Bind 9 - Présentation

service DNS

Exemple de configuration

Serveur de nom cache uniquement

```
// 2 sous réseaux autorisés à effectués des requêtes:
acl corpnets { 192.168.4.0/24 ; 192.168.7.0/24 ; }
options {
 directory { "/etc/namedb" ; };
 allow-query { corpnets ; };
};

zone "0.0.127.in-addr.arpa" {
 type master;
 file "localhost.rev";
 notify no;
};
```

Serveur de nom autoritatif uniquement :

```
options {
 directory "/etc/namedb";
 allow-query-cache { none ; };
 allow-query { any; };
 recursive no;
};

zone "0.0.127.in-addr.arpa" {
 type master;
 file "localhost.rev";
 notify no;
};

zone "example.com";
 type master;
 file "example.com.db";
 allow-transfert {
  192.168.4.14;
  192.168.5.53;
 };
};

zone "eng.example.com" {
 type slave;
 file "eng.example.com.bk";
 masters { 192.168.4.12; };
};
```

Load Balancing

Le load balancing peut-être fait en utilisant plusieurs enregistrements pour un seul nom.

Notification

La notification DNS est un mécanisme qui permet aux serveurs maître de notifier leurs esclaves des changements de données de zone. En réponse à une notification d'un maître, l'esclave va vérifier si sa version de la zone est la version courante, et dans le cas contraire, initier un transfert de zone.

Mise à jour automatique

Dynamic update est une méthode pour ajouter, remplacer ou supprimer des enregistrements dans un serveur maître en envoyant des messages DNS spécifique.

Le dynamic update est activé avec options **allow-update** ou une clause **update-policy** dans la déclaration de zone.

Si update-policy est à **local**, les mises à jour de la zone seront permises pour la clé **local-ddns**, qui sera généré par named au démarrage.

Les clauses **tkey-gssapi-credential** et **tkey-domain** dans les options autorisent le serveur de négocier les clés qui peut correspondre à celles dans update-policy ou allow-update.

Le fichier journal

Tous les changements faits dans une zone en utilisant le dynamic update sont stockés dans un fichier journal de zone. Ce fichier est automatiquement crée par le serveur lors de la première mise à jour automatique effectuée. Ce journal possède l'extension .jnl et est au format binaire.

Le serveur va occasionnellement écrire le contenu complet de la zone mise à jour dans son fichier de zone (toutes les 15 minutes). Quand un serveur redémarre après un crash, il va relire le journal et mettre à jour la zone si besoin.

Les fichiers de zone dynamique ne peuvent pas être édités manuellement. La seul manière de s'assurer que le fichier de zone d'une zone dynamique est à jour est d'utiliser rndc stop.

Si vous voulez effectuer des changements dans une zone dynamique manuellement, la procédure suivante fonctionne : désactiver le dynamic update de la zone en utilisant **rndc freeze ZONE**, ce qui va supprimer le journal et mettre à jour le fichier master. après modification, lancer **rndc thaw ZONE** pour recharger le zone changée et réactiver le dynamic update.

Transfert de zone incrémental

Le protocole **ICFR** est une manière pour les serveurs esclaves de transférer uniquement les données changées, au lieu de transférer tout la zone. En agissant comme master, BIND 9 supporte ce protocole pour ces zones où l'historique des changements est disponible. Cela inclus les zones maîtres maintenues par dynamic update et les zones esclaves dont les données ont été obtenues par **IXFR**. Pour les zones maîtres maintenues manuellement, et pour les zones esclaves obtenues par transfert de zone complet (**AXFR**), **IXFR** est supporté seulement si l'option **ixfr-from-differences** est à yes. En agissant comme esclave, BIND 9 va tenter d'utiliser IXFR par défaut.

Split DNS

Le split DNS consiste à paramétrer différentes vues, ou visibilitées, de l'espace DNS pour les resorvers interne et externe. Exemple de split DNS avec 2 zones public et 2 zone interne :

Configuration du serveur DNS interne :

```
acl internals { 172.16.72.0/24 ; 192.168.1.0/24 ; };
acl externals { bastion-ips-go-here ; };

options {
 ...
 ...
 forward only;
 forwarders { bastion-ips-go-here ; };
 allow-transfer { none ; };
 allow-query { internals; externals; };
 allow-recursion { internals ; };
 ...
 ...
};

zone "site1.example.com" {
 type master;
 file "site1.example.com";
 forwarders {};
 allow-query { internals ; externals ; };
 allow-transfer { internals ; };
};

zone "site2.example.com" {
 type slave;
 file "site2.example.com";
 masters { 172.16.72.3; };
 forwarders {};
 allow-query { internals; externals; };
 allow-transfer { internals; };
};

zone "site1.internal" {
 type master ;
 file "site1.internal" ;
 forwarders {};
 allow-query { internals ; };
 allow-transfer { internals ; };
;

zone "site2.internal" {
 type slave ;
 file "site2.internal" ;
 masters { 172.16.72.3 ; };
 forwarders {};
 allow-query { internals ; };
 allow-transfer { internals ; };
};
```

