Le service DNS

Table des matières

Table des illustrations

Présentation des services DNS

On utilise tous les jours sur nos ordinateurs, tablettes ou smartphones des applications qui se connectent sur l'internet. Pour accéder à ces ressources, l'homme emploie des noms permettant d'identifier facilement les machines. Mais les systèmes informatiques ont besoin d'adresses TCP/IP pour pouvoir s'y connecter.

Comme il n'est pas possible de se rappeler des adresses comme 157.26.190.200, il est nécessaire de disposer d'un annuaire permettant la transformation du nom, facile à retenir pour l'homme, en adresse de 32 bits.

Il ne s'agit donc pas d'un problème technique, Internet fonctionne très bien avec des adresses IP, mais d'un problème de nommage pour permettre un accès simplifié à Internet pour nous tous. Ce système de nom s'appelle le Domain Name System (DNS).

Nous pouvons d'ailleurs faire l'analogie avec le bottin de téléphone. En effet, il est plus facile de chercher un numéro d'un correspondant à l'aide de son nom dans l'annuaire que de retenir par cœur tous les numéros.

Figure 1:annuaire de téléphone

Téléphonie :

Annuaire	Lieu	Nom	Numéro
Fribourg	Rossens	EvoLink SA	0264258070

Informatique :

TLD	Domaine	Nom	Adresse TCP/IP
.ch	evolink	www	93.88.240.82
.ch	evolink	evo-fw002	85.90.1.66

Le DNS est donc un protocole indispensable au fonctionnement d'Internet, mais non pas d'un point de vue technique, mais essentiellement d'un point de vue de son utilisation. Il n'est pas concevable d'utiliser des adresses TCP/IP en lieu et place des noms des sites web pour aller sur Internet. Se rappeler de 93.88.240.82 est compliqué mais quand vous surfez sur des dizaines de sites par jour, il y aura trop d'adresses à retenir.

Objectifs essentiels :

Recherche d'une adresse TCP/IP en fonction du nom : **@TCP/IP = f(nom)**

Recherche d'un nom en fonction d'une adresse TCP/IP : **nom = f(@TCP/IP)**

Exemple de recherche directe et inversée avec la commande nslookup :

```
Microsoft Windows [version 10.0.19041.572]
(c) 2020 Microsoft Corporation. Tous droits réservés.

C:\Users\Nicolas Borowy>nslookup evo-fw002.evolink.ch
Serveur :   myrouter.local
Address:  192.168.9.1

Réponse ne faisant pas autorité :
Nom :    evo-fw002.evolink.ch
Address:  85.90.1.66


C:\Users\Nicolas Borowy>nslookup 85.90.1.66
Serveur :   myrouter.local
Address:  192.168.9.1

Nom :    66.1.90.85.reverse.netplusfr.net
Address:  85.90.1.66


C:\Users\Nicolas Borowy>nslookup 93.88.240.82
Serveur :   myrouter.local
Address:  192.168.9.1

Nom :    h2web1.infomaniak.ch
Address:  93.88.240.82


C:\Users\Nicolas Borowy>_
```

Un arbre avec des branches

Le système DNS, vous l'utilisez tous les jours quand vous naviguez sur Internet. Lorsque vous voulez accéder à l'hôte « www.evolink.ch », le système DNS se charge de convertir (on parle de résolution) le nom du site web demandé en adresse IP.

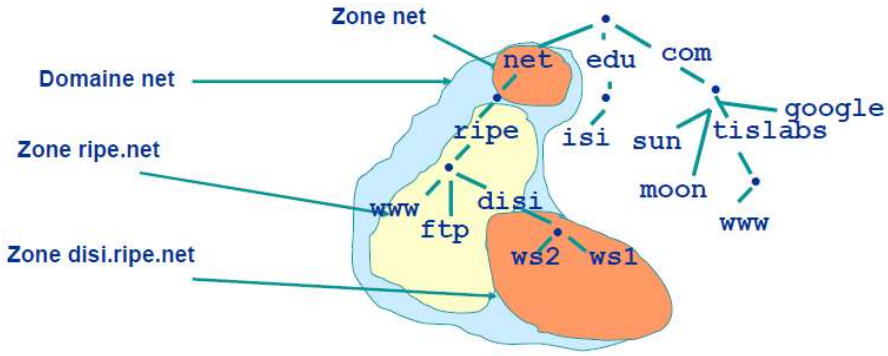


Figure 2: arbre avec des branches (Illustration : Alain Patrick, AINA)

