DNS

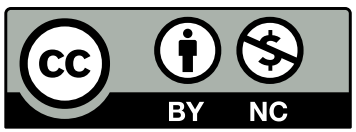
Pierre David
pda@unistra.fr

Université de Strasbourg – Master CSMI

2023 – 2024

Plan

Licence d'utilisation



Plan

Nommage des nœuds Internet

Problème : les adresses IP ne sont pas faciles à mémoriser.
Solution : faire une table de correspondance (fichier `hosts.txt` original)

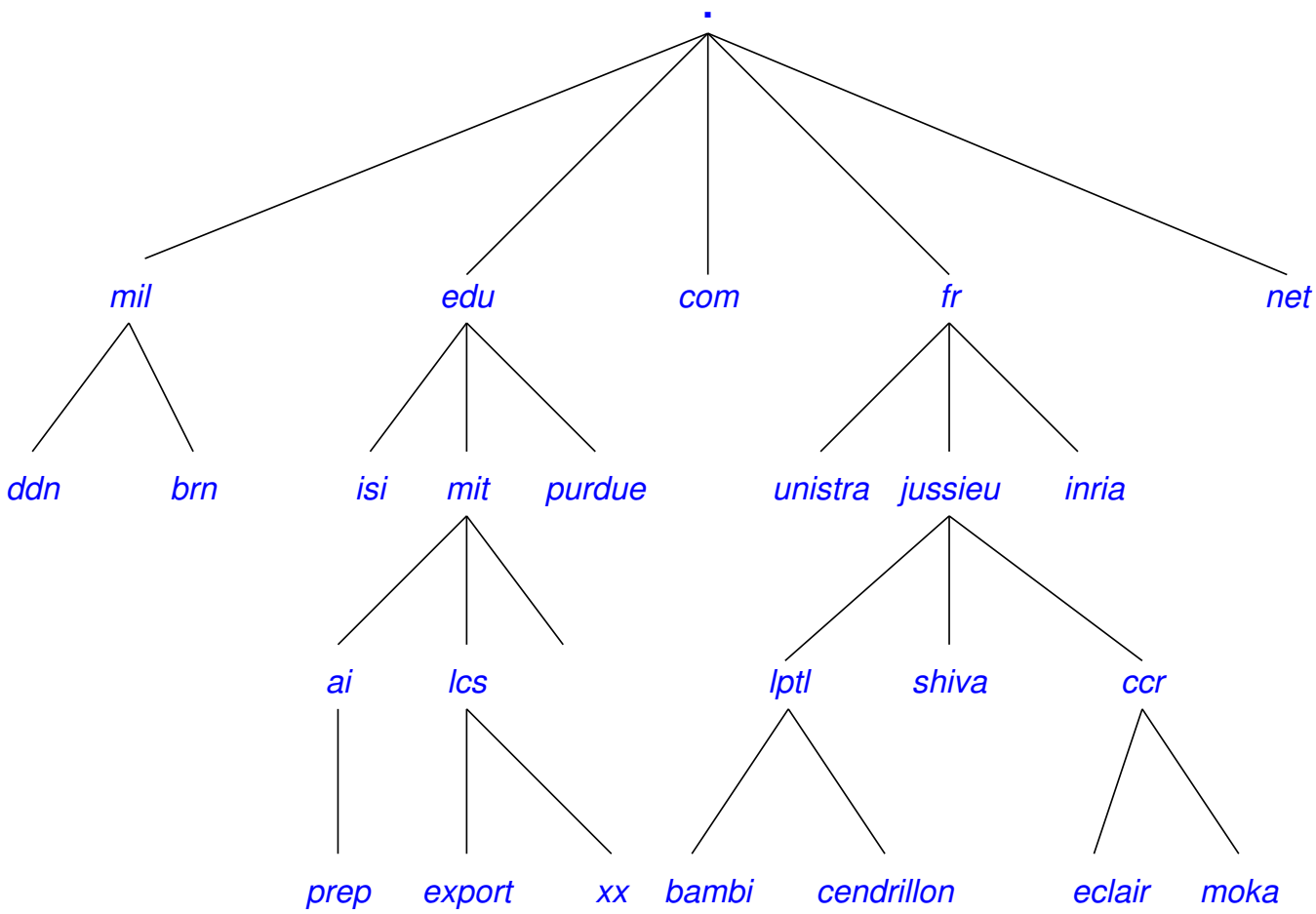
Problèmes :

- impraticable au delà de quelques centaines de machines
  $\Rightarrow$ administration impossible
- ambiguités inévitables
  Exemple : des milliers de machines nommées « *venus* »

Solution : espace de noms hiérarchique et distribué

Nommage

Espace de noms hiérarchique :



Nommage des nœuds Internet

Top-level domains :

com	sociétés privées
edu	universités
gov	agences gouvernementales
mil	departement of defense
net	organisations de réseaux
org	autres organisations
arpa	domaine de service
pays (2 lettres)	toutes organisations dans le pays
info, xxx, etc.	nouveaux TLD attribués par l'IANA

Nommage – Noms absolus

Exemples :

- shiva.jussieu.fr.
- cendrillon.lptl.jussieu.fr.
- prep.ai.mit.edu.

Nommage – Noms relatifs

... relatifs au domaine courant (ex : `lptl.jussieu.fr`)

- cendrillon
- cendrillon.lptl
- cendrillon.lptl.jussieu
- cendrillon.lptl.jussieu.fr
- shiva
- eclair.ccr

Algorithme de *Domain completion* :

1. Entrée : requête = x, domaine courant = $d_1.d_2 \ldots d_n.$
2. Algorithme :

   pour i variant de 1 à n
   sortir si $x.d_i \ldots d_n.$ est trouvé
   fin pour

Nommage

Exemples (avec le domaine courant `lptl.jussieu.fr`) :

Requête = cendrillon.lptl	
cendrillon.lptl.lptl.jussieu.fr.	$\Rightarrow$ échec
cendrillon.lptl.jussieu.fr.	$\Rightarrow$ succès

Requête = www.unistra	
www.unistra.lptl.jussieu.fr.	$\Rightarrow$ échec
www.unistra.jussieu.fr.	$\Rightarrow$ échec
www.unistra.fr.	$\Rightarrow$ succès

Requête = prep.ai.mit.edu	
prep.ai.mit.edu.lptl.jussieu.fr.	$\Rightarrow$ échec
prep.ai.mit.edu.jussieu.fr.	$\Rightarrow$ échec
prep.ai.mit.edu.fr.	$\Rightarrow$ échec
prep.ai.mit.edu.	$\Rightarrow$ succès

Nommage

La *domain completion* entraîne une surcharge du réseau avec des requêtes la plupart du temps inutiles.
$\Rightarrow$ obsolète

Implémentations actuelles :

- un ou plusieurs domaines de recherche explicites
  - Fichier `/etc/resolv.conf` sur Unix (configuré statiquement ou dynamiquement via DHCP)
  - Exemple : `search unistra.fr u-strasbg.fr`
- recherche avec le nom seul, puis avec les différents domaines cités

Plan

Zones

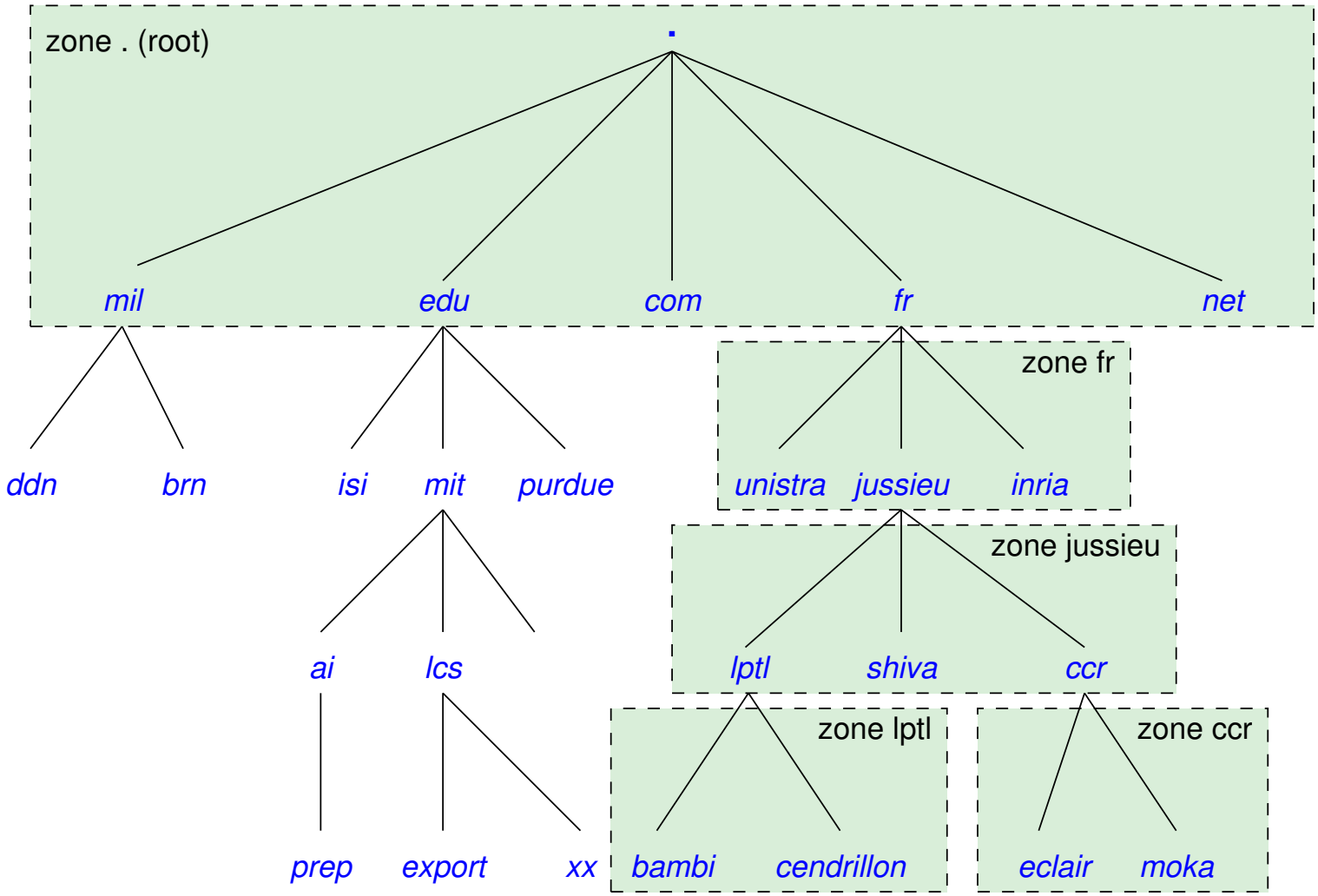
Espace de noms distribué : découpage en zones

