Déployer DNSSEC, comment, quoi, où ?

(Version 3 – Juin 2015)

Association Française pour le Nommage Internet en Coopération | **www.afnic.fr** | **contact@afnic.fr**
Twitter : **@AFNIC** | Facebook : **afnic.fr**

Table des matières

Déployer DNSSEC, comment, quoi, où ?

Avertissement

Le contenu de ce document est fourni «TEL QUEL».

Les informations présentes sont susceptibles de contenir des inexactitudes techniques, erreurs typographiques ou des informations obsolètes. Ce document peut être mis à jour ou modifié sans préavis, à tout moment, dans son format électronique en ligne sur le site web de l'Afnic.

L'utilisation de l'information contenue dans ce document est donc à vos propres risques.
En aucun cas, l'Afnic ne pourra être tenue responsable de toute perte ou dommage, y compris mais sans s'y limiter, aux pertes indirectes ou dommages indirects, ou de toute perte ou dommage consécutifs à des opérations liées à l'utilisation ou reproduction des exemples présents dans ce document.

Ce document ne dispense pas son lecteur des formations techniques nécessaires à la bonne compréhension et l'utilisation des ressources et logiciels mentionnés.

Préambule

En tant qu'acteur-clé de l'Internet français, l'Afnic souhaite jouer un rôle de premier plan dans le déploiement de DNSSEC (*Domain Name System Security Extensions*) en France. Ainsi, l'Office d'enregistrement du .fr a élaboré un plan stratégique pluriannuel, visant à accélérer le déploiement de cette technologie sous la zone .fr.

Les objectifs liés à ce déploiement sont les suivants :

- rendre l'Internet français plus sûr ;
- encourager la création de nouveaux services innovants tirant parti de cette chaîne de confiance ;
- aider à renforcer l'image des acteurs du .fr comme acteurs de référence dans la sécurité des infrastructures de communications sur Internet.

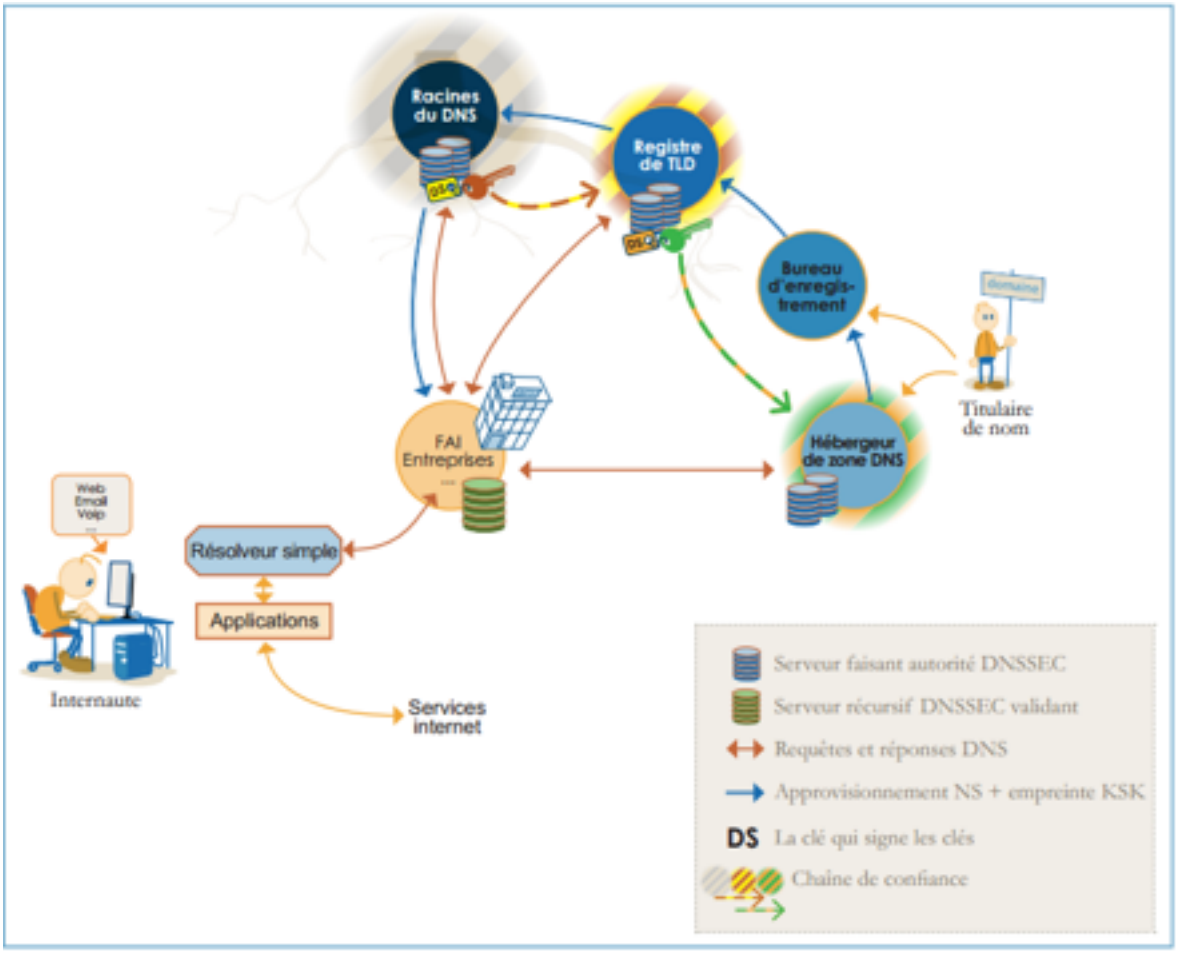
Cette documentation se veut opérationnelle, enrichie d'exemples concrets, et a pour objectif d'aider les différents hébergeurs de DNS dans la mise en oeuvre de DNSSEC.