Un nom d'hôte se décompose en plusieurs parties. Prenons l'exemple suivant :

ws1.disi.ripe.net

Chacune des parties est séparée par un point. Pour revenir à la référence de l'arbre, on commence à lire ce nom par la droite. Ainsi, on trouve :

- **.net :** Le Top Level Domain (TLD). Il existe des TLD nationaux (ch, fr, it, de, es, etc.) ou ccTLD (country code TLD), des TLD génériques (com, org, net, biz, etc.) ou gTLD, et des TLD d'infrastructure (in-addr.arpa) ou infra TLD.
- **.ripe :** Le domaine. Celui-ci peut être acheté, ou plutôt loué par un paiement annuel auprès d'un registrar affilié au TLD.
- **.disi :** Le sous-domaine. Il est défini par le propriétaire du domaine. On peut ajouter autant de sous-domaine que l'on souhaite, mais cela peut conduire à avoir un nom de domaine relativement long.
- **ws1 :** le nom d'hôte. Il désigne la machine.

La séquence complète « **ws1.disi.rip.net.** » s'appelle un FQDN (Fully Qualified Domain Name) Attention, il n'est pas possible d'avoir un nom d'hôte complètement défini plus long que 255 caractères. Il ne peut pas non plus de dépasser 63 caractères par partie. Enfin, ce nom est unique dans le système DNS.

Par convention, un FQDN se finit par un **point** (.), car au-dessus des TLD il y a la racine du DNS, tout en haut de l'arbre. Ce point disparaît lorsque vous utilisez les noms d'hôte avec votre navigateur (URL), mais il est très important lors de la configuration des fichiers des serveurs DNS.

Au niveau DNS, « **ws1.disi.rip.net »** n'est donc pas un FQDN, car il manque le point à la fin. Tout FQDN sur Internet doit obligatoirement se finir par un point car on est sûr qu'il n'y a pas de domaine au-dessus. Il s'agira donc d'une référence absolue par rapport à la racine.

Architectures DNS

Voici une toute petite partie de l'arborescence des noms Internet :

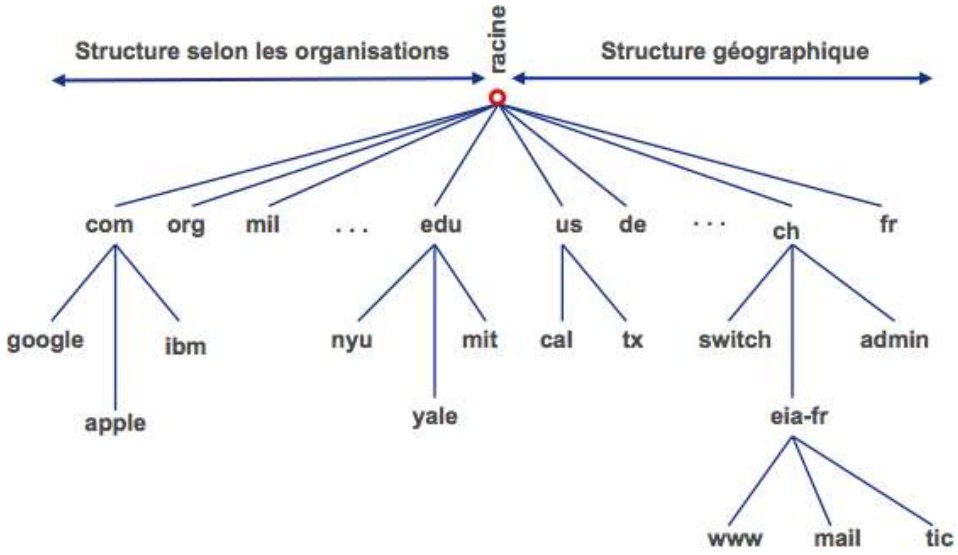


Figure 3: arborescence des noms internet (Illustration : Antoine Delley, EIAFR)

Dans l'architecture du service DNS, chaque label est responsable du niveau directement en dessous et uniquement de celui-ci. La racine est responsable du domaine .com, le .com de google.com et google.com de www.google.com, etc.