Chaque zone est servie par plusieurs *serveurs de noms*

- serveur maître (primaire) pour la zone :
  source de la zone
- serveurs esclaves (secondaires) pour la zone :
  disposent d'une copie (automatique) de la zone

Un serveur peut servir plusieurs zones

Zones


zone . (root)
.
mil
edu
com
fr
net
ddn
brn
isi
mit
purdue
zone fr
unistra
jussieu
inria
zone jussieu
ai
lcs
lptl
shiva
ccr
zone lptl
zone ccr
prep
export
xx
bambi
cendrillon
eclair
moka

Enregistrements – Resource records

Une zone est un ensemble de *resource records*. Par exemple :

```
lptl.jussieu.fr.            IN NS      cendrillon.lptl.jussieu.fr.
lptl.jussieu.fr.            IN SOA     ...
lptl.jussieu.fr.            IN MX      100 shiva.jussieu.fr.
bambi.lptl.jussieu.fr.      IN A       134.157.8.33
bambi.lptl.jussieu.fr.      IN AAAA    2001::660:...
bambi.lptl.jussieu.fr.      IN WKS     TCP smtp telnet ftp
bambi.lptl.jussieu.fr.      IN HINFO   SUN4/SUNOS
mailhost.lptl.jussieu.fr.   IN CNAME   bambi.lptl.jussieu.fr.
```

Une requête est formulée par un *resolver*.
Le protocole utilisé repose sur UDP, avec basculement sur TCP si le volume des requêtes ou des réponses est trop important.

Délégation

Une zone est déléguée à des serveurs (primaire et secondaire) grâce à l'enregistrement de type `NS` :

Zone racine		
`fr.`	`IN NS`	`ns1.nic.fr.`
	`IN NS`	`d.ext.nic.fr.`
Zone fr		
`jussieu.fr.`	`IN NS`	`shiva.jussieu.fr.`
	`IN NS`	`soleil.uvsq.fr.`
Zone jussieu.fr		
`lptl.jussieu.fr.`	`IN NS`	`cendrillon.lptl.jussieu.fr.`
	`IN NS`	`parthe.lpthe.jussieu.fr.`

Ces serveurs **font autorité** : ils délivrent une information de référence (par opposition aux informations issues d'un cache)

Délégation

Problème : pour répondre à des requêtes dans `jussieu.fr`, il faut s'adresser à `shiva.jussieu.fr`.
Solution : « enregistrements collants » (*glue records*)

Zone fr

`jussieu.fr.`	`IN NS`	`shiva.jussieu.fr.`
`shiva.jussieu.fr.`	`IN A`	`134.157.0.129`

Un enregistrement collant :

- a le type `A` ou `AAAA` (adresse IP)
- est placé dans la zone parente (zone mère, grand-mère, etc.) où il est nommé comme `NS`
- donne l'adresse IP du serveur s'il n'y a pas de moyen de l'avoir « normalement »

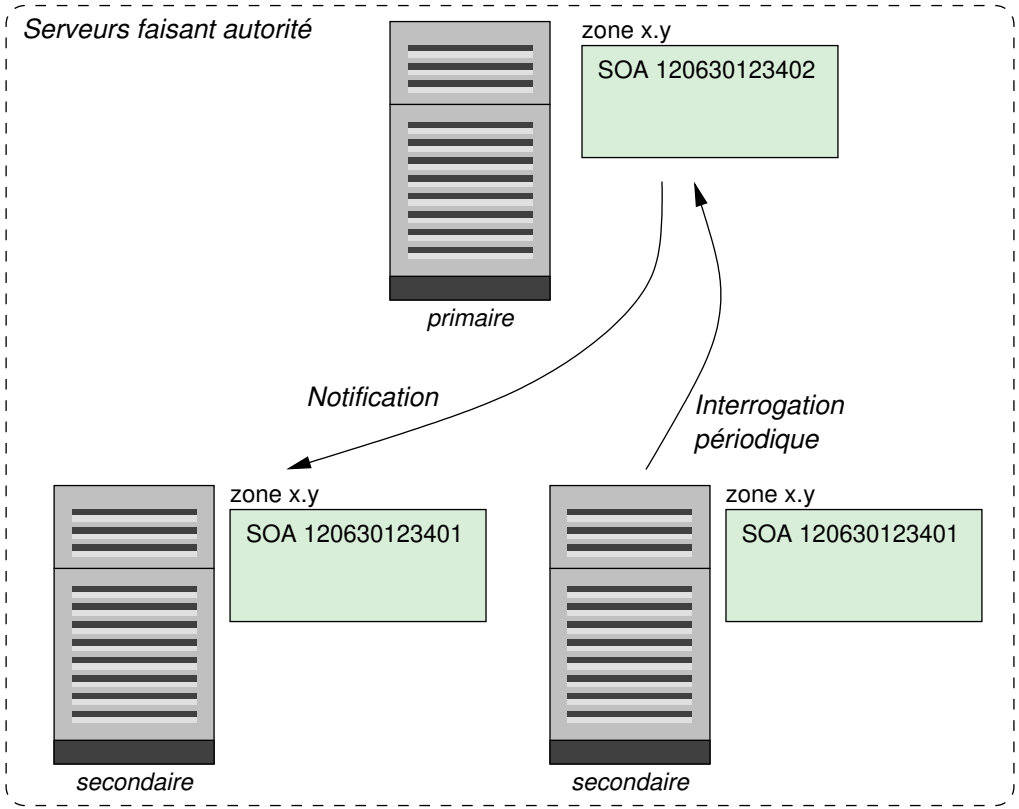
Serveurs primaire et secondaires

Les serveurs primaire et secondaires :

- font autorité
  $\Rightarrow$ délivrent une information de référence
- ne sont pas distingués lors des requêtes
  $\Rightarrow$ primaire non privilégié par rapport aux secondaires
- sont automatiquement répliqués
  $\Rightarrow$ grâce au « numéro de série » de la zone

Serveurs primaire et secondaires

Réplication du primaire vers les secondaires :



L'interrogation périodique et le transfert de zones sont faits automatiquement par les serveurs secondaires

- le serveur primaire peut également provoquer une notification des secondaires en cas de changement de la zone