Configuration du serveur DNS externe :

```
acl internals { 172.16.72.0/24 ; 192.168.1.0/24 ; };
```

```
acl externals { bastion-ips-go-here ; };

options {
 ...
 ...
 allow-transfer { none ; };
 allow-query { any ; };
 allow-query-cache { internals ; externals ; };
 allow-recursion { internals ; externals ; };
 ...
 ...
};

zone "site1.example.com" {
 type master;
 file "site1.example.com";
 allow-transfer { internals; externals; };
};

zone "site2.example.com" {
 type slave;
 file "site2.example.com";
 masters { another_bastion_host_maybe ; };
 allow-transfer { internals; externals; };
};
```

TSIG

BIND supporte TSIG principalement pour les communication serveur à serveur. Cela inclus le transfert de zone, notification, et message query récursifs. TSIG peut être également utile pour le dynamic update. Le programme nsupdate supporte TSIG. Un secret partagé est généré pour être partagé entre 2 hôtes. Un nom de clé arbitraire est choisis "host1-host2".

La commande suivante va générer une clé 128-bits HMAC-SHA256. la longueur max est de 256-bits.
dnssec-keygen -a hmac-sha256 -b 128 -n HOST host1-host2
la clé est dans le fichier Khost1-host2.+163+00000.private. Rien n'utilise directement ce fichier, mais le chaîne encodée en base64 suivant "Key :" peut être extrait de ce fichier et utilisé comme clé partagées.

Imaginons 2 serveurs host1 et host2. Ajouter dans named.conf de chaque serveur :

```
key host1-host2. {
 algorithm hmac-sha256;
 secret "La/E5CjG9O+os1jq0a2jdA==";
};
```

Vu que les clés sont partagées entre 2 hôtes uniquement, le serveur doit savoir quand utiliser la clé. Ajouter dans named.conf

```
server 10.1.2.3 {
 keys { host1-host2. ; };
};
```

Contrôle d'accès basé sur les clés TSIG

BIND permet aux adresses et plages IP d'être spécifiées dans des ACL et les directives allow- query | transfer | update. Cela a été étendue pour les clés TSIG également :

allow-update { key host1-host2. ; } ;

Cela permet aux dynamic update de s'effectuer uniquement si la requête est signée par al clé spécifiée.

TKEY

TKEY est un mécanisme pour générer automatiquement un secret partagée entre 2 hôtes. Il y'a de nombreux modes de TKEY qui spécifient comment la clé est générée ou assignée. BIND 9 implémente seulement un de ces modes, l'échange de clé diffie-Hellman. Les 2 hôtes requièrent d'avoir un record KEY diffie-hellman. Le processus TKEY doit utiliser des messages signés, signés soit par TSIG ou SIG(0). Le résultat de TKEY est un secret partagé qui peut être aussi utilisé pour supprimer des secrets partagées.

Le processus TKEY est initialisé par un client ou un serveur en envoyant un query TKEY signé. La réponse du serveur, en cas de succès, contient un record TKEY et un clé appropriée. Après cet échange, tous les participants ont suffisamment d'information pour déterminer le secret partagé. Le processus dépend du mode TKEY. En utilisant le mode Diffie-Hellman, les clés Diffie-Hellman sont échangées, et le secret partagé est dérivé par les participants.

SIG(0)

BIND 9 supporte partiellement les signatures de transaction DNSSEC SIG(0). SIG(0) utilise des clés public/privée pour authentifier les messages. Le contrôle d'accès est effectué de la même manière que les clé TSIG ; Les privilèges peuvent être donnée ou refusés en fonction du nom de clé. Quand un message signé SIG(0) est reçu, il va seulement être vérifié si la clé est connue et sûre par le serveur, le serveur ne tentera pas de validation de la clé.

DNSSEC

L'authentification cryptographique d'information DNS est possible au travers des extensions DNSSEC. Pour paramétrer une zone DNSSEC, il y'a une série d'étapes qui doivent être suivies. BIND 9 intègre plusieurs outils qui sont utilisés dans ce processus.

