Web & DNS

UE LU3IN033 Réseaux

2024-2025

Bruno Baynat

Bruno.Baynat@sorbonne-universite.fr

Plan du cours

- HTTP Hypertext Transfer Protocol
  - Les ingrédients du Web
    - les URL, le langage HTML et le protocole HTTP
  - Le protocole HTTP
    - format des requêtes et des réponses HTTP
  - Interactions HTTP / TCP
  - Les proxys Web
  - HTTPS
- DNS Dynamic Naming System
  - Nommage et adressage
  - Base de données répartie
  - Organisation hiérarchique des serveurs DNS
  - Le protocole DNS
    - format des requêtes et des réponses DNS
  - Les caches DNS
  - Interactions HTTP / DNS

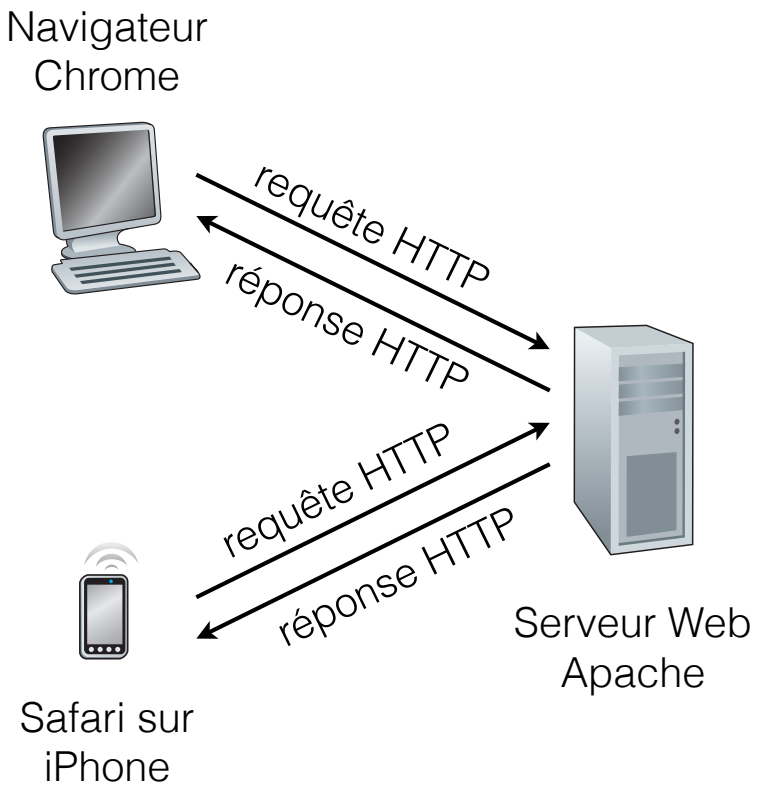
Web

Les 3 composants du Web

- URL : *Uniform Resource Locator*
  - « Adresse » (série de caractères) standardisée permettant d'identifier de manière unique une ressource sur internet
    - une page Web, un fichier, une image, une vidéo, …
  - Ex : http://www.exemple.com/dossier/page.html
    - méthode (protocole) utilisée pour accéder à la ressource : http://
    - nom de domaine : www.exemple.com
    - chemin d'accès : /dossier/page.html
- HTML : *HyperText Markup Language*
  - Langage le plus utilisé pour créer des pages Web
    - interprété par un navigateur pour afficher une page
  - HTML est un langage de description de pages
    - il permet de décrire la structure sémantique d'une page Web en utilisant des « balises » (entourées de crochets angulaires < >)
  - HTML n'est pas un langage de programmation
    - il ne contient pas de logique (comme des boucles ou des conditions)
- HTTP : *HyperText Transfer Protocol*
  - Protocole de communication utilisé pour le transfert d'informations sur le World Wide Web
  - Protocole de la couche application (5) utilisé entre un navigateur et un serveur Web

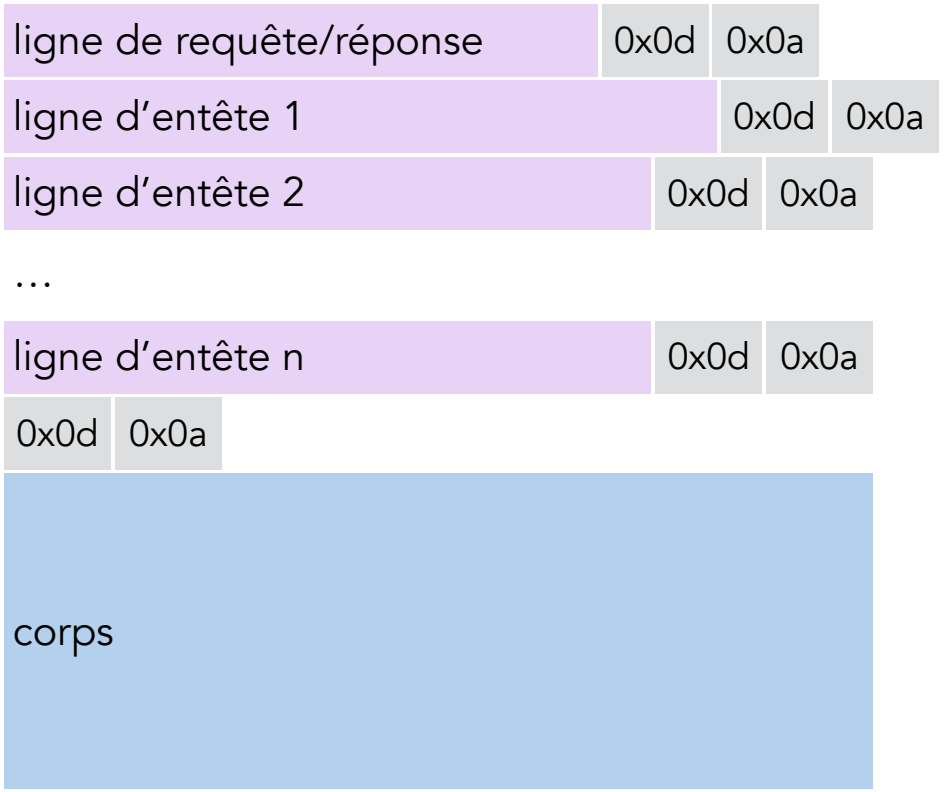
Protocole HTTP

- HTTP: *HyperText Transfer Protocol*
- Protocole de couche 5 utilisé par le Web
- 3 versions principales
  - HTTP/1.0 : RFC 1945 (1996)
  - HTTP/1.1 : RFC 2068 (1999)
  - HTTP/2.0 : RFC 7540 (2015)
  - HTTP/3.0 : RFC 9114 (2022)
- Toutes les versions de HTTP (jusqu'à HTTP/2.0) utilisent le protocole TCP pour fiabiliser les échanges
- Protocole textuel (pour HTTP/1.0 et 1.1)
  - Les champs d'entête et leurs valeurs sont codés en ASCII
- Modèle Client/Serveur
  - Client : le navigateur envoie des requêtes pour demander des objets (pages Web et éléments inclus)
  - Serveur : le serveur Web répond au client en lui retournant les objets demandés
- Mode non connecté (pour HTTP/1.0 et 1.1)

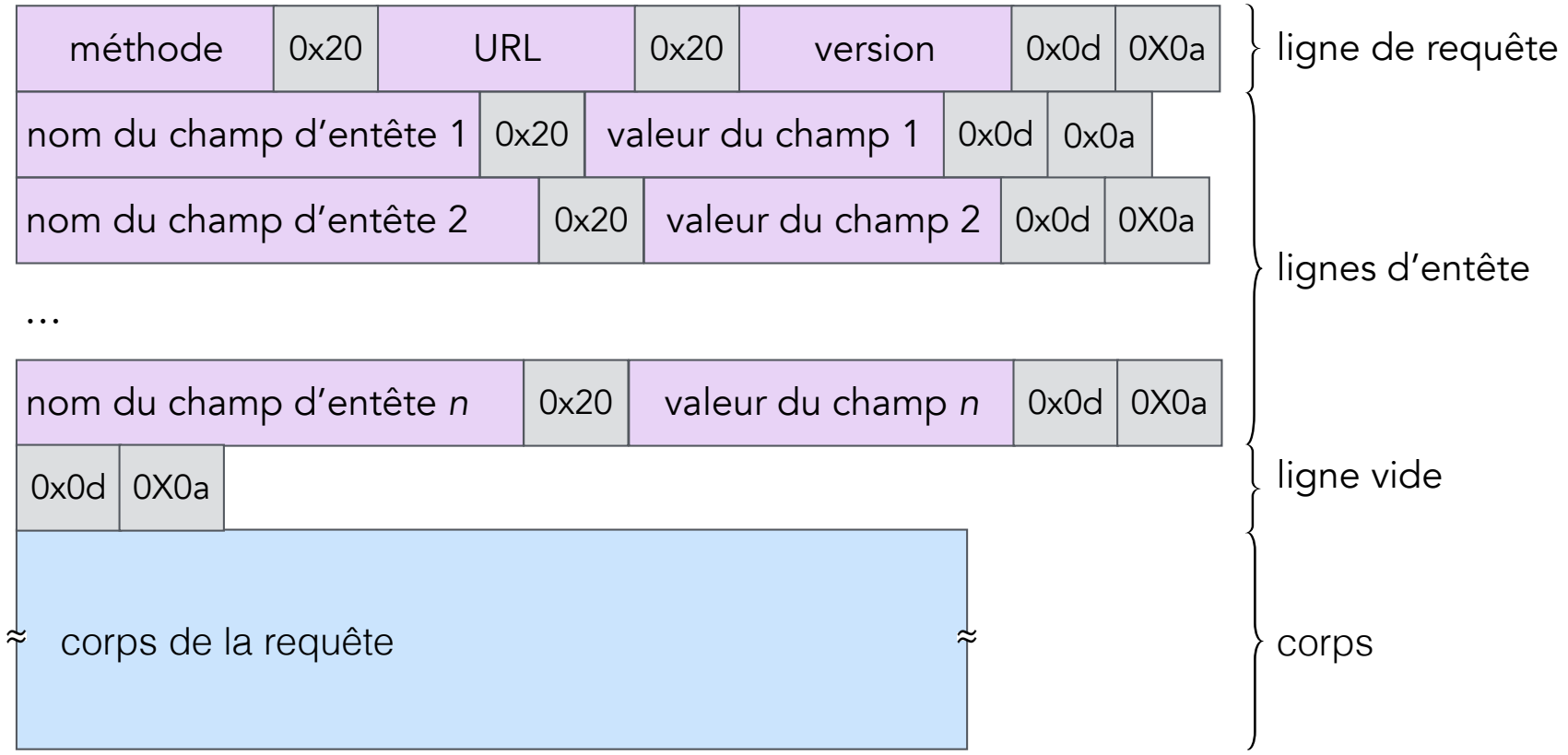


Format des messages HTTP

- Message HTTP
  - Client : requête
  - Serveur : réponse
- Formé
  - d'une entête
    - une première ligne de requête/réponse
    - des lignes supplémentaires d'entête
  - d'un corps
    - éventuellement vide
- Les lignes de l'entête sont séparées par un « retour chariot » et un « saut de ligne »
  - <CR><LF> (\r\n) soit en ASCII 0x0d0a
- L'entête et le corps de la requête (même vide) sont séparés par un « retour chariot » et un « saut de ligne » (en plus de ceux de la dernière ligne d'entête)
  - <CR><LF> (\r\n) soit en ASCII 0x0d0a



Format des requêtes HTTP



- La ligne de requête (la première) comporte 3 champs : méthode, URL et version
- Les lignes d'entête (les suivantes) commencent par un nom suivi de « : » et d'un espace
- Les champs d'une ligne sont séparés par un espace
  - <SP> (\s) soit en ascii 0x20

Types de requête : les méthodes

- GET
  - permet de réclamer au serveur l'objet identifié par la valeur du champ URL
- HEAD
  - permet de réclamer au serveur une réponse sans corps (seulement l'entête)
- POST
  - permet de transmettre au serveur des données passées dans le corps de la requête
- PUT
  - permet de mettre à jour une ressource existante ou de créer une nouvelle ressource si elle n'existe pas déjà à l'emplacement spécifié par le champ URL
- DELETE
  - permet de supprimer une ressource située à l'emplacement spécifié par le champ URL
- ...