Serveurs primaire et secondaires

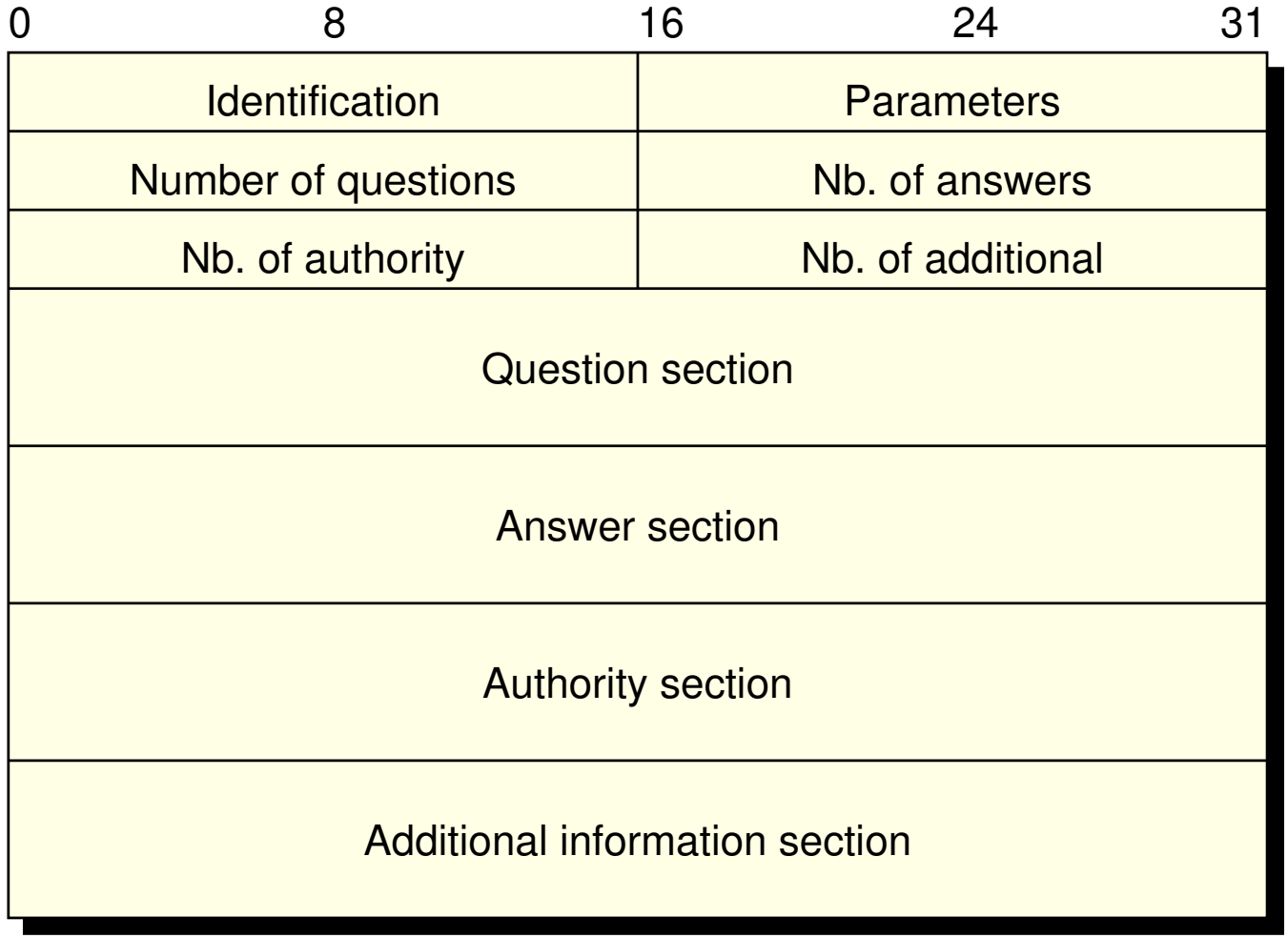
Paramètres de la réplication = enregistrement de type `SOA`, dont les données sont :

primary	nom du serveur primaire
responsible	adresse électronique du responsable de la zone (« @ » remplacé par « . »)
serial	numéro de série de la zone (fonction monotone strictement croissante)
refresh	temps entre deux interrogations du primaire par le secondaire
retry	temps entre deux interrogations si *refresh* dépassé
expire	temps au bout duquel le secondaire renonce à faire autorité
min ttl	TTL minimum des enregistrements de la zone renvoyés par le serveur

Plan

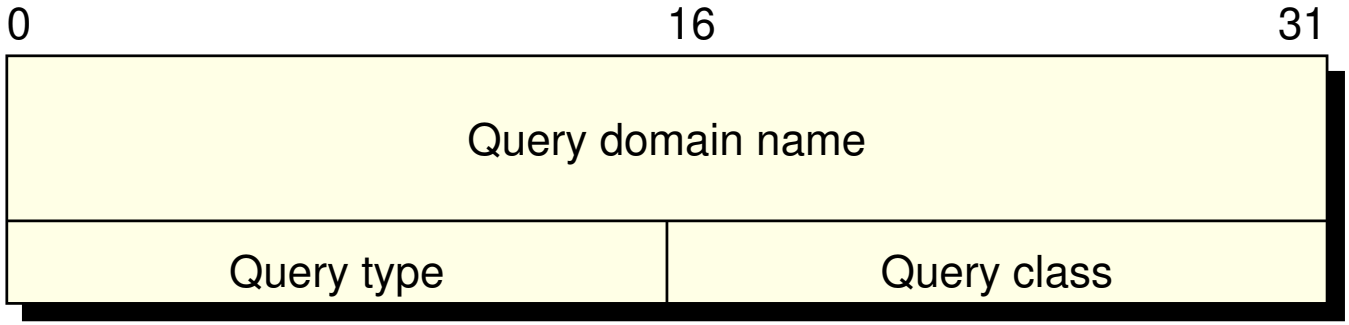
Protocole DNS

Format des paquets DNS :

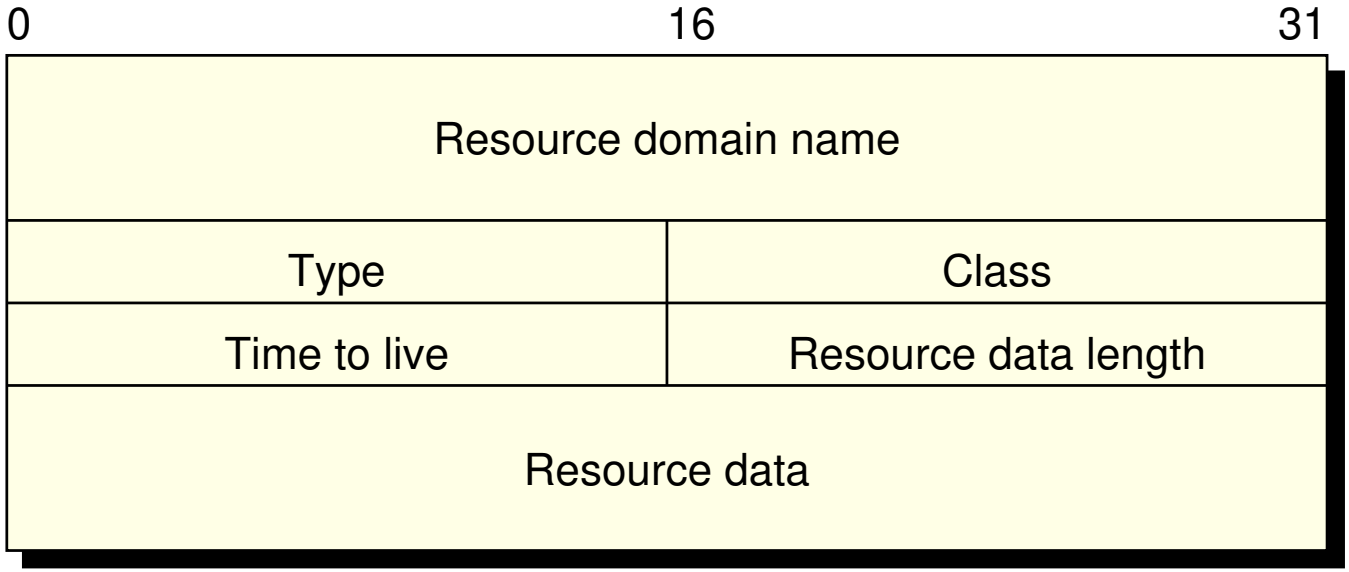


Protocole DNS

Format de la section *Question* :

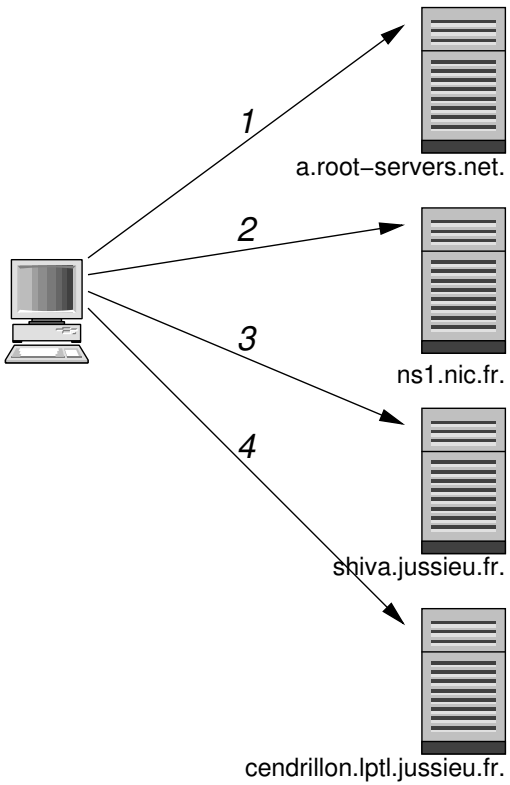


Format des *Resource Records* (autres sections) :



Requête 1

On souhaite connaître l'adresse IP de bambi.lptl.jussieu.fr :