Génération des clés

le programme **dnssec-keygen** est utilisé pour générer les clés. Une zone sécurisée doit contenir une ou plusieurs clés de zone. Les clés de zone vont signer tous les enregistrements dans la zone. Les clés de zone doivent avoir le même nom que la zone. Actuellement le seul algorithme utilisé est RSASHA1.

Cette commande génère une clé 768-bits RSASHA1 pourla zone child.example.zone :

dnssec-keygen -a RSASHA1 -b 768 -n ZONE child.example.

2 fichiers de sortie sont produits : **Kchild.example.+005+12345.key** et **Kchild.example.+005+12345.private**. Le nom des fichiers de clé contient le nom de clé, l'algorithme (3 pour DSA, 1 pour RSAMD5 et 5 pour RSASHA1) et le tag de clé. La clé privée est utilisée pour générer des signatures, et la clé publique est utilisée pour vérifier la signature. Pour générer une autre clé avec les même propriétés (mais avec un tag de clé différent), répéter la commande. le programme **dnssec-keyfromlabel** est utilisé pour obtenir une paire de clé depuis une cryptographie hardware et construire les fichiers de clé. il fonctionne similairement à **dnssec-keygen**. Les clés publiques devraient être insérées dans le fichier de zone en incluant les fichiers .key en utilisant $INCLUDE.

Signer la zone

dnssec-signzone est utilisé pour signer une zone. Les fichiers keyset correspondant à une sous-zone sécurisée devraient être présents. Le signataire de zone va générer des records NSEC, NSEC3 et RRSIG pour la zone, et un DS pour les zones enfants si -g est spécifié.

La commande suivante signe la zone, assumant qu'il est dans un fichier appelé zone.child.example. Par défaut, toutes les clés de zones qui ont une clé privée disponible sont utilisés pour générer les signatures

dnssec-signzone -o child.example zone.child.example

Un fichier zone.child.example.signed sera produit. Ce fichier devrait être référencé par named.conf comme fichier d'entrée pour la zone.

dnssec-signzone va aussi produire des fichiers keyset et dsset et optionnellement un fichier dlvset. Ils sont utilisés pour la zone parent avec les DNSKEY (ou leur record DS correspondant) qui sont le point d'entrée sécurisé de la zone.

Configurer les serveurs

Pour permettre à named de répondre aux requêtes depuis des clients DNSSEC, **dnssec-enable** doit être à yes. Pour permettre à named de valider les réponses depuis d'autres serveurs, **dnssec-enable** et **dnssec-validation** doivent être à yes, et au moins en groupe trust doit être configuré avec les options **trusted-keys** ou **managed-keys**.

Les trusted-keys sont des copies des **DNSKEY RR** pour les zones qui sont utilisés pour former le premier lien dans la chaîne cryptographique. Toutes les clés listées dans **trusted-keys** sont considérées pour exister et seul les clés listées seront utilisés pour valider le DNSKEY RRset.

managed-keys sont des clés de confiance qui sont automatiquement mis à jours via un groupe de maintenance de confiance.