Pour toute information supplémentaire sur les actions de l'AFNIC relatives à DNSSEC ou commentaires sur ce document, n'hésitez pas à contacter votre Chargé de Clientèle ou écrire à support@afnic.fr.

1. Introduction

DNSSEC a pour vocation de protéger la résolution DNS contre un certain nombre d'attaques. Cette technologie reste aujourd'hui la seule solution reconnue et effective contre une attaque de type **empoisonnement de cache** (*cache poisoning)*. Néanmoins, son déploiement nécessite un certain nombre de précautions, car, comme toute technique de sécurisation, une mauvaise mise en oeuvre risque de provoquer des incidents techniques/opérationnels majeurs.

Ce document est destiné aux administrateurs système d'un hébergeur DNS[1] souhaitant déployer DNSSEC afin de signer les zones qu'ils gèrent en minimisant le risque de problèmes. Pour cela, l'orientation de ce guide se veut didactique et privilégiant les solutions résistantes aux erreurs d'exploitation plutôt que celles qui résistent aux attaques massives. Certains choix pourront donc être simplifiés dans cette documentation afin de permettre une mise en place rapide.



Composantes de la chaine de confiance DNSSEC

[1] Qui peut être ou ne pas être un Bureau d'Enregistrement (Registrar).

Le déroulé de ce guide partira des prérequis qu'il convient de valider avant d'envisager un déploiement de DNSSEC en production, puis exposera deux exemples concrets de gestion de DNSSEC avec les logiciels OpenDNSSEC+NSD ou avec BIND.

Ensuite le document décrira les étapes nécessaires à la mise en place de DNSSEC en production.

En effet la mise en œuvre de DNSSEC implique pour chaque zone de :

- Créer des clés.
- Signer ses enregistrements.
- Publier la zone signée.
- Gérer des périodes de validité.
- Gérer les publications du résumé de la clé dans la zone parente avec chaque rotation de KSK.
- Contrôler la publication d'une nouvelle clé avant de l'utiliser pour signer.

Attention : La configuration de DNSSEC sur les résolveurs DNS (activer la validation) n'est pas traitée dans ce document, qui se concentre sur les serveurs faisant autorité.

2. Prérequis

Cette section est nécessaire dans la mesure où elle indique des conditions à satisfaire préalablement à un déploiement de DNSSEC en production.

À ce stade, un bagage minimum de connaissances théoriques et de compétences pratiques est nécessaire.

Pour les connaissances théoriques, il convient de consulter certains documents qui expliquent ce que DNSSEC est et ce qu'il apporte en matière d'extensions de sécurité au DNS. À titre indicatif et non limitatif, le lecteur peut se référer au dossier thématique Afnic (*afnic.dnssec-thema* : §5.Bibliographie) et au document publié aux JRES (*bortzmeyer.dnssec-jres* : §5.Bibliographie).

Quant aux compétences pratiques, outre l'auto-formation, le lecteur peut bénéficier de la **formation DNSSEC proposée en partenariat Afnic-HSC, qui allie cours théoriques et travaux pratiques**[2].

DNSSEC change le modèle de fonctionnement du DNS.

Sans DNSSEC, le DNS est en général dit *statique*. Il est initialement configuré, testé avec un logiciel comme Zonemaster, puis ne nécessite plus d'interventions particulières en dehors des modifications volontaires.

[2] **https://www.afnic.fr/fr/produits-et-services/formations/formation-dnssec-en-partenariat-avec-hsc-3.html**

DNSSEC oblige à penser différemment.

En raison de l'expiration des signatures DNSSEC[3], il est important de re-signer périodiquement. En outre, il peut être utile de changer les clés de temps en temps, notamment pour être prêt si un changement est soudainement nécessaire. Si on ne fait jamais de changements, on sera fort dépourvu le jour où un changement s'impose (en cas de perte ou de compromission d'une clé, par exemple). Dans ce cas, il faut aussi gérer le remplacement de ces clés en complément des signatures.

