RFC 7719 : DNS Terminology

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 19 décembre 2015

Date de publication du RFC : Décembre 2015

https://www.bortzmeyer.org/7719.html

Comme beaucoup de protocoles très utilisés sur l'Internet, le DNS est ancien. Très ancien (la première norme, le RFC 882 [1], date de 1983). Les normes techniques vieillissent, avec l'expérience, on comprend mieux, on change les points de vue et donc, pour la plupart des protocoles, on se lance de temps en temps dans une révision de la norme. Mais le DNS est une exception : la norme actuelle reste fondée sur des textes de 1987, les RFC 1034 et RFC 1035. Ces documents ne sont plus à jour, modifiés qu'ils ont été par des dizaines d'autres RFC. Bref, aujourd'hui, pour comprendre le DNS, il faut s'apprêter à lire de nombreux documents. En attendant qu'un courageux et charismatique participant à l'IETF se lance dans la tâche herculéenne de faire un jeu de documents propre et à jour, ce nouveau RFC 7719 se limitait à une ambition plus modeste : fixer la terminologie du DNS. Depuis, il a été remplacé par un document plus récent, le RFC 8499.

En effet, chacun peut constater que les discussions portant sur le DNS sont difficiles : on manque de terminologie standard, et celle des RFC officielles ne suffit pas, loin de là. Souvent, le DNS a tellement changé que le RFC officiel est même trompeur : les mots ne veulent plus dire la même chose. D'autres protocoles ont connu des mises à jour de la norme. Cela a été le cas de SMTP, passé successivement du RFC 772 à l'actuel RFC 5321, en passant par plusieurs révisions successives. Ou de XMPP, qui a vu sa norme originale mise à jour dans le RFC 6120. Et bien sûr de HTTP, qui a connu récemment un toilettage complet <https://www.bortzmeyer.org/http-11-reecrit.html>. Mais personne n'a encore osé faire pareil pour le DNS. Au moins, ce nouveau RFC 7719 traite l'un des problèmes les plus criants, celui du vocabulaire. Le RFC est évidemment en anglais, les traductions proposées dans cet article, et qui n'ont pas de valeur « officielle » sont de moi seul.

Notre RFC 7719 rassemble donc des définitions pour des termes qui étaient parfois précisément définis dans d'autres RFC (et il fournit alors un lien vers ce RFC original), mais aussi qui n'étaient

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc882.txt

définis qu'approximativement ou parfois qui n'étaient pas définis du tout (et ce RFC fournit alors cette définition). Du fait du flou de certains RFC anciens, et des changements qui ont eu lieu depuis, certaines définitions sont différentes de l'original. Le document a fait l'objet d'un consensus relatif auprès des experts DNS mais quelques termes restent délicats. Notez aussi que d'autres organisations définissent des termes liés au DNS par exemple le W3C a sa propre définition de « domaine » <https://specs.webplatform.org/url/webspecs/develop/#concept-domain>.

Ironiquement, un des termes les plus difficiles à définir est « DNS » lui-même. D'accord, c'est le sigle de "*Domain Name System*" mais ça veut dire quoi ? « DNS » peut désigner le schéma de nommage (les noms de domaine comme `signal.eu.org`, leur syntaxe, leurs contraintes), la base de données répartie (et faiblement cohérente) qui associe à ces noms des informations (comme des certificats, des adresses IP, etc), ou le protocole requête/réponse (utilisant le port 53) qui permet d'interroger cette base. Parfois, « DNS » désigne uniquement le protocole, parfois, c'est une combinaison des trois éléments indiqués plus haut (personnellement, quand j'utilise « DNS », cela désigne uniquement le protocole).

