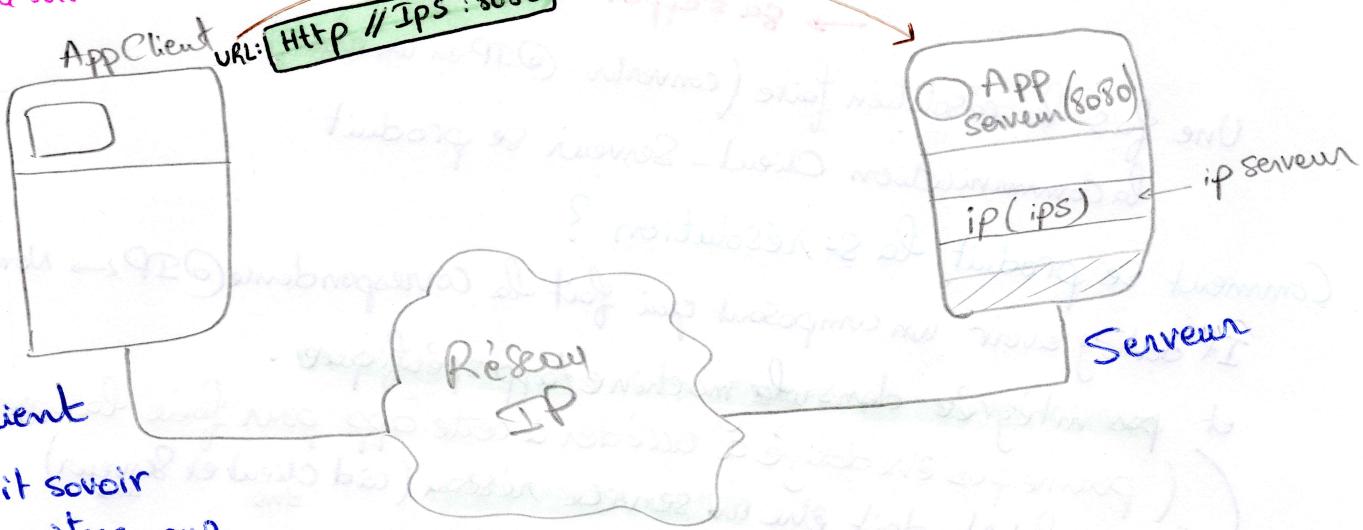


# Serveurs DNS (Domain Name System)

L'app Client fait le paramétrage de 4 éléments fondamentaux

- l'@IP
- l'@Masq
- la Gateway par défaut (la gateway associée à la destination 0.0.0.0/0)
- le serveur DNS qui va se connecter à

Pourquoi le serveur DNS ?



Client  
doit savoir  
3 paramètres pour  
pouvoir communiquer avec le serveur

- @IP serveur (ips)
- Num de port (8080)
- Protocole (Http)

(Les protocoles de communication : Http / Https  
FTP / FTPS)

\* Notre objectif est de simplifier cette @URL aux maximum, pour faciliter la communication

→ 1<sup>ère</sup> simplification : on associe à chaque protocole de communication un port par défaut

pour un serveur web (Http) on a le port par défaut 80.

→ URL : Http://Ips : 80

mais on note que le port par défaut est connu partout le monde qui ?  
représente le protocole http.

→ 2<sup>ème</sup> simplification : on élimine 80 (on ne peut pas éliminer le protocole)

→ URL : Http://Ips

update  
Client  
doit Savoir 2 paramètres pour communiquer avec le serveur {  
- ips  
- protocole de communication}

- \* pour le ips (@IPs)
  - on peut le garder tel qu'il est
  - mais ça sera plus facile à l'utilisateur si la machine était identifiée par un nom significatif au lieu d'une @IP