Bien entendu, Google veut gérer lui-même le domaine google.com. L'organisme qui gère le domaine .com délègue donc la gestion de ce nom de domaine à Google.

Ainsi, chaque personne qui veut posséder un domaine sur Internet peut l'acheter, mais devra ensuite gérer un serveur DNS pour publier ses adresses. Cependant, la plupart des entreprises qui vendent des noms de domaine (qu'on appelle registrar) proposent de gérer elles-mêmes vos entrées DNS)

Nous savons donc que le DNS est organisé sous forme d'une grosse arborescence, et que chaque partie de l'arborescence peut être gérée par la personne qui la possède.

Il est intéressant de noter que les gTLD et les ccTLD sont gérés par des registrar et que c'est vers ces entités qu'il faut annoncer les entrées de résolution directes (recherche d'une adresse TCP/IP en fonction du nom). Par contre, pour les résolutions inversées, ce sera en général auprès du fournisseur d'accès à l'internet qu'il faudra annoncer les entrées DNS.

Mais comment fait-on pour savoir qui possède telle ou telle partie et où sont stockées les informations que l'on recherche ?

Le mécanisme de résolution

Vous êtes connectés à votre réseau, votre serveur DHCP vous a donné une adresse IP, un masque de sous-réseau et une passerelle par défaut, mais également un serveur DNS. Imaginez que vous entrez www.google.com dans votre navigateur. Lorsque vous entrez ce nom, votre machine doit commencer par le résoudre en une adresse IP. Vous allez donc demander une résolution au serveur DNS que vous avez reçu par le DHCP (Resolver). Celui-ci a deux moyens pour vous fournir la réponse :

- Il connaît lui-même la réponse,
- Il doit la demander à un autre serveur, car il ne la connaît pas.

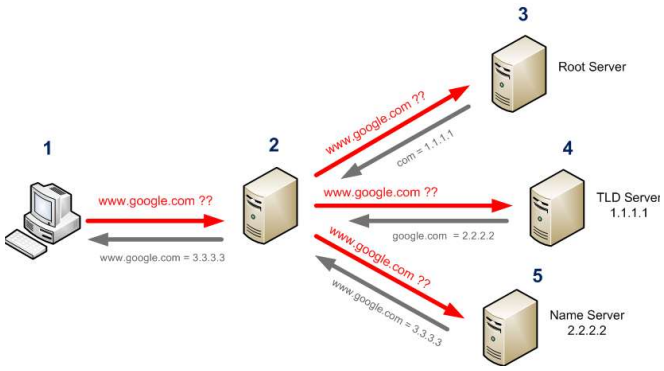


Figure 4: résolution du nom www.google.com

La plupart du temps, votre serveur DNS dispose de peu d'informations et demande à un autre serveur de lui donner la réponse. En effet, chaque serveur DNS étant responsable d'un domaine ou d'un petit nombre de domaines, la résolution consiste à aller chercher la bonne information sur le bon serveur.

Exemple d'une recherche directe

Dans la figure ci-dessus, on cherche le site www.google.com et voilà ce que va faire le serveur DNS qui reçoit la requête du poste client :

- Tout d'abord, il est évident que cette information ne se trouve pas sur notre serveur (2), car ce n'est pas lui qui est en charge du site google.com Pour obtenir cette résolution, notre serveur va procéder de façon rigoureuse et commencer par là où il a le plus de chance d'obtenir l'information, c'est-à-dire au point de départ de notre arborescence.
- Il va demander aux serveurs racine (3) l'adresse IP de www.google.com. Mais comme les serveurs racine ne sont pas responsables de ce domaine, ils vont le rediriger vers un autre serveur qui peut lui donner une information et qui dépend de la racine, le serveur DNS de la zone .com.
- Il demande ensuite au serveur DNS de .com (4) l'adresse IP de www.google.com. Mais comme auparavant, le serveur .com renvoie l'adresse IP du serveur DNS qui dépend de lui, le serveur DNS de la zone google.com.
- Il demande au serveur DNS de google.com (5) l'adresse IP de www.google.com et là, le serveur de google.com connaît l'adresse IP correspondante et peut la renvoyer.

