résolution de noms récursive/itérative

Résolution récursive. la machine qui demande la résolution de nom contacte un serveur DNS et attend que ce dernier lui retourne la réponse désirée.

Résolution itérative. le serveur de noms contacté fournit en réponse le nom d'un autre serveur DNS à contacter pour avancer dans la résolution.

Dans une résolution de nom, certaines requêtes peuvent être itératives, d'autres récursives.

Résolution récursive : `ssh.ens-lyon.fr`→?



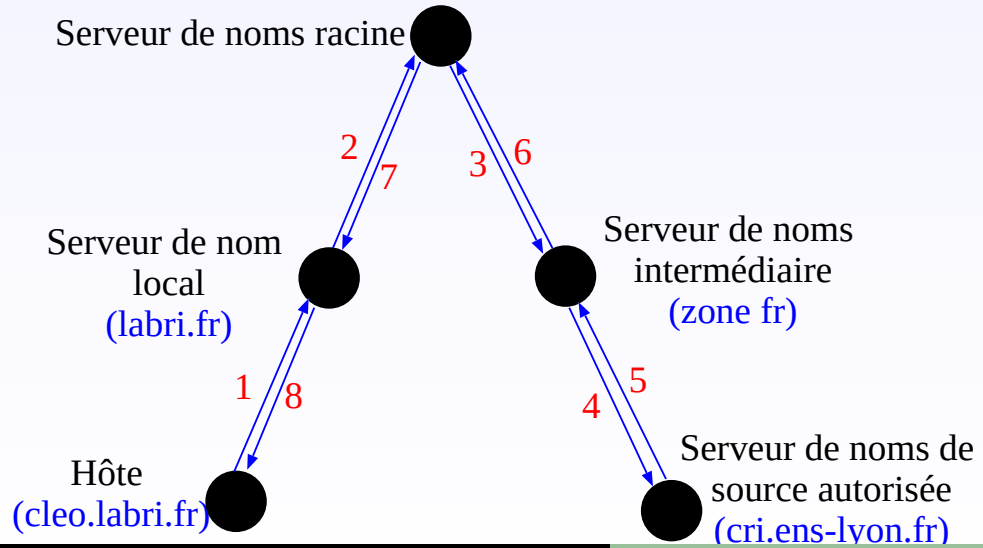
Résolution de noms récursive/itérative

Résolution récursive. la machine qui demande la résolution de nom contacte un serveur DNS et attend que ce dernier lui retourne la réponse désirée.

Résolution itérative. le serveur de noms contacté fournit en réponse le nom d'un autre serveur DNS à contacter pour avancer dans la résolution.

Dans une résolution de nom, certaines requêtes peuvent être itératives, d'autres récursives.

Résolution récursive : `ssh.ens-lyon.fr`→?



Le serveur de noms racine connaît un serveur qui le rapprochera de la réponse (il peut aussi connaître directement le serveur de source autorisée)