Bon, et ces définitions rigoureuses et qui vont mettre fin aux discussions, ça vient ? Chaque section du RFC correspond à une catégorie particulière. D'abord, en section 2, les noms eux-même, ces fameux noms de domaine :

— **Nom de domaine** ("*domain name*") : on reprend la définition du RFC 1034, section 3.1, dans une structure arborescente, le nom est une suite de composants ("*labels*"), la racine de l'arbre étant à la fin. Aucune restriction n'est imposée dans ces composants (tant pis pour la légende comme quoi le DNS serait limité à ASCII). Comme on peut représenter les noms de domaine sous forme d'un arbre, ou bien sous forme texte (`www.madmoizelle.com`), le vocabulaire s'en ressent. Par exemple, on va dire que `com` est « au-dessus de `madmoizelle.com` » (vision arborescente) ou bien « à la fin de `www.madmoizelle.com` » (vision texte). Notez aussi que la représentation des noms de domaine dans les paquets IP n'a rien à voir avec leur représentation texte (par exemple, les points n'apparaissent pas).
— **FQDN** ("*Fully Qualified Domain Name*", nom de domaine complet) : apparu dans le RFC 819, ce terme désigne un nom de domaine où tous les composants sont cités (par exemple, `ldap.potamochère.fr.` est un FQDN alors que `ldap` tout court ne l'est pas). En toute rigueur, un FQDN devrait toujours s'écrire avec un point à la fin (pour représenter la racine) mais ce n'est pas toujours le cas.
— **Composant** ("*label*") : un nœud de l'arbre des noms de domaine, dans la chaîne qui compose un FQDN. Dans `www.laquadrature.net`, il y a trois composants, `www`, `laquadrature` et `net`.
— **Nom de machine** ("*host name*") : ce n'est pas la même chose qu'un nom de domaine <https://www.bortzmeyer.org/host-vs-domain.html>. Utilisé dans de nombreux RFC (par exemple RFC 952) mais jamais défini, un nom de machine est un nom de domaine avec une syntaxe plus restrictive : uniquement des lettres, chiffres, points et le tiret. Ainsi, `brienne.tarth.got.example` peut être un nom de machine mais `www.&#$%?.example` ne peut pas l'être. Le terme de « nom de machine » est parfois aussi utilisé pour parler du premier composant d'un nom de domaine (`brienne` dans `brienne.tarth.got.example`).
— **TLD** ("*Top Level Domain*", domaine de premier niveau ou domaine de tête) : le dernier composant d'un nom de domaine, celui juste avant (ou juste en dessous) de la racine. Ainsi, `fr` ou `name` sont des TLD. N'utilisez surtout pas le terme erroné d'« extension » <https://www.bortzmeyer.org/parties-nom-domaine.html>.
— **IDN** ("*Internationalized Domain Name*", nom de domaine internationalisé) : un nom de domaine en Unicode, normalisé dans le RFC 5890.
— **Sous-domaine** ("*subdomain*") : domaine situé sous un autre, dans l'arbre des noms de domaines. Sous forme texte, un domaine est sous-domaine d'un autre si cet autre est un suffixe. Ainsi, `www.cl.cam.ac.uk` est un sous-domaine de `cl.cam.ac.uk`, qui est un sous-domaine de `cam.ac.uk` et ainsi de suite, jusqu'à la racine, le seul domaine à n'être sous-domaine de personne.
— **Alias** ("*alias*") : attention, il y a un piège. Le DNS permet à un nom d'être un alias d'un autre, avec le type d'enregistrement CNAME (voir la définition suivante). L'alias est le terme de gauche de l'enregistrement CNAME. Ainsi, si on met dans un fichier de zone `vader IN CNAME anakin`, l'alias est `vader` (et c'est une erreur de dire que c'est « le CNAME »).

— **CNAME** ("*Canonical Name*", nom canonique, le « vrai » nom) : le membre droit dans l'enregistrement CNAME. Dans l'exemple de la définition précédente, `anakin` est le CNAME, le « nom canonique ».
— **Suffixe public** ("*public suffix*") : ce terme pas très officiel est parfois utilisé pour désigner un suffixe de noms de domaine qui est contrôlé par un registre public (au sens où il accepte des enregistrements du public). Le terme est ancien mais est apparu pour la première fois dans un RFC avec le RFC 6265, section 5.3. `com`, `co.uk` et `eu.org` sont des suffixes publics. Rien dans la syntaxe du nom n'indique qu'un nom de domaine est un suffixe public, puisque ce statut ne dépend que d'une politique d'enregistrement (qui peut changer).