Champs d'entête

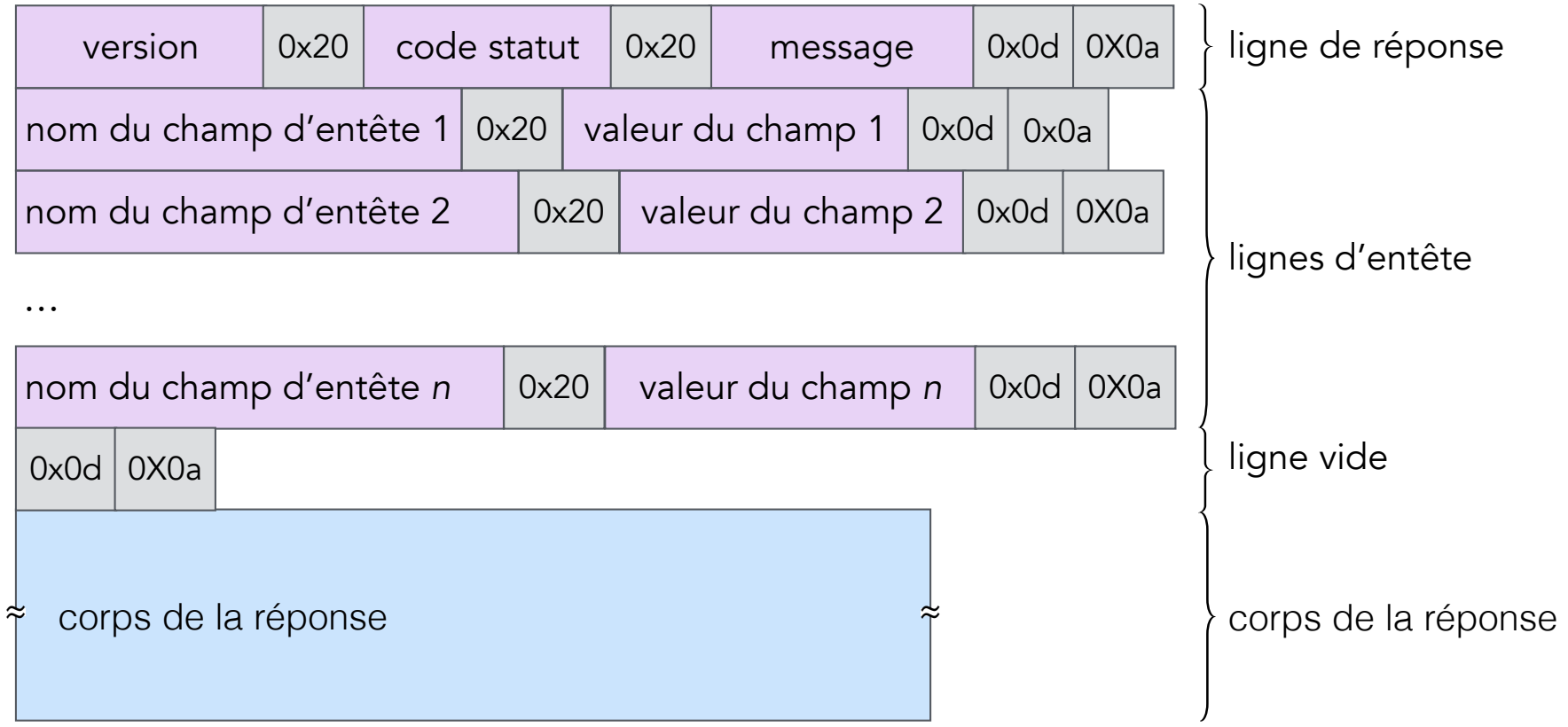
- Host:
  - nom de domaine du serveur
  - obligatoire dans HTTP/1.1 (et les versions ultérieures)
- Connection:
  - indique si le client souhaite que la connexion soit maintenue ou fermée après la transaction
  - Ex : « Connection: keep-alive » : connexion persistante
  - Ex : « Connection: close » : connexion non persistante
- Date:
  - date et heure d'émission du message
- User-Agent:
  - fournit des informations sur le client (généralement le navigateur ou l'application)
- Accept:
  - indique les types de médias que le client peut traiter
  - Ex : « Accept: text/html »
- Accept-Language:
  - spécifie les langues préférées par le client pour les réponses
- …

Exemple : requête

Entête

```
GET /team-members/index.html HTTP/1.1<CR><LF>
Host: www-npa.lip6.fr<CR><LF>
User-agent: Mozilla/5.0<CR><LF>
Connection: close<CR><LF>
Accept: text/html, image/jpeg<CR><LF>
Accept-language: fr<CR><LF>
<CR><LF>
```

Format des réponses HTTP



- La ligne de réponse (la première) comporte 3 champs : version, code statut et message
- Les lignes d'entête (les suivantes) commencent par un nom suivi de « : » et d'un espace
- Les champs d'une ligne sont séparés par un espace
  - <SP> (\s) soit en ascii 0x20

Code de statut

- 1XX : informations
  - 100 Continue
    - le serveur a reçu la requête et le client peut continuer le processus
- 2XX : succès
  - 200 OK
    - requête en succès : l'objet demandé se trouve dans le corps de la réponse
- 3XX : redirections
  - 301 Moved Permanently
    - l'objet demandé a changé d'emplacement
    - la nouvelle URL de l'objet se trouve dans le corps de la réponse
  - 304 Not Modified
    - en réponse à une requête conditionnelle (champ If-Modified-Since) pour un objet non modifié
- 4XX : erreurs du client
  - 400 Bad Request
    - la requête est mal formulée (erreur de syntaxe)
  - 404 Not Found
    - l'objet demandé n'a pas été trouvé à l'URL demandé
- 5XX : erreurs du serveur
  - 505 HTTP Version Not Supported
    - la version du protocole HTTP n'est pas supportée par le serveur

Champs d'entête

- Server:
  - fournit des informations sur le serveur Web qui a généré la réponse
- Connection:
  - indique si la connexion doit être maintenue ou fermée après la transaction
- Date:
  - indique la date et l'heure d'émission du message
- Last-Modified:
  - indique la date et l'heure à laquelle la ressource a été modifiée pour la dernière fois
- Content-Length:
  - indique la taille (en octets) du corps de la réponse
- Content-Type:
  - spécifie le type de média du corps de la réponse
  - Ex : « Content-Type: text/html »
- …

Exemple : réponse

Entête

```
HTTP/1.1 200 OK<CR><LF>
Connection: close<CR><LF>
Date: Thu, 6 Aug 2023 12:00:15 GMT<CR><LF>
Server: Apache/1.3.0 (Unix)<CR><LF>
Last-Modified: Mon, 6 Feb 2023 11:12:23 GMT<CR><LF>
Content-Length: 6821<CR><LF>
Content-Type: text/html<CR><LF>
<CR><LF>
```

Corps

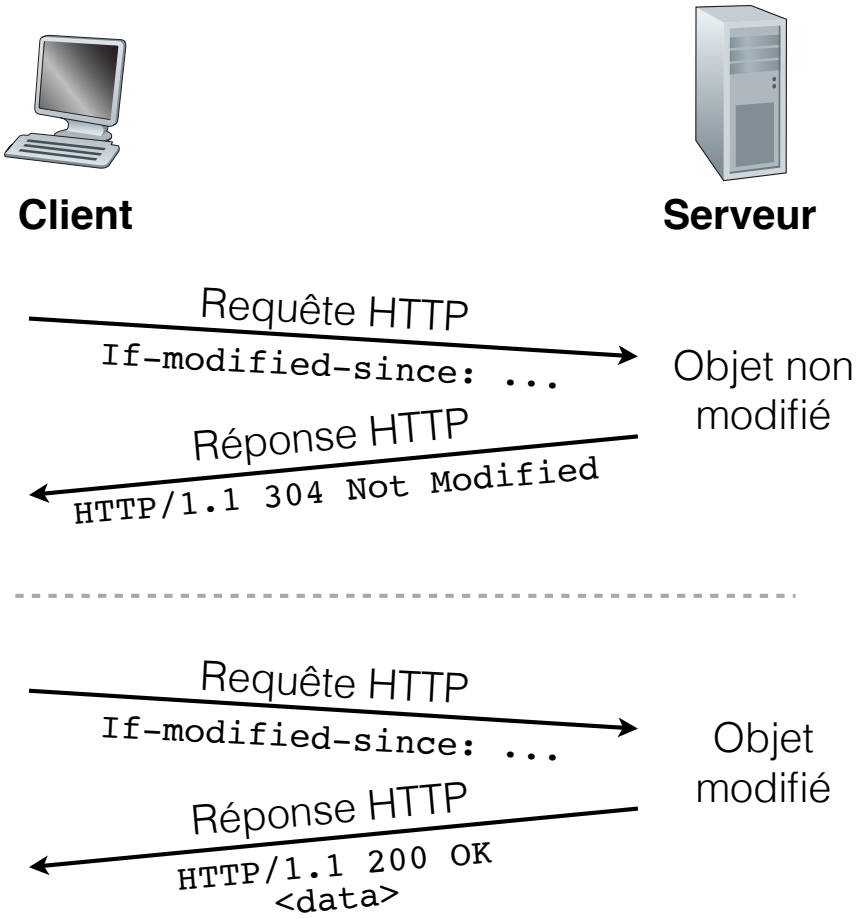
```
data data data data data...
```

Requête conditionnelle

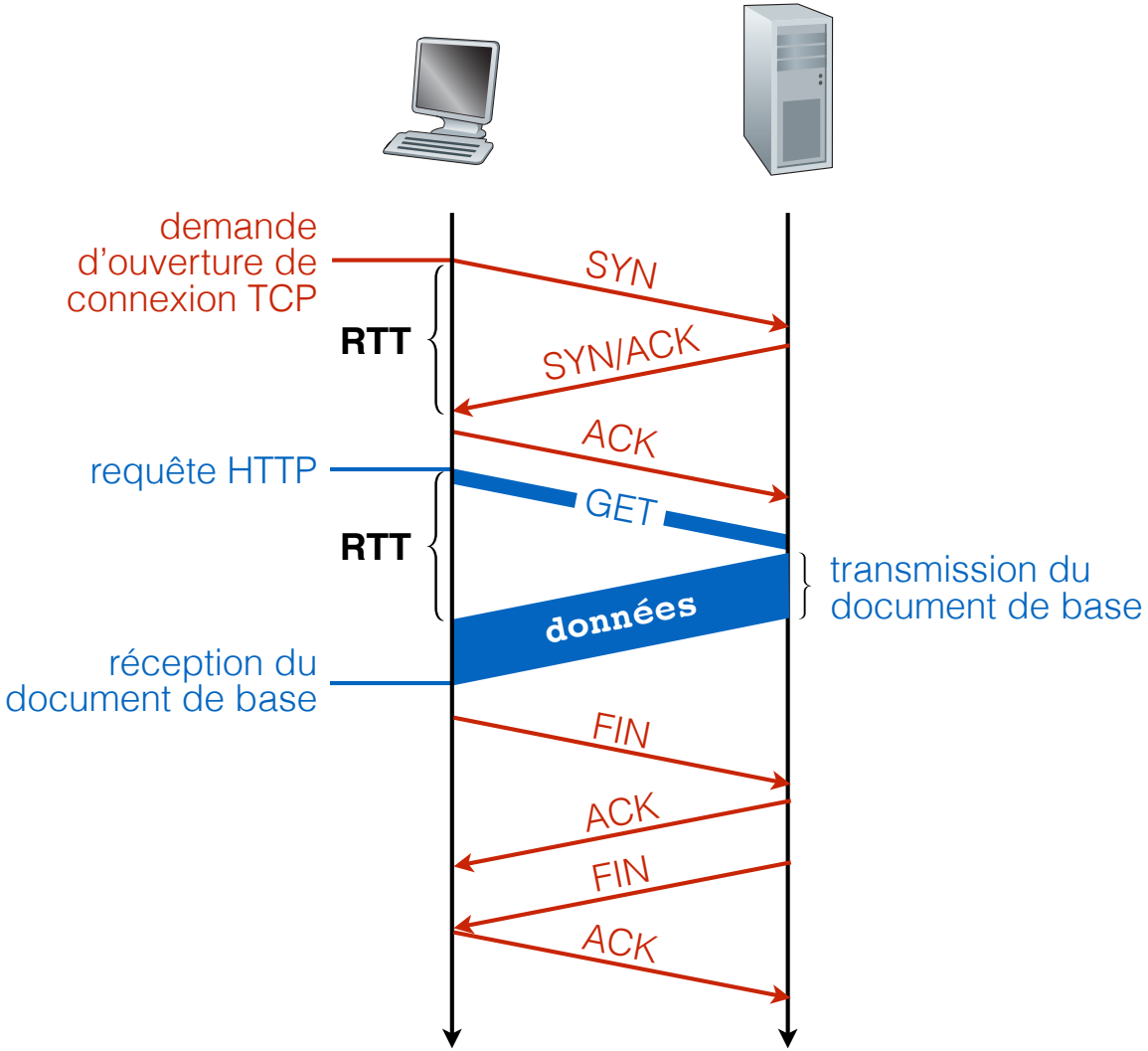
- Un objet est retourné uniquement si l'objet a changé sur le serveur

```
GET /team-members/index.html HTTP/1.1
Host: www-npa.lip6.fr
User-Agent: Mozilla/4.03
If-Modified-Since: Mon, 6 Feb 2023 11:12:23 GMT
<CRLF>
```