→ 3ème simplification

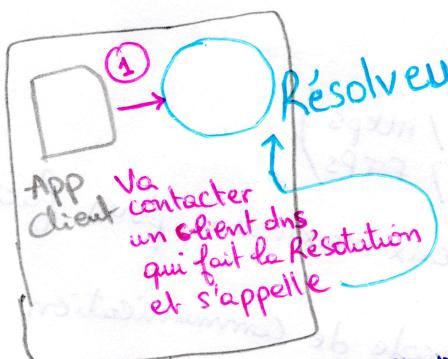
URL: http:// Nom.Serveur

↳ qui représente une @IP  
 ↳ il faut faire la correspondance @IP → Nom  
 → ça s'appelle: La Résolution

Une fois la résolution faite (convertir @IP en un Nom)  
 la communication Client - Serveur se produit

Comment se produit la résolution ?

Il doit y avoir un composant qui fait la correspondance @IP → Nom et pas intégrée dans une machine/app spécifique.  
 (pour ne pas être obligé d'accéder à cette app pour faire la correspondance)  
 ↪ mais plutôt doit être un service réseau (càd Client et Serveur) dns dns



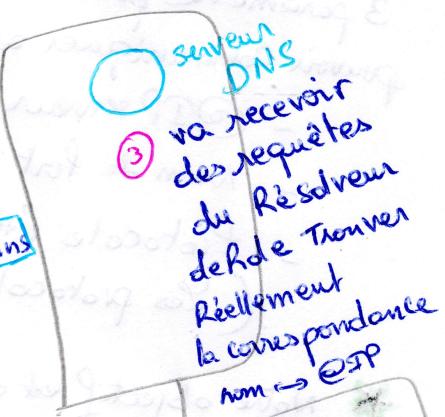
Résolveur: process, composante logicielle du système d'exploitation qui a pour rôle de résolution des noms en contactant un serveur dns

et pour URL: http:// serveur  
 Résolveur contacte un serveur dns:

“donne moi une @IP qui correspond à serveur 1”

La résolution est stockée dans la mémoire cache (temporaire) du résolveur, ainsi si une autre app client demande la résolution du même nom elle est prête

notons bien que la résolution prend beaucoup de temps.



⇒ Résolveur intermédiaire entre Client et Serveur DNS

Pourquoi ne pas avoir communication directe entre Client et Serveur ?

Supposons qu'on a 2 App locales dans la même machine veulent accéder au même nom dans 2 instants diff → Résolution 2 fois.

### \* par contre avec le Résolveur :

- la Résolution se produit une seule fois
- Résultat stocké dans la mémoire cache peut être exploité plusieurs fois

→ On utilise le Résolveur pour

- minimiser la charge sur le serveur DNS.
- optimiser la communication Client → serveur DNS
- simplifier le code de l'app Client.

un Protocole DNS implémenté entre

Résolveur et Serveur DNS

- . Résolveur doit Savoir { de NumPort du serveur DNS 53 }  
{ @IP serveur DNS  
G doit avoir une config }

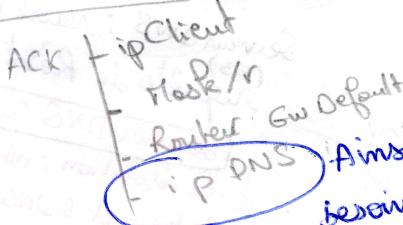
Revenons à la config du Client :

config → manuelle  
ou bien  
avec DHCP

paramètres Réseaux :

- ipClient/r
- Table de routage
- @IP serveur DNS

Dest | Gateway  
0.0.0.0 Default Gateway



Ainsi Résolveur soit que : quand il a besoin de faire une autre résolution il va contacter ip DNS.

. Communication entre Résolveur et

Serveur DNS via UDP.  
config statique dans l'application  
ou DHCP

Résolution prête, le Résolveur reçoit les requêtes du serveur DNS en saupoudre (requêtes contenant la correspondance Nom - @IP)

Résolveur doit renvoyer une réponse au serveur DNS.

Ce serveur DNS doit avoir la table de correspondance Nom - OIP mais avec un Internet (Réseau Très grand)

Ce n'est pas possible d'avoir la BD de toutes les résolutions dans une seule machine

→ il faut diviser la BD à des parties dans des serveur 1, serveur 2, ..., serveur n

pour cela on a suivi l'exemple des fichiers et répertoires.