Aussi bien pour les signatures que pour les clés, les changements doivent se faire en respectant les délais imposés par les caches DNS. Ces caches DNS peuvent en effet garder l'information en mémoire et empêchent donc tout changement immédiat.

Avec DNSSEC, il faut avoir une perspective temporelle.

Les signatures doivent être re-générées au moins $TTL[4] avant leur expiration, pour tenir compte des problématiques de cache précitées. Cette opération est difficilement réalisable à la main[5] et nécessite donc une automatisation du processus.

La suite des prérequis s'adresse aux personnes souhaitant un déploiement en production. Dans le cas d'une simple expérimentation de la technologie DNSSEC, vous pouvez directement passer aux exemples concrets.

2.1.Ayez une configuration technique correcte

Avec le DNS « traditionnel », tout marche même en cas d'erreur, en raison de la robustesse de ce protocole.
DNSSEC représente un compromis différent : on diminue un peu la robustesse au profit de meilleures garanties d'intégrité. Il est donc nécessaire de bien comprendre et se former à DNSSEC.

Un prérequis fondamental : tester l'exactitude de votre configuration DNS.
Il existe une multitude d'outils[6] en ligne pour cela. Nous nous concentrerons sur **Zonemaster.**[7]

Attention : Vérifiez bien tous les points indiqués par Zonemaster. Si certains vous semblent obscurs, prenez le temps de les étudier et de les comprendre. Rappelez-vous, DNSSEC nécessite une bonne compréhension de l'ensemble des mécanismes.

---

[3] Mais pas les clés : c'est une grosse différence avec X.509.

[4] $TTL : le *Time To Live* des enregistrements DNS.

[5] Des exemples des problèmes se produisant en pratique figure dans **bortzmeyer.dnssec-satin.**

[6] **http://www.bortzmeyer.org/tests-dns.html**

[7] **http://www.zonemaster.fr/**

Notamment, les points suivants doivent être testés :

- que les serveurs faisant autorité répondent bien tous en EDNS ;
- qu'ils peuvent effectivement envoyer des réponses de taille supérieure à 512 octets[8] ;
- qu'ils peuvent effectivement envoyer des réponses de taille supérieure à la MTU du lien ;
- qu'ils peuvent répondre en TCP.

DNSSEC est un protocole relativement ancien (la norme actuelle date de 2005) mais certains logiciels n'ont intégré DNSSEC complètement, et sans bogues que depuis deux ou trois ans.
Il est donc essentiel de n'utiliser que des logiciels récents.
Si, pour une raison d'administration système, vous êtes provisoirement contraint de n'utiliser que des versions anciennes, il est alors plus prudent de requalifier / replanifier vos projets de mise en oeuvre de DNSSEC.

2.2.Supervisez...

Un système DNS fiable nécessite une supervision.
Même s'il est parfaitement configuré, les choses peuvent changer par la suite, un serveur peut « crasher », un pare-feu être (mal) reconfiguré, etc.
Tous les hébergeurs DNS ont une supervision en place avec des outils comme Nagios par exemple. Cette supervision s'avère encore plus nécessaire avec DNSSEC.

Un exemple de problème détectable grâce à des outils de supervision est par exemple le filtrage d'accès : si un serveur esclave ne se met plus à jour en raison d'un filtrage accidentel de son accès, avec le DNS traditionnel, la seule conséquence sera la distribution d'informations éventuellement obsolètes. Mais, avec DNSSEC, lorsque les signatures stockées sur ce serveur auront expiré, toute la zone sera invalidée. Il faut donc être informé rapidement afin de pouvoir y remédier. Des tests supplémentaires doivent aussi être prévus tels que :

- tous les serveurs répondent avec les signatures ;
- que les signatures ne soient pas proches de l'expiration ;
- etc.

Un exemple de configuration de la supervision figure dans la section TODO.

2.3.Synchronisez vos horloges

DNSSEC dépend d'horloges qui doivent être à l'heure exacte puisque les signatures contiennent une date de début et une date de fin de validité. Si vous signez sur une machine dont l'horloge est en retard, vos signatures pourraient être considérées comme expirées avant même que vous ne les publiiez. Bien sûr, avoir des machines

[8] Ancienne limite de taille du DNS, supprimée en 1999.

à l'heure fait partie des bonnes pratiques depuis de nombreuses années mais, avant DNSSEC, le non-respect de ces recommandations n'avait pas forcément de conséquences visibles.

Les horloges doivent donc être maintenues à l'heure, par exemple via NTP, et ce maintien doit être supervisé par un logiciel de supervision[9].

2.4. Sécurisez votre stockage

DNSSEC utilise **la cryptographie** et, à cet égard, reprend les mêmes problématiques que d'autres systèmes de cryptographie, par exemple les certificats X.509. Ainsi, il faut garder les clés privées... privées, afin d'éviter leur copie par un attaquant, tout en ayant des sauvegardes qui permettront de les récupérer si le système de stockage est défaillant.