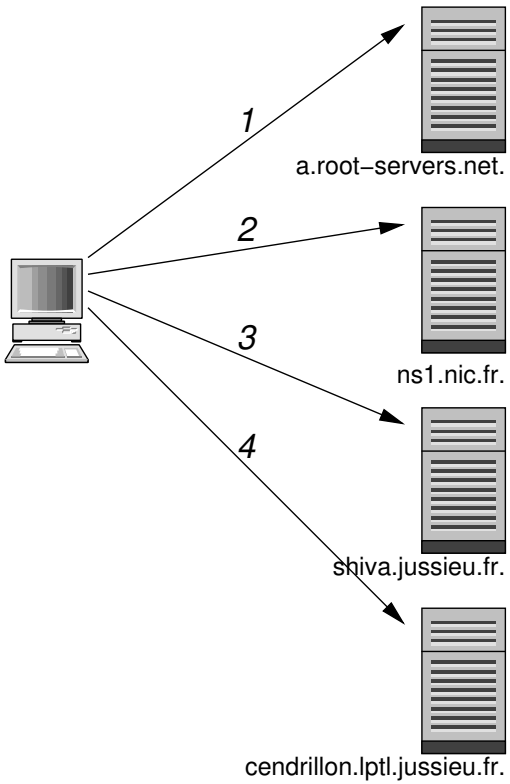
1. Question : bambi.lptl.jussieu.fr/A ?
   - réponse = (*rien*)
   - autorité : `fr IN NS ns1.nic.fr`
   - additionnel : `ns1.nic.fr IN A 1.2.3.4`
2. Question : bambi.lptl.jussieu.fr/A ?
   - réponse = (*rien*)
   - autorité : `jussieu.fr IN NS shiva.jussieu.fr`
   - additionnel : `shiva.jussieu.fr IN A 134.157.0.129`
3. Question : bambi.lptl.jussieu.fr/A ?
   - réponse = (*rien*)
   - autorité : `lptl.jussieu.fr IN NS cendrillon.lptl.jussieu.fr`
   - additionnel : `cendrillon.lptl.jussieu.fr IN A 134.157.8.24`
4. Question : bambi.lptl.jussieu.fr/A ?
   - réponse = `bambi.lptl.jussieu.fr IN A 134.147.8.33`
   - autorité : (*rien*)
   - additionnel : (*rien*)

Toutes les réponses vont dans le cache du client

Note : chaque client DNS (*resolveur*) connaît les noms et adresses IP des serveurs de la zone racine

Requête 2

On souhaite ensuite connaître l'adresse IP de donald.lptl.jussieu.fr :



Les requêtes 1 à 3 sont inutiles car le cache du client contient déjà les informations :

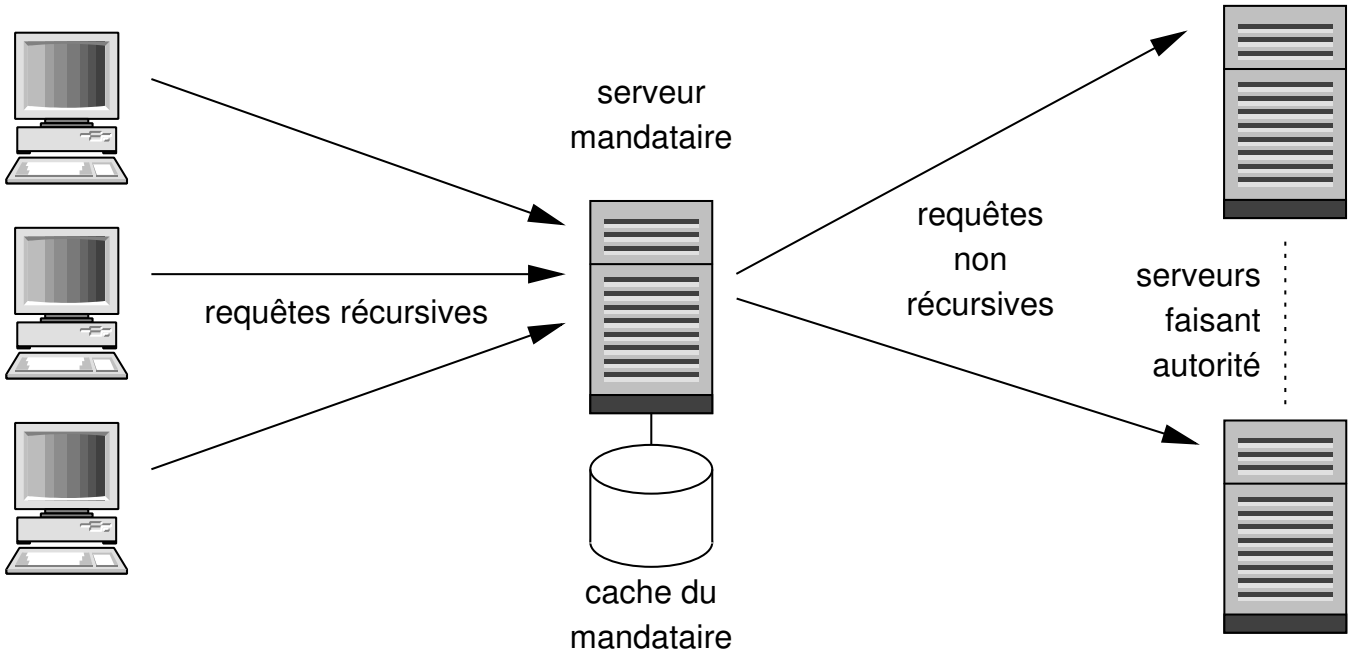
```
lptl.jussieu.fr              IN NS    cendrillon.lptl.jussieu.fr
cendrillon.lptl.jussieu.fr   IN A     134.157.8.24
```

4. Question : donald.lptl.jussieu.fr/A ?
   - réponse = `donald.lptl.jussieu.fr IN A 134.147.8.23`
   - autorité : (*rien*)
   - additionnel : (*rien*)

⇒ optimisation importante grâce au cache du resolveur

Serveur mandataire

Dans la pratique, le cache n'est pas dans chaque client
$\Rightarrow$ serveur mandataire



Même protocole entre clients et serveur mandataire qu'entre serveurs (excepté le bit *recursion*).

Serveur mandataire

Intérêts d'un serveur mandataire au niveau d'une organisation :

- mutualiser le cache entre les différents clients
  - si quelqu'un cherche un moteur de recherche bien connu, l'information est vraisemblablement déjà dans le cache du mandataire
- simplifier l'implémentation des clients
  - le client n'a pas à gérer toutes les requêtes individuelles aux serveurs faisant autorité
    (timeout, panne, sélection du meilleur serveur, etc.)

En pratique, chaque opérateur ou grande organisation a souvent un serveur mandataire

Requêtes

Interrogation DNS

Commande `nslookup` : accès facile au serveur de noms

```
> nslookup bambi.lptl.jussieu.fr          # requête de type A par défaut
Server:         127.0.0.53                  # serveur DNS mandataire interrogé
Address:        127.0.0.53

Non-authoritative answer:                   # réponse venant du cache du mandataire
Name:   bambi.lptl.jussieu.fr
Address: 134.157.8.33

> nslookup -norecurse -q=ns unistra.fr d.nic.fr   # requête de type NS envoyée au serveur de fr
Server:         d.nic.fr                    # serveur contacté
Address:        2001:678:c::1               # son adresse (IPv6 seulement)

Non-authoritative answer:
*** Cant find unistra.fr: No answer        # le serveur ne connait pas la réponse...

Authoritative answers can be found from: # ... mais peut nous indiquer où la demander
unistra.fr      nameserver = ns1.u-strasbg.fr.
unistra.fr      nameserver = shiva.jussieu.fr.
unistra.fr      nameserver = ns2.u-strasbg.fr.
ns2.u-strasbg.fr        has AAAA address 2001:660:2402::3
ns1.u-strasbg.fr        has AAAA address 2001:660:2402::1
shiva.jussieu.fr        internet address = 134.157.0.129
ns2.u-strasbg.fr        internet address = 130.79.200.3
ns1.u-strasbg.fr        internet address = 130.79.200.1
```

Requêtes

Interrogation DNS

Commande `dig` : utilitaire de plus bas niveau, proche du protocole DNS

```
> dig ns unistra.fr @d.nic.fr    # interrogation de type NS envoyée au serveur de .fr
...
;; Got answer:                   # chic, on a reçu une réponse !
# informations sur la réponse : flags du paquet, nombre de RR dans chaque section (à noter ici : aucune réponse)
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 3, ADDITIONAL: 6

;; QUESTION SECTION:             # rappel de la question (première partie du paquet)
;unistra.fr.                IN NS

;; AUTHORITY SECTION:            # les informations d'autorité (3e partie du paquet)
unistra.fr.        172800 IN NS    ns2.u-strasbg.fr.
unistra.fr.        172800 IN NS    ns1.u-strasbg.fr.
unistra.fr.        172800 IN NS    shiva.jussieu.fr.

;; ADDITIONAL SECTION:           # les informations additionnelles (4e partie du paquet)
ns2.u-strasbg.fr. 172800 IN AAAA 2001:660:2402::3
ns1.u-strasbg.fr. 172800 IN AAAA 2001:660:2402::1
shiva.jussieu.fr. 172800 IN A    134.157.0.129
ns2.u-strasbg.fr. 172800 IN A    130.79.200.3
ns1.u-strasbg.fr. 172800 IN A    130.79.200.1

;; Query time: 12 msec
;; SERVER: 2001:678:c::1#53(d.nic.fr) (UDP)
```

Requêtes

L'envers du décor

Le site `https://messwithdns.net/` permet de :

- créer des enregistrements de test dans une zone réelle
- de visualiser les requêtes arrivant sur le serveur DNS
- de simuler des situations typiques (ou atypiques)