- Le serveur économise les ressources nécessaires à l'envoi d'un objet
  - en inspectant la valeur « last modified » de l'objet
  - en comparant cette valeur à celle du champ « if-modified-since »
  - en retournant « 304 Not Modified » si l'objet n'a pas changé…
  - … ou « 200 OK » avec la version la plus récente de l'objet



Interaction HTTP/TCP

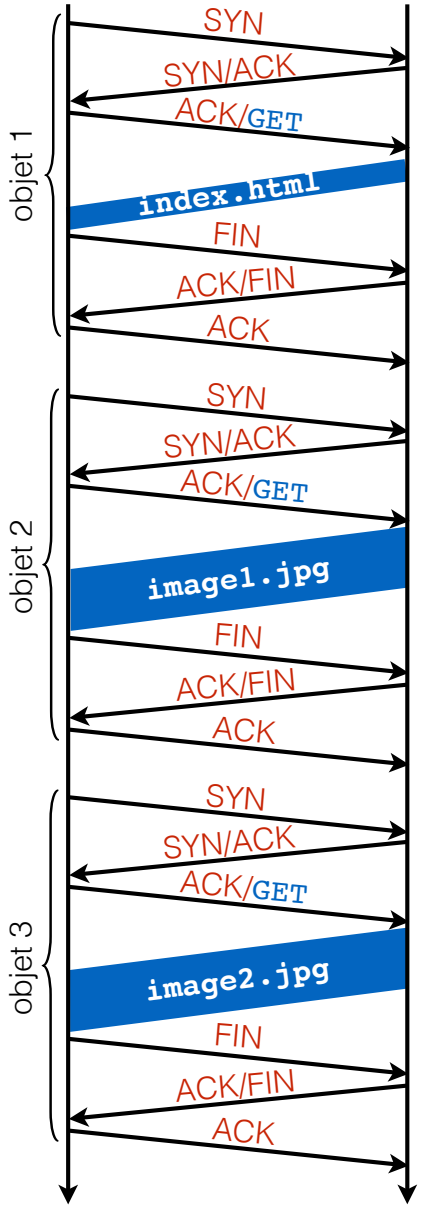


Interaction HTTP/TCP

- Une page web est généralement composée de plusieurs objets
  - le document de base (index) qui inclut le texte de la page (au format HTML)
  - des objets inclus : images, vidéos, feuille de style (CSS), animations (JavaScript), …
- HTTP nécessite une requête et une réponse par objet
  - une première requête pour le document de base
  - et autant de requêtes supplémentaires que d'objets référencés dans le document de base
- Mais combien de connexions TCP sont nécessaires ?
  - une par objet : mode non persistant
  - une pour l'ensemble des objets de la page : mode persistant
- En mode persistant, doit-on attendre la réception d'un objet avant de demander le suivant ?
  - oui : mode persistant sans pipelining
  - non : mode persistant avec pipelining

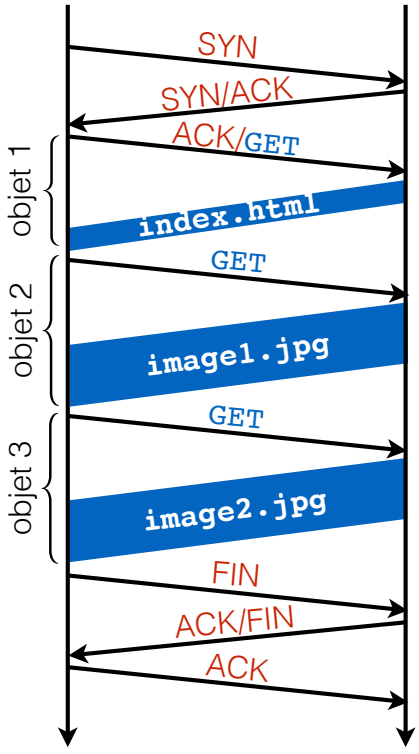
Les modes de fonctionnement de HTTP

Mode non persistant (HTTP/1.0)

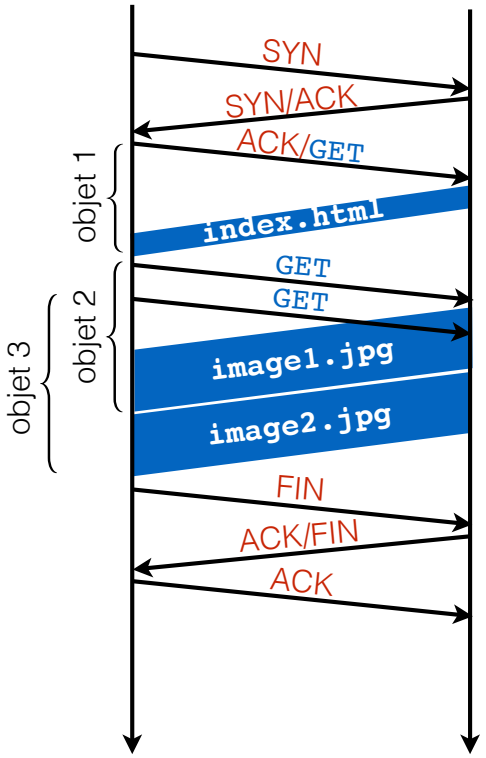


Mode persistant (HTTP/1.1)

sans pipelining

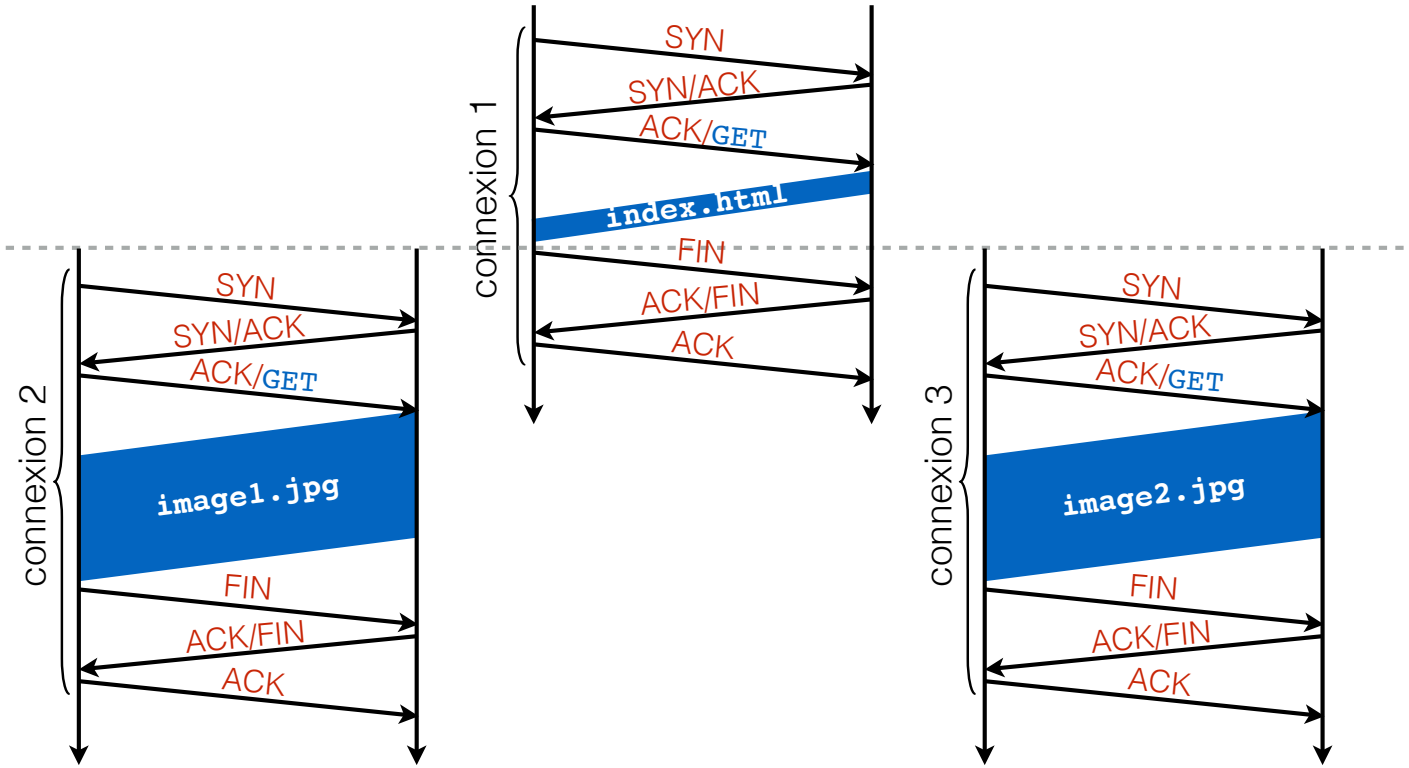


avec pipelining



Les modes de fonctionnement de HTTP

Mode non persistant et connexions TCP parallèles

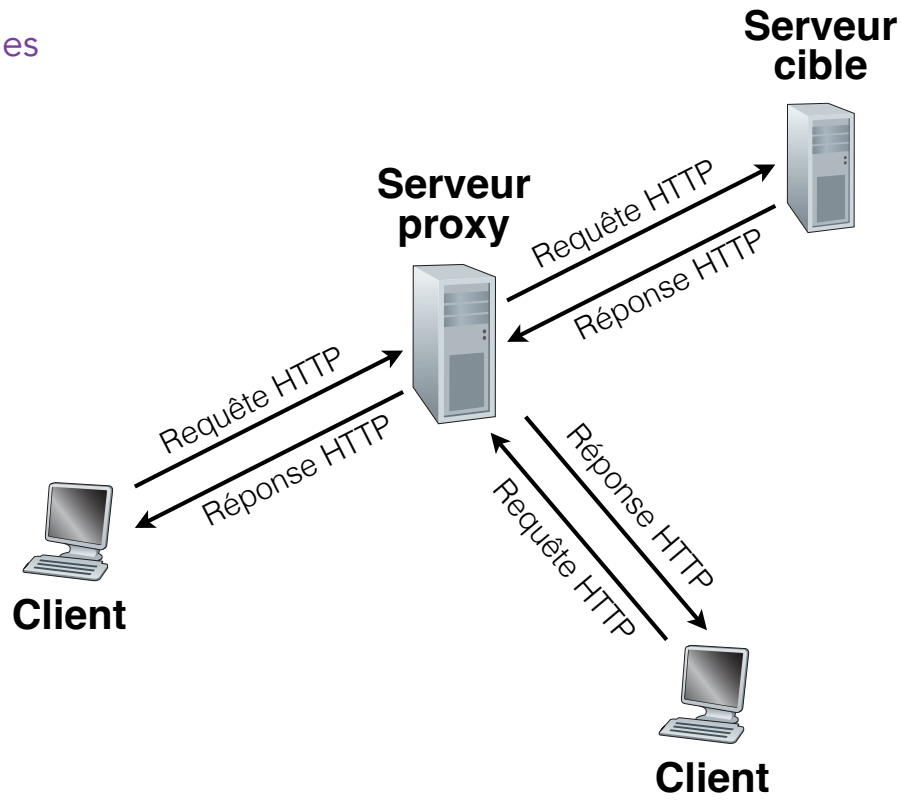


Mode persistant vs non persistant

- Mode non persistant : HTTP/1.0 (et HTTP/1.1)
  - Le navigateur ouvre généralement plusieurs connexions TCP concurrentes
  - Inconvénients
    - surcharge liée à la gestion des connexions TCP
    - iniquité vis-à-vis des trafics utilisant une seule connexion TCP empruntant le même chemin
- Mode persistant : HTTP/1.1
  - Le navigateur ouvre une seule connexion TCP dite « longévive »
  - Avantages
    - gestion simplifiée
    - permet une meilleure estimation du RTT
    - atteint un débit d'émission acceptable
  - Inconvénient
    - problème du HOL (*Head of the line*) : le premier objet demandé, si volumineux, bloque l'affichage du reste de la page
    - HTTP/2.0 résout ce problème en introduisant le « multiplexage de flux » : plusieurs requêtes et réponses peuvent être envoyées et reçues simultanément sur la même connexion
  - En pratique, le pipelining n'est pas utilisé
    - les serveurs Web ne répondent pas toujours selon l'ordre de réception des requêtes

Proxy Web

- Intermédiaire entre les utilisateurs et les ressources auxquelles ils souhaitent accéder sur Internet
  - agit à la fois comme serveur (pour le client) et comme client (pour le serveur cible)
    - reçoit les demandes des clients
    - les transmet au serveur cible
    - renvoie les réponses au client
- Fonctionnalités
  - Mise en Cache
    - des réponses à des demandes
    - permet d'accélérer les futures demandes identiques
  - Filtrage du contenu
    - en fonction de règles prédéfinies
    - permet de bloquer ou d'autoriser l'accès à certaines ressources
  - Anonymat
    - pour masquer l'adresse IP réelle de l'utilisateur
  - Contrôle d'Accès
    - permet de restreindre l'accès à certaines ressources en fonction de politiques de contrôle d'accès
  - Sécurité (pare-feu)
    - permet d'inspecter le trafic entrant et sortant et d'empêcher l'accès à des contenus malveillants ou à des sites dangereux