Si vous gérez déjà des systèmes cryptographiques, DNSSEC ne vous surprendra pas.

Si vous débutez la cryptographie avec DNSSEC, prudence. Une clé DNSSEC privée stockée en mode « lecture pour tous » sur une machine qui est également un serveur Web public avec plein de scripts programmés sans souci de sécurité est sans doute à éviter.

3.Exemples concrets

3.1.Des choix à faire

3.1.1.La gestion des clés

Deux choix importants seront à faire au sujet de la gestion des clés.

D'abord, faut-il une clé ou deux pour la zone ? À lire beaucoup de textes sur DNSSEC, on peut avoir l'impression que la séparation en deux clés, la KSK (*Key Signing Key*) et la ZSK (*Zone Signing Key*) est obligatoire. Mais ce n'est pas le cas[10]. On peut très bien n'avoir qu'une seule clé[11]. Les conseils que nous donnons sont « si vous gérez vos zones à la main, comme dans l'exemple BIND ci-dessous, n'utilisez qu'une seule clé » et « si vous utilisez un outil de gestion de clés comme OpenDNSSEC, n'hésitez pas à avoir deux clés, l'outil s'occupera de tout ». Il est à noter que le choix du logiciel BIND ou

9 Par exemple avec le script `check_ntp_peer` pour Nagios et les compatibles.

10 La plus grosse zone avec une seule clé est co.uk.

11 Qui est parfois appelée CSK (*Combined Signing Key*).

d'OpenDNSSEC est a priori indépendant du choix du nombre de clés, les deux outils permettant d'implanter les deux stratégies de gestion de clés[12].

Seconde décision à prendre sur la gestion des clés : où conserve-t-on les clés ? La méthode la moins sûre est lorsque les clés sont simplement stockées sur le serveur qui fait les signatures. La plus sûre est celle d'un HSM (*Hardware Security Module*), un ordinateur spécialisé et durci qui stocke les clés et réalise les signatures. Entre les deux, on peut avoir des clés qui sont gardées sur un support USB enfermé dans un coffre-fort et sorti les jours où l'on signe (soit parce qu'on a modifié les données, soit parce que l'on doit re-signer avant expiration).

L'usage des HSM nécessite un investissement pour leur acquisition et leur appropriation technique (formation). La décision de les utiliser ou pas est a priori guidée par un arbitrage sur les coûts associés aux risques à couvrir/ accepter. En pratique, dans le cas de zones considérées importantes/critiques par l'opérateur DNS, ce dernier investit dans des HSM. Dans le cas contraire (zones « ordinaires »), d'autres solutions sont privilégiées.

La solution du coffre-fort est tentante, mais manque de souplesse, elle interdit d'utiliser les outils de re-signature automatiques, comme ceux présentés dans la prochaine section. Cette solution augmente certes le niveau de protection de la clé de signature contre le vol/compromission, mais elle induit un sur-coût en tâches d'administration : mise en ligne manuelle et périodique de la clé pour les signatures et installation d'un système de rappel/surveillance des signatures pour éviter l'oubli de re-signer[13].

Une solution, la plus pratique dans le cas de zones ordinaires, est de garder les clés sur le serveur de signature. Certes, c'est la moins sûre parmi toutes celles mentionnées dans ce document, mais, actuellement, cette solution a l'avantage de la simplicité ainsi que celui d'éviter de garder le statu quo (pas de signature et donc pas de sécurité du tout). Bien sûr, la clé doit être protégée par les mécanismes habituels de sécurité du serveur (ne pas la mettre dans un répertoire partagé, veiller à ses protections) et le serveur lui-même doit être administré selon les bonnes pratiques de sécurité (ne donner le mot de passe de root qu'aux seules personnes autorisées, appliquer les correctifs de sécurité...).

3.1.2.Les logiciels

[12] En effet, le choix d'utiliser une ZSK et une KSK dans le cas d'OpenDNSSEC vient de deux raisons : c'est le choix par défaut du logiciel, celui qui provoquera le moins de surprises, et le coût en complexité est faible puisque OpenDNSSEC se charge de tout.

[13]Nous rappelons ici la section 2.2 du document qui préconise la mise en place d'un système de supervision. Si la solution de coffre-fort est celle retenue, le système de supervision doit intégrer un rappel automatique pour re-signer périodiquement la zone, conformément aux bonnes pratiques opérationnelles en la matière (voir par exemple la section 11.4 de *[nist.dnssec-deployment],* §5.Bibliographie).

Il existe évidemment un grand choix de systèmes et de logiciels pour héberger les fonctions DNSSEC. Nous nous focaliserons sur la plate-forme la plus utilisée pour l'hébergement, Unix.