Fini avec les noms, passons à l'en-tête des messages DNS et aux codes qu'il peut contenir. Cet en-tête est défini dans le RFC 1035, section 4.1. Il donne des noms aux champs mais pas forcément aux codes. Ainsi, le code de réponse 3 indiquant qu'un domaine demandé n'existe pas est juste décrit comme "*name error*" et n'a reçu son mnémonique de NXDOMAIN ("*No Such Domain*") que plus tard. Notre RFC définit également :

— **NODATA** : un mnémonique pour une réponse où le nom de domaine demandé existe bien mais ne contient aucun enregistrement du type souhaité. Le code de retour est 0, NOERROR, et le nombre de réponses (ANCOUNT pour "*Answer Count*") est nul.
— **Réponse négative** ("*negative answer*") : le terme recouvre deux choses, une réponse disant que le nom de domaine demandé n'existe pas, ou bien une réponse indiquant que le serveur ne peut pas répondre (code de retour SERVFAIL ou REFUSED). Voir le RFC 2308.
— **Renvoi** ("*referral*") : le DNS étant décentralisé, il arrive qu'on pose une question à un serveur qui ne fait pas autorité pour le domaine demandé, mais qui sait vous renvoyer à un serveur plus proche. Ces renvois sont indiqués dans la section "*Authority*" d'une réponse.

Voici un renvoi depuis la racine vers `.fr` :

```
% dig @l.root-servers.net A blog.imirhil.fr
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16572
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 5, ADDITIONAL: 11
...
;; AUTHORITY SECTION:
fr. 172800 IN NS d.ext.nic.fr.
fr. 172800 IN NS d.nic.fr.
fr. 172800 IN NS e.ext.nic.fr.
fr. 172800 IN NS f.ext.nic.fr.
fr. 172800 IN NS g.ext.nic.fr.
```

Passons maintenant aux enregistrements DNS, stockés dans cette base de données répartie (section 4 du RFC) :

— **RR** ("*Resource Record*", un enregistrement DNS).
— **Ensemble d'enregistrements** (RRset pour "*Resource Record set*") : un ensemble d'enregistrements ayant le même nom (la même clé d'accès à la base), la même classe, le même type et le même TTL. Cette notion avait été introduite par le RFC 2181. Notez que la définition originale, reprise par notre RFC, parle malheureusement de "*label*" dans un sens incorrect. La terminologie du DNS est vraiment compliquée !
— **EDNS** ("*Extension for DNS*", également appelé EDNS0) : normalisé dans le RFC 6891, EDNS permet d'étendre l'en-tête du DNS, en spécifiant de nouvelles options, en faisant sauter l'antique limitation de taille de réponses à 512 octets, etc.
— **OPT** (pour "*Option*") : une astuce d'EDNS pour encoder les informations de l'en-tête étendu. C'est un enregistrement DNS un peu spécial, défini dans le RFC 6891, section 6.1.1.
— **Propriétaire** ("*owner*" ou "*owner name*") : le nom de domaine d'un enregistrement. Ce terme est très rarement utilisé, même par les experts.

— **Champs du SOA** (*"SOA field names"*) : ces noms des enregistrements SOA (comme MNAME ou RNAME) sont peu connus et peu utilisés (RFC 1035, section 3.3.13). Notez que la sémantique du champ MINIMUM a complètement changé avec le RFC 2308.
— **TTL** (*"Time To Live"*) : la durée de vie maximale d'un enregistrement dans les caches des résolveurs. C'est un entier non signé (même si le RFC 1035 dit le contraire), en secondes.

Voici un ensemble d'enregistrements (*"RRset"*), comptant ici deux enregistrements :

```
rue89.com. 600 IN MX 50 mx2.typhon.net.
rue89.com. 600 IN MX 10 mx1.typhon.net.
```

Et voici un pseudo-enregistrement OPT, tel qu'affiché par dig, avec une indication de la taille maximale et l'option *"client subnet"* (RFC pas encore publié) :

```
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 512
; CLIENT-SUBNET: 13.176.144.0/24/0
```

Ensuite, un sujet chaud où le vocabulaire est particulièrement peu défini, et très mal utilisé (voir les forums grand public sur le DNS où les discussions prennent un temps fou car les gens utilisent mal les mots) : les différents types de serveurs et clients DNS (section 5) :

