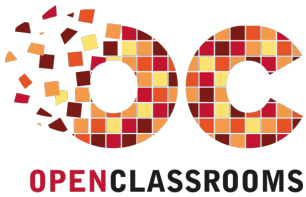
Installer un serveur DNS avec djbdns sous debian

Par elalitte



www.openclassrooms.com

Dernière mise à jour le 18/07/2011

Sommaire

Installer un serveur DNS avec djbdns sous debian

Par elalitte

Mise à jour : 18/07/2011
Difficulté : Difficile

Pourquoi ce tutoriel ?

Le DNS est un service vital sur Internet. Il est bien plus complexe que l'on peut le penser au premier abord, et sa mise en œuvre nécessite une attention particulière. Pour ceux qui ne sont pas convaincus, il y a un cours en elearning sur le DNS à votre disposition.

Par ailleurs, de nombreuses failles de sécurité ont existé, et existeront, sur les outils serveur proposés car ceux-ci sont vieux et peu évolutifs. Le choix de djbdns est donc judicieux, tant par sa simplicité d'utilisation que sa robustesse au niveau sécurité.

Enfin, je n'ai rencontré aucun tutoriel digne de ce nom qui soit capable d'expliquer en détail le fonctionnement et l'installation d'un serveur avec djbdns. Je vais donc m'efforcer d'être clair et d'expliquer les différentes commandes utilisées pour la mise en place, ainsi que les commandes permettant de tester l'installation au fur et à mesure.

Pour essayer d'expliquer les éléments importants, cela rend ce tutoriel un peu long, mais il vaut mieux passer un peu plus de temps à comprendre ce que l'on fait que mettre en place des services que l'on ne maîtrise pas !
Vous pouvez retrouver tous mes tutos sur www.lalitte.com.

Pourquoi djbdns ?

djbdns est un excellent programme. Il a été conçu par D.J.Bernstein, un enseignant en mathématiques américain très original et trublion dans le monde Internet.
Je trouve que djbdns est excellent car :

- Il marche !
- Il a été pensé et programmé pour être sécurisé.
- Il sépare les services de domaine et de cache qui n'ont rien à voir.

Rien que pour ces trois raisons, il mérite qu'on s'y attarde. Surtout quand on connait le passé de Bind, le serveur dns historique, qui n'a pas une très bonne réputation et qui continue de faire parler de lui après 30 ans de bons et loyaux services. Donc, que vous ayez déjà un serveur bind en place, ou que vous vouliez mettre en place un nouveau serveur, il vous faut installer djbdns.

Pré-requis

Ce tutoriel est de **difficulté élevée**, il est donc destiné à des zéros qui ont déjà quelques connaissances en réseau.
Cela implique notamment:

- De bonnes connaissances sur les bases des réseau TCP/IP, que vous pourrez apprendre ici
- De bonnes connaissances sur les principes du DNS, que vous pourrez apprendre ici
- Et enfin de bonnes connaissances sur les systèmes unix et notamment Debian ou Ubuntu, que vous pourrez apprendre ici

N'hésitez pas à poser des questions dans les commentaires si vous ne vous en sortez pas.

Installation et configuration

Environnement et périmètre

Je vous propose de faire l'installation de djbdns sur une distribution debian de linux. Tout simplement car cette distribution est excellente pour l'installation de services sécurisés et fiables, et que l'installation d'applications y est plutôt simple. Dans ce tutoriel je me contenterai de vous parler de l'installation d'un serveur dns de domaine, c'est à dire du module tinydns de djbdns. Je vous invite à lire le site de djbdns si vous voulez installer un serveur de cache avec dnscache. Nous allons faire nos commandes en tant que root pendant l'installation, nos amis sous ubuntu pourront le faire grâce à la commande sudo, ou tout simplement en passant root avec sudo -s.