HTTPS

- Les données transférées en HTTP ne sont pas chiffrées et peuvent être interceptées par des tiers
  - vulnérabilité aux attaques
- HTTPS (Hypertext Transfer Protocol Secure) utilise une couche de chiffrement pour sécuriser les données échangées entre le client et le serveur
  - les informations transitant en HTTPS sont cryptées
  - chiffrement SSL/TLS (Secure Sockets Layer/Transport Layer Security)
    - certificat SSL/TLS émis par une autorité de certification (CA) pour établir une connexion sécurisée
    - le certificat garantit l'authenticité du serveur auprès du client
    - les messages HTTPS, sont chiffrées et encapsulées dans des enregistrements SSL/TSL
  - le protocole de transport TCP est toujours utilisé
    - le certificat garantit l'authenticité du serveur auprès du client
- HTTPS utilise le port 443 par défaut (80 pour HTTP)
- Les URL commencent par « https:// » (au lieu de « http:// » pour HTTP)

DNS
Domain Name System

Noms vs Adresses IP

- Dans Internet, les machines sont identifiées par
  - Une adresse IP (Ex : 74.125.133.94)
    - adresses numériques utilisées par les routeurs pour acheminer les paquets jusqu'à leur destination
    - longueur fixe (32 bits pour IPv4 ou 128 bits pour IPv6)
    - hiérarchiques en rapport avec la localisation des hôtes
  - Un nom (Ex : www.google.fr)
    - noms mnémoniques utilisée par les humains
    - longueur variable, caractères alpha-numériques
    - donnent peu (voir aucune) d'information sur la localisation

Nommage et adressage

- Les noms sont plus simples à retenir
  - Ex : www.google.fr vs 74.125.133.94
- L'adresse peut changer tout en conservant le nom
  - Ex : déplacer le serveur www.google.fr à l'adresse 74.125.133.100
  - pérennise l'utilisation du nom
- Un nom peut cacher plusieurs adresses IP
  - Ex : le serveur Web www.google.fr est répliqué sur plusieurs machines
  - permet de retourner celle qui est le plus proche de l'utilisateur et donc de réduire la latence
- Plusieurs noms pour une même adresse IP
  - Ex : alias tels que www.sorbonne-universite.fr et www.jussieu.fr
  - utiliser des alias (noms alternatifs) permet plus de souplesse

DNS qu'est ce que c'est ?

- Le DNS (Domain Name System) définit
  - le format des noms utilisés pour identifier les machines dans Internet
  - la méthode d'attribution des noms garantissant leur unicité
  - une base de données distribuée contenant les correspondances entre noms et adresses IP
  - une collection de serveurs qui gère chacun, une partie de ces correspondances
  - un protocole de résolution des noms en adresses IP

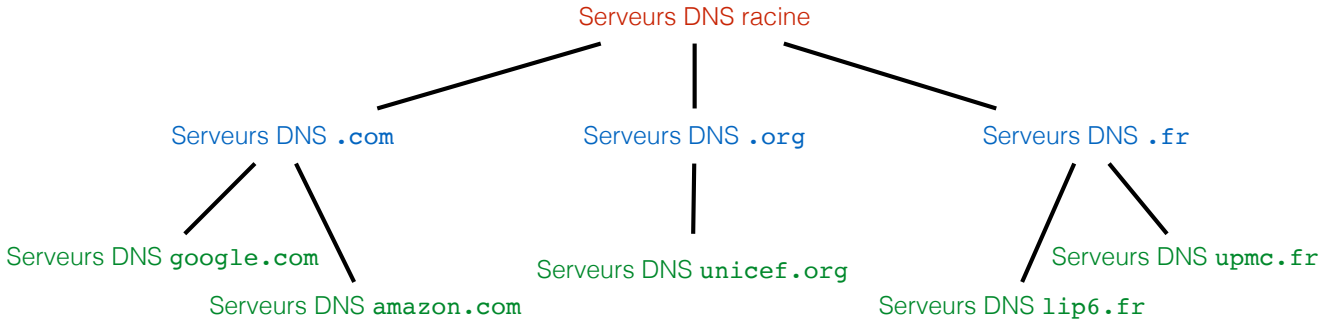
Format des noms

- Structure hiérarchique basée sur des étiquettes (labels)
  - Ex : ppti-14-308-02.ufr-info-p6.jussieu.fr
  - chaque étiquette
    - doit commencer par une lettre ou un chiffre
    - ne peut pas être vide
    - ne peut pas se terminer par un tiret
  - les étiquettes sont séparées par des points
  - caractères autorisés dans les étiquettes
    - les lettres de l'alphabet (a à z, A à Z)
    - les chiffres (0 à 9)
    - le tiret (-)
  - longueur totale maximum : 255 caractères
  - insensibles à la casse

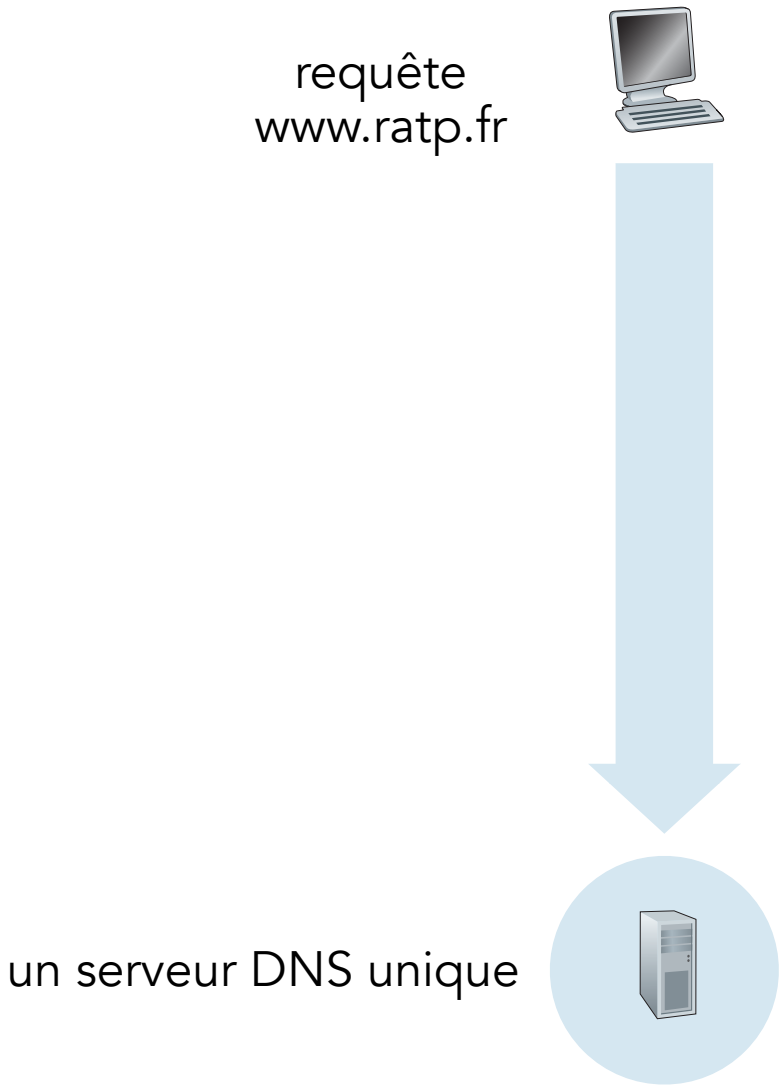
Serveurs DNS

Base de données mondiale distribuée

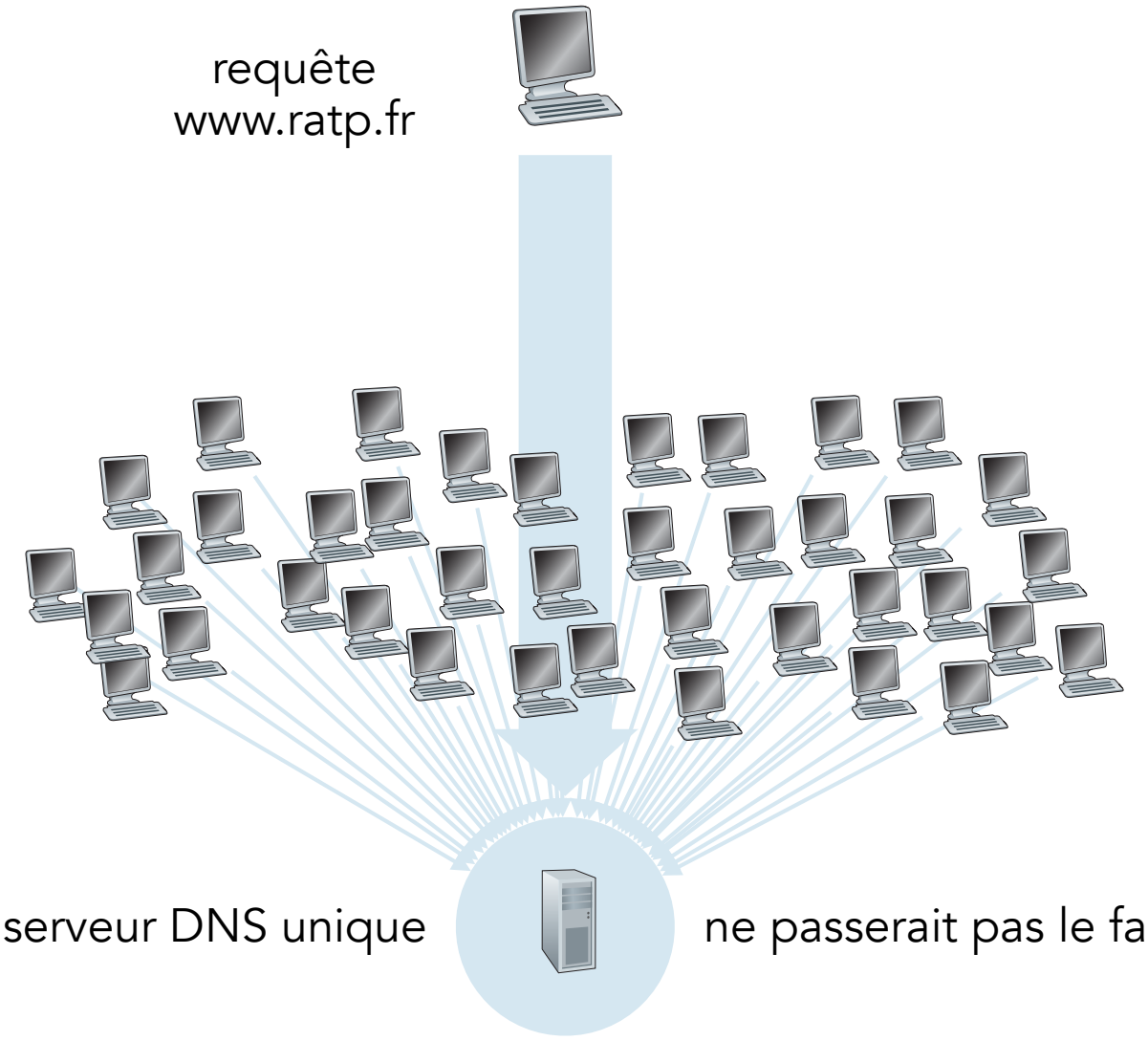
- Espace de nommage hiérarchique divisé en zones
  - Les noms d'une même zone sont gérés par un serveur DNS ayant autorité sur le domaine (généralement redondé)
- Hiérarchie de serveurs DNS
  - Serveurs DNS racine (*root*)
  - Serveurs DNS TLD (*Top-Level Domain*) ou de premier niveau
  - Serveurs DNS d'autorité (*Authoritative*)



- Niveau utilisateur
  - Serveurs DNS locaux (*local DNS*)