/rep/rep<sub>1</sub>/f<sub>1</sub> : chemin absolu

pour accéder à un nom précis.

Comment ?

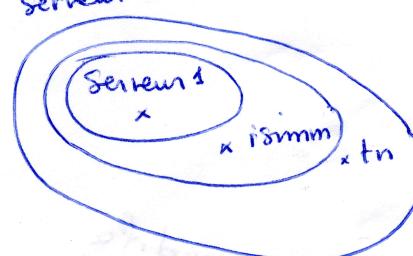
{ La machine ∈ Domaine  
Le nom → qui peut-être un sous-domaine d'un domaine supérieur

on associe à chaque domaine un serveur DNS

et enfin un domaine racine qui regroupe tous les sous-domaine.

### Exemple :

serveur 1 ∈ Domaine



nom Complet de serveur:

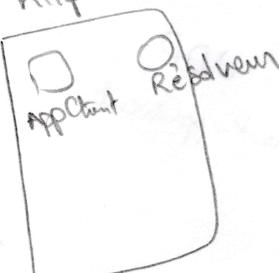
FQDN: serveur 1 . isimm . tn

rout

∈ Domaine

∈ Domaine

http://: serveur 1 . isimm . tn



Server DNS : reçoit les requêtes du Résolveur

(53) Server DNS Cache qui fait la résolution ; intermédiaire entre résolveur et S.DNS primaire

Réseau IP

ips

Autre : Server DNS reçoit les requêtes de serveur DNS (53)

contient une BD qui contient table de correspondance

serveur DNS primaire qui gère la zone isimm . tn

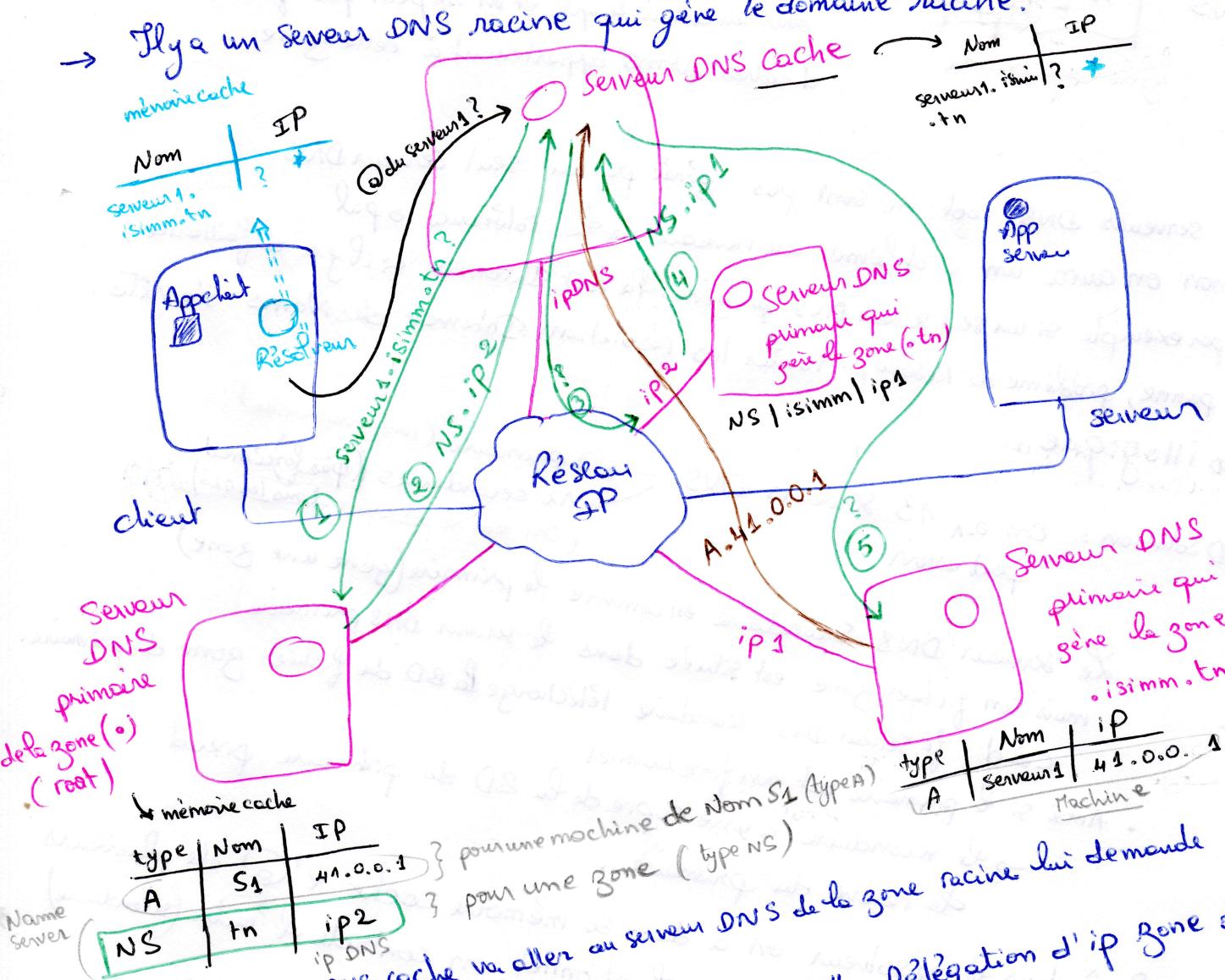
Nom | IP  
serveur 1 | 192.0.1.1 d'un domaine