$\Rightarrow$ très pédagogique pour comprendre l'« envers du décor »

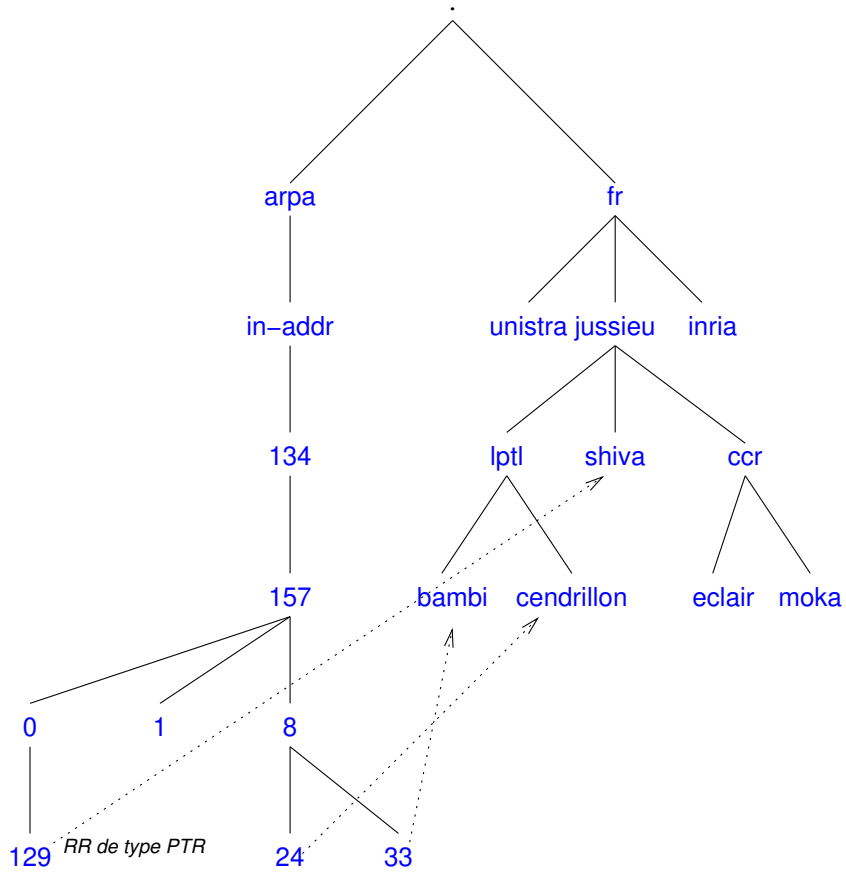
Plan

Conversion inverse

Conversion d'adresses IP en noms

Exemple :
`24.8.157.134.in-addr.arpa.`

Similaire pour IPv6 avec :
`...1.0.0.2.ip6.arpa`



Utilise des enregistrements de type `PTR`

Plan

Autres usages

En plus des conversions nom $\leftrightarrow$ adresse IP, le DNS peut être utilisé pour :

- annonce de propriétés (géolocalisation, clef SSH, etc.)
- routage du courrier
- annonce de services
- autres choses (Projet Athena...)

Géolocalisation

Le DNS permet de publier la localisation d'une machine

```
serveur.domaine.fr IN LOC 48 6 55.000 N 1 38 19.000 W 30.00m 1m 10000m 10m
```

Décodage :

latitude	48˚6'55" nord
longitude	1˚38'19" ouest
altitude	30 m
taille	1 m
précision horizontale	10 km
précision verticale	10m

$\Rightarrow$ très peu utilisé en pratique
(sécurité + méconnaissance + manque d'intérêt pratique)

Clef publique de serveur SSH

Le DNS permet de publier les empreintes de clefs publiques de serveurs SSH

```
serveur.domaine.fr IN SSHFP 2 1 empreinte
```

- 2 = algorithme de chiffrement (DSS)
- 1 = type d'empreinte (SHA-1)
- empreinte convertie en base64

Nécessite une confiance dans la sécurité du DNS
$\Rightarrow$ DNSSEC

Routage de messagerie

Client SMTP doit acheminer un courriel à *local@remote* :

1. requête DNS *remote* MX
   $\Rightarrow$ réponse = liste de RR de type `MX` avec <*poids*, *nom*>
2. pour chaque RR trié par *poids* croissant
   - requête DNS *nom* AAAA/A
   - pour chaque adresse obtenue
     - tenter une connexion sur le port SMTP
     - si succès : le dialogue SMTP s'effectue
3. si pas de RR de type `MX` :
   - requête DNS *remote* AAAA/A
   - pour chaque adresse obtenue
     - tenter une connexion sur le port SMTP
     - si succès : le dialogue SMTP s'effectue
4. si aucune réponse : « *Host not found* »

Routage de messagerie

Exemple : je souhaite envoyer un courriel à pda@unistra.fr

1. requête DNS « unistra.fr MX » ?
   ⇒ réponse :

```
unistra.fr   MX 20 mailhost-v6.u-strasbg.fr
unistra.fr   MX 10 mailhost.u-strasbg.fr
```

2. requêtes DNS « mailhost.u-strasbg.fr AAAA/A » ?
   ⇒ réponse :

```
mailhost.u-strasbg.fr   A 130.79.222.211
mailhost.u-strasbg.fr   A 130.79.222.212
...
```

3. connexion SMTP à la première adresse IP qui répond
4. si échec : recommencer avec mailhost-v6.u-strasbg.fr

Annonce de services – SRV

Format :

_service._proto.name SRV prio poids port cible

avec:

service	nom du service (cf `/etc/services`)
proto	`tcp`, `udp`, etc.
prio	priorité (0 = plus haute priorité)
poids	pour l'équilibrage de charge
port	numéro de port
cible	nom DNS de la cible

Exemples:

```
_imaps._tcp.gmail.com        SRV 5 0 993 imap.gmail.com.
_submission._tcp.gmail.com   SRV 5 0 587 smtp.gmail.com.
_http._tcp.pkg.freebsd.org   SRV 10 10 80 pkg0.isc.freebsd.org.
```

Usages amusants

Pourquoi se limiter à des usages banals ?

Un serveur peut répondre à des requêtes peu « classiques » :

```
> dig +short 1234.words @dns.toys                    # ou nslookup 1234.words dns.toys
"1234 = one thousand, two hundred thirty four"

> dig strasbourg/fr.weather @dns.toys
strasbourg. 1 IN TXT "Strasbourg (FR)" "3.20C" "66% hu." "clearsky_day" "11:00, Wed"
strasbourg. 1 IN TXT "Strasbourg (FR)" "5.70C" "59% hu." "clearsky_day" "13:00, Wed"
strasbourg. 1 IN TXT "Strasbourg (FR)" "7.10C" "55% hu." "clearsky_day" "15:00, Wed"

> nslookup -q=txt pi. dns.toys
pi      text = "3.14159265358979323846264338327950288419716 9"

> dig +short TXT 67400.cp.bortzmeyer.fr              # TXT ou LOC ou URI
"ILLKIRCH GRAFFENSTADEN"
```

Cf. `https://dns.toys` et d'autres...

Attention : le protocole peut être détourné à des fins moins amusantes

⇒ DNS menteurs (cf. `https://www.bortzmeyer.org/dns-menteur.html`)

Plan

DNS poisoning

Empoisonnement des résolutions DNS (« *DNS poisoning* »).

Objectif : remplir le cache DNS d'un serveur DNS vulnérable avec de fausses informations.

Méthode :

1. amener le serveur DNS vulnérable à interroger un serveur DNS malicieux
   Exemple : envoyer un message avec un « From » égal à `toto@pirate.com`.