Conventions

Toutes les commandes unix utilisées utiliseront la typologie suivante encadrée sur fond noir :

Code : Console

```
# whoami
```

Je considèrerai par ailleurs que vous avez une relativement bonne connaissance des systèmes unix pour pouvoir comprendre les commandes que j'indiquerai, même si j'en explique la plupart. Il vous faudra aussi un minimum de connaissances sur l'application DNS.

Installation de djbdns

Installation des paquets

Depuis la release nouvelle version de Debian, la 6.0 Squeeze, djbdns a été retiré de la liste des packages. Il sera remplacé par dbndns, une version de djbdns à la sauce debian, dans une future version.

Nous allons donc récupérer la future version de djbdns sous forme de package, et nous allons l'installer comme un package normal.

Tout d'abord, il faut récupérer le package:

Code : Console

```
# cd /opt
# wget http://ftp.fr.debian.org/debian/pool/main/d/djbdns/dbndns_1.05-
4+lenny1_i386.deb
```

Et on commence l'installation en récupérant les packages nécessaires au fonctionnement de djbdns.

On commence par mettre à jour notre liste locale (un bon réflexe à prendre)

Code : Console

```
# apt-get update
```

Et on installe la totale des packages liés à djbdns.

Code : Console

```
# apt-get install daemontools daemontools-run ucspi-tcp
```

daemontools est un ensemble d'application créé par D.J.Bernstein pour gérer les services unix, comme par exemple leur lancement ou les logs d'information.
ucspi-tcp fournit des outils permettant de créer des communication client-serveur de façon simple. C'est donc lui qui va servir à mettre notre application djbdns en écoute sur le réseau pour répondre aux requêtes dns.
Pour vérifier que vos paquets sont bien installés, vous pouvez tenter

Code : Console

```
# dpkg -l | grep daemon
```

Ce qui va lister tous les packages installés, mais n'afficher que les lignes qui contiennent la chaîne de caractères daemon
Vous devriez avoir un truc du genre :

Code : Console

```
ii  daemontools                          1:0.76-
3                     a collection of tools for managing UNIX services
ii  daemontools-run                      1:0.76-
3                     daemontools service supervision
```

Le ii indiquant bien que le paquet est sélectionné et installé.
Il nous reste à installer dbndns que nous avons récupéré.

Code : Console

```
# dpkg -i /opt/dbndns_1.05-4+lenny1_i386.deb
```

djbdns est installé !
Nous pouvons donc dès maintenant utiliser les commandes fournies avec djbdns, et nous n'allons pas nous en priver.

Création des utilisateurs

Comme djbdns a été pensé pour la sécurité, il ne tourne pas bêtement avec les droits root pour qu'en cas de compromission le vilain pirate ait encore un peu de boulot. Pour cela, il faut donc créer des comptes utilisateurs qui seront utilisés par djbdns pour lancer les services (root ne sera utilisé que temporairement pour lancer la socket réseau)
On prend par ailleurs bien garde à ne pas donner de shell valide à ces deux comptes en cas de compromission.

Code : Console

```
useradd -s /bin/false tinydns
useradd -s /bin/false dnslog
```

On peut vérifier que les comptes ont bien été créés en regardant le fichier /etc/passwd qui contient la liste des utilisateurs :

Code : Console

```
tail /etc/passwd
```

Et vous devriez voir quelque chose comme :

Code : Console

```
tinydns:x:1001:100::/home/tinydns:/bin/false
dnslog:x:1002:100::/home/dnslog:/bin/false
```

Ce qui prouve que vos utilisateurs sont bel et bien présents à l'appel.

Création des dossiers et répertoires

Nous allons utiliser la commande tinydns-conf qui va nous créer le répertoire de configuration de tinydns.