Les exemples donnés ici sont pour une machine utilisant le système d'exploitation Debian et devront être légèrement adaptés pour d'autres systèmes.

Pour les logiciels, la priorité a été donnée aux logiciels libres, avec un accent particulier sur deux solutions que nous connaissons et utilisons :

- OpenDNSSEC et NSD,
- BIND avec auto-dnssec.

3.1.3.Aspects cryptographiques pratiques

Ce document n'est pas un cours de cryptographie et passera donc rapidement sur cette section. Notez bien que, pour faire du DNSSEC, vous n'avez besoin que des aspects opérationnels de la cryptographie, ses facettes mathématiques ne sont pas nécessaires.

Pour les non-cryptographes, le seul choix pratique est « NSEC3 ou pas ». NSEC3, normalisé dans le RFC 5155, permet de limiter les risques d'énumération du contenu de la zone, via le *zone walking*. Si vos zones sont déjà publiques (transfert de zone autorisé) ou si le contenu de vos zones est trivial à énumérer (par exemple, il n'y a que l'*apex* et www), alors NSEC3 est inutile. Mais le cas inverse est bien plus fréquent et nous recommandons donc d'utiliser NSEC3.

Au final, nous recommandons le choix de l'algorithme RSA + SHA256 (numéro 8 dans le **registre IANA des algorithmes**[14]). C'est celui qui est utilisé dans les exemples suivants.

[14] **http://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xhtml**

3.2. Un exemple avec OpenDNSSEC et NSD

3.2.1. Signature initiale

Les opérations de changement des clés DNSSEC sont complexes et des problèmes peuvent apparaître[15].

Si on souhaite changer régulièrement les clés, il est alors très recommandé d'automatiser ces opérations.
C'est ce que fait le logiciel OpenDNSSEC. Il prend en charge toute la gestion des clés (leur création, leur retrait, en respectant les délais) et la signature (et re-signature).
Sur une machine Debian, on l'installe avec :
`aptitude install opendnssec softhsm`

Puis, lancer les démons OpenDNSSEC avec la commande :
`ods-control start`
Ces démons effectuent des contrôles et les historisent.

OpenDNSSEC permet d'utiliser des HSM, ou bien son propre système de stockage, nommé SoftHSM (en fait, une simple base de données SQLite locale, qu'il faut protéger contre les accès non autorisés).

Note : Vérifiez que le compte ods a bien les droits en écriture sur le répertoire et contenu SoftHSM.

On doit d'abord initialiser SoftHSM :

```
% softhsm --init-token --slot 0 --label "OpenDNSSEC"
The SO PIN must have a length between 4 and 255 characters.
Enter SO PIN:
The user PIN must have a length between 4 and 255 characters.
Enter user PIN:
The token has been initialized.
```

Remarque : `SO` signifie `Security Officer`.

On configure alors OpenDNSSEC pour utiliser SoftHSM en éditant `/etc/opendnssec/conf.xml` :

```
<Repository name="softHSM">
                <Module>/usr/lib/softhsm/libsofthsm.so</Module>
                <TokenLabel>OpenDNSSEC</TokenLabel>
                <PIN>cestressecret</PIN>
                <RequireBackup/>
</Repository>
```

[15] http://conferences.npl.co.uk/satin/papers/satin2011-Bortzmeyer.pdf

Où l'élément `PIN` est celui indiqué à l'initialisation de SoftHSM. On peut alors tester que OpenDNSSEC interagit avec sa base de données :

```
% ods-hsmutil list
Listing keys in all repositories.
0 key found.

Repository            ID                                Type
----------            --                                ----
```

Notez qu'aucune clé n'est présente mais, à ce stade, c'est normal (OpenDNSSEC les créera tout seul).

On définit ensuite la politique de clés et de signatures, dans le fichier **`/etc/opendnssec/kasp.xml`**.

Voici la politique par défaut recommandée par ce document (suivant **la documentation officielle**[16] et le **format de dates d'OpenDNSSEC**[17]).

Des commentaires ont été ajoutés pour les choix qui méritent le plus de discussion :

```
<Policy name="default">
                <Description>Politique recomandée par le HOWTO DNSSEC
                de l'Afnic</Description>
                <Signatures>
                        <Resign>PT2H</Resign>
                        <Refresh>P3D</Refresh>
                        <!-- Signatures valables un mois. Cela
                        augmente légèrement le risque d'attaque par
                        re-jeu (par rapport à des durées plus courtes)
                        mais cela donne plus de marge de manœuvre en
                        cas de problème : si le processus de signature
                        a un problème, nous aurons 30 jours pour
                        réparer. -->
                        <Validity>
                                <Default>P30D</Default>
                                <Denial>P30D</Denial>
                        </Validity>
                        <Jitter>PT12H</Jitter>
                        <InceptionOffset>PT3600S</InceptionOffset>
                </Signatures>
                 <Denial>
                        <NSEC3>
                                <Resalt>P100D</Resalt>
                                <Hash>
                                        <Algorithm>1</Algorithm>
                                        <Iterations>1</Iterations>
                                        <Salt length="8"/>
                                </Hash>
                        </NSEC3>
                </Denial>
                <Keys>
                        <TTL>PT12H</TTL>
```

