http://clx.asso.fr/spip/?Gerer-une-zone-alias

Bind

Gérer une zone "alias"

- Documentations - Technique -

Date de mise en ligne : mercredi 15 octobre 2003

Le *resolver* de **Bind**, que l'on trouve sur tous les unices, fait la correspondance entre un nom de serveur pleinement qualifié (nom de la machine serveur + nom de domaine - machine.domaine.com) et l'adresse IP associée (par exemple 211.211.211.211).

Un mécanisme particulier permet d'éviter de saisir le nom complet du serveur à atteindre, mais uniquement le nom du serveur. Le *resolver* complÚte tout seul le nom de domaine grâce à une liste de domaines prédéfinis dans son fichier de configuration `/etc/resolv.conf`.
Ainsi, une requête sur portant « clx » (par exemple `ping clx`) donnera le résultat « clx.anet.fr » lorsque le fichier `/etc/resolv.conf` contient la directive `search anet.fr`. Ce « raccourci » est bien pratique pour désigner rapidement une machine.

Or, dans un environnement contenant plusieurs zones, des requêtes sont effectuées sur les différents domaines qui les composent. Mais le nombre de domaines affectés à la directive *search* est limité en général à 6. Vous ne pouvez pas utiliser de tous les « raccourcis » dont vous pourriez avoir besoin, vous obligeant ainsi à taper les noms complets.

Cet article décrit le fonctionnement de la résolution de nom, et permet - si vous avez un DNS local - de contourner cette limitation du champ *search* du fichier `resolv.conf`.

Fonctionnement de la résolution DNS

Tout d'abord, un mot sur le fonctionnement de *Bind*.
Si pensez tout savoir de la résolution de nom, passez directement au cœur du problème.

Bind sert à *résoudre des noms*, c'est à dire de transformer une adresse de la forme `clx.anet.fr` en adresse IP (80.118.2.10).
De même, il permet de retrouver le nom `clx.anet.fr` à partir de son adresse IP. On appelle ce nom, un *Fully Qualified Domain Name*, FQDN en abrégé, c'est-à-dire un « Nom de Domaine Complet » (ou nom de domaine pleinement qualifié).

Pratiquement, cela permet - entre autres - de déplacer un serveur sans en changer son nom FQDN [1], et surtout, évite aux utilisateurs de retenir une adresse IP sur 4 octets, ce qu'ils ne pourraient de toute façon, vraisemblablement pas faire pour tous les sites Web qu'ils consultent.

Les noms de domaines sont gérés zone par zone, de manière hiérarchique.
Le '.' sépare les zones.
Le découpage hiérarchique des zones DNS se fait de droite à gauche, le nom de la machine en dernier. Ainsi, `clx.anet.fr` représente le serveur nommé *clx*, dans le domaine *anet*, du suffixe *fr*.

Concrêtement, l'Afnic gère l'attribution des zones dans le suffixe .fr, VeriSign s'occupe de l'attribution des zones pour le suffixe .com et .net, nous venons d'en entendre beaucoup parler (http://linuxfr.org/2003/09/17/13947.html et http://www.transfert.net/a9294).

Pour effectuer la résolution, le *resolver* procèdera donc par requêtes successives, comme ceci :
â€" *DNS racine, qui est le DNS du suffixe* `fr` ?
â€" `ns1.nic.fr.`
â€" *DNS de l'Afnic, qui est le DNS du domaine* `anet.fr` ?
â€" `ns1.cyberouest.fr. et b612.anet.fr.`
â€" *DNS de cyberouest, quelle est l'adresse IP du serveur* `clx.anet.fr` ?
â€" `clx.anet.fr` est un alias vers le serveur `gaia.anet.fr.`, lequel a l'adresse 80.118.2.10. [2]
[http://clx.asso.fr/spip/local/cache-vignettes/L400xH297/dns-anim-b95ab.png] **Requête DNS simplifiée** Ici, pour simplifier le schéma, nous interrogeons b612.anet.fr. En réalité c'est un peu plus complexe, et il vaut mieux interroger ns1.cyberouest.fr.

D'une manière similaire, pour connaître le nom FQDN d'une adresse IP particulière, on fera des requêtes successives sur les serveurs DNS gérant les zones de résolution inverse des différents réseaux IP.

Les fichiers de zone de BIND

Concrètement, pour résoudre les noms de domaine, *Bind* va utiliser deux zones.

La première, vouée à la **résolution directe** de la zone `anet.fr`, contient toutes les informations relatives à la zone (domaine) `anet.fr`, notamment les adresses IP des serveurs, et les informations sur les sous domaines potentiels.