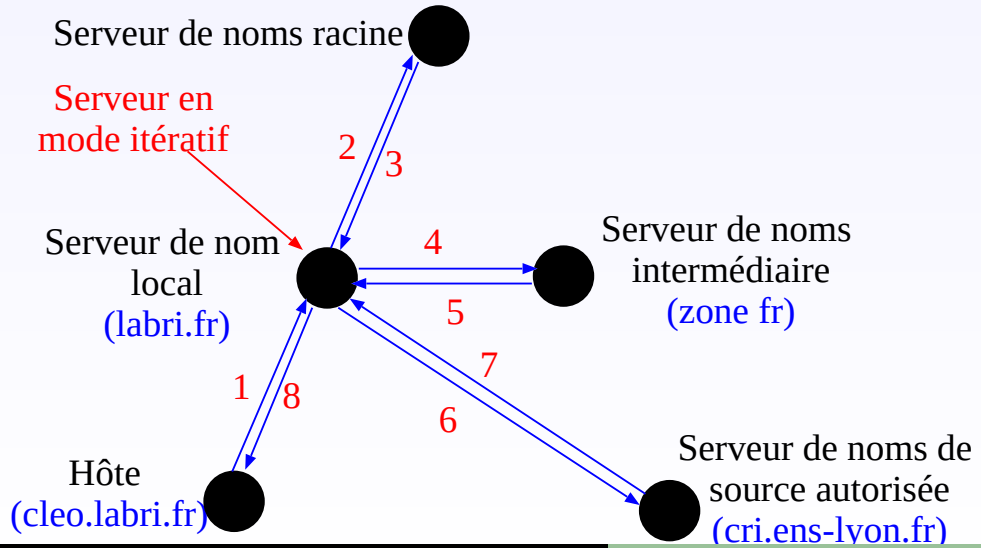
Résolution de noms récursive/itérative

Résolution récursive. la machine qui demande la résolution de nom contacte un serveur DNS et attend que ce dernier lui retourne la réponse désirée.

Résolution itérative. le serveur de noms contacté fournit en réponse le nom d'un autre serveur DNS à contacter pour avancer dans la résolution.

Dans une résolution de nom, certaines requêtes peuvent être itératives, d'autres récursives.

Résolution itérative : `ssh.ens-lyon.fr`→?



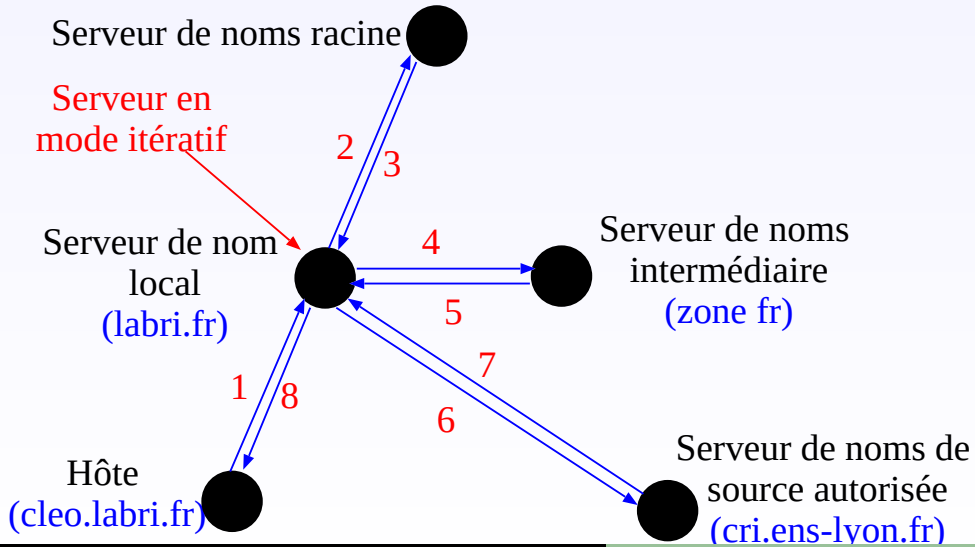
Résolution de noms récursive/itérative

Résolution récursive. la machine qui demande la résolution de nom contacte un serveur DNS et attend que ce dernier lui retourne la réponse désirée.

Résolution itérative. le serveur de noms contacté fournit en réponse le nom d'un autre serveur DNS à contacter pour avancer dans la résolution.

Dans une résolution de nom, certaines requêtes peuvent être itératives, d'autres récursives.

Résolution itérative : `ssh.ens-lyon.fr`→?



généralement toutes les requêtes sont récursives sauf celles entre le serveur local et le serveur racine

⇒ Permet de moins solliciter le serveur racine

Cache DNS

Objectif : Réduire le temps de réponse d'une résolution de nom

- ★ diminuer le nombre de messages DNS nécessaires
- ★ le serveur de noms (quelconque) stocke dans son cache les informations récentes
  - ▸ comme la mémoire n'est pas infinie et que les données peuvent ne plus être valables au bon d'un certain temps, les données "sortent" du cache après un certain temps (TTL d'environ 2 jours)
- ★ Un serveur DNS qui mémorise dans son cache un enregistrement DNS n'a pas autorité dessus