[16] https://wiki.opendnssec.org/display/DOCS/kasp.xml

[17] https://wiki.opendnssec.org/display/DOCS/Date+Time+durations

```
                        <RetireSafety>PT3H</RetireSafety>
                        <PublishSafety>PT3H</PublishSafety>
                        <Purge>P14D</Purge>

                        <KSK>
                                <!-- RSA + SHA-256 -->
                                <Algorithm length="2048">8</Algorithm>
                                <Lifetime>P6Y</Lifetime>
                                <Repository>SoftHSM</Repository>
                                <!-- Pas de remplacement automatique
                                (puisque, de toute façon, il faut une
                                interaction avec la zone parente. -->
                                <ManualRollover/>
                        </KSK>

                        <ZSK>
                                <Algorithm length="1024">8</Algorithm>
                                <Lifetime>P60D</Lifetime>
                                <Repository>SoftHSM</Repository>
                        </ZSK>
                </Keys>
                <Zone>
                        <PropagationDelay>PT3600S</PropagationDelay>
                        <SOA>
                                <TTL>PT7200S</TTL>
                                <Minimum>PT3600S</Minimum>
                                <Serial>datecounter</Serial>
                        </SOA>
                </Zone>
                <Parent>
                        <PropagationDelay>PT3600S</PropagationDelay>
                        <DS>
                                <!-- La vraie valeur dépend de votre
                                zone parente. Les chiffres ici sont
                                pour .FR -->
                                <TTL>PT3600S</TTL>
                        </DS>
                        <SOA>
                                <TTL>PT172800S</TTL>
                                <Minimum>PT5400S</Minimum>
                        </SOA>
                </Parent>
</Policy>
```

On indique ensuite à OpenDNSSEC, dans **`/etc/opendnssec/zonelist.xml`**, quelles zones il doit gérer :

```
<Zone name="dnssec.fr">
        <Policy>default</Policy>
        ...
        <Adapters>
                <Input>
                        <File>/var/opendnssec/unsigned/dnssec.fr</File>
                </Input>
                <Output>
                        <File>/var/opendnssec/signed/dnssec.fr</File>
                </Output>
...
```

On note qu'on utilise la politique définie plus haut, `default`.

Il faut ensuite créer les clés en utilisant la commande suivante :
`ods-ksmutil key generate --policy default --interval 1Y`.

Après, on initialise la base d'OpenDNSSEC avec `ods-ksmutil setup`. OpenDNSSEC va créer les clés correspondant à la politique (nommée default ici) :

```
% ods-hsmutil list
Listing keys in all repositories.
4 keys found.

Repository            ID                                Type
----------            --                                ----
softHSM               cc59d2b16e421c57eec35ed4ceae0099  RSA/1024
softHSM               e17b1d0465e6976536119c872a353911  RSA/1024
softHSM               fa8cdfc8da319311283b66fe55839212  RSA/2048
softHSM               60b57dff6604cc35ec6fdd8aef7710a2  RSA/2048
```

On a (le résultat exact dépend de la politique configurée) deux ZSK (1024 bits par défaut) et deux KSK (2048 bits par défaut), une active et une prête à assurer le remplacement futur. Pour voir les clés plus en détail, on utilise `ods-ksmutil` :

```
% sudo ods-ksmutil key list --zone dnssec.fr  --verbose
SQLite database set to: /var/lib/opendnssec/db/kasp.db
Keys:
Zone:                           Keytype:      State:    Date of next
transition:  Repository:                       Keytag:
dnssec.fr                       KSK           active    2013-07-31
15:11:45       SoftHSM                           8680
dnssec.fr                       ZSK           active    2013-09-17
15:15:18       SoftHSM                           23283
```

Note : Si vous ne voyez pas de clés dans liste, éventuellement, redémarrez le démon `ods-control`.

L'état des clés (ici, `active`) peut avoir les valeurs suivantes :

- `Generated` : la clé a été créée (OpenDNSSEC permet de créer des clés à l'avance, par exemple à des fins de sauvegarde, avec `ods-ksmutil key generate --policy test --interval 1Y`, où `1Y` veut dire de générer des clés pour l'année qui vient) ;
- `Publish` : la clé est publiée dans le DNS, sous la forme d'un enregistrement DNSKEY mais elle n'a pas encore forcément atteint tous les résolveurs (car ils peuvent avoir une vieille version de l'ensemble des DNSKEYs dans leur cache) ;
- `Ready` : la clé est publiée et a atteint tous les résolveurs ;
- `Active` : la clé est utilisée pour signer ;
- `Retired` : la clé n'est plus utilisée pour signer mais est encore publiée car des vieilles signatures faites avec cette clé sont peut-être encore dans des caches ;
- `Dead` : la clé est encore publiée mais elle ne devrait plus servir à rien, toutes les signatures ont quitté les caches ;
- `Removed` : la clé est complètement supprimée ;

- `Revoked` : il désigne une clé qui a été annulée explicitement « à la main ».