Mais comment le serveur DNS Cache peut savoir que la zone `iisimm.tn` est gérée par `ip1` ?! pour savoir la correspondance

d'`@IP` au serveur 1

Domaine

→ Il y a un serveur DNS racine qui gère le domaine racine.



type	Nom	IP
A	serveur1.iisimm.tn	41.0.0.1

Rqve: Les réponses NS ne doivent pas être mises dans la mémoire cache.

type	nom	IP
NS	isimm.tn	ips

Dans ce cas au moment de l'expiration de cette résolution ips va changer ainsi la zone isimm.tn devient inaccessible pour un certain temps et on ne peut pas faire des résolutions d'autres Noms appartenant à cette zone.

*Zone inaccessible*

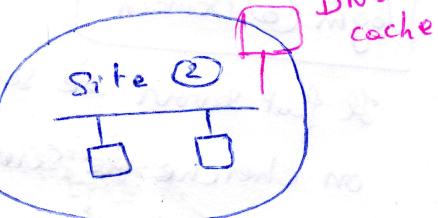
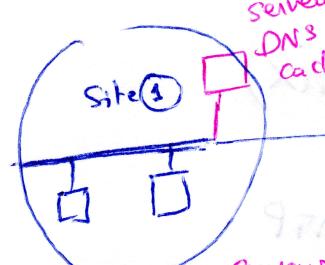
Les serveurs DNS root ne sont pas générés par un seul serveur DNS sinon on aura un problème au niveau de la tolérance opal par exemple si un seul serveur DNS qui gère la root Internet se défaillait, panne, problème de Réseau, ... toutes les résolutions Internet des Noms s'arrête. => illogique,

=> Solution: On a 13 serveurs DNS → 1 primaire (config manuelle) → 12 secondaires (Max) (On peut ne pas avoir de secondaire)

Le serveur DNS secondaire est comme le primaire (génère une zone) mais son fichier zone est située dans le serveur DNS primaire. c'est le serveur DNS secondaire télécharge la BD du fichier zone du primaire. Ainsi si le primaire est non fonctionnel → le secondaire a une copie de la BD du primaire prend la relève du primaire.

De même dans le résolveur on a dans sa mémoire cache l'@ de plusieurs serveurs DNS, si l'un n'est pas fonctionnel on contacte l'autre (secondaire).