La seconde, vouée à la **résolution inverse** (zone reverse, en franglais courant), indique le FQDN des serveurs ayant des adresses IP dans le réseau `80.118.2.0`, ainsi que ses sous réseaux. Cette zone sera notée, pour éviter tout ambiguïté, à l'envers, c'est à dire dans le même ordre hiérarchique que pour la zone directe. Le plus important est situé à la fin :
`2.118.80.in-addr.arpa` avec un suffixe particulier indiquant qu'il s'agit d'une zone de résolution inverse.

Le fichier de zone contient plusieurs champs, permettant de fixer :
[-] le serveur DNS primaire, gérant la zone ;
[-] des informations sur la zone, comportant notamment le numéro de révision du fichier de zone (il sert à savoir s'il y a eu des modifications) ; on appelle ce champ le "SOA", pour *Start Of Authority* ;
[-] la liste des sous domaines éventuels ;
[-] la liste des serveurs, associés à leur adresse IP ;
[-] d'autres champs éventuels, qui sortent du cadre de cet article.

Ainsi, un fichier de zone s'écrira :

```
 $ORIGIN anet.fr.
@    IN SOA ns2.cyberouset.fr. zoneadmin.cyberouest.fr. {
              2003092501    ; numéro de série de la zone
              10800         ; A rafraîchir toutes les 3h (3x3600")
              3600          ; Délai avant réessai sur erreur : 1h (3600")
              604800        ; Informations périmées après une semaine (24x7x3600")
              86400         ; Informations périmées au minimum 1 après une journée (24x3600")
            }
@      IN NS    ns2.cyberouest.fr.
                 ns3.cyberouest.fr.
                 b512.anet.fr.
clx    IN CNAME gaia
cim    IN CNAME gaia
gaia   IN A     80.118.2.10
puck   IN A     80.118.2.11
.../...
```

On voit donc les champs :
[-] "$ORIGIN" qui permet de préciser de quelle zone il s'agit [3] ;
[-] SOA serveur mail-responsable.serveur *liste de paramètres* ;
[-] NS pour les serveurs de référence ;
[-] A pour les serveurs de la zone avec leur adresse IP ;
[-] CNAME pour les alias dont nous allons maintenant parler. Nous ne détaillerons pas la syntaxe précise, pour cela, référez-vous au *NAG* [4] cité dans la bibliographie, mais là, l'essentiel est dit.

Les alias

Il est souvent commode d'appeler un serveur par plusieurs noms, ne serait-ce que pour des raisons de limitation du nombre d'adresses IP en IPv4 (le standard actuel) ; c'est à ça que sert le champ *CNAME*.
Vous le voyez, pour définir un alias, c'est très simple. Il suffit de donner un nom d'alias à créer, "IN CNAME", et la machine sur laquelle pointe l'alias. Notez que les FQDN sont **toujours** suivis d'un point « . », sinon, il s'agit d'un nom relatif à la zone, c'est à dire qu'il sera suffixé du champ *$ORIGIN* [5].

Ainsi, pour créer clx.anet.fr, qui est un alias pointant vers gaia.anet.fr, il faut écrire :

```
clx          IN CNAME gaia
```

dans la zone anet.fr,
Ou, en utilisant les FQDN :

```
clx.anet.fr. IN CNAME gaia.anet.fr.
```

Là, cela devient intéressant : il est tout à fait possible de définir un alias pointant sur un serveur *extérieur* à la zone. Ainsi, nous pourrions créer un alias de la sorte :

```
www.ryckeboer.org. IN CNAME asr.free.fr.
```

dans la zone « ryckeboer.org ».

Cela nous sera utile un peu plus loin dans l'article.

Le resolver unix

Une fois connu le fonctionnement des requêtes DNS et de *Bind*, penchons-nous sur la façon dont le système fera appel aux serveurs DNS.

Deux fichiers, qui cachent la complexité du procédé sont utilisés. Le premier porte le doux nom de `/etc/nsswitch.conf`, et indique de quelle façon le système trouvera les adresses IP associées aux noms des machines, et vice-versa.
Deux moyens courants sont en effet à la disposition du système : le fichier `/etc/hosts`, et le DNS. Pour utiliser le DNS, nous allons donc vérifier la présence de la ligne :

```
hosts:    files dns
```

dans le fichier `/etc/nsswitch.conf`

Toutes les variations sont possibles, mais sachez que l'ordre des mots-clefs désigne l'ordre dans lequel le système effectuera ses requêtes. D'autres mots-clefs peuvent être présents dans ce fichier, `nis`, par exemple, afin d'utiliser les *pages jaunes* ou *YellowPages* dont nous ne parlerons pas ici.