Et la signature ? Elle est automatique. OpenDNSSEC va prendre le fichier à l'endroit indiqué (`/var/opendnssec/unsigned/dnssec.fr` ici) et le déposer là où on lui a dit (`/var/opendnssec/signed/dnssec.fr` ici). OpenDNSSEC gérera également les re-signatures.

3.2.2. Maintenance

Il reste à recharger le serveur de noms lorsque la signature a été faite ou refaite. La méthode recommandée est d'utiliser l'option `NotifyCommand` d'OpenDNSSEC dans `conf.xml` :

```
<NotifyCommand>/usr/local/sbin/opendnssec-nsd-reload</NotifyCommand>
```

Mais que doit contenir le script `/usr/local/sbin/opendnssec-nsd-reload` ?

NSD, jusqu'à la version 3 incluse[18], n'a pas de mécanisme permettant la communication avec le serveur. Il faut donc créer un script qui appelle un programme qui va effectuer le rechargement. Cela peut se faire par un programme setuid ou bien en utilisant sudo. Ici, nous utilisons sudo. Le script peut contenir :

```
#!/bin/sh
logger -i -t OpenDNSSEC-signer -p daemon.info "Reloading NSD,
modification in zone $1 (file $2)"
sudo -u nsd /usr/sbin/nsdc rebuild
sudo -u nsd /usr/sbin/nsdc reload
sudo -u nsd /usr/sbin/nsdc notify
```

La configuration de sudo (`/etc/sudoers`) doit contenir :

```
Defaults:opendnssec !requiretty
…
opendnssec ALL = (nsd) NOPASSWD: /usr/sbin/nsdc reload, /usr/sbin/nsdc
rebuild, /usr/sbin/nsdc notify
```

On va alors voir dans le journal les notifications de rechargement :

```
Jul 26 02:54:59 aetius OpenDNSSEC-signer[15803]: Reloading NSD,
modification in zone dnssec.fr (file /var/opendnssec/signed/dnssec.fr)
```

Et lorsqu'on modifie la zone ? Il faut l'indiquer à OpenDNSSEC :

```
# ods-signer sign dnssec.fr
Zone dnssec.fr scheduled for immediate re-sign.
```

18 Cette fonction apparaîtra dans la version 4.

3.2.3. Transmission de l'enregistrement DS et changements

Une signature de zone se termine par l'envoi de l'enregistrement DS au gestionnaire de la zone parente (typiquement le registre du TLD). Pour connaître les DS à transmettre, la commande est :

```
# ods-ksmutil key export --ds dnssec.fr
;ready KSK DS record (SHA1):
dnssec.fr.      3600    IN      DS      8680 8 1
050a641297fbd70912eec6d6e1e056354635b370

;ready KSK DS record (SHA256):
dnssec.fr.      3600    IN      DS      8680 8 2
317348d9d1b567df63d038cdef48b0a26d1c5594aa7ba253ad622d84872baa72
```

Il faut alors transmettre ces DS au registre :

- dans le cas d'un hébergeur de DNS est un Bureau d'enregistrement, la transmission de l'enregistrement DS peut se réaliser via l'interface que le registre fournit (EPP ou bien interface Web).
- dans le cas d'un hébergeur de DNS qui n'est pas un bureau d'enregistrement, la transmission de l'enregistrement DS doit être mis à disposition via l'interface fournie par le Bureau d'enregistrement à ses clients (interface web, API, autre)

Si aucun DS n'apparaît lorsqu'on tape cette commande, c'est souvent parce que vous avez été trop impatient : OpenDNSSEC gère les délais et n'accepte pas d'exporter les enregistrements DS tant que l'enregistrement DNSKEY n'est pas présent sur tous les résolveurs (`ods-ksmutil key list --verbose` vous l'affiche dans le champ *Date of next transition*.)
Maintenant, ajoutons une zone à cette configuration. On édite `zonelist.xml` pour ajouter la zone, ici `dnssec.pm`. On signale à OpenDNSSEC que les choses ont changé :

```
# ods-ksmutil update all
...
Zone dnssec.pm found
Policy set to default.
Added zone dnssec.pm to database
```

Et OpenDNSSEC la prend en charge :

```
# ods-ksmutil key list --zone dnssec.pm
Keys:
Zone:               Keytype:      State:    Date of next transition:
dnssec.pm           KSK           publish   2013-07-27 01:40:03
dnssec.pm           ZSK           active    2013-08-25 11:40:03
```