Code : Console

```
# tinydns-conf tinydns dnslog /etc/tinydns 192.168.0.1
```

Bien sûr, vous mettrez l'adresse IP de votre machine à la place de 192.168.0.1 .
Cette commande va créer tous les répertoires nécessaires au fonctionnement de tinydns dans /etc/tinydns. Le service sera lancé avec les droits de l'utilisateur indiqué dans la commande (tinydns ici) et chrooté dans le répertoire /etc/tinydns/root/. Si vous n'avez aucune idée de ce qu'est le chroot, sachez simplement que c'est une mesure de sécurité, ou faites des recherches dessus pour en savoir plus.
Elle va aussi créer des logs dans /etc/tinydns/log/main/ gérés par l'utilisateur dnslog. Vous pourrez ainsi y voir des messages d'erreur s'il y a un quelconque problème.

Lancement du service

Vous avez peut-être l'habitude de lancer des services sous linux avec des scripts de démarrage dans /etc/init.d/. Et bien pour tinydns, le mode de fonctionnement est extrêmement différent puisque le service est lancé est supervisé par daemontools qui a un fonctionnement original, mais très efficace et utile, notamment car il supervise un processus en permanence et tente de le relancer s'il est arrêté.
Nous allons créer un lien symbolique vers le répertoire d'installation de tinydns.

Code : Console

```
# mkdir /etc/service
# ln -s /etc/tinydns /etc/service/
```

On peut alors voir le lien créé :

Code : Console

```
# ls -la /etc/service/
```

Code : Console

```
lrwxrwxrwx  1 root root   12 2007-06-27 13:30 tinydns -> /etc/tinydns
```

Une fois que cela est fait, votre service est magiquement lancé !

Comment donc ? En fait, la commande svscan de daemontools scrute le répertoire /service et cherche dans les répertoires contenus un fichier run. Si ce fichier est présent, il l'exécute.
On peut voir le processus svscan lancé :

Code : Console

```
# ps auxw
root      2639  0.0  0.0    140    28 ?
        S     2007   0:05 svscan /etc/service
```

Dans notre cas, le répertoire /etc/service/tinydns/ contient le script run suivant :

Code : Console

```
#!/bin/sh
exec 2>&1
exec envuidgid tinydns envdir ./env softlimit -
d300000 /usr/local/bin/tinydns
```

Dans le langage de daemontools, cela veut dire de lancer le programme /usr/local/bin/tinydns sous l'identité de tinydns en prenant en compte les variables du répertoire /etc/tinydns/env/. Donc si tout se passe bien, notre service est lancé. On peut le voir avec ps :

Code : Console

```
# ps auwx
root      2640  0.0  0.0     96    16 ?
        S     2007   0:00 readproctitle service errors: ..........................
root      2641  0.0  0.0    108    24 ?        S     2007   0:00 supervise tinydns
```

Ou avec la commande adéquate :

Code : Console

```
# svstat /etc/service/*
/etc/service/tinydns: up (pid 2644) 61511978 seconds
```

Si vous voyez 0 ou 1 seconde, c'est mauvais signe, quelque chose se passe mal et tinydns redémarre en boucle. Dans ce cas allez voir les logs pour y trouver de l'information :

Code : Console

```
# tail -n 30 /etc/tinydns/log/main/current
```

Cela est souvent dû au fait d'avoir un problème de bind de la socket car l'adresse IP spécifiée n'existe pas ou un service DNS sur le port 53 UDP tourne déjà...
Et donc, si tout s'est bien passé, nous devrions voir le service en écoute sur la machine :

Code : Console

```
# netstat -anpe | grep 53
udp        0      0 88.191.51.73:53         0.0.0.0:*                           0
```

Cette commande affiche la liste des processus en écoute ainsi que le port et le protocole utilisés. Ici, tinydns est bien en écoute sur le port 53/UDP qui sert au DNS.

Configurer tinydns

Données de zone

Maintenant que nous avons vérifié que notre service était installé et qu'il était bien en écoute, il va nous falloir le configurer pour diffuser nos informations DNS.
Imaginons que nous possédions le domaine intechinfo.fr et que nous faisions tourner un serveur web, un serveur de messagerie et bien sûr, deux serveurs DNS. Nous allons voir comment configurer ce domaine sur notre serveur tinydns.