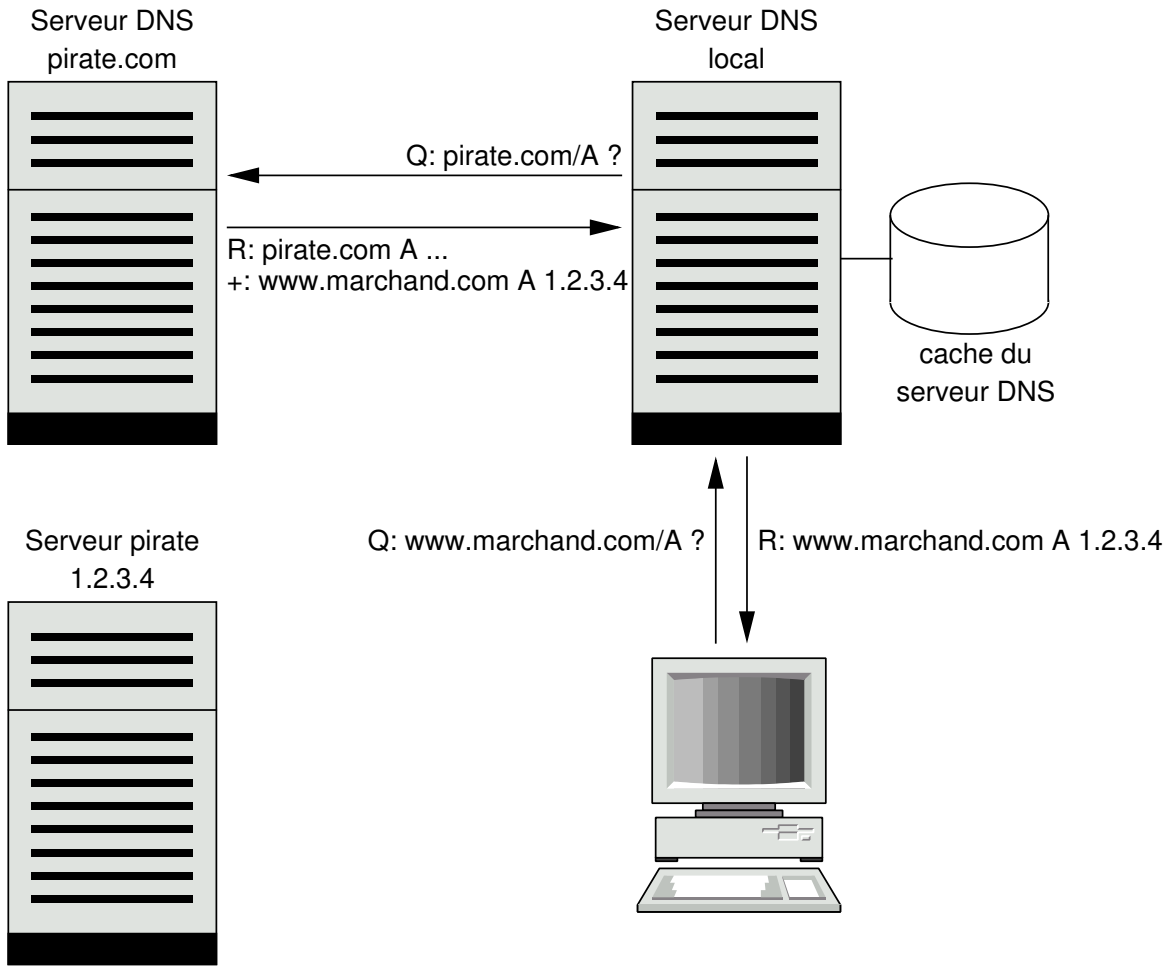
DNS poisoning

2. le serveur DNS malicieux (mais légitime pour pirate.com) répond normalement, mais ajoute des informations additionnelles
   Exemple : www.marchand.com = adresse d'un serveur pirate
3. le serveur DNS vulnérable ajoute aveuglément ces informations dans son cache. La prochaine fois que la victime essaye d'accéder à www.marchand.com, elle obtient l'adresse du pirate et pas l'adresse légitime.
4. tant qu'il est dans le cache, le poison continue à se répandre

Remède : utiliser un serveur DNS à jour

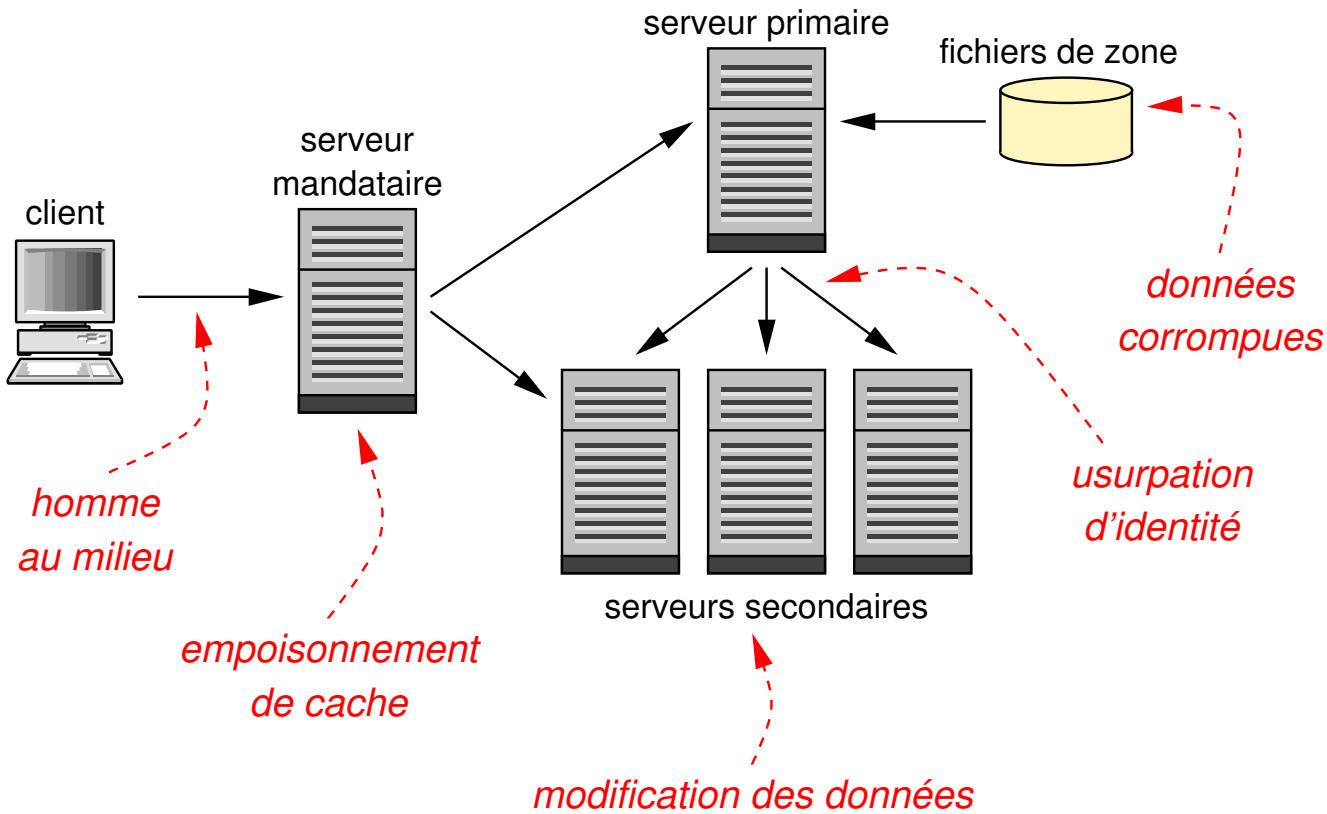
DNS poisoning

Empoisonnement des résolutions DNS (« *DNS poisoning* »).



Vulnérabilités DNS

Récapitulatif des vulnérabilités de l'infrastructure DNS



DNSSEC

Extensions de sécurité pour le DNS : DNSSEC

- utilisation de la crytographie pour assurer la vérification des réponses depuis la racine
- nouveaux types de resource records
  - signature des enregistrements de la zone : `DNSKEY`, `RRSIG`
  - absence d'enregistrement : `NSEC3`
  - délégation sécurisée : `DS`