— **Résolveur** (*"resolver"*) : un client DNS qui va produire une réponse finale (pas juste un renvoi, pas juste une information ponctuelle comme le fait dig). Il existe plusieurs types de résolveurs (voir ci-dessous) et, en pratique, quand on dit « résolveur » tout court, c'est en général un « résolveur complet ».
— **Résolveur minimum** (*"stub resolver"*) : un résolveur qui ne sait pas suivre les renvois et qui dépend donc d'un ou de plusieurs résolveurs complets pour faire son travail. Ce type est défini dans la section 6.1.3.1 du RFC 1123. C'est ce résolveur minimum qu'appelent les applications lorsqu'elles font un `getaddrinfo()` ou `getnameinfo()`. Sur Unix, le résolveur minimum fait en général partie de la libc et trouve l'adresse du ou des résolveurs complets dans `/etc/resolv.conf`.
— **Résolveur complet** (*"full resolver"*) : un résolveur qui sait suivre les renvois et donc fournir un service de résolution complet. Des logiciels comme Unbound ou PowerDNS Resolver assurent cette fonction. Le résolveur complet est en général situé chez le FAI ou dans le réseau local de l'organisation où on travaille, mais il peut aussi être sur la machine locale <`https://www.bortzmeyer.org/son-propre-resolveur-dns.html`>. On le nomme aussi « résolveur récursif ».
— **Serveur faisant autorité** (*"authoritative server"* et traduire ce terme par « serveur autoritaire » montre une grande ignorance de l'anglais, un adjudant est autoritaire, un serveur DNS fait autorité) : un serveur DNS qui connait une partie des données du DNS (il « fait autorité » pour une ou plusieurs zones) et peut donc y répondre (RFC 2182, section 2). Ainsi, au moment de l'écriture de cet article, `f.root-servers.net` fait autorité pour la racine, `d.nic.fr` fait autorité pour `pm`, etc. Des logiciels comme NSD ou Knot assurent cette fonction. Les serveurs faisant autorité sont gérés par divers acteurs, les registres, les hébergeurs DNS (qui sont souvent en même temps bureaux d'enregistrement), mais aussi par M. Michu. La commande `dig NS $ZONE` vous donnera la liste des serveurs faisant autorité pour la zone `$ZONE`.
— Serveur mixte : ce terme n'est **pas** dans ce RFC. Autrefois, il était courant que des serveurs DNS fassent à la fois résolveur et serveur faisant autorité. Cette pratique est fortement déconseillée depuis de nombreuses années (entre autres parce qu'elle complique sérieusement le déboguage, mais aussi pour des raisons de sécurité parce qu'elle mène à du code plus complexe) et n'est donc pas dans ce RFC.