  → spécifie *"no authoritative"* dans la réponse

Les messages DNS (RFC 1034, 1035)

Un message de réponse DNS contient un ou plusieurs RR (*Resource record*)

- l'unité de stockage d'une correspondance dans le cache :
  `(Nom, Type, Classe, TTL, Valeur)`
- `Type` représente le type de l'enregistrement; la signification de `Nom` et `Valeur` dépend de la valeur de `Type`.
  - `Type=A` adresse IPv4
  - `Type=NS` serveur de noms de source autorisée
  - `Type=MX` alias réservé au serveur de mail
  - `Type=PTR` sert à la résolution inverse
  - ...
- `Classe` représente la famille de protocoles; `Classe=1` pour internet (`IN`)
- `TTL` représente la durée de vie de l'entrée dans le cache (en secondes)

Un message de type requête DNS est de la forme :

`(Nom, Type, Classe)`

Clients et Serveur DNS

Côté client :

- le *resolver* est en charge des résolutions de noms à chaque fois que cela est nécessaire
- deux fichiers de configurations :
  - `/etc/resolv.conf`. permet de paramétrer les requêtes DNS effectuées
  - `/etc/host.conf`. permet de configurer les *resolver* (en particulier l'ordre de la résolution) :
    ```
    order hosts,bind,nis
    ```
  - `/etc/nsswitch.conf`. est consulté avant `host.conf` pour la configuration de l'ordre de la recherche
- commandes de test : `host`, `nslookup`, `dig`, ...

Côté serveur : Serveur `BIND` (*Berkeley Internet Name Domain*)

- Le démon répondant aux requêtes est `named`
  - fichier de configuration → `/etc/named.conf`
  - fichiers décrivant les zones administrées stockées dans `/etc/bind/`

Plan

1. DNS
   - Introduction
   - Fonctionnement
   - DNS & Linux/UNIX

2. DHCP
   - Introduction
   - Le protocole DHCP
   - DHCP & Linux/UNIX

DHCP

DHCP → Dynamic Host Configuration Protocol (RFC 2131)

- Protocole client/serveur
- Le serveur DHCP fournit des paramètres de configuration aux clients (généralement des paramètres nécessaires à la configuration du réseau)
- Permet l'attribution automatique d'adresses IP
- Successeur de BOOTP (protocole d'obtention automatique d'adresse IP utilisé pour les stations *"diskless"*)
- Compatible avec BOOTP

DHCP

DHCP → Dynamic Host Configuration Protocol (RFC 2131)

- Protocole client/serveur
- Le serveur DHCP fournit des paramètres de configuration aux clients (généralement des paramètres nécessaires à la configuration du réseau)
- Permet l'attribution automatique d'adresses IP
- Successeur de BOOTP (protocole d'obtention automatique d'adresse IP utilisé pour les stations *"diskless"*)
- Compatible avec BOOTP

3 méthodes de gestion des adresses IP :

Allocation manuelle. Le serveur DHCP attribue l'adresse IP en se basant sur une table d'adresses MAC prédéfinie

Allocation automatique. Le serveur DHCP attribue de manière permanente une adresse IP (parmi l'ensemble des adresses "libres") à un client

Allocation dynamique. Méthode permettant la réutilisation d'adresses IP (basée sur un mécanisme de "bail")

Protocole DHCP (1/2)

- Protocole utilisant les ports 67/UDP (serveur) et 68/UDP (client)
- Échange de messages entre client et serveurs pour l'attribution d'une adresse IP pour une durée donnée
- Protocole basé sur un mécanisme de *broadcast*
- Utilisation de différents types de messages : `DHCP Discover`, `DHCP Offer`, `DHCP Request`, `DHCP Acknowledge`, `DHCP Inform` et `DHCP Release`

Protocole DHCP (1/2)

- Protocole utilisant les ports 67/UDP (serveur) et 68/UDP (client)
- Échange de messages entre client et serveurs pour l'attribution d'une adresse IP pour une durée donnée
- Protocole basé sur un mécanisme de *broadcast*
- Utilisation de différents types de messages : `DHCP Discover`, `DHCP Offer`, `DHCP Request`, `DHCP Acknowledge`, `DHCP Inform` et `DHCP Release`

DHCP Discover.