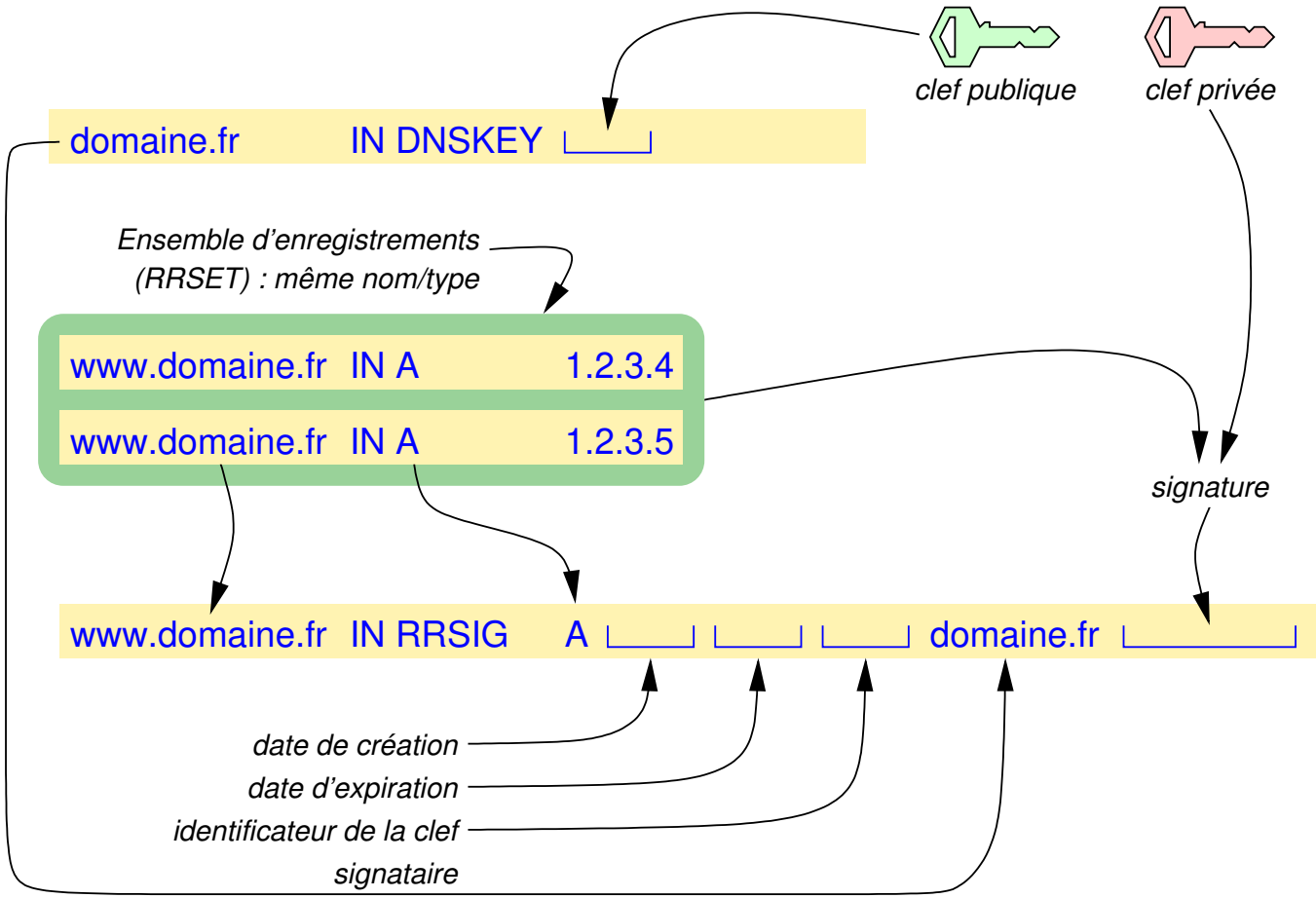
DNSSEC – Signature des enregistrements

Signature des enregistrements d'une zone :

- `DNSKEY` : clef(s) publique(s) de la zone
- `RRSIG` : signature d'un ensemble d'enregistrements (mêmes nom et type) avec une clef privée
- `NSEC3` : non-existence de l'information

DNSSEC – Signature des enregistrements

Signature des enregistrements d'une zone :



DNSSEC – Preuve de non existence

Preuve de non existence ?
$\Rightarrow$ renvoyer une réponse vide ne permet pas de prouver (cryptographiquement) l'absence de réponse.

Solution : enregistrement de type `NSEC3`

Sur le serveur de noms :

1. calcul de l'empreinte (hash) de tous les noms des enregistrements de la zone
2. tri des empreintes, tel que $h_1 < h_2 < \ldots < h_n$
3. génération des enregistrements :

   h_i `IN NSEC3` $h_{i+1}\ t_1 \ldots t_{n_i}$

   h_i `IN RRSIG` ...

   où les t_j sont les types des enregistrements du nom correspondant à h_i (ex: `A`, `RRSIG`, etc.)
4. le « suivant » du dernier (h_n) est h_1

DNSSEC – Preuve de non existence

Lorsqu'une requête parvient au serveur pour un nom de domaine d inexistant, le serveur renvoie les enregistrements :

$$h_i \quad \texttt{IN NSEC3} \quad h_{i+1}\ t_1 \ldots t_{n_i}$$
$$h_i \quad \texttt{IN RRSIG} \quad \ldots$$

tels que $h_i < \text{empreinte}(d) < h_{i+1}$

Le client peut vérifier que $h_i < \text{empreinte}(d) < h_{i+1}$
$\Rightarrow$ preuve de non-existence

DNSSEC – Preuve de non existence

Lorsqu'une requête parvient au serveur pour un nom d_i existant mais sans le type demandé, le serveur renvoie les enregistrements :

$$h_i \quad \texttt{IN NSEC3} \quad h_{i+1}\ t_1 \ldots t_{n_i}$$
$$h_i \quad \texttt{IN RRSIG} \quad \ldots$$

tels que h_i = empreinte (d_i)

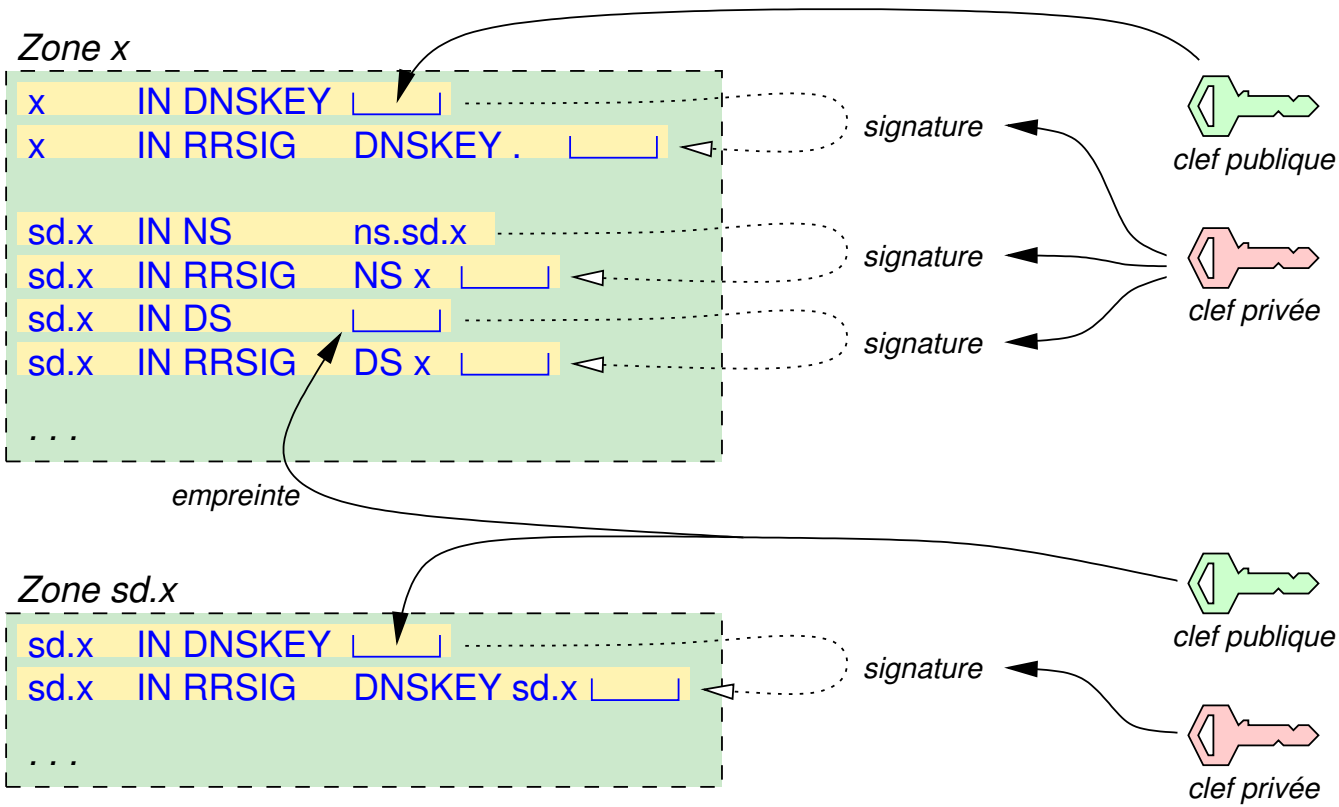
Le client peut vérifier que h_i correspond et que le type demandé n'est pas cité dans les t_j
$\Rightarrow$ preuve de non-existence

DNSSEC – Délégation sécurisée

Nouveau type d'enregistrement : `DS`

⇒ empreinte de la clef publique de la zone fille

⇒ avec `RRSIG` : signée par la clef privée de la zone parente



DNSSEC – Exploitation

Problème opérationnel de DNSSEC : la rotation des clefs

- interaction avec la zone parente (DS)
  coordination, ou interface d'administration nécessaire
- obsolescence de la clef
  $\Rightarrow$ les enregistrements de la zone deviennent invalides
- vérification par les clients (ou les serveurs mandataires)
  $\Rightarrow$ les enregistrements de la zone n'existent plus

DNSSEC – Bilan

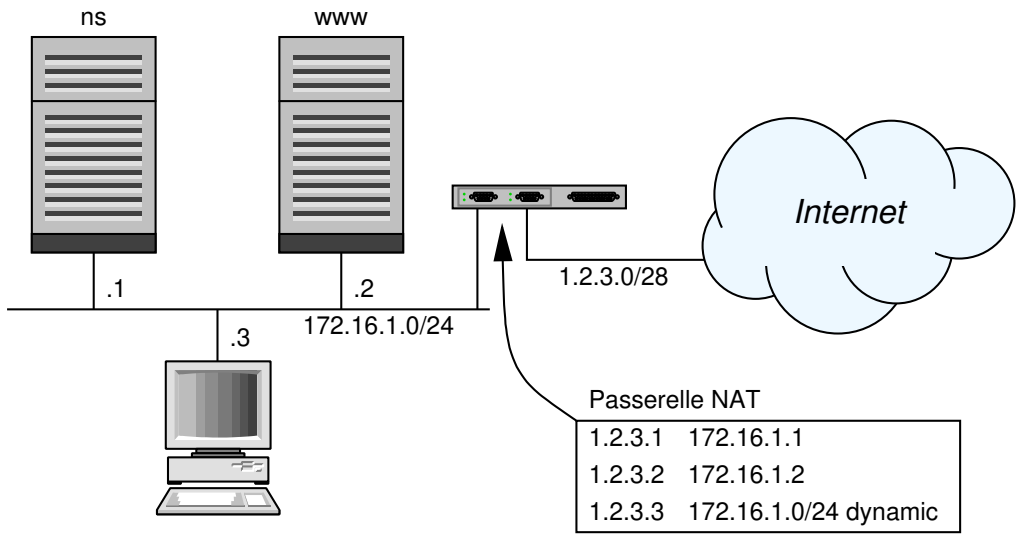
- extensions spécifiées dès 2003
- peu d'adoption tant que la racine n'était pas signée
- peu d'adoption depuis...
- fragilité du système de nommage (rotation des clefs)