— **Initialisation** (*"priming"*) : le processus par lequel un résolveur complet vérifie l'information sur les serveurs de la racine. Au démarrage, le résolveur ne sait rien. Pour pouvoir commencer la résolution de noms, il doit demander aux serveurs de la racine. Il a donc dans sa configuration leur liste. Mais ces configurations ne sont pas forcément mises à jour souvent. La liste peut donc être trop vieille. La première chose que fait un résolveur est donc d'envoyer une requête « `NS .` » à un des serveurs de sa liste. Ainsi, tant qu'un moins un des serveurs de la vieille liste répond, le résolveur est sûr d'apprendre la liste actuelle.
— **Mémorisation des négations** (*"negative caching"*, ou « cache négatif ») : mémoriser le fait qu'il n'y a pas eu de réponse, ou bien une réponse disant qu'un nom de domaine n'existe pas.
— **Serveur primaire** (*"primary server"* mais on dit aussi *"master server"*) : un serveur faisant autorité qui a accès à la source des données (fichier de zone, base de données, etc). Attention, il peut y avoir plusieurs serveurs primaires (autrefois, ce n'était pas clair et beaucoup de gens croient qu'il y a un serveur primaire, et qu'il est indiqué dans l'enregistrement SOA). Attention bis, le terme ne s'applique qu'à des serveurs faisant autorité, l'utiliser pour les résolveurs (« on met en premier dans `/etc/resolv.conf` le serveur primaire ») n'a pas de sens.
— **Serveur secondaire** (*"secondary server"* mais on dit aussi « serveur esclave », *"slave server"*) : un serveur faisant autorité qui n'est pas la source des données, qui les a prises d'un serveur primaire (dit aussi serveur maître), via un transfert de zone (RFC 5936).
— **Serveur furtif** (*"stealth server"*) : un serveur faisant autorité mais qui n'apparait pas dans l'ensemble des enregistrements NS. (Définition du RFC 1996, section 2.1.)
— **Maître caché** (*"hidden master"*) : un serveur primaire qui n'est pas annoncé publiquement (et n'est donc accessible qu'aux secondaires). C'est notamment utile avec DNSSEC : s'il signe, et donc a une copie de la clé privée, il vaut mieux qu'il ne soit pas accessible de tout l'Internet (RFC 6781, section 3.4.3).
— **Transmission** (*"forwarding"*) : le fait, pour un résolveur, de faire suivre les requêtes à un autre résolveur (probablement mieux connecté et ayant un cache partagé plus grand). On distingue parfois (mais ce n'est pas forcément clair, même dans le RFC 5625) la transmission, où on émet une nouvelle requête, du simple relayage de requêtes sans modification.
— **Transmetteur** (*"forwarder"*) : le terme est confus (et a suscité plein de débats dans le groupe de travail DNSOP lors de la mise au point de ce RFC). Il désigne parfois la machine qui transmet une requête et parfois celle vers laquelle on transmet (c'est dans ce sens qu'il est utilisé dans la configuration de BIND, avec la directive `forwarders`).
— **Résolveur politique** (*"policy-implementing resolver"*) : un résolveur qui modifie les réponses reçues, selon sa politique. On dit aussi, moins diplomatiquement, un « résolveur menteur ». C'est ce que permet, par exemple, le système RPZ <`https://www.bortzmeyer.org/rpz-faire-mentir-resolveur-dn` `html`>. Sur l'utilisation de ces « résolveurs politiques » pour mettre en œuvre la censure, voir entre autres mon article aux RIPE Labs <`https://labs.ripe.net/Members/stephane_` `bortzmeyer/dns-censorship-dns-lies-seen-by-atlas-probes`>. Notez que le résolveur politique a pu être choisi librement par l'utilisateur (par exemple comme élément d'une solution de blocage des publicités) ou bien qu'il lui a été imposé.
— **Résolveur ouvert** (*"open resolver"*) : un résolveur qui accepte des requêtes DNS depuis tout l'Internet. C'est une mauvaise pratique <`https://www.bortzmeyer.org/fermer-les-recursifs-ouverts.` `html`> (cf. RFC 5358) et la plupart de ces résolveurs ouverts sont des erreurs de configuration. Quand ils sont délibérement ouverts, comme Google Public DNS <`https://www.bortzmeyer.` `org/google-dns.html`>, on parle plutôt de **résolveurs publics**.
— **Collecte DNS passive** (*"passive DNS"*) : désigne les systèmes qui écoutent le trafic DNS, et stockent tout ou partie des messages DNS échangés. Le cas le plus courant est celui où le système de collecte ne garde que les réponses (ignorant donc les adresses IP des clients et serveurs), afin de constituer une base historique du contenu du DNS (c'est ce que font DNSDB <`https://www.` `bortzmeyer.org/dnsdb.html`> ou le système de PassiveDNS.cn <`https://www.bortzmeyer.` `org/passivedns-cn.html`>).
— **Anycast** : le fait d'avoir un service en plusieurs sites physiques, chacun annonçant la même adresse IP de service (RFC 4786). Cela résiste mieux à la charge, et permet davantage de résilience en cas d'attaque par déni de service. Les serveurs de la racine, ceux des « grands » TLD, et ceux des importants hébergeurs DNS sont ainsi « *anycastés* ».

Voici, vu par tcpdump, un exemple d'**initialisation** d'un résolveur BIND utilisant la racineYeti <`https:`

```
//www.afnic.fr/fr/ressources/blog/le-projet-yeti-d-experimentation-d-une-racine-dns-2
html> :


15:07:36.736031 IP6 2a01:e35:8bd9:8bb0:21e:8cff:fe76:29b6.35721 > 2001:6d0:6d06::53.53: \
       21476% [1au] NS? . (28)
15:07:36.801982 IP6 2001:6d0:6d06::53.53 > 2a01:e35:8bd9:8bb0:21e:8cff:fe76:29b6.35721: \
       21476*- 16/0/1 NS yeti-ns.tisf.net., NS yeti-ns.lab.nic.cl., NS yeti-ns.wide.ad.jp., NS yeti.ipv6.err
```

La question était « `NS .` » (quels sont les serveurs de la racine) et la réponse contenait les noms des seize serveurs racine.