Il suffit de faire une mise à jour dans le fichier zone du DNS primaire et tous les S.DNS secondaires seront mis à jour automatiquement. (pas de problème de synchronisation des configurations).

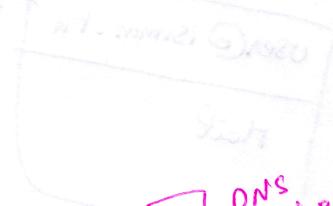
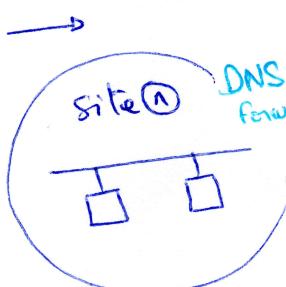


on ajoute les serveurs DNS cache pour :

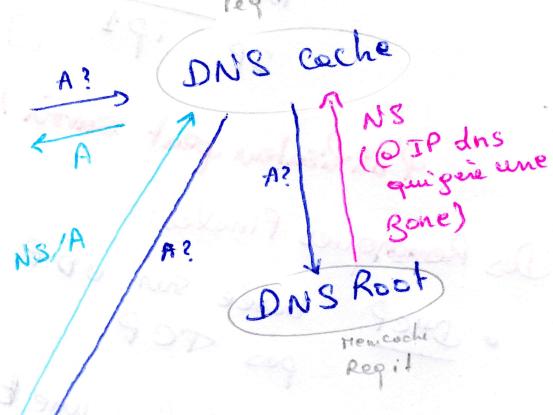
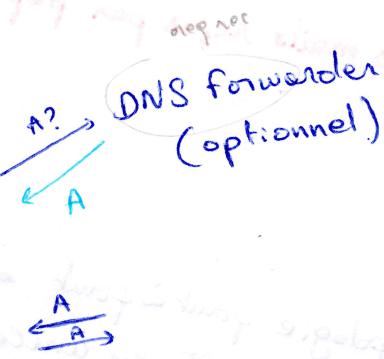
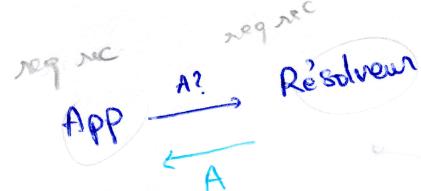
Accélérer la résolution mais pas optimiser

sites visité dans sites 1 peuvent être les même que site 2

pour ne pas refaire la résolution et faire



DNS Cache



req rec : requête récursive  
req it : requête itérative  
(réponse partielle)

mémoire cache : mémoire temporaire tampon pour enregistrer de manière temporaire. Contient un champ TTL (temps de validité).

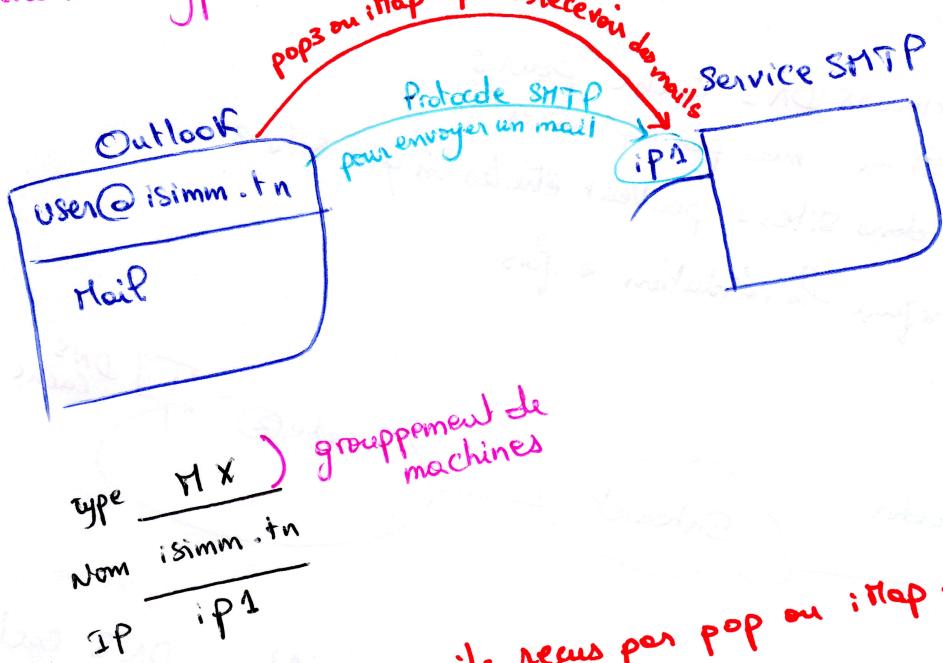
Rq: le serveur DNS racine ne peut jamais envoyer au résolveur des requêtes car → Résolveur émet des requêtes récursive