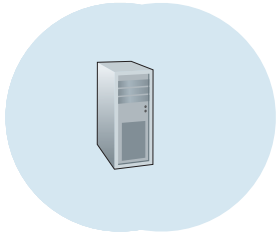

requête
www.ratp.fr
un serveur DNS unique

requête
www.ratp.fr



un serveur DNS unique ne passerait pas le facteur d'échelle

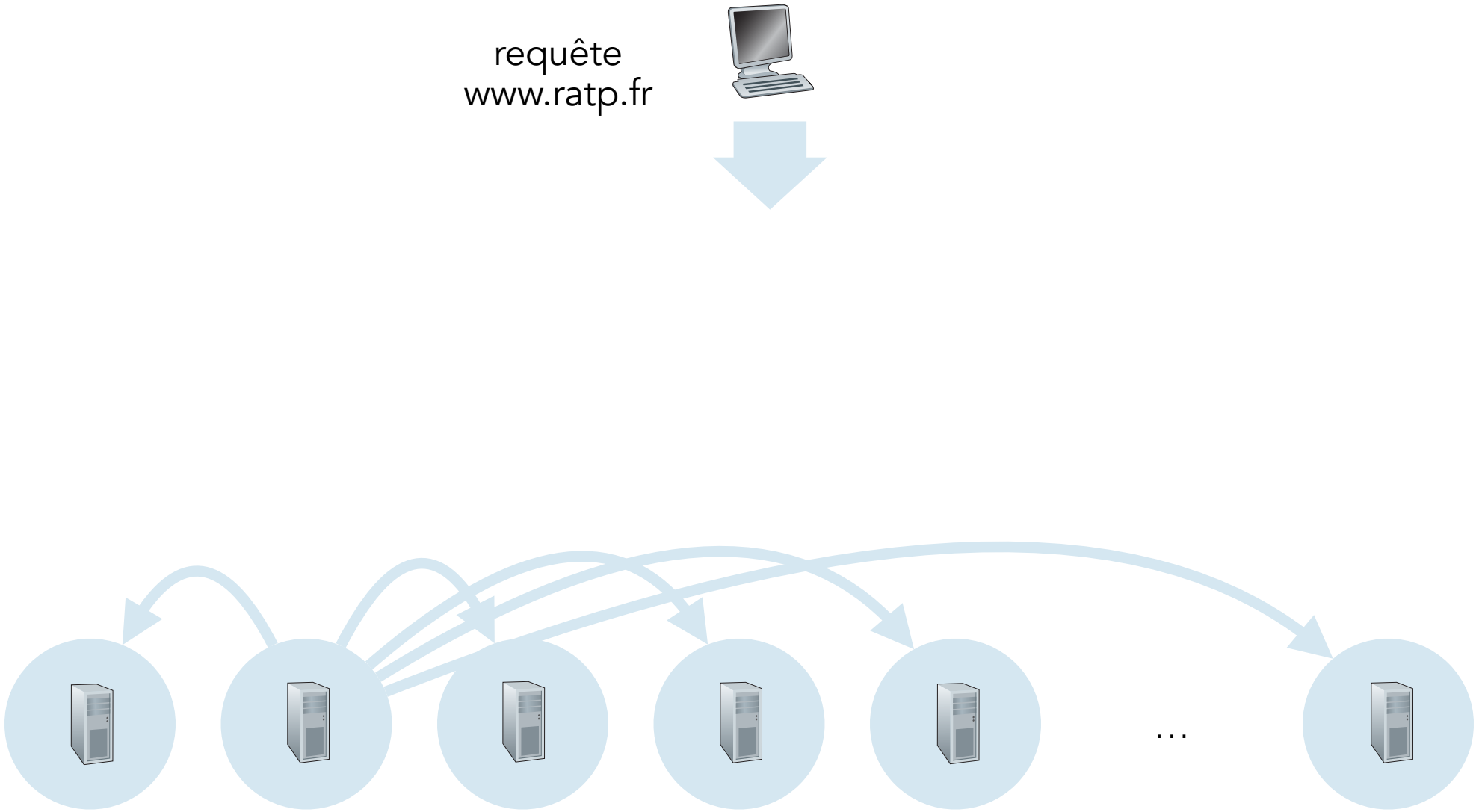
requête
www.ratp.fr



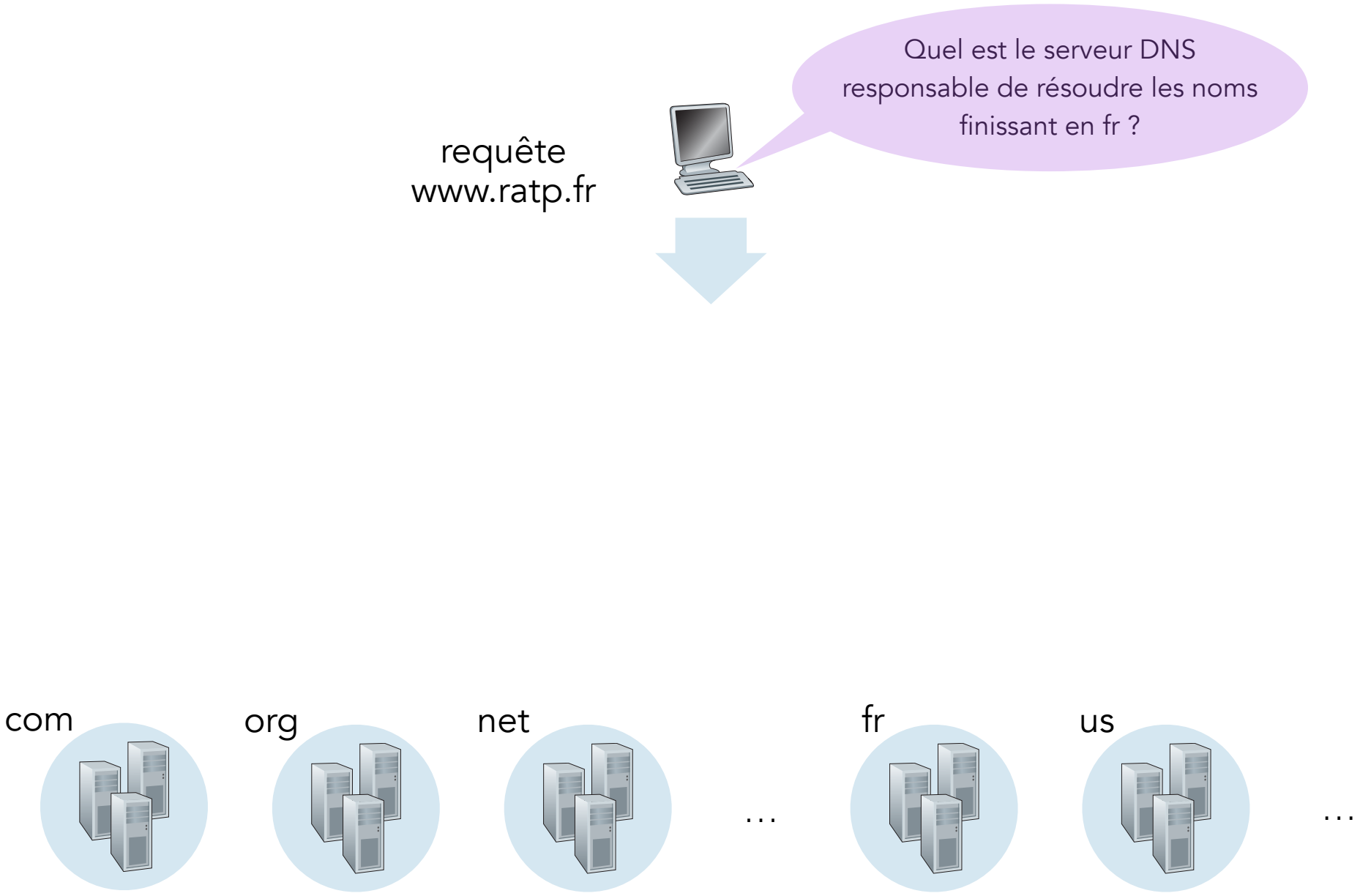
…

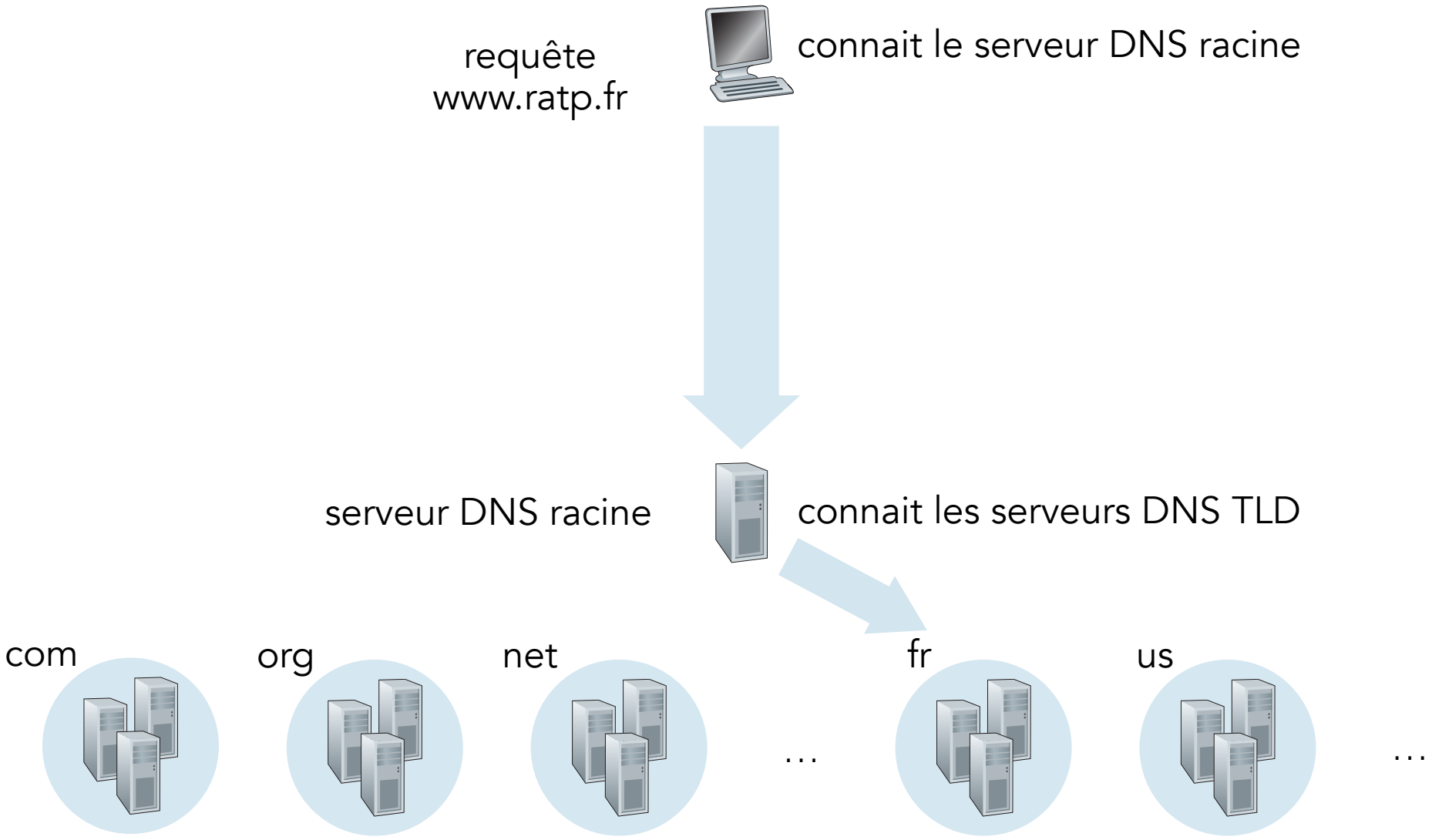
répliquer le même serveur DNS

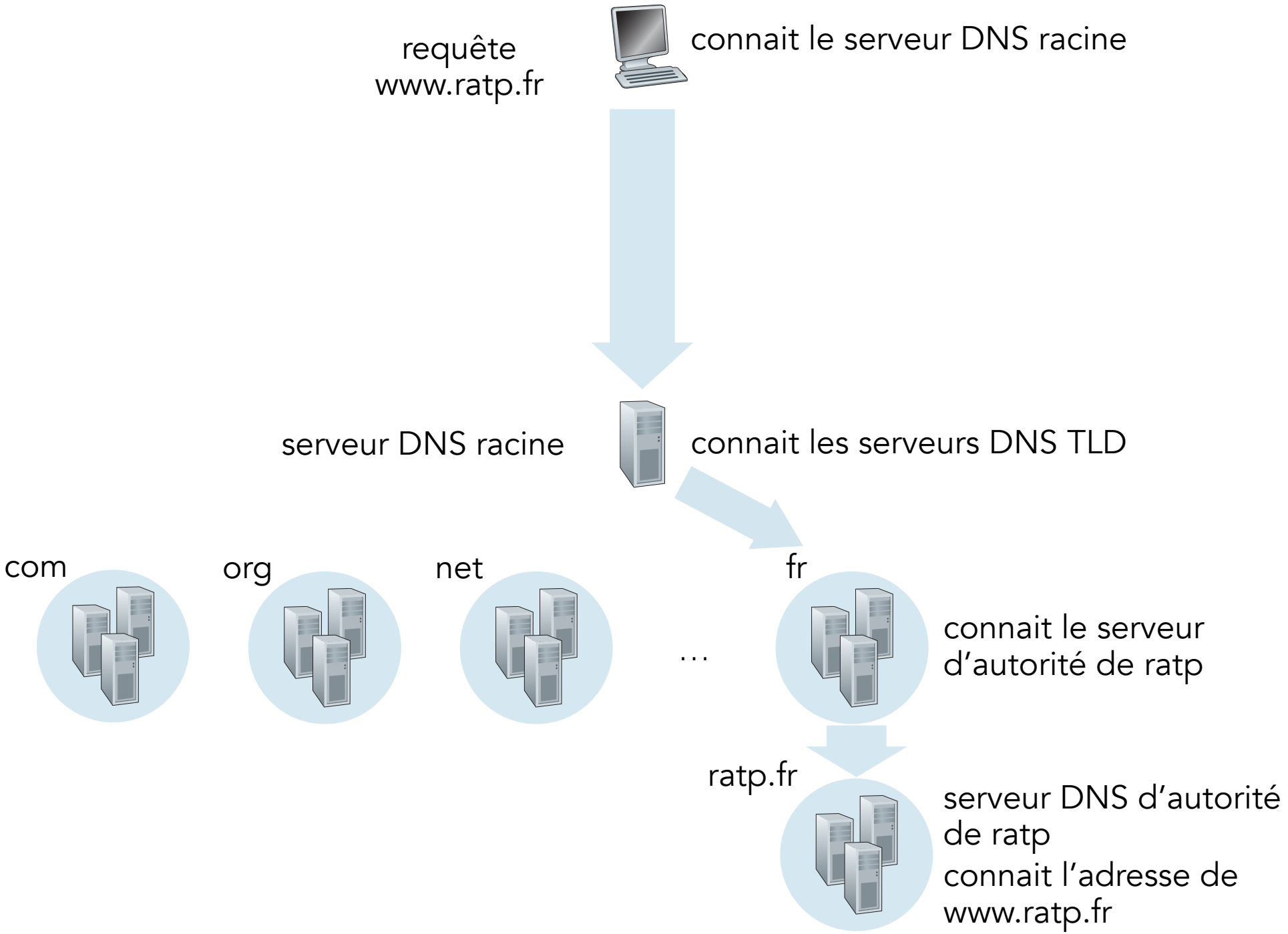
requête
www.ratp.fr

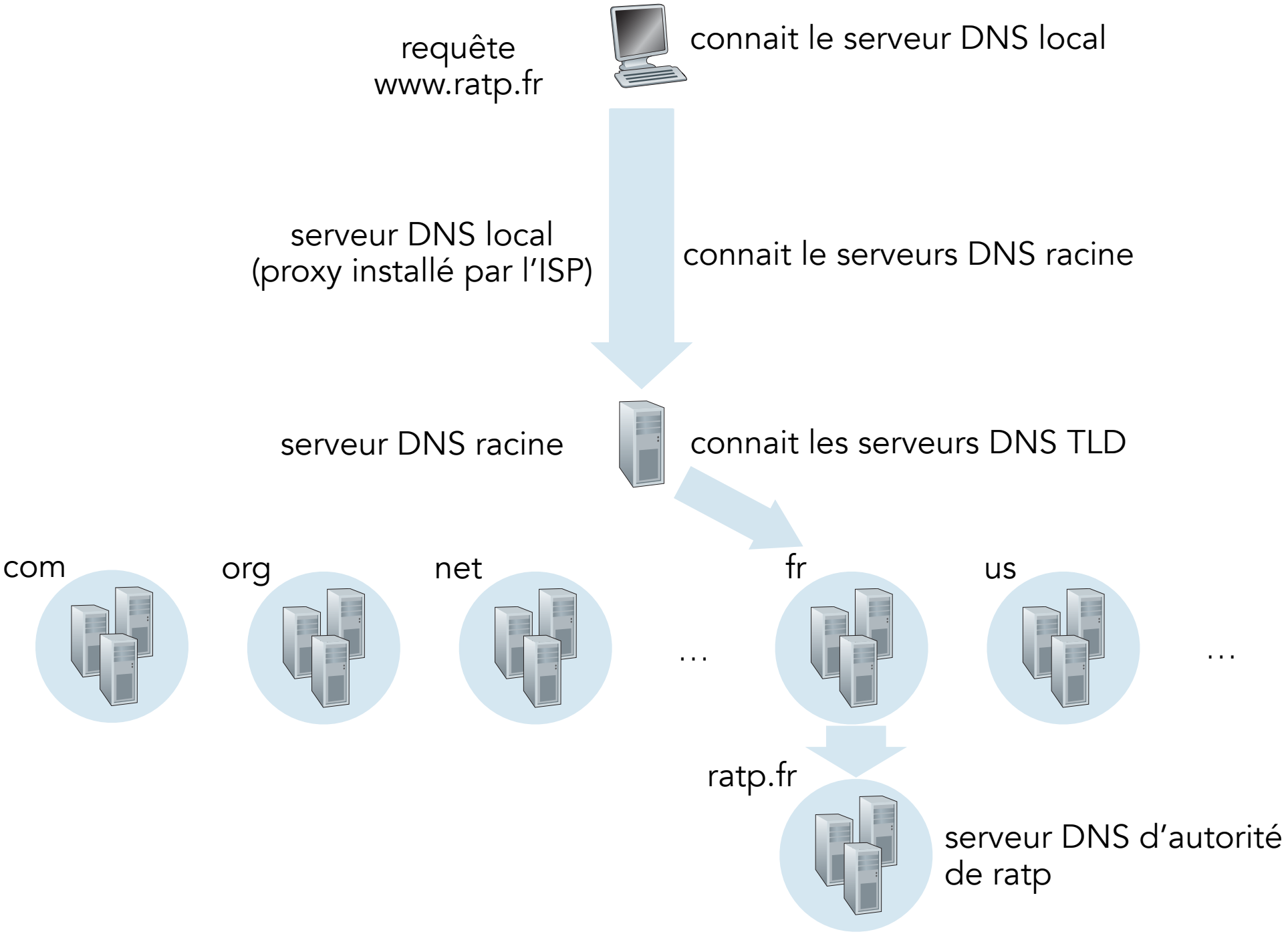


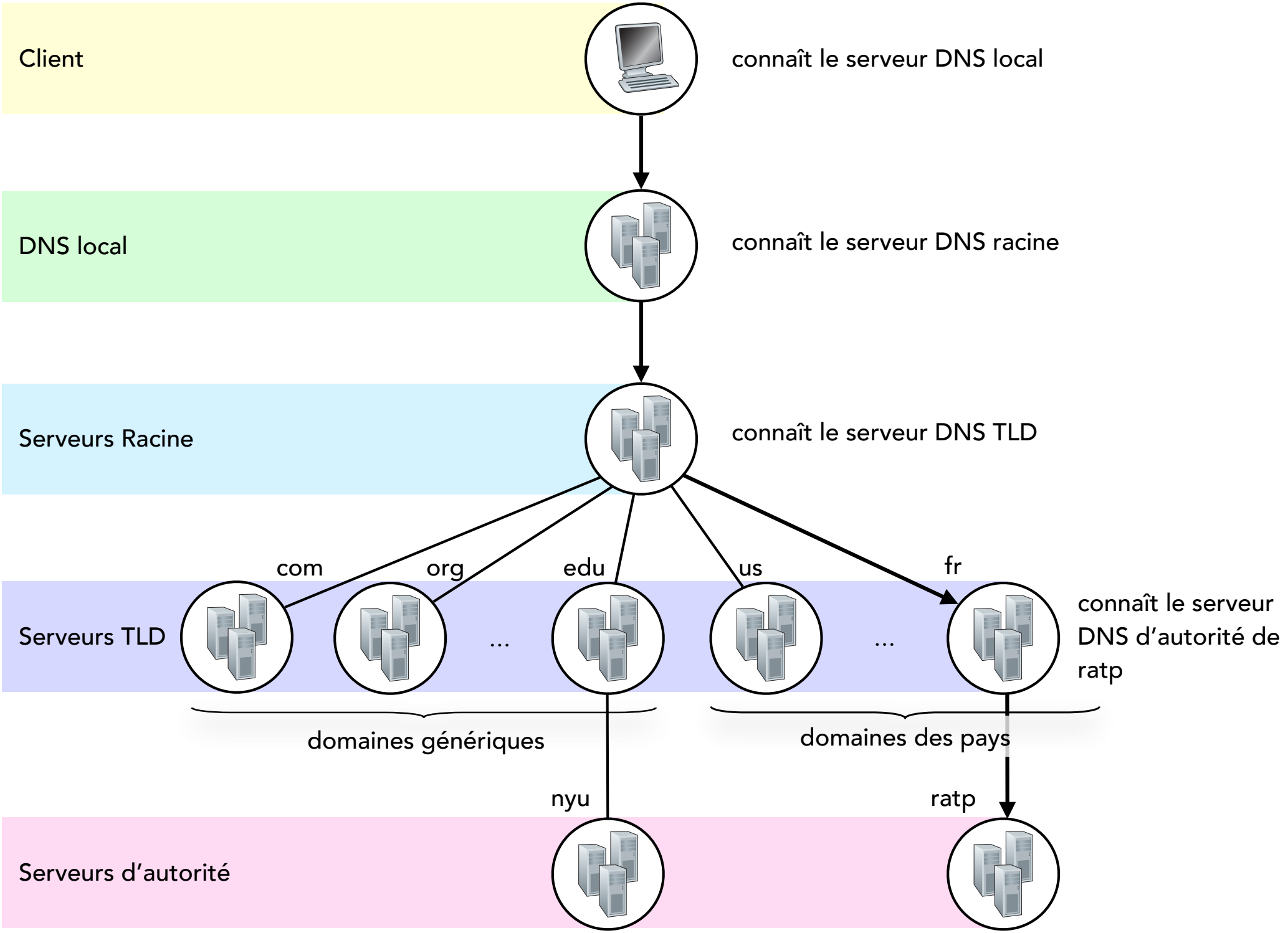
répliquer le même serveur DNS
nécessite de synchroniser les replicas


Quel est le serveur DNS responsable de résoudre les noms finissant en fr ?
requête
www.ratp.fr