Voici aussi des exemples de résultats avec un résolveur ou bien avec un serveur faisant autorité. Si je demande à un serveur faisant autorité (ici, un serveur racine), avec mon client DNS qui, par défaut, demande un service récursif (”*flag*” RD, ”*Recursion Desired*”) :

```
% dig @2001:620:0:ff::29 AAAA www.iab.org
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 54197
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 9, ADDITIONAL: 13
;; WARNING: recursion requested but not available
...
;; AUTHORITY SECTION:
org. 172800 IN NS b0.org.afilias-nst.org.
...
```

C’est pour cela que dig affiche ”*WARNING : recursion requested but not available*”. Notez aussi que le serveur, ne faisant autorité que pour la racine, n’a pas donné la réponse mais juste un **renvoi** aux serveurs d’Afilias. Maintenant, interrogeons un serveur récursif (le service de résolveur public Yandex DNS <`https://dns.yandex.com/`>) :

```
% dig @2a02:6b8::feed:0ff AAAA www.iab.org
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63304
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
...
;; ANSWER SECTION:
www.iab.org. 1800 IN AAAA 2001:1900:3001:11::2c
```

Cette fois, j’ai obtenu une réponse, et avec le ”*flag*” RA, ”*Recursion Available*”. Si je pose une question sans le ”*flag*” RD (”*Recursion Desired*”, avec l’option `+norecurse` de dig) :

```
% dig +norecurse @2a02:6b8::feed:0ff AAAA www.gq.com
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 59438
;; flags: qr ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
...
;; ANSWER SECTION:
www.gq.com. 293 IN CNAME condenast.map.fastly.net.
```

J'ai obtenu ici une réponse car l'information était déjà dans le cache (la mémoire) de Yandex DNS (on le voit au TTL, qui n'est pas un chiffre rond, il a été décrémenté du temps passé dans le cache). Si l'information n'est pas dans le cache :

```
% dig +norecurse @2a02:6b8::feed:0ff AAAA blog.keltia.net
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19893
;; flags: qr ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0
...
```

Je n'obtiens alors pas de réponse (ANSWER : 0). Si je demande au serveur faisant autorité pour cette zone :

```
% dig +norecurse @2a01:e0d:1:3:58bf:fa61:0:1 AAAA blog.keltia.net
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62908
;; flags: qr aa; QUERY: 1, ANSWER: 2, AUTHORITY: 6, ADDITIONAL: 15
...
;; ANSWER SECTION:
blog.keltia.net. 86400 IN AAAA 2a01:240:fe5c:1::2
...
```

J'ai évidemment une réponse et, comme il s'agit d'un serveur faisant autorité, elle porte le "*flag*" AA ("*Authoritative Answer*", qu'un résolveur ne mettrait pas). Notez aussi le TTL qui est un chiffre rond (et qui ne change pas si on rejoue la commande).

Passons maintenant à un concept relativement peu connu, celui de **zones**, et le vocabulaire associé :

— **Zone** : un groupe de domaines contigus et gérés ensemble, par le même ensemble de serveurs de noms (ma définition, celle du RFC étant très abstraite). Beaucoup de gens croient que tout domaine est une zone, mais c'est faux. Ainsi, au moment de la publication de ce RFC, `gouv.fr` n'est pas une zone séparée, il est dans la même zone que `fr` (cela se teste facilement : `gouv.fr` n'a pas d'enregistrement NS ou de SOA).
— **Parent** : le domaine « du dessus ». Ainsi, le parent de `wikipedia.org` est `org`.
— **Apex** : le sommet d'une zone, là où on trouve les enregistrements NS et SOA. Si la zone ne comprend qu'un domaine, l'apex est ce domaine. Si la zone est plus complexe, l'apex est le domaine le plus court.
— **Coupure de zone** ("*zone cut*") : l'endoit où on passe d'une zone à l'autre. Au-dessus de la coupure, la zone parente, en dessous, la zone fille.
— **Délégation** ("*delegation*") : un concept évidemment central dans le DNS, qui est un système décentralisé. En ajoutant un ensemble d'enregistrements NS pointant vers les serveurs de la zone fille, une zone parente **délègue** une partie de l'arbre des noms de domaine à une aure entité. L'endroit où se fait la délégation est donc une coupure de zone.
— **Colle** ("*glue records*") : lorsqu'une zone est déléguée à des serveurs dont le nom est dans la zone fille, la résolution DNS se heurte à un problème d'œuf et de poule. Pour trouver l'adresse de `ns1.mazone.example`, le résolveur doit passer par les serveurs de `mazone.example`, qui est déléguée à `ns1.mazone.example` et ainsi de suite... On rompt ce cercle vicieux en ajoutant, dans la zone parente, des données qui ne font pas autorité sur les adresses de ces serveurs (RFC 1034, section 4.2.1). Il faut donc bien veiller à les garder synchrones avec la zone fille. (Tanguy Ortolo me suggère d'utiliser « enregistrement de raccord » plutôt que « colle ». Cela décrit bien leur rôle, en effet.)

