Interface de configuration du futur BIND 10

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 8 octobre 2009

https://www.bortzmeyer.org/bind10-interface-config.html

Le développement de la prochaine version du serveur DNS BIND continue à suivre son petit bonhomme de chemin <https://www.bortzmeyer.org/bind10-avance.html>. Cette semaine, la discussion portait surtout sur l'interface de configuration, posant quelques problèmes intéressants, notamment pour les cas où BIND sert un très grand nombre de zones.

BIND est utilisé dans des cas très variés. Du serveur de noms de la racine, avec très peu de données mais énormément de trafic et un rôle critique, au serveur d'un fournisseur de services Internet, qui héberge 100 000 zones pour ses clients, en passant par le petit résolveur du petit réseau local de la petite entreprise. Il est difficile de concevoir une interface qui convient à tous les cas. Aujourd'hui, BIND version 9 se configure essentiellement via le fichier `named.conf` où on indique à la fois des informations de **contrôle** (comme `recursion no;` ou `listen-on { 127.0.0.1; };`) et des **données** comme la liste des zones servies. Dans la plupart des cas, ces données sont de taille raisonnable mais, pour un gros hébergeur DNS, il peut y avoir des centaines de milliers de zones servies (soit comme serveur maître, soit comme esclave), et la lecture du fichier de configuration, au démarrage comme au rechargement, prend un temps fou :

```
zone "example.net" IN {
type master;
file "pri/example.net.zone";
};
zone "example.org" IN {
type master;
file "pri/example.org.zone";
};
zone "example.com" IN {
type master;
file "pri/example.com.zone";
};
...
```

La configuration via un fichier, traditionnelle sur Unix, surtout pour un démon, a d'autres défauts. Tout changement doit se faire par la modification d'un fichier, puis un rechargement (opération, on l'a vu, parfois assez lente et pendant laquelle le serveur ne répond pas, ou plus lentement). Cela n'est pas pratique par rapport à une configuration interactive, comme celle des routeurs Cisco utilisant IOS.

La proposition des développeurs de BIND 10 était donc de remplacer le traditionnel `named.conf` par une telle interface interactive. L'administrateur système aurait alors tapé des requêtes de configuration, que BIND aurait sauvegardé sous un format non texte, par exemple une base de données SQLite.

Ce projet, présenté dans des réunions physiques en marge de la réunion RIPE à Lisbonne, a suscité, c'est le moins qu'on puisse dire, des réserves. Outre le fait qu'un tel projet heurtait des habitudes ancestrales, outre le fait qu'il mettait BIND à part de tous les autres démons Unix, ce mécanisme avait d'autres inconvénients comme de rendre impossible l'échange de fichiers de configuration (par exemple pour signaler une bogue ou un problème sur une liste de diffusion). Et les gérants de grands parcs de serveurs BIND craignaient la perspective de ne plus pouvoir générer un fichier de configuration avant de le pousser sur toutes les machines.

Des tas d'approches alternatives ont été discutées : par exemple s'inspirer de JunOS plutôt que de IOS, pour une interface interactive (avec navigation hiérarchique) ou bien permettre une exportation de la base de données sous forme texte pour les sauvegardes ou l'échange d'informations.

Mais l'idée qui est revenue le plus souvent était de regarder ce que font les autres démons Unix qui gèrent des milliers de domaines. En général, ils gardent le principe du fichier de configuration pour le contrôle et y renoncent pour les données.

Ainsi, Apache a plusieurs méthodes pour configurer divers domaines. La méthode de base est de les mettre dans les fichiers de configuration :

```
NameVirtualHost 198.51.100.44
<VirtualHost 198.51.100.44>
ServerName www.customer-1.com
DocumentRoot /www/hosts/www.customer-1.com/docs
</VirtualHost>

<VirtualHost 198.51.100.44>
ServerName www.customer-2.com
DocumentRoot /www/hosts/www.customer-2.com/docs
</VirtualHost>

# ...

<VirtualHost 198.51.100.44>
ServerName www.customer-N.com
DocumentRoot /www/hosts/www.customer-N.com/docs
</VirtualHost>
```

Mais il en existe d'autres, très bien décrites dans la documentation <http://httpd.apache.org/docs/2.0/vhosts/mass.html>. Avec le module Apache `mod_vhost_alias`, on utilise le système de fichiers comme base de données : la présence d'un répertoire indique la gestion d'un domaine de même nom, sans avoir besoin d'éditer un fichier ou de recharger Apache :

```
VirtualDocumentRoot /www/hosts/%0/docs

# Et il n'y a plus qu'à créer /www/hosts/www.customer-1.com,
# /www/hosts/www.customer-1.com, etc
```

Pour le serveur de courrier Postfix, il existe également plusieurs méthodes, décrites dans la documentation <http://www.postfix.org/VIRTUAL_README.html>. Toutes reposent sur l'usage des "*maps*", des tables qui associent une clé (typiquement une adresse de courrier) à une valeur, par exemple :

```
virtual_alias_domains = example.com ...
virtual_alias_maps = hash:/etc/postfix/virtual
...
# /etc/postfix/virtual
postmaster@example.com postmaster
info@example.com       joe
sales@example.com      jane
```

Ces tables peuvent être des fichiers texte, ou bien distribuées par des SGBD, LDAP, etc.

Bref, il en existe des méthodes pour configurer un serveur Internet.