L'autre fichier important pour la résolution DNS s'appelle `/etc/resolv.conf`. Il sert à paramétrer le *resolver* unix. On y trouve les champs principaux :
[-] `domain`, qui indique le nom de domaine dans lequel se trouve votre machine ;
[-] `nameserver`, qui indique un serveur DNS sur lequel effectuer les requêtes ;
[-] `search`, qui permet de préciser un ou des domaines sur lesquels effectuer les recherches si elles n'aboutissent pas pour le domaine précisé par `domain`.

Sur mon serveur, jabba.nowhere.null, j'ai un fichier */etc/resolv.conf* qui contient :

```
 domain  domain.nowhere.null
nameserver 172.0.0.1 # serveur DNS local
nameserver 80.118.2.10 # serveur DNS anet.fr
nameserver 212.27.35.34 # serveur DNS free
search domain.nowhere.null nowhere.null ailleurs.null anet.fr ryckeboer.org virtual-net.fr
```

Ce qui veut dire que :
[-] mon domaine local est *domain.nowhere.null* ;
[-] j'utilise dans l'ordre trois serveurs de noms, c'est-à-dire le serveur local, le serveur anet.fr, et le serveur free ;
[-] je lance des requêtes sur quatre domaines (dans l'ordre) en cas d'échec.

Notez domain.nowhere.null et nowhere.null ; il a été dit plus haut que le *resolver* "ajoutait" les domaines de `search` à votre requête, ce qui veut dire qu'il ne fera pas de requête récursive. Autrement dit, pour rechercher sur les sous domaines de `nowhere.null`, il faut les indiquer tous dans le champ *search*.

Sauf que... on ne peut pas mettre plus de 6 domaines dans le champ *search*. Et c'est heureux, cela ralentit considérablement le temps de traitement en cas d'erreur, c'est-à-dire entre le moment où vous faites la requête, et le moment où le *resolver* répond que l'hôte demandé n'existe pas.

Pour contourner cela, il faut créer une zone particulière sur laquelle vous ferez les requêtes, et qui ne contiendra que des alias vers les vrais zones.

Vous avez un DNS local

Admettons que les fichiers de vos zones primaires soient stockées dans `/var/named/primary`.
Vous souhaitez faire des requêtes sur les noms de domaine directement, sans taper le nom de domaine complet.

Nous allons créer une zone contenant le nom des machines de toutes les zones sur lesquelles vous faites les recherches, déclarées comme alias vers le nom réel.

Par exemple, je veux que « ping jabba » fonctionne correctement, et examine via un *ping* la machine `jabba.starwars.null`, et que ping bilbo effectue correctement un *ping* sur `bilbo.middleearth.org`.

Il faut tout simplement ajouter une zone contenant la liste des alias, de la sorte :

```
 $ORIGIN aliases.nowhere.null.
bilbo    IN CNAME bilbo.middleearth.org.
yoda     IN CNAME yoda.starwars.null.
jabba    IN CNAME jabba.starwars.null.
clx      IN CNAME clx.anet.fr.
leeloo   IN CNAME leeloo.lee-loo.net.
epp      IN CNAME www.epplug.org.
```

etc.

Dans mon `/etc/resolv.conf`, je n'ai plus besoin que de la zone aliases.nowhere.null :

```
search aliases.nowhere.null
```

Deux méthodes s'offrent à vous. Soit vous tapez tout à la main, si vous ne gérez pas vous-même vos zones, par exemple, soit vous écrivez un petit script qui va lire les zones locales, et créer les alias corespondants. Pour cela, bien sûr, il faut avoir à disposition les fichiers des zones que vous souhaîtez « aliaser ».

Lire les fichiers de zone et créer la zone alias

Le principe est simple :
[-] On prend tous les fichiers de zone que l'on veut traiter, qui sont stockées dans un fichier. `zone.list`, par exemple.
[-] Pour chaque fichier de la zone, on recherche les champs « IN A », et on le transforme en « IN CNAME ».
[-] Enfin, on écrit tout dans le fichier `alias.nowhere.null`.

Personnellement, je suis fainéant, j'ai donc écrit un script dont voici l'explication.