— **Dans le bailliage** (*"in bailiwick"*) : terme absent des textes DNS originaux et qui peut désigner plusieurs choses. Il est (très rarement, selon mon expérience) parfois utilisé pour parler d'un serveur de noms dont le nom est dans la zone servie (et qui nécessite donc de la colle, voir la définition précédente), mais le sens le plus courant désigne des données pour lesquelles le serveur qui a répondu fait autorité, soit pour la zone, soit pour un ancêtre de cette zone. L'idée est qu'il est normal dans la réponse d'un serveur de trouver des données situées dans le bailliage et, par contre, que les données hors-bailliages sont suspectes (elles peuvent être là suite à une tentative d'empoisonnement DNS). Un résolveur DNS prudent ignorera donc les données hors-bailliage.
— **ENT** (*"Empty Non-Terminal"* pour nœud non-feuille mais vide) : un domaine qui n'a pas d'enregistrements mais a des sous-domaines. C'est fréquent, par exemple, sous `ip6.arpa` ou sous les domaines très longs de certains CDN.
— **Zone de délégation** (*"delegation-centric zone"*) : zone composée essentiellement de délégations vers d'autres zones. C'est typiquement le cas des TLD et autres suffixes publics.
— **Joker** (*"wildcard"*) : une source de confusion considérable depuis les débuts du DNS. Si on pouvait refaire le DNS en partant de zéro, ces jokers seraient la première chose à supprimer. Pour les résumer, le nom `*` dans une zone déclenche la synthèse automatique de réponses pour les noms qui n'existent pas dans la zone. Si la zone `foo.example` contient `bar.foo.example` et `*.foo.example`, une requête pour `thing.foo.example` renverra le contenu de l'enregistrement avec le joker, une requête pour `bar.foo.example` donnera les données de `bar.foo.example`.
— **Changement rapide** (*"fast flux"*) : une technique notamment utilisée par les botnets pour mettre leur centre de commande à l'abri des filtrages ou destructions. Elle consiste à avoir des enregistrements d'adresses IP avec des TTL très courts et à en changer fréquemment.

Voyons ici la colle retournée par un serveur faisant autorité (en l'occurrence un serveur de `.net`) :

```
% dig @a.gtld-servers.net AAAA labs.ripe.net
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18272
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 8, ADDITIONAL: 9
...
;; AUTHORITY SECTION:
ripe.net. 172800 IN NS ns3.nic.fr.
ripe.net. 172800 IN NS sec1.apnic.net.
ripe.net. 172800 IN NS sec3.apnic.net.
ripe.net. 172800 IN NS tinnie.arin.net.
ripe.net. 172800 IN NS sns-pb.isc.org.
ripe.net. 172800 IN NS pri.authdns.ripe.net.
...
;; ADDITIONAL SECTION:
sec1.apnic.net. 172800 IN AAAA 2001:dc0:2001:a:4608::59
sec1.apnic.net. 172800 IN A 202.12.29.59
sec3.apnic.net. 172800 IN AAAA 2001:dc0:1:0:4777::140
sec3.apnic.net. 172800 IN A 202.12.28.140
tinnie.arin.net. 172800 IN A 199.212.0.53
tinnie.arin.net. 172800 IN AAAA 2001:500:13::c7d4:35
pri.authdns.ripe.net. 172800 IN A 193.0.9.5
pri.authdns.ripe.net. 172800 IN AAAA 2001:67c:e0::5
```