Le serveur (2) dispose enfin de la réponse et peut transmettre la réponse au poste client.

Type de requête

Le type de demande entre le poste client et les serveurs comme illustré ci-dessous est désigné comme une **requête récursive.**

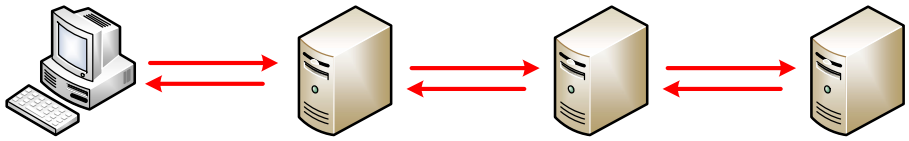


Figure 5: requête récursive

Le type de demande entre les serveurs comme illustré ci-dessous est désigné comme une **requête itérative.**

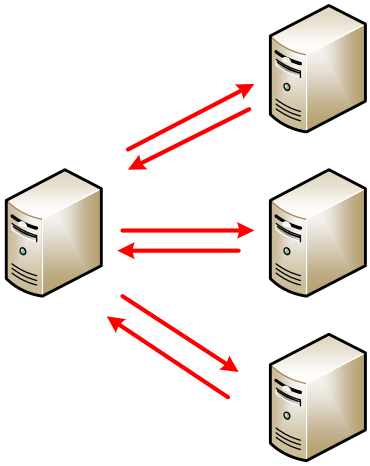


Figure 6: requête itérative

Serveur cache

On dit d'un serveur permettant la résolution d'un nom de domaine sans avoir eu à demander l'information à un autre système, qu'il fait autorité.

Les serveurs DNS utilisent un système de cache pour ne pas avoir à demander une information de manière répétitive, mais ils ne font pas autorité pour autant. En effet, l'information stockée en cache peut ne plus être valide après un certain temps.

Résolution inverse

Il est possible également de chercher un nom en fonction d'une adresse TCP/IP. On parle alors de reverse DNS et de résolution inverse. C'est relativement peu utilisé, mais on retrouve ce type de résolution sur des systèmes antispam ou de contrôle d'accès pour des raisons de sécurité.

Gestion internationale des noms de domaine

Le système de noms de domaine est géré par un organisme américain appelé l'ICANN. Celui-ci dépend directement du Département du Commerce des États-Unis. L'ICANN est responsable de la gestion des 13 serveurs DNS qui gèrent la racine du DNS. Ces 13 serveurs connaissent les adresses IP des serveurs DNS gérant les TLD (les .ch, .fr, .com, org, etc.)

Après plusieurs attaques sur les serveurs racine, on s'est rendu compte de la faiblesse de n'avoir que 13 serveurs et de la menace que cela pouvait représenter pour le fonctionnement d'Internet. L'ICANN a donc mis en place un système qui duplique les 13 serveurs en différents endroits d'Internet. Il y a donc réellement aujourd'hui plusieurs centaines de serveurs racine (environ 120) qui dupliquent les informations des 13 serveurs d'origine. Le mécanisme qui permet cette duplication de serveurs, et notamment d'adresses IP, s'appelle l'anycast, mais il fait appel à des notions réseau avancées que nous n'exposerons pas ici.

L'ICANN autorise la création d'une nouvelle extension, comme le .swiss il y a plusieurs mois ou l'utilisation de caractères non-latins (arabes, chinois, japonais, etc.), il y a quelques années.