com
org
net
…
fr
us
…


requête
www.ratp.fr
connait le serveur DNS racine
serveur DNS racine
connait les serveurs DNS TLD
com
org
net
…
fr
us
…

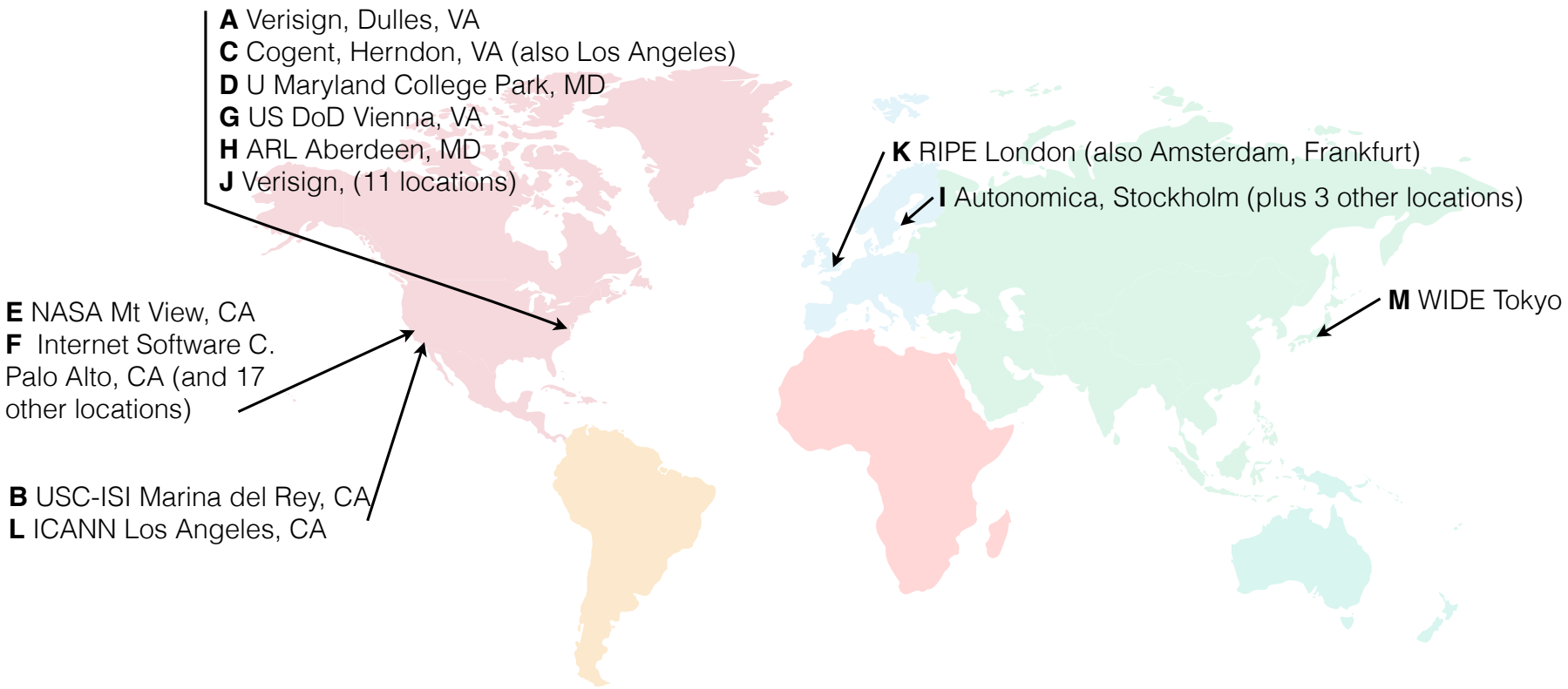

requête
www.ratp.fr
connait le serveur DNS racine
serveur DNS racine
connait les serveurs DNS TLD
com
org
net
...
fr
connait le serveur
d'autorité de ratp
ratp.fr
serveur DNS d'autorité
de ratp
connait l'adresse de
www.ratp.fr


requête
www.ratp.fr
connait le serveur DNS local
serveur DNS local
(proxy installé par l'ISP)
connait le serveurs DNS racine
serveur DNS racine
connait les serveurs DNS TLD
com
org
net
…
fr
us
…
ratp.fr
serveur DNS d'autorité
de ratp