[-] **1)** Tout d'abord quelques variables relatives à l'emplacement des fichiers de zone :

```
 $server="ns.nowhere.null"
$named_path="/var/named/primary"
$aliaszone="aliases.nowhere.null
$zonelist="$named_path/zones.list"
```

[-] **2)** On lit les fichiers de zone, de la forme

```
sousdomaine.domaine.suffixe.hosts
```

; `$zonelist` est le fichier contenant la liste des zones à traiter :

```
 for i in `cat $zonelist`
do
cat $i | perl -pe "
   chomp;
   # On récupère l'enregistrement 'toto IN A addresse_ip'
   if (/([^\\. \\t]+)(\\..*)?[ \\t]+[Ii][nN][ \\t]+[aA][\\t ]+(.+)/)
   {
      \$_ = \"\$1\\tIN CNAME\\t\$1.${i%.hosts}.\\n\"
   } else {
      # Sinon, c'est peut-être un commentaire ?
      unless (/^[ \t]*;/) {
        \$_ = '';
      } else {
        # Si ce n'est pas un commentaire, on ignore.
        \$_ .=\"\\n\";
      }
   }"
done
```

[-] **3)** Pour créer un fichier de zone cohérent, il faut également un *SOA*, écrit de la sorte :

```
 @ IN SOA $server. root.$server. (
      $version_date$rev      ; numero de serie de la zone
      10800                  ; refresh every 3 hours
      3600                   ; retry every hours
      604800                 ; expire after a week
      86400 )                ; minimum TTL of 1 day
```

et le serveur de nom primaire :

```
 IN      NS      $server.
```

[-] **4)** Le numéro de série de la zone doit toujours s'incrémenter. Sinon, les changements ne sont pas repérés. Voici le bout de code qui recherche le numéro de série, et l'incrémente. C'est ce numéro de série matérialisé par les variables `$version_date` et `$rev`. En général, on l'écrit sous la forme :

```
ANNEE MOIS JOUR REVISION
```

Ainsi, pour la 3eme révision de la zone, ces trois révisions ayant ayant eu lieu le 7 octobre 2003, on écrira *2003.10.07 03*, sans les espaces, et les points, soit **2003100703**

Voici le code en question :

```
 rev=01
version_date=`date +"%Y%m%d"`

if [ -f $named_path$aliaszone ] ; then
  # First, get old revision number for the zone file
  # split date and revision
  oldver=`grep "; serial" $named_path$aliaszone | tr -s " " | cut -d " " -f 2`
  oldrev=`echo $oldver | perl -pe 's/........(..)/$1/'`
  # compare current date and zone date
  [ ! "a$oldrev" = "a" ] && if [ $version_date = `echo $oldver | perl -pe 's/(........)../$1/'` ] ; then
    # ++ revision number for the day
    if [ $rev -le $oldrev ] ; then
      if [ $oldrev -ge 10 ] ; then
      # handle 2 digits
        rev=`expr $oldrev + 1`
      else
        rev="0`expr "$oldrev" + 1`"
      fi
    fi
  fi
fi
```

Texte - 2.5Â ko **Le script complet**

Placez ce script dans `/usr/local/bin`, en supprimant l'extension .txt. N'oubliez pas de mettre les droits en écriture dessus :

```
# mv /usr/local/bin/mkaliaszone.txt /usr/local/bin/mkaliaszone
 # chmod ugo+x /usr/local/bin/mkaliaszone
```

En savoir plus

Chez *O'Reilly*, on trouve :
[-] **DNS & BIND** par *Paul Albitz* et *Cricket Liu*, ISBN 2-84177-150-4, pour découvrir (beaucoup) plus sur *Bind* et le fonctionnement des DNS ;
[-] **TCP-IP Administration en réseau**, par *Craig Hunt* ISBN 2-84177-221-7, pour tout savoir sur le fonctionnement de TCP/IP, notamment au niveau de l'adressage IP ;
[-] **Administration réseau sous Linux** par *Olaf Kirch* et *Terry Dawson*, ISBN 2-84177-125-3 qui est la version papier et traduite en français du NAG, **Network Administrator's Guide**, que l'on trouve sur le Linux Documentation Project. Cet ouvrage est très instructif dans la mesure où il apporte une première approche des réseaux en TCP/IP de conception classique, tels que vous en rencontrerez partout dans les petites/moyennes structures.

---

[1] sans trop s'étendre sur le sujet, disons simplement que les adresses IP « appartiennent » à l'hébergeur chez qui vous stockez votre serveur/votre site. Si vous changez d'hébergeur, votre serveur/site change d'adresse IP.

[2] En pratique c'est un peu plus complexe, mais vous connaissez maintenant les grandes lignes.

[3] En fait, c'est un raccourci rapide. `$ORIGIN` sert à créer un « raccourci », qui sera ajouté ensuite aux noms non terminés par un point. Ainsi, il serait théoriquement valable d'écrire :

```
 $ORIGIN fr
clx.anet    IN CNAME gaia.anet.fr.
```

Notez également que le mot clef `$ORIGIN` peut être mis n'importe où dans le fichier de zone.

[4] Network Administrator Guide.

[5] Voir la note 3.