Plan

Vues DNS – Split DNS

Réponses différentes en fonction de l'adresse du client DNS

Exemple :



Si la requête vient de 172.16.1.0/24 :

```
ns.truc.fr    A 172.16.1.1
www.truc.fr   A 172.16.1.2
pc.truc.fr    A 172.16.1.3
```

Si la requête vient d'ailleurs :

```
ns.truc.fr        A 1.2.3.1
www.truc.fr       A 1.2.3.2
postes.truc.fr    A 1.2.3.3
```

Plan

Bonjour

Bonjour :

- système d'autoconfiguration et de découverte de services
- centré sur les les réseaux locaux et ad-hoc
- initialement développé par Apple
  Appelé « RendezVous » avant 2004
- repris par le Working Group « ZeroConf » de l'IETF
  (groupe fermé en 2004 sans avoir publié de RFC)

Implémentations :

- Apple : Bonjour
- Open Source : Avahi

Bonjour

3 niveaux :

- adressage : DHCP ou adresse lien-local (IPv4LL ou IPv6)
- nommage : mDNS (Multicast DNS)
- découverte de service : DNS-SD (Service Discovery)

Bonjour – mDNS

Multicast DNS: `http://www.multicastdns.org`

Similaire au DNS, à l'exception de :

- utilisation de l'adresse multicast lien-local pour les requêtes comme pour les réponses
  (addresse = `224.0.0.251` ou `ff02::fb`)
- restreint au domaine spécial « `.local` »
  (et `254.169.in-addr.arpa` et `8` $\rightarrow$ `b.e.f.ip6.arpa`)
- port UDP 5353
- les requêtes peuvent inclure plus d'une question
- les réponses doivent inclure toutes les réponses connues

Bonjour – mDNS

Exemple: adresse IPv4 de truc.local ?

1. requête avec la question
   `truc.local/A` ?
2. truc reçoit la question et répond :
   `truc.local A 169.254.12.34`
3. chaque machine qui a reçu cette réponse met à jour son cache

Bonjour – Cache mDNS

Chaque machine :

- a un cache mDNS par interface
- est à l'écoute des requêtes mDNS :
  - répond quand elle est citée
  - se souvient de ne pas re-poser la même question
- écoute les réponses mDNS et met à jour son cache

Quand le TTL d'un enregistrement va expirer :

- s'il y a un intérêt à cet enregistrement...
- ... alors la machine doit émettre des requêtes (p. ex. à 80 %, 85 %, 90 % and 95 %) pour le rafraîchir

Bonjour – Optimisations de mDNS

- Groupement de requêtes
  $\Rightarrow$ peut-être plus d'une question dans une requête
- Suppression des réponses connues
  $\Rightarrow$ les requêtes doivent inclure les réponses déjà obtenues
- Suppression des questions dupliquées
  $\Rightarrow$ écoute constante des requêtes
- Suppression des réponses dupliquées
  $\Rightarrow$ écoute constante des réponses
  $\Rightarrow$ ne pas répondre trop vite (délai aléatoire $\in$ [0..20] ms)
- Limitation de trafic
  $\Rightarrow$ délai exponentiel (questions/réponses) : [1..3600] s

Bonjour – Obtention d'un nom mDNS

Je suis une imprimante et je veux m'annoncer comme truc :

1. à t_0, j'attends un délai aléatoire $d \in [0..250]$ms
2. à $t_1 = t_0 + d$, j'émets une requête mDNS : `truc.local/ANY` ?
3. si je n'obtiens pas de réponse, je recommence à $t_2 = t_1 + 250$ms, puis à $t_3 = t_2 + 250$ms
4. si j'obtiens une réponse avant $t_4 = t_3 + 250$ms, je dois choisir un autre nom
5. si je n'obtiens aucune réponse, je peux m'annoncer : j'envoie 2 réponses non sollicitées (avec un délai de 1 s) avec mon enregistrement :
   `truc.local A 169.254.12.34`

Délai court $[t_0..t_4] \Rightarrow$ pas de délai aléatoire dans la réponse aux tests

Bonjour – Obtention d'un nom mDNS

Concurrence lorsque 2 machines testent simultanément :

- chacune attend une réponse
- quand le délai (t_4) expire, elles s'annoncent toutes les deux

Solution :

- chaque machine teste avec l'enregistrement proposé dans la section « autorité » de la requête DNS
- quand les machines détectent un test, elles comparent les enregistrements proposés : l'enregistrement lexicographiquement le plus grand gagne !
  `truc.local A 169.254.12.34` $<$ `A 169.254.12.35`
- la machine la plus basse lexicographiquement attend 1 s et recommence
- la machine la plus haute lexicographiquement ignore le test de l'autre

Bonjour – mDNS unicast

Utilisation de l'unicast :

- un nœud se connecte à un réseau, il doit apprendre son environnement
- son cache mDNS est vide
- utilisation de l'unicast pour éviter une surcharge réseau
  - requête avec l'adresse de destination multicast habituelle et le flag QU (au lieu de QM habituellement)
  - les machines qui répondent doivent utiliser l'adresse unicast de l'émetteur pour répondre directement (si dans le même sous-réseau)

Utilisation de requêtes dirigées : également possible

$\Rightarrow$ applications spécialisées, gestion réseau, etc.

Bonjour – DNS-SD

Comment configurer automatiquement certains services ?

- comment indiquer le serveur de courrier électronique d'une organisation ?
- comment indiquer où mon système doit aller chercher ses mises à jour ?

$\Rightarrow$ DNS-SD = Service Discovery `http://dns-sd.org/`

Bonjour – DNS-SD

DNS-SD utilise les enregistrements `SRV` du DNS

Mais problème avec les enregistrements `SRV` :

- ils sont parfaits si tous les `SRV` sont équivalents
  $\Rightarrow$ idéal pour obtenir n'importe quel serveur IMAPS
- ils sont inadaptés si je veux une imprimante en particulier
  $\Rightarrow$ tous les `SRV _ipp._tcp` ne sont pas équivalents

$\Rightarrow$ DNS-SD introduit une chaîne de caractères lisible par un humain dans un enregistrement `PTR` :

```
_service._proto.dom PTR instance._service._proto.dom
```

- « instance » est une chaîne UTF-8 (max 63 octets)
- Exemple (fictif) :

```
_ipp._tcp.unistra.fr PTR Ma Laserjet._ipp._tcp.unistra.fr
```

Bonjour – DNS-SD

Étapes après la requête `PTR` :

1. demander le `SRV` pour `instance._service._proto.domain`
   $\Rightarrow$ obtenir le nom DNS et le numéro de port
   (un seul `SRV` par instance)
2. demander le `TXT` pour `instance._service._proto.domain`
   $\Rightarrow$ informations complémentaires (file d'impression, espace disque, etc.)
   jusqu'à 65535 octets, mais limite = taille des paquets

Bonjour – DNS-SD

Je veux imprimer quelque chose sur mon imprimante :

1. requête : `_ipp._tcp.local/PTR` ?
   ⇒ `_ipp._tcp.local PTR Ma Laserjet._ipp._tcp.local`
2. requête : `Ma Laserjet._ipp._tcp.local/SRV` ?
   ⇒ `Ma Laserjet._ipp._tcp.local SRV 0 0 631 truc.local`
   ⇒ `truc.local A/AAAA ...`
3. requête : `Ma Laserjet._ipp._tcp.local/TXT` ?
   ⇒ `Ma Laserjet._ipp._tcp.local TXT Papersize=A4`

En pratique, les requêtes 2 et 3 n'existent pas : l'information est déjà renvoyée dans la section « additionnelle » de la réponse 1

Bonjour – DNS-SD

Comment obtenir tous les services disponibles ?

Requête: `_services._dns-sd._udp.domain/PTR` ?

⇒ `_services._dns-sd._udp.domain PTR _ssh._tcp.domain.`

⇒ `_services._dns-sd._udp.domain PTR _ipp._tcp.domain.`

⇒ `_services._dns-sd._udp.domain PTR _http._tcp.domain.`

`...`

⇒ pour le diagnostic

Bonjour – Wide Area Bonjour

DNS-SD peut utiliser le DNS traditionnel avec l'arborescence globale

Énumération pour le domaine d :

`b._dns-sd._udp.d`	liste des domaines pour le browsing
`db._dns-sd._udp.d`	domaine de browsing par défaut
`r._dns-sd._udp.d`	liste des domaines pour l'enregistrement de services
`dr._dns-sd._udp.d`	domaine d'enregistrement par défaut
`lb._dns-sd._udp.d`	domaine de browsing legacy

Exemples:

- `db._dns-sd._udp.dns-sd.org. PTR dns-sd.org.`
- `b._dns-sd._udp.dns-sd.org. PTR IL 2 4th. Floor South.dns-sd.org.`