Une fois que DNSSEC est établit, une configuration DNSSEC typique va ressembler à la configuration ci-dessous. Il y'a une ou plusieurs clés publiques pour le root. Cela permet aux réponses en dehors de l'organisation d'être validées. Il y'a aussi plusieurs clés pour les parties de l'espace de nom que l'organisation contrôle.

```
managed-keys {
 /* root Key */
 "." initial-key 257 3 3 "BNY4wrWM1nCfJ+CXd0rVXyYmobt7sEEfK3clRbGaTwS
  JxrGkxJWoZu6I7PzJu/E9gx4UC1zGAHlXKdE4zYIpRh
  aBKnvcC2U9mZhkdUpd1Vso/HAdjNe8LmMlnzY3zy2Xy
  4klWOADTPzSv9eamj8V18PHGjBLaVtYvk/ln5ZApjYg
  hf+6fElrmLkdaz MQ2OCnACR817DF4BBa7UR/beDHyp
  5iWTXWSi6XmoJLbG9Scqc7l70KDqlvXR3M/lUUVRbke
  g1IPJSidmK3ZyCllh4XSKbje/45SKucHgnwU5jefMtq
  66gKodQj+MiA21AfUVe7u99WzTLzY3qlxDhxYQQ20FQ
  97S+LKUTpQcq27R7AT3/V5hRQxScINqwcz4jYqZD2fQ
  dgxbcDTClU0CRBdiieyLMNzXG3";
};
trusted-keys {
 /* Key for our organization's forward zone */
 example.com. 257 3 5 "AwEAAaxPMcR2x0HbQV4WeZB6oEDX+r0QM6
  5KbhTjrW1ZaARmPhEZZe3Y9ifgEuq7vZ/z
  GZUdEGNWy+JZzus0lUptwgjGwhUS1558Hb
  4JKUbbOTcM8pwXlj0EiX3oDFVmjHO444gL
  kBOUKUf/mC7HvfwYH/Be22GnClrinKJp1O
  g4ywzO9WglMk7jbfW33gUKvirTHr25GL7S
  TQUzBb5Usxt8lgnyTUHs1t3JwCY5hKZ6Cq
  FxmAVZP20igTixin/1LcrgX/KMEGd/biuv
  F4qJCyduieHukuY3H4XMAcR+xia2nIUPvm
  /oyWR8BW/hWdzOvnSCThlHf3xiYleDbt/o
  1OTQ09A0=";
/bin /boot /dev /etc /home /lib /lib64 /lost+found /media /mnt /opt /proc /root /run /sbin /srv /sys
/@System.solv /tmp /usr /var Key for our reverse zone. */
```

```
2.0.192.IN-ADDRPA.NET. 257 3 5 "AQOnS4xn/IgOUpBPJ3bogzwc
  xOdNax071L18QqZnQQQAVVr+i
  LhGTnNGp3HoWQLUIzKrJVZ3zg
  gy3WwNT6kZo6c0tszYqbtvchm
  gQC8CzKojM/W16i6MG/eafGU3
  siaOdS0yOI6BgPsw+YZdzlYMa
  IJGf4M4dyoKIhzdZyQ2bYQrjy
  Q4LB0lC7aOnsMyYKHHYeRvPxj
  IQXmdqgOJGq+vsevG06zW+1xg
  YJh9rCIfnm1GX/KMgxLPG2vXT
  D/RnLX+D3T3UL7HJYHJhAZD5L
  59VvjSPsZJHeDCUyWYrvPZesZ
  DIRvhDD52SKvbheeTJUm6Ehkz
  ytNN2SN96QRk8j/iI8ib";
};
options {
 ...
 dnssec-enable yes;
 dnssec-validation yes;
};
```

DNSSEC, zones dynamiques, et auto-signature

Il est possible de changer une zone dynamique non-sécurisée vers une zone signée, et inversement. Une zone sécurisée peut utiliser soit NSEC soit NSEC3.

Convertir en zone sécurisée

Celà peut être fait de 2 manières : utiliser un DNS dynamic update, ou l'option de zone auto-dnssec.

```
zone example.net {
 type master;
 update-policy local;
 file "dynamic/example.net";
 key.directory "/dynamic";
};
```

si un KSK et une clé ZSK DNSKEY ont été générés, cette configuration va forcer tous les records dans la zone à être signés avec ZSK, et le DNSKEY RRset d'être signé avec KSK.

Méthode de mise à jour DNS Dynamic

Pour insérer les clés via dynamic update

% nsupdate
> ttl 3600
> update add example.net DNSKEY 256 3 7 AwEAAZn17pUF0KpbPA2c7Gz76Vb18v0teKT3E
> update add example.net DNSKEY 257 3 7 AwEAAd/7odU/64o2LGsifbLtQmtO8dFDtTAZX
> send

La requête va se compléter immédiatement, lazone ne sera pas complètement signée jusqu'à ce que named ait eu le temps de traverser la

zone et générer les records NSEC et RRSIG.

Si vous voulez utiliser NSEC3 au lieu de NSEC, vous devriez ajouter un record NSEC3PARAM à la requête de mise à jour initiale. Si vous que la chaîne NSEC3 ai le bit OPTOUT mis, le définir dans les champs de flag du record NSEC3PARAM.

```
% nsupdate
> ttl 3600
> update add example.net DNSKEY 256 3 7 AwEAAZn17pUF0KpbPA2c7Gz76Vb18v0teKT3EyAGfB
> update add example.net DNSKEY 257 3 7 AwEAAd/7odU/64o2LGsifbLtQmtO8dFDtTAZXSX2+X
> update add example.net NSEC3PARAM 1 1 100 1234567890
> send
```

De même, cette requête sera complétée immédiatement ; cependant le record se sera pas visible tant que named n'a pas eu une chance de construire/supprimer le chaîne.

Signer une zone automatiquement

Pour activer la signture automatique, ajouter **auto-dnssec** à la déclaration de zone. Cette options a 2 arguments possibles : allow et maintain.

Avec **auto-dnssec allow**, named peut rechercher le répertoire de clé pour les clés correspondant à la zone, les insérer dans la zone, et les utiliser pour signer la zone. Il le fera uniquement quand il recevra la commande **rndc sign <zone-name>**

auto-dnssec maintain inclue la même fonctionnalité, mais va également ajuster automatiquement les records DNSKEY de la zone en accord