L'ICANN délègue ensuite les domaines de premier niveau à divers organismes. Pour l'Europe, c'est le RIPE, qui délègue lui-même à Switch (qui est responsable du domaine .ch). Pour le domaine .com, c'est VeriSign qui s'en occupe. Les labels inférieurs correspondent généralement à des sites ou à des entreprises, et la gestion du nom de domaine leur revient.

Type d'enregistrement

Dans ce fichier de zone (base de données contenant toutes les informations d'un domaine), nous allons indiquer des enregistrements. Il en existe de plusieurs types :

- A : c'est le type le plus courant, il fait correspondre un nom d'hôte à une adresse IPv4,
- AAAA : fait correspondre un nom d'hôte à une adresse IPv6,
- CNAME : permet de créer un alias pointant sur un autre nom d'hôte,
- NS : définit le ou les serveurs DNS du domaine,
- MX : définit le ou les serveurs de mail du domaine,
- PTR : fait correspondre une IP à un nom d'hôte. Il n'est utilisé que dans le cas d'une zone inverse, que nous verrons plus loin,
- SOA : donne les infos de la zone, comme le serveur DNS principal, l'adresse mail de l'administrateur de la zone, le numéro de série de la zone et des durées que nous détaillerons.

Il en existe d'autres comme TXT mais pas forcément utiles ou intéressants pour ce cours.

Exemple d'un fichier de zone

```
console$TTL 604800     ; 1 semaine
$ORIGIN test.local.
@       IN SOA  ns1.reseau.fr. admin.test.local. (
                                2013020905 ;serial
                                3600       ; refresh (1 hour)
                                3000       ; retry (50 minutes)
                                4619200    ; expire (7 weeks 4 days 11 hours 6 minutes 40 seconds)
                                604800     ; minimum (1 week)
                                )

@               IN      NS      ns1.test.local.
@               IN      NS      ns2
@               IN      MX      10 mx1
@               IN      MX      20 mx2
ns1             IN      A       192.168.0.1
ns2             IN      A       192.168.0.2
mx1             IN      A       192.168.0.3
mx2             IN      A       192.168.0.4
tuto            IN      A       192.168.0.5
www             IN      A       192.168.0.6
blog            IN      CNAME   www
```

Figure 7: fichier de zone

Examinons chacune de ces informations.

La première info est un TTL (Time to Live). Quand quelqu'un va interroger votre serveur DNS pour obtenir des informations, ces informations vont être stockées en cache chez cette personne (dans la mémoire de son serveur DNS, pour éviter qu'il vienne nous réinterroger de nombreuses fois s'il a de nouveau besoin d'une information). Ce TTL est la durée pendant laquelle les informations sont conservées en cache. Ce délai passé, une nouvelle demande devra être faite au serveur. Le TTL est défini ici sur 1 semaine. En fonction de la fréquence de vos mises à jour, vous pouvez décider de baisser cette valeur pour que vos clients aient leurs informations à jour.

La deuxième info est la variable ORIGIN. Celle-ci est optionnelle. Les @ prennent la valeur de la variable ORIGIN. En l'absence de variable, ils prendront la valeur du nom de votre zone défini dans le fichier named.conf (test.local ici) qui est un fichier de référence sur la machine UNIX par exemple.

Vient ensuite le premier enregistrement, c'est un enregistrement de type SOA (Start Of Authority). Le type SOA est suivi de deux informations. La première est le nom du serveur de domaine principal (master) et la seconde est l'adresse mail de l'administrateur du domaine (en remplaçant l'arrobase par un point). Suivent entre parenthèses différentes valeurs.

- Le serial peut être comparé à un numéro de version de votre zone. Il doit être incrémenté à chaque modification. Cela indique à votre serveur que votre zone a été mise à jour et qu'il faut envoyer la notification à vos serveurs esclaves. Les best practices recommandent une syntaxe particulière pour le serial de la forme AAAAMMJJXX (où XX est la version du jour en question). Cela vous permet entre autres de savoir la date de la dernière mise à jour de votre zone.
- Refresh est le temps au bout duquel les enregistrements sont stockés sur le serveur slave. Passé ce délai, le serveur slave demandera une nouvelle mise à jour au serveur master.
- Retry est le temps qu'attendra le serveur slave dans le cas où le serveur master contacté n'est pas joignable pour faire un nouvel essai.
- Expire est le temps pendant lequel le serveur slave continuera à essayer de contacter le serveur master.
- Minimum est la durée minimale du cache ; elle est en général égale à Refresh.