On notera :

— La section *"ANSWER"* est vide, c'est un **renvoi**.
— Le serveur indique la colle pour `pri.authdns.ripe.net` : ce serveur étant dans la zone qu'il sert, sans son adresse IP, on ne pourrait jamais le joindre.
— Le serveur envoie également les adresses IP d'autres machines comme `sec1.apnic.net`. Ce n'est pas strictement indispensable (on pourrait l'obtenir par une nouvelle requête), juste une optimisation.

— Les adresses de `ns3.nic.fr` et `sns-pb.isc.org` ne sont pas renvoyées. Le serveur ne les connait probablement pas et, de toute façon, elles seraient hors-bailliage.

Voyons maintenant, un **ENT**, `gouv.fr` :

```
% dig @d.nic.fr ANY gouv.fr
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 42219
;; flags: qr aa rd; QUERY: 1, ANSWER: 0, AUTHORITY: 2, ADDITIONAL: 1
```

Le serveur fait bien autorité pour ce domaine ("*flag*" AA dans la réponse), le domaine existe (autrement, le "*status*" aurait été NXDOMAIN, pas NOERROR) mais il n'y a aucun enregistrement (ANSWER : 0).

Allez courage, ne faiblissons pas, il reste encore la question de l'enregistrement des noms de domaine (section 7) :

— **Registre** ("*registry*") : l'organisation ou la personne qui gère les délégations d'une zone. Le terme est parfois employé uniquement pour les gérants de grandes zones de délégation, mais ce n'est pas obligatoire. Je suis à moi tout seul le registre de `bortzmeyer.org` :-) C'est le registre qui décide de la politique d'enregistrement, qui peut être très variable selon les zones (sauf dans celles contrôlées par l'ICANN, où une certaine uniformité est imposée). Mes lecteurs français noteront que, comme le terme « registre » est court et largement utilisé, le gouvernement a inventé un nouveau mot <`http://www.legifrance.gouv.fr/affichCodeArticle.do?cidTexte=LEGITEXT000006070987&idArticle=LEGIARTI000006465470&dateTexte=&categorieLien=cid`>, plus long et jamais vu auparavant, « office d'enregistrement ».

— **Titulaire** ("*registrant*", parfois "*holder*") : la personne ou l'organisation qui a enregistré un nom de domaine.

— **Bureau d'enregistrement** ("*registrar*") : dans le modèle RRR ("*Registry-Registrar-Registrant*", mais ce modèle n'est pas obligatoire et pas universel), c'est un intermédiaire entre le titulaire et le registre.

— **Hébergeur DNS** ("*DNS operator*") : la personne ou l'organisation qui gère les serveurs DNS. Cela peut être le titulaire, son bureau d'enregistrement, ou encore un acteur spécialisé dans cette gestion.

— **EPP** ("*Extensible Provisioning Protocol*") : normalisé dans le RFC 5730, c'est le protocole standard entre bureau d'enregistrement et registre. Ce protocole n'a pas de lien avec le DNS, et tous les registres ne l'utilisent pas.

— **Whois** (nommé d'après la question "*Who Is ?*") : un protocole réseau, sans lien avec le DNS, normalisé dans le RFC 3912. Il permet d'interroger les bases de données du registre pour trouver les informations qui ne sont pas dans le DNS (comme le nom du titulaire, et des moyens de le contacter). Les termes de « base Whois » ou de « données Whois » sont parfois utilisés mais ils sont erronés puisque les mêmes bases peuvent être interrogées par d'autres protocoles que Whois, comme RDAP (voir définition suivante).

— **RDAP** ("*Registration Data Access Protocol*") : un protocole concurrent de Whois <`https://www.bortzmeyer.org/weirds-rdap.html`>, mais plus moderne.

Enfin, pour terminer, la section 8 de notre RFC couvre DNSSEC. Pas grand'chose de nouveau ici, DNSSEC étant plus récent et donc mieux défini.

Je rappelle que ce RFC n'est plus le document final, ce rôle est désormais tenu par son successeur, le RFC 8499.