→ DNS racine émet des requêtes itérative (avec la réponse finale)

et envoi une réponse avec un TTL (temp de validité)

## login @ Domain

Il faut savoir le serveur (@IP du serveur) qui gère le domaine.  
on cherche le serveur de messagerie d'un domaine.  
Résolution de type MX (enregistrement de type MX)



L'utilisateur peut savoir les mails reçus par POP ou IMAP.

## Des Remarques finales:

- DHCP basée sur UDP  
car TCP utilise une topologie point à point  
donc la communication doit être en unicast  
alors que DHCP communique en Broadcast.
- TCP est un protocole en mode connecté qui teste la joignabilité de l'app.  
la 3phase principale :
  1. ouverture (SYN)
  2. transfert des données (SYN-ACK)
  3. fermeture (ACK)segmenté dans le bon ordre  
→ il rétablit la date.  
→ détecte les pertes des segments par acquittement (accusé de Réception)
- DNS utilise UDP → UDP : pour la résolution (si un serveur cache ne répond pas on contacte un autre)  
TCP : pour la synchronisation du fichier zone  
du DNS primaire et des DNS secondaires  
(voir l'exercice suivant pour comprendre les composantes du fichier zone)

## Exercice :

Quel sont les paramètres qui doivent être configurés sur

- un résolveur @ ip(s) Serveur DNS / forwarder ou DN Cache

- DNS Cache (@ primarie / 12 sec) @ ip(s) DNS Root

- DNS secondaire @ ip du DNS Primaire (copie de Primarie pour que si Primarie défaillent serveur' ont recourt au secondaire)

- DNS forwarder @ ip(s) DNS cache / forwarder

- DNS primaire

• Nom du Domaine (isimm.tn)

• Fichier zone de isimm.tn

un Enregistrement

pour faire la délégation

qui contient:

Type	Nom	IP /	TTL
A	www	41.0.0.4	
NS	etudiant	41.0.0.3	
MX	isimm.tn	41.0.0.3	P ip serv Messagie
	Nom Domaine		ip serveur dns → primarie → secondaire

DNS root(.)

NS	tn	41.0.0.4
NS	tn	41.0.0.2
NS	tn	41.0.0.3

Round  
Robin

si on veut ajouter un autre serveur DNS tn  
il faut qu'il soit délégué par le root(.)

on augmente le TTL pour diminuer le nbr de requêtes refusées par le DNS cache.

Type Nom IP TTL  
A | www | 41.0.0.1 | 2J

Appartient à une migration de .1 à .10

Si on le change directement en 41.0.0.10  
après 2J elle sera renouvelée  
et l'@ip est indiqué par 41.0.0.1  
→ site inaccessible pendant 2J (trop)  
→ dom inaccessible pendant ej (trop)  
(www)

→ A | www | 41.0.0.1 | 2h on change TTL

on attend 2J jusqu'à la durée de renouvellement de went 2h  
→ site inaccessible pendant 2h seulement on attend 2h  
→ @ip migrée on peut renouveler 2h en 2J