Nous trouvons ensuite les enregistrements, du moins ceux qui sont intéressants !

Les enregistrements se découpent en 4 parties sur une ligne (parfois 5 pour des enregistrements spécifiques).

- La première information, c'est l'hôte du domaine. Nous avons parlé du @ tout à l'heure qui est remplacé par la valeur de $ORIGIN (le cas échéant par le nom de votre zone). Notez qu'on peut ne rien mettre du tout si on veut parler du domaine entier. Rien, @, ou un nom de machine ou de sous-domaine au choix.
- Le second, représente la classe. Ici, elle spécifie qu'il s'agit d'un enregistrement concernant Internet. Il existe d'autres valeurs mais elles ne sont pas utilisées, donc on met toujours IN.
- Le troisième spécifie le type d'enregistrement dont on a détaillé les différents types précédemment.
- Enfin, le dernier spécifie la valeur de l'enregistrement dépendant du type. Un type A attendra une adresse IP, un type PTR attendra un nom d'hôte, etc.

On trouve parfois, juste avant cette dernière valeur, un nombre qui indique le "poids" d'un enregistrement. On verra plus loin dans quel cas c'est utile.

On commence généralement par les enregistrements des serveurs gérant notre domaine et les services associés (le mail en l'occurrence). Dans notre cas il s'agit des types NS et MX. On utilise l'@ parce que ces enregistrements ne déterminent pas un hôte en particulier, mais bien le domaine entier.

```
console@                IN      NS      ns1.test.local.
```

Cette ligne se traduit donc par : "ns1.test.local est un serveur de nom de domaine de test.local"

Attention sur le "." situé à la fin de « ns1.test.local. », car celui-ci est extrêmement important. Cette valeur doit être un FQDN et le FQDN contient le "." représentant la racine du DNS. Si vous aviez écrit ns1.test.local sans le ".", votre serveur aurait pu automatiquement ajouté à la fin le FQDN de votre zone, ce qui aurait donné « ns1.test.local.test.local. » !

Ceci étant, réécrire à chaque fois le FQDN c'est un peu contraignant. Et comme on sait que, ne pas finir sa ligne par un "." rajoute au FQDN de votre zone, on peut se permettre de n'écrire que "ns1". Ainsi, votre serveur rajoutera "test.local." et on aura le FQDN que l'on cherchait à obtenir.

Voyez la deuxième ligne qui utilise cette syntaxe raccourcie.

Les enregistrements MX utilisent la même syntaxe que pour les NS et indiquent l'adresse IP d'un serveur de messagerie, à cela près que nous avons rajouté un chiffre devant "mx1". Nous avons dit tout à l'heure que ce chiffre déterminait le "poids" d'un enregistrement, on parle aussi de priorité. Nous avons deux serveurs MX : mx1 et mx2 ; cette valeur va permettre de déterminer lequel des deux doit être utilisé en priorité. Plus elle est basse, plus le serveur est prioritaire.

Mais nous avons aussi deux serveurs NS ! Comment se passe cette priorité, étant donné qu'il n'y a pas de valeur pour les départager ? Dans ce cas, c'est chacun son tour. Cela s'appelle du Round-Robin, c'est une méthode qui permet d'équilibrer la charge entre les deux serveurs pour ne pas les surcharger, car un serveur sera autant consulté que les autres serveurs du même type.

Très bien, maintenant on sait que les serveurs mail de notre domaine sont mx1.test.local et mx2.test.local. Cependant, on ne sait toujours pas leurs adresses IP alors que c'est quand même le but d'un serveur DNS. Il faut donc ajouter un enregistrement A pour le nom de ce serveur.