Client
connaît le serveur DNS local
DNS local
connaît le serveur DNS racine
Serveurs Racine
connaît le serveur DNS TLD
com
org
edu
us
fr
Serveurs TLD
...
...
connaît le serveur DNS d'autorité de ratp
domaines génériques
domaines des pays
nyu
ratp
Serveurs d'autorité

Serveurs DNS racine

- Serveurs DNS racine
  - 13 serveurs dans le monde (http://www.root-servers.org/)
  - connus des serveurs DNS locaux



Serveurs TLD

- Serveurs DNS TLD (*Top-Level Domain*)
  - serveurs de premier niveau
  - responsables des noms de domaine
    - génériques : .com, .org, .edu, …
    - de pays : .fr, .us, .jp, …
  - gérés par des sociétés privées ou des associations à but non lucratif
    - Ex : Network Solutions est en charge du nom de domaine .com
      l'AFNIC est en charge du nom de domaine .fr
      Educause est en charge du nom de domaine .edu
  - connus des serveurs DNS racine
  - connaissent les serveurs DNS d'autorité qui se terminent par le label dont ils sont responsables

Serveurs d'autorité

- Serveurs DNS d'autorité (*Authoritative*)
  - serveurs de deuxième (troisième, …) niveau
  - responsables des noms de la zone sur laquelle ils ont autorité
  - gérés par une institution (ou par un fournisseur de services pour le compte de l'institution)
  - connus des serveurs TLD
  - connaissent les correspondances pour les machines appartenant à l'institution qu'ils gèrent (Ex : serveurs mail, Web, …)
- Ce sont les seuls serveurs DNS à connaitre les adresses IP des machines

Serveurs DNS locaux

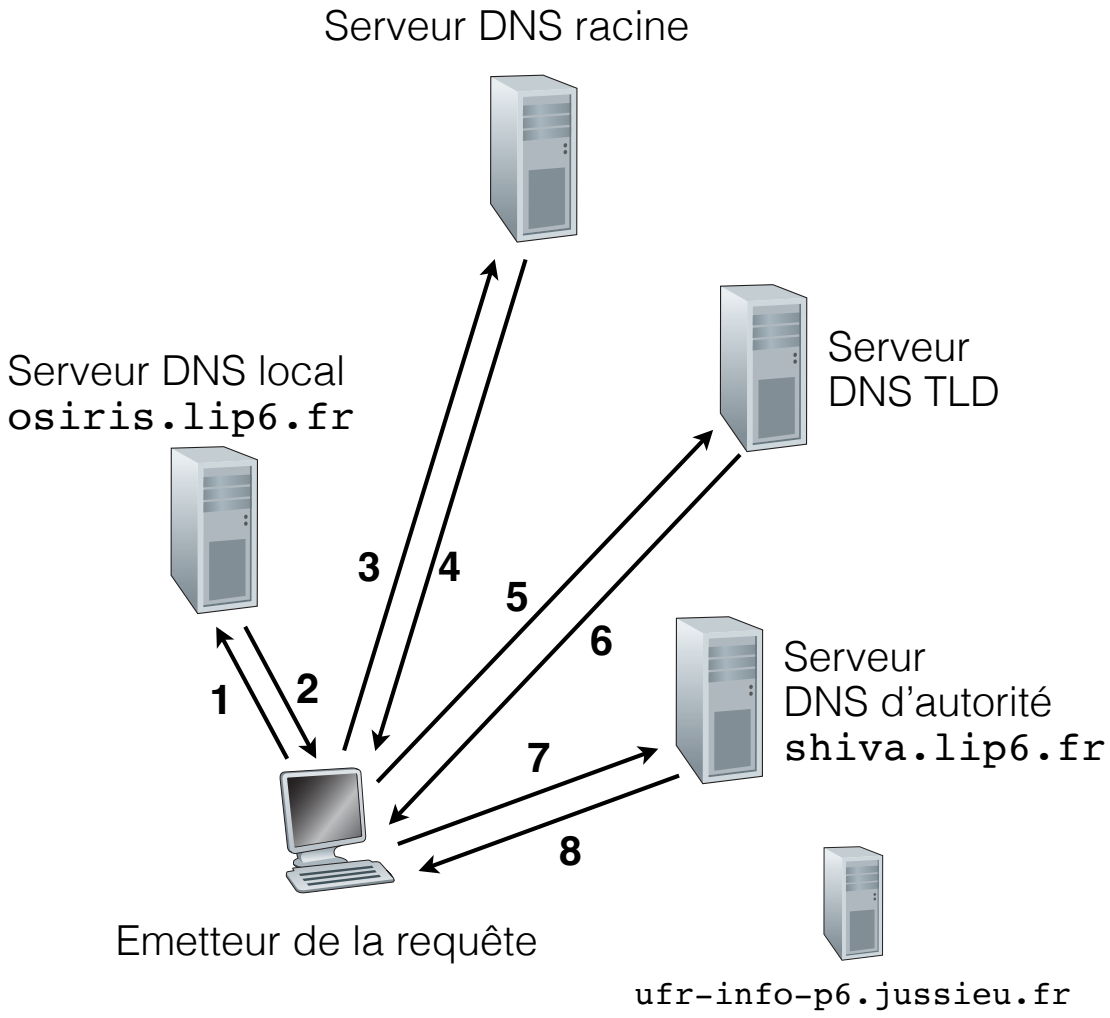
N'appartient pas vraiment à la hiérarchie DNS

- Serveurs DNS locaux
  - mis à disposition par l'institution gérant le réseau (FAI, entreprise, université)
  - aussi appelés « serveurs de nom par défaut » (*default name servers*)
  - connus des machines hôtes par configuration manuelle ou DHCP
  - les seuls à connaitre les 13 serveurs DNS racine
- Fonctionnement
  - l'hôte émet toutes ses requêtes DNS au serveur DNS local
  - le DNS local gère un cache contenant les correspondances récemment obtenues
  - il joue le rôle de proxy en acheminant les requêtes pour le compte des clients du réseau

Résolution de nom

Requête itérative

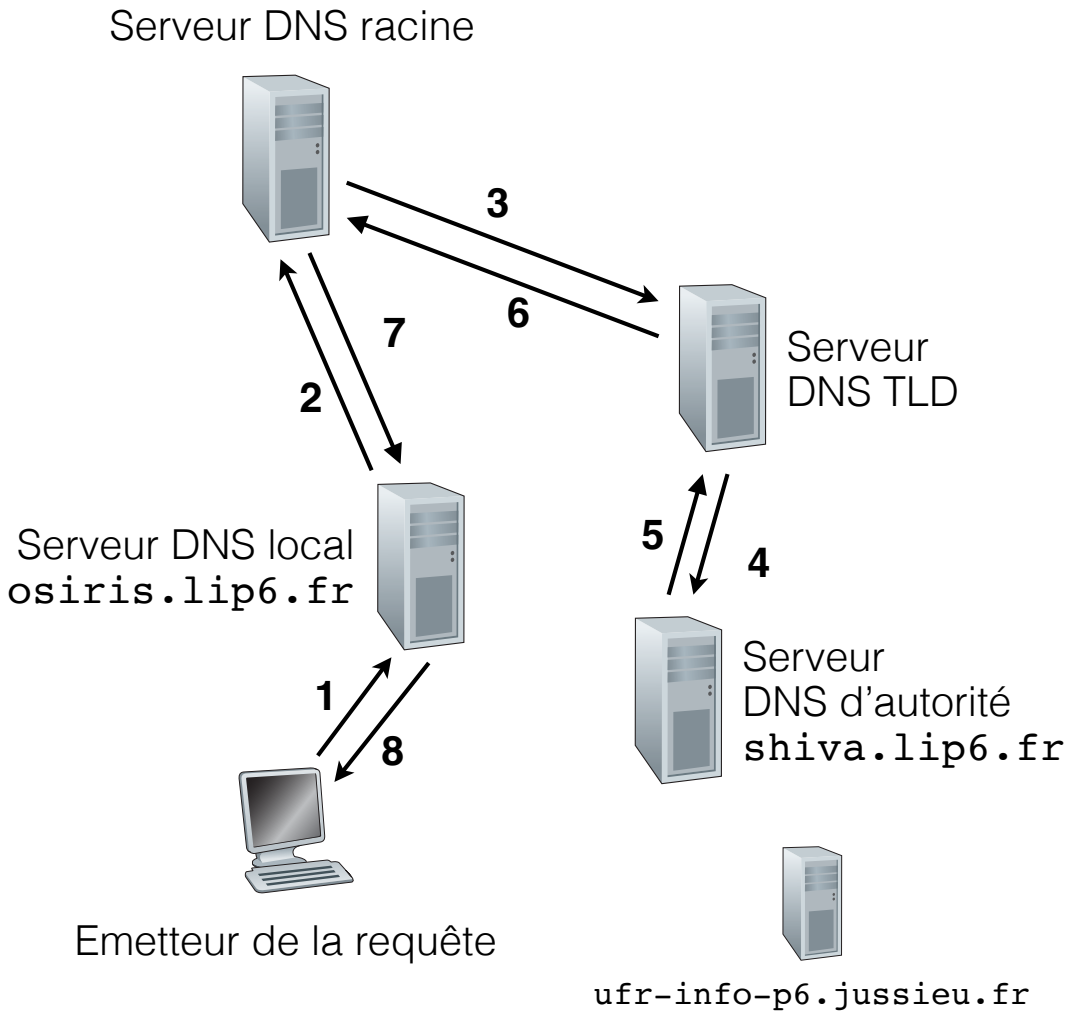
- Un hôte veut obtenir l'adresse IP du serveur web ufr-info-p6.jussieu.fr
  - l'hôte prend en charge lui-même la résolution du nom en contactant successivement les serveurs DNS appropriés
  - les serveurs DNS contactés répondent avec l'adresse IP du prochain serveur à contacter dans la hiérarchie DNS…
  - … le dernier qui fournit la réponse