- Le client diffuse un message dans le réseau local pour trouver les serveurs disponibles.
- La diffusion est faite vers l'adresse 255.255.255.255.
- De plus Le client peut ajouter sa dernière adresse IP obtenue à l'aide de DHCP.

Protocole DHCP (1/2)

- Protocole utilisant les ports 67/UDP (serveur) et 68/UDP (client)
- Échange de messages entre client et serveurs pour l'attribution d'une adresse IP pour une durée donnée
- Protocole basé sur un mécanisme de *broadcast*
- Utilisation de différents types de messages : `DHCP Discover`, `DHCP Offer`, `DHCP Request`, `DHCP Acknowledge`, `DHCP Inform` et `DHCP Release`

DHCP Offer. Le serveur choisit une configuration pour le client (en se basant éventuellement sur l'adresse MAC contenue dans le message du client)

Protocole DHCP (1/2)

- Protocole utilisant les ports 67/UDP (serveur) et 68/UDP (client)
- Échange de messages entre client et serveurs pour l'attribution d'une adresse IP pour une durée donnée
- Protocole basé sur un mécanisme de *broadcast*
- Utilisation de différents types de messages : `DHCP Discover`, `DHCP Offer`, `DHCP Request`, `DHCP Acknowledge`, `DHCP Inform` et `DHCP Release`

DHCP Request.

- Le client sélectionne une configuration parmi celles qu'il a reçues (paquets `DHCP Offer`) et la diffuse dans le réseau.
- Le client met dans le message l'adresse que le serveur lui a proposée.
- S'il a reçu plusieurs "offres" il spécifie le serveur qu'il a choisi.

Protocole DHCP (1/2)

- Protocole utilisant les ports 67/UDP (serveur) et 68/UDP (client)
- Échange de messages entre client et serveurs pour l'attribution d'une adresse IP pour une durée donnée
- Protocole basé sur un mécanisme de *broadcast*
- Utilisation de différents types de messages : `DHCP Discover`, `DHCP Offer`, `DHCP Request`, `DHCP Acknowledge`, `DHCP Inform` et `DHCP Release`

DHCP Acknowledgement.

- Le serveur confirme l'attribution de l'adresse IP au client (le message contient toutes les informations nécessaires à la configuration plus éventuellement un bail).
- Le client peut configurer son "réseau" à la réception de ce message

Protocole DHCP (1/2)

- Protocole utilisant les ports 67/UDP (serveur) et 68/UDP (client)
- Échange de messages entre client et serveurs pour l'attribution d'une adresse IP pour une durée donnée
- Protocole basé sur un mécanisme de *broadcast*
- Utilisation de différents types de messages : `DHCP Discover`, `DHCP Offer`, `DHCP Request`, `DHCP Acknowledge`, `DHCP Inform` et `DHCP Release`