On retrouve donc les enregistrements les plus courants, ceux de type A (et AAAA quand on a de l'IPv6). En effet, le rôle principal du DNS est de faire correspondre un nom d'hôte avec son adresse IP, et c'est ce que fait le type A.

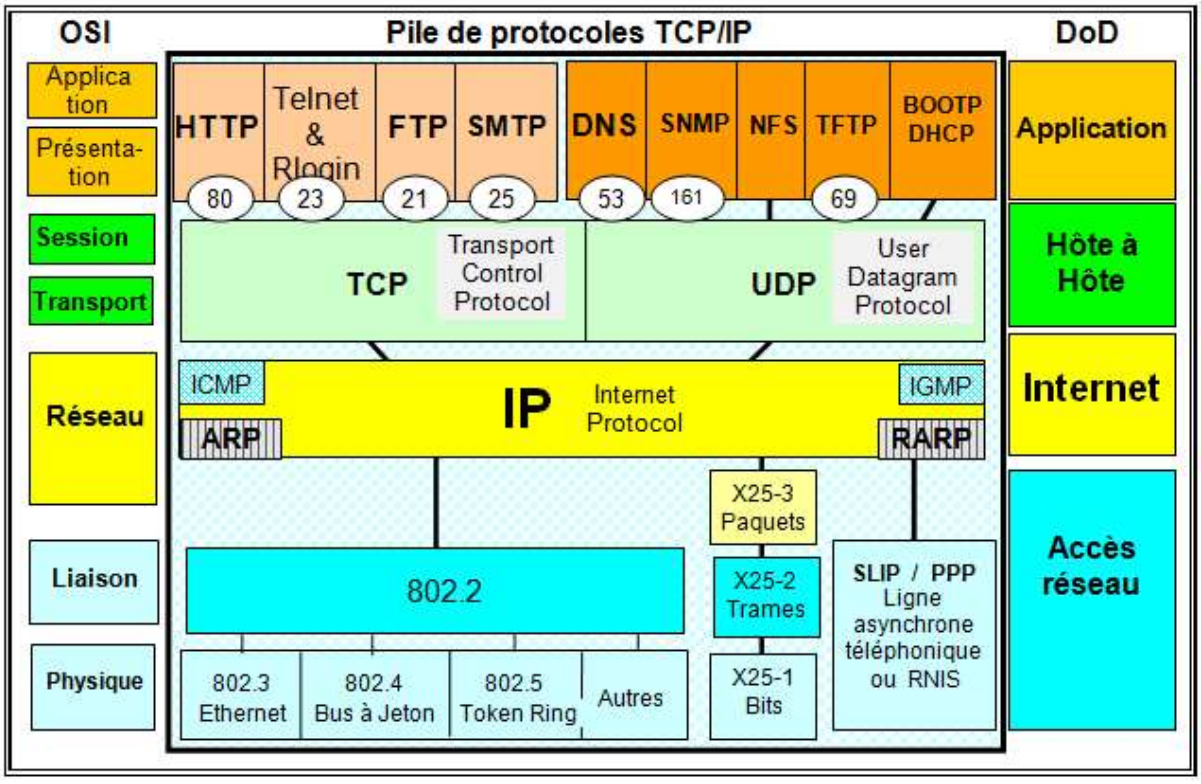
La syntaxe est relativement simple comme vous pouvez le voir :

consoletuto IN A 192.168.0.5

Comme pour les autres enregistrements, "tuto" ou "tuto.test.local" revient au même. N'oubliez pas le point si vous optez pour le FQDN.

Le type CNAME est aussi simple à comprendre. On fait correspondre un nom d'hôte à un autre nom d'hôte. Bien sûr, si "blog" pointe sur "www", l'enregistrement www doit exister.

Tableau modèle OSI avec les différentes couches :



Le protocole DNS fait donc partie de la couche application.

Exercices

Trouvez les noms et adresses IP des 13 serveurs racine ;

Trouvez la ou les adresses IP de www.cpmb.ch ;

Trouvez la ou les adresses IP de www.google.ch ;

Trouvez les adresses IP des serveurs DNS de www.evolink.ch et support.evolink.ch.