Résolution de nom

Requête récursive

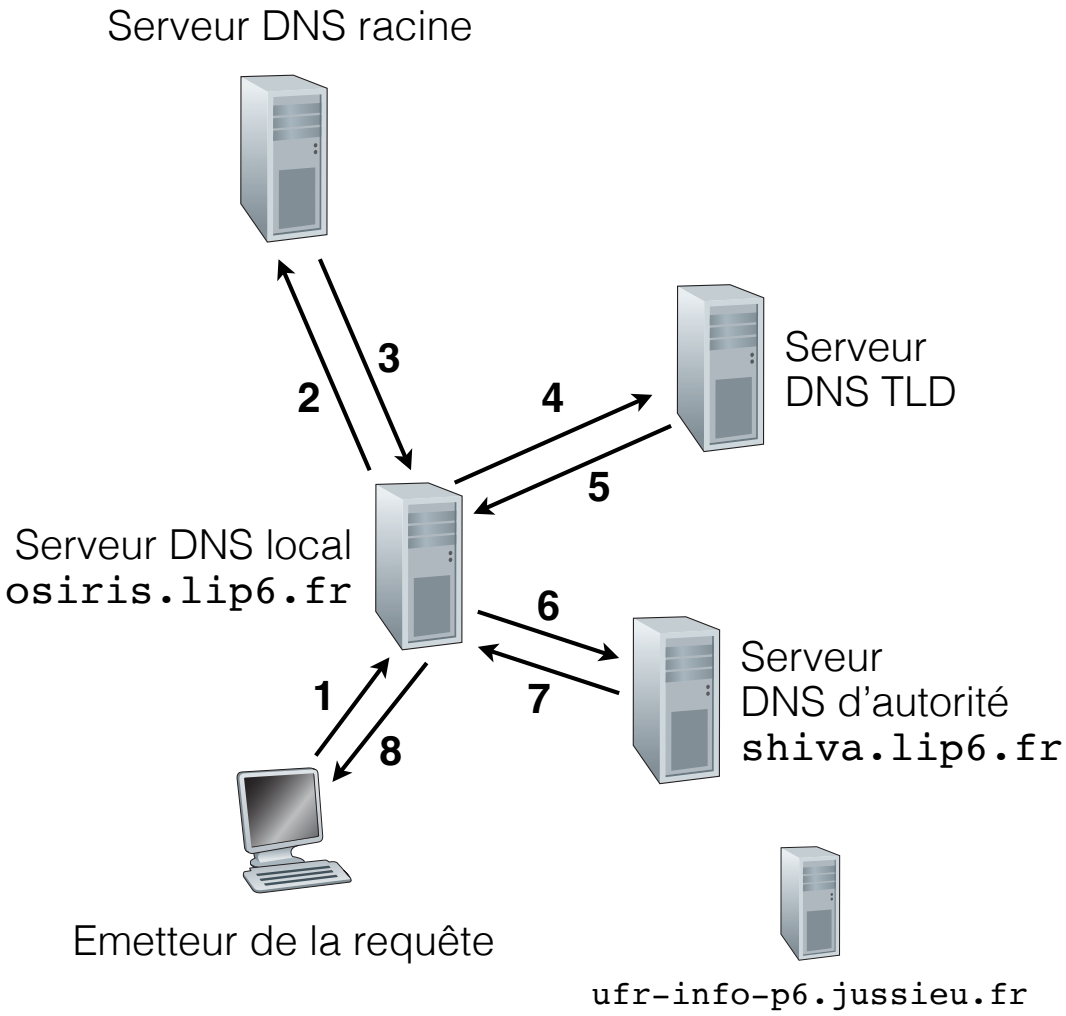
- Un hôte veut obtenir l'adresse IP du serveur web ufr-info-p6.jussieu.fr
  - l'hôte soumet une requête « récursive » à son serveur DNS local et attend de lui la réponse
  - Le serveur DNS local soumet à son tour une requête récursive au DNS racine…
  - … jusqu'au dernier DNS qui renvoie la réponse



Résolution de nom

Requête hybride

- Un hôte veut obtenir l'adresse IP du serveur web ufr-info-p6.jussieu.fr
  - l'hôte soumet une requête « récursive » à son serveur DNS local et attend de lui la réponse
  - le serveur DNS local fait une requête itérative en interrogeant successivement les serveurs DNS approprié



DNS et caches

- La résolution de toutes ces requêtes prend du temps
  - mais elle est nécessaire pour que la communication ait lieu
  - Ex : 1 seconde de latence avant de télécharger un fichier Web
- Mettre en cache les correspondances permet de réduire cette latence
  - certains sites populaires sont souvent visités (Ex : www.google.fr)
  - les adresses IP des serveurs DNS TLD ne changent quasiment jamais
    - les serveurs racines ne sont donc presque jamais visités
- Mise en place d'un cache DNS
  - contenant les correspondances récemment obtenues
    - incluant un champ TTL (*Time To Live*)
- Utilisation possible d'un cache négatif
  - pour se rappeler des noms qui n'ont pu être résolus
    - Ex : erreurs de frappe **wwww**.google.fr ou www.g**ooo**gle.fr
  - protège les serveurs DNS (racine et TLD) des requêtes inutiles

Fiabilité de DNS

- DNS utilise le protocole de transport UDP
  - la fiabilité est prise en charge par DNS
- Les serveurs DNS (racine, TLD et d'autorité) sont répliqués
  - le service de résolution de nom est disponible si au moins un replica est disponible
  - la charge des requêtes est répartie parmi les replicas (*load balancing*)
- Retransmission de la requête à l'expiration d'un temporisateur (*timeout*)
  - sur des serveurs DNS alternatifs
  - sur le même serveur (attente de type *Exponential Backoff*)
  - … un certain nombre de fois (typiquement 16)
- Même identifiant pour toutes les requêtes
  - Le serveur qui répond n'a pas d'importance

DNS Resource Records

DNS est une base de données distribuée qui contient des **Resource Records** (**RR**)

`(name, type, class, ttl, value)`

`type = A`

`name` est le nom d'un hôte

`value` est l'adresse IPv4 de cet hôte

`type = AAAA`

`name` est le nom d'un hôte

`value` est l'adresse IPv6 de cet hôte

`type = NS`

`name` est le nom d'un domaine

`value` est le nom du serveur DNS d'autorité

`type = MX`

`name` est le nom d'un domaine

`value` est le nom du serveur de mail du domaine

`type = CNAME`

`name` est un nom d'alias associé à un nom canonique (réel)

`value` est le nom canonique

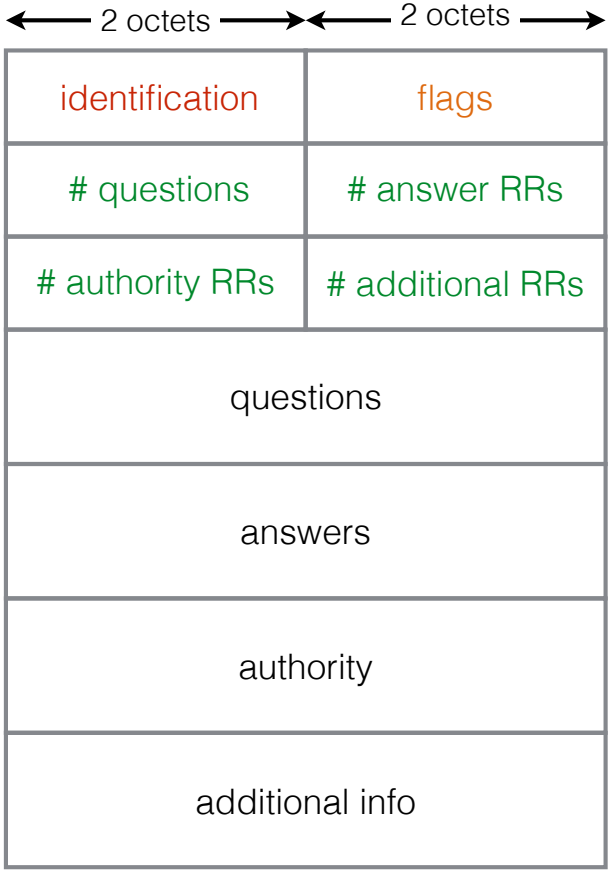
`class = IN`

`IN` pour Internet

Protocole DNS

Les messages de requête et de réponse ont le même format

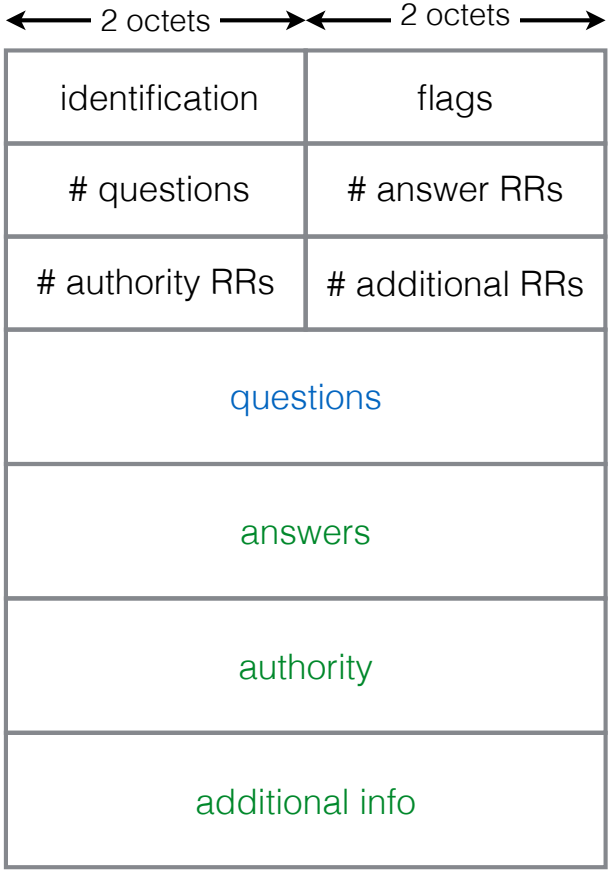
- Identification
  - nombre codé sur 16 bits pour identifier la requête
  - la réponse utilise le même identifiant
- Flags
  - bit QR : requête ou réponse
  - bit RD : récursion désirée ou non
  - bit RA (réponse) : récursion disponible ou non
  - bit AA (réponse) : réponse provenant d'un serveur ayant autorité sur le domaine ou non
  - …
- Nombre d'entrées
  - contenues dans les 4 sections « questions », « answers », « authority » et « additional info »



Protocole DNS

Les messages de requête et de réponse ont le même format

- Questions
  - contient une ou plusieurs questions DNS
  - format : (name, type, class)
- Answers
  - contient les réponses aux questions posées dans la section Question
  - format : (name, type, class, ttl, value)
- Authority
  - contient des informations sur des serveurs d'autorité pour la zone interrogée
- Additional info
  - contient des informations supplémentaires qui peuvent être utiles, comme des adresses IP supplémentaires pour les serveurs



Création et enregistrement d'un nom de domaine

- Choisir un nom de domaine et vérifier sa disponibilité
  - Ex : la startup « Network Utopia, Inc » souhaite enregistrer un domaine networkutopia.com
- Choisir un Fournisseur de Services d'Enregistrement (*registrar* DNS)
  - Ex : Network Solutions
- Lui fournir les noms et adresses IP des serveurs de noms (DNS primaire et secondaire) ayant autorité sur le domaine
  - Le *registrar* DNS se charge d'ajouter deux RRs dans la base de donnée du serveur DNS TLD .com
  - Ex : type NS pour le nom des DNS d'autorité du domaine networkutopia.com
    type A pour l'adresse IP de ces serveurs DNS
- Créer des RRs dans la base de donnée du serveur d'autorité de la startup
  - Ex : type MX pour le nom du serveur de mail du domaine
    type A pour l'adresse IP de www.networkutopia.com et du serveur de mail

Commande Unix dig

- La commande Unix dig permet d'interroger les serveurs DNS afin d'obtenir des informations associées à un nom (de domaine ou d'hôte)
  - adresse IP associée à l'hôte : type A (IPv4) ou AAAA (IPv6)
  - nom du serveur DNS ayant autorité sur le domaine : type NS
  - nom du serveur mail du domaine : type MX
- Syntaxe

dig [@serveur] nom [type] [options]

  - @serveur (optionnel) : de spécifier un serveur DNS particulier à interroger
    - si non spécifié : adressé au DNS local
  - type (optionnel) : A, AAAA, NS, MX, …
    - si non spécifié : type = A
  - options courantes :
    - +short : affiche une sortie simplifiée
    - +noall +answer : affiche uniquement les résultats de la section « answer »
    - +trace : effectue une résolution DNS complète (traçage) en partant de la racine DNS

Interactions HTTP/DNS

- Un utilisateur saisit une URL dans son navigateur
  - Ex : http://www.google.fr/maps
- Le navigateur extrait le nom du site web (Ex : www.google.fr)
- Il doit tout d'abord résoudre le nom pour obtenir l'adresse IP du serveur
  - il envoie une requête DNS au serveur DNS local
  - qui retourne l'adresse IP du serveur web www.google.fr
- Le navigateur établit une connexion TCP et envoie sa requête HTTP
- Il reçoit la réponse du serveur contenant le document de base (index) de la page
- Il examine le document et en extrait la liste des objets inclus dans la page
- Pour chaque objet
  - il envoie une requête DNS au serveur DNS local pour obtenir l'adresse IP du serveur hébergeant l'objet (si différent du serveur hébergeant le document de base)
  - il envoie une requête HTTP pour obtenir l'objet
    - dans une nouvelle connexion TCP si mode non persistant
    - dans la même connexion TCP si mode persistant

Conclusions

- HTTP (*Hypertext Transfer Protocol*)
  - Protocole en mode non connecté
  - Les pages contiennent plusieurs objets
  - Les performances de HTTP dépendent de TCP et du DNS
  - Utilisation de proxys cache pour améliorer les performances
- DNS (*Domain Name System*)
  - Mécanisme d'allocation et de résolution des noms dans Internet
  - Base de données répartie sur plusieurs serveurs
  - Organisation en arborescence des serveurs DNS
  - Utilisation de proxys et de caches pour améliorer les performances

A faire

- Cours 11
  - à relire attentivement
- Devoir 11 sur Moodle (le dernier !)
  - date de rendu : dimanche 1er décembre