Préparation du fichier de zone

Les informations concernant notre zone vont se situer dans le fichier /etc/tinydns/root/data. Le format de ce fichier est un peu particulier. En gros, le premier caractère d'une ligne indique le type d'enregistrement DNS, et la suite le nom utilisé, l'adresse IP, etc.
Dans notre cas, il va nous falloir :

- Un enregistrement SOA pour définir notre zone
- Deux enregistrements NS pour nos deux serveurs dns de zone
- Un enregistrement A pour notre serveur web
- Un enregistrement MX pour notre serveur de messagerie
- Un enregistrement A pour le nom de notre serveur de messagerie

Il y a deux façons de créer le fichier de zone.

- En l'éditant directement à la main
- En utilisant les commandes fournies par djbdns

Nous allons faire les deux pour en avoir un aperçu.

Configuration à l'aide des commandes djbdns

Nous avons plusieurs commandes utiles dans le répertoire /etc/tinydns/root/, donc le nom est évocateur :

- add-ns (ajouter un serveur dns, soit un enregistrement NS)
- add-alias (ajouter un alias, soit un enregistrement CNAME)
- add-host (ajouter un host, soit un enregistrement A)
- add-mx (ajouter un serveur de messagerie, soit un enregistrement MX)
- add-childns (ajouter une zone fille, si on veut créer une sous-zone, par exemple etudiants.intechinfo.fr)

Nous allons donc les utiliser pour créer notre zone. En tapant la commande sans argument, nous obtenons une aide :

Code : Console

```
# cd /etc/tinydns/root
# ./add-ns
tinydns-edit: usage: tinydns-
edit data data.new add [ns|childns|host|alias|mx] domain a.b.c.d
```

Nous pouvons maintenant ajouter nos serveurs dns :

Code : Console

```
# ./add-ns intechinfo.fr 192.168.0.1
```

Et voir le résultat dans le fichier data :

Code : Console

```
# cat data
.intechinfo.fr:192.168.0.1:a:259200
```

Le . en début de ligne indique un enregistrement NS, puis il y a l'adresse IP de ce serveur, le nom associé et le timeout associé à cet enregistrement (3 jours ici, en secondes)
Vous pouvez utiliser ces commandes à votre guise, et nous allons voir maintenant comment entrer les informations directement dans le fichier de zone.

Configuration à la main

La syntaxe du fichier data n'est pas très complexe, et son contenu est expliqué sur le site de D.J.Bernstein. Nous allons utiliser vi pour configurer notre fichier data :

Code : Console

```
# vi /etc/tinydns/root/data
```

Et entrer les informations suivantes :

Code : Console

```
Zintechinfo.fr:sd-8131.dedibox.fr:lalitte.esiea.fr:2005042201:::::
.intechinfo.fr:192.168.0.1:dns1.intechinfo.fr:259200
.intechinfo.fr:192.168.0.2:dns2.intechinfo.fr:259200
+intechinfo.fr:192.168.0.1:86400
+www.intechinfo.fr:192.168.0.1:86400
@intechinfo.fr:192.168.0.1:mail.intechinfo.fr::86400
```

Nous avons sur la première ligne l'enregistrement SOA, puis les deux enregistrements NS, deux enregistrements A pour le nom du domaine et le serveur web, et enfin un enregistrement MX pour le serveur de messagerie (l'enregistrement A pour le nom du serveur de messagerie est automatiquement créé, comme pour les enregistrements NS d'ailleurs)
Et voilà, nous sommes enfin prêts ! ou presque...
Car malheureusement, tinydns ne parle pas notre langage et le fichier data a besoin d'être compilé pour être compris par tinydns.