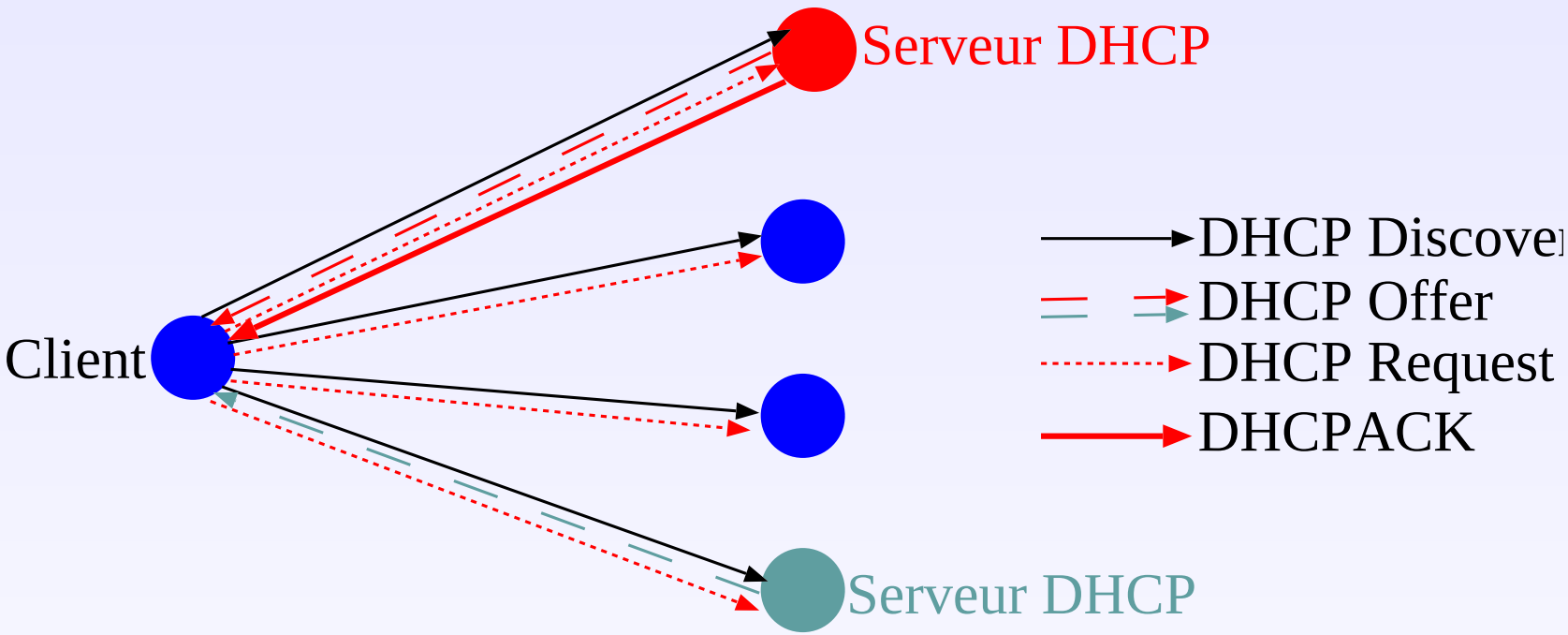
DHCP Inform. Le client demande des informations complémentaires au serveur DHCP (complément par rapport au DHCPACK ou demande spécifique pour une application donnée)

Protocole DHCP (1/2)

- Protocole utilisant les ports 67/UDP (serveur) et 68/UDP (client)
- Échange de messages entre client et serveurs pour l'attribution d'une adresse IP pour une durée donnée
- Protocole basé sur un mécanisme de *broadcast*
- Utilisation de différents types de messages : `DHCP Discover`, `DHCP Offer`, `DHCP Request`, `DHCP Acknowledge`, `DHCP Inform` et `DHCP Release`

DHCP Release. Le client envoie un message au serveur pour libérer son adresse IP. Ce message n'est pas nécessaire (l'adresse IP est récupérée par le serveur lorsque le bail expire et qu'il n'est pas renouvelé)

Protocole DHCP (2/2)



Ordre des messages :

1- DHCP Discover
2- DHCP Offer
3- DHCP Request
4- DHCPACK

DHCP & Linux/UNIX

- DHCPD est le démon qui implémente le serveur DHCP
- Configuration via le fichier `/etc/dhcpd.conf`
  Exemple simple :

```
subnet 192.168.0.0 netmask 255.255.255.0 {
default-lease-time 345600; # bail par défaut
max-lease-time 691200; # bail maximal
option domain-name “domainelocal.com”;
option domain-name-servers 192.168.0.2;
option routers 192.168.0.2;
option broadcast-address 192.168.0.255;
range 192.168.0.20 192.168.0.30;
}
```

- Possibilité d'ajouter dynamiquement les nouveaux hôtes enregistrés via DHCP au DNS (Dynamic DNS en utilisant BIND et DHCPD).
- Configuration du resolver via `/etc/resolv.conf`
- Plusieurs clients DHCP disponibles : `dhclient`, `pump`,...