Compilation du fichier data

Pour compiler, rien de plus simple, il suffit de faire la commande make (en étant dans le répertoire /etc/tinydns/root/)

Code : Console

```
# cd /etc/tinydns/root/
# make
```

Vous devriez avoir un nouveau fichier data.cdb qui est apparu :

Code : Console

```
# ls -la
drwxr-sr-x 2 root root  4096 2009-10-26 14:22 .
drwxr-sr-t 6 root root  4096 2007-06-27 17:54 ..
-rwxr-xr-x 1 root root    77 2007-06-27 17:54 add-alias
-rwxr-xr-x 1 root root    79 2007-06-27 17:54 add-childns
-rwxr-xr-x 1 root root    76 2007-06-27 17:54 add-host
-rwxr-xr-x 1 root root    74 2007-06-27 17:54 add-mx
-rwxr-xr-x 1 root root    74 2007-06-27 17:54 add-ns
-rw-r--r-- 1 root root 16620 2009-10-26 14:22 data
-rw-r--r-- 1 root root 39460 2009-09-08 10:52 data.cdb
-rw-r--r-- 1 root root   234 2008-06-23 18:17 Makefile
```

Ce fichier est illisible, mais il contient les données compilées du fichier data.
Nous pouvons maintenant tester si notre serveur dns répond bien à des requêtes pour notre domaine.

Interrogation du serveur

Il y a de nombreux outils pour interroger un serveur dns. host ou dig sous unix sont excellent pour cela, nous allons donc utiliser la commande dig. La syntaxe est relativement simple :

Code : Console

```
dig @@IPserveur requête type
```

Attention, il faut bien indiquer le @ suivi de l'adresse Ip du serveur ! Ce qui donne par exemple pour faire une requête NS sur notre serveur pour le domaine intechinfo.fr

Code : Console

```
# dig @192.168.0.1 intechinfo.fr NS
; <<>> DiG 9.3.4 <<>> @192.168.0.1. intechinfo.fr NS
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 56372
;; flags: qr aa rd; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 3
;; QUESTION SECTION:
;intechinfo.fr.                         IN        NS
;; ANSWER SECTION:
intechinfo.fr.              259200        IN        NS        ns1.intechinfo.fr.
intechinfo.fr.              259200        IN        NS        ns2.intechinfo.fr.
;; ADDITIONAL SECTION:
ns1.intechinfo.fr.        259200        IN        A        192.168.0.1
ns2.intechinfo.fr.        259200        IN        A        192.168.0.2
```

```
;; Query time: 1 msec
;; SERVER: 192.168.0.1#53(192.168.0.1)
;; WHEN: Mon Oct 26 14:49:22 2009
;; MSG SIZE  rcvd: 127
```

Ça marche ! Notre serveur répond bien pour notre zone intechinfo.fr.

Arrêter ou relancer le service

Normalement, si vous avez fait l'installation avec apt, un script de démarrage aura été placé dans /etc/init.d et vous pourrez facilement relancer tinydns comme d'habitude sous Debian avec un bon :

Code : Console

```
# /etc/init.d/tinydns restart
```

Mais si vous l'avez installé par compilation, ou que vous voulez respecter l'esprit de D.J.Bernstein, vous pouvez utiliser les commandes sv* pour gérer votre processus tinydns.
Pour arrêter le service :

Code : Console

```
svc -d /service/tinydns
```

Pour le relancer :

Code : Console

```
svc -u /service/tinydns
```

Pour recharger la configuration :

Code : Console

```
svc -h /service/tinydns
```

Il ne nous reste plus qu'à acheter un nom de domaine et à le gérer !
Pour ceux qui ont l'habitude de bind, djbdns est très déroutant car différent, aussi bien dans son installation que sa configuration ou son fonctionnement.

Cependant, cela vaut le coup de s'y essayer, surtout quand on voit les failles successives auxquelles bind a encore affaire...

Partager