Si on veut remplacer explicitement la KSK, par exemple parce qu'on soupçonne qu'elle a été compromise ? On peut demander à OpenDNSSEC un remplacement :

```
% ods-ksmutil key rollover --zone dnssec.fr --keytype KSK
```

On voit tout de suite la nouvelle clé apparaître, en état **publish**. Elle sera publiée dans le DNS mais n'est pas tout de suite utilisable, tous les caches ne l'ayant pas forcément encore :

```
% ods-ksmutil key list --zone dnssec.fr  --verbose
...
dnssec.fr                       KSK           publish    2013-08-01
05:11:46       c12c000741d3d79c2ce22cfaa3567e9c  SoftHSM
                           34292
```

Au bout d'un temps qui dépend des TTL choisi, la clé va passer dans l'état **ready** :

```
% ods-ksmutil key list --zone dnssec.fr  --verbose
...
dnssec.fr                   KSK           ready     waiting for ds-seen
...
```

On peut alors extraire l'enregistrement DS et le transmettre à la zone parente, comme expliqué plus haut. Une fois le DS effectivement mis dans la zone parente (on testera avec dig), on peut dire à OpenDNSSEC que le DS a été vu :

```
% ods-ksmutil key ds-seen --zone dnssec.fr --keytag 34292
```

La clé passera alors en état **active**. Il n'y a rien à faire pour l'ancienne clé, elle sera retirée automatiquement lorsque cela sera sûr de le faire.

3.3. Un exemple avec BIND

BIND dispose, depuis la version 9.9, de la capacité de gérer les signatures et re-signatures. Il reste à faire la gestion des clés mais on a vu plus haut qu'un remplacement systématique et automatique des clés n'est nullement obligatoire.

Le principe avec BIND est donc le suivant :

1. créer la ou les clés ;
2. indiquer à BIND de gérer seul les signatures et re-signatures de la zone.

On installe le seul logiciel nécessaire dans cet exemple, BIND, par exemple (sur Debian) avec :
aptitude install bind9.
Puis la première étape consiste à appeler **dnssec-keygen**. Ici, pour le cas d'une seule clé :

```
% dnssec-keygen -a RSASHA256 -3 -f KSK -b 2048 dnssec.fr
Generating key pair.........++
+ ..........................................+++
Kdnssec.fr.+008+49734
```

Remarque : On peut ajouter l'option `-n ZONE` mais c'est la valeur par défaut, de toute façon.

Si l'opération prend longtemps et que vous constatez avec un programme comme **top** que la machine ne fait pas grand-chose, c'est normal : la génération d'une clé nécessite de l'entropie pour fabriquer des nombres vraiment aléatoires et votre machine ne produit peut-être pas assez d'entropie. Paradoxalement, il faut faire travailler votre machine pour que cela aille plus vite (compiler un très gros programme, par exemple[19]), les entrées/sorties sur le disque étant la principale source d'entropie.
Lorsque `dnssec-keygen` se termine, il crée deux fichiers :

```
% ls -lt Kdnssec.fr*
-rw-r--r-- 1 bortzmeyer bortzmeyer  598 Jul 19 10:53 Kdnssec.fr.
+008+49734.key
-rw------- 1 bortzmeyer bortzmeyer 1776 Jul 19 10:53 Kdnssec.fr.
+008+49734.private
```

La clé privée est précieuse : elle doit être mise en sécurité, sur une machine fiable, tout en ayant des sauvegardes au cas où cette machine aurait une défaillance. Cette gestion des clés est une partie nouvelle et importante de DNSSEC.
La clé publique doit être mise dans un répertoire où BIND la trouvera (`key-directory` voir plus loin)[20].

Ensuite, il faut configurer BIND :

```
options {
       directory "/etc/bind";
       key-directory "/var/bind/keys";
       recursion no;
       dnssec-enable yes;
};

zone "dnssec.fr" in {
       inline-signing yes;
       auto-dnssec maintain;
       update-policy local; # Necessary, says the ARM (otherwise, you
cannot freeze/thaw)
       type master;
       file "dnssec.fr";
       ...
}
```

19 Sur Unix, un `find / -type f` est également une bonne solution.

20 Si vous lisez des anciennes documentations trouvées sur le Web, vous verrez peut-être des indications comme quoi il faut mettre cette clé publique dans le fichier de zone. Cela n'est plus nécessaire avec la méthode présentée ici.

Et cela suffit. BIND trouvera les clés dans `key-directory` et signera tout seul[21]. Et BIND re-signera lorsque cela sera nécessaire[22].

Vous noterez que la zone a été signée avec NSEC. Pour NSEC3, il va falloir l'expliquer à BIND, et lui fournir un sel, un nombre aléatoire qui servira à rendre les attaques par dictionnaire plus difficiles :

```
% dd if=/dev/random bs=1 count=10 | md5sum | cut -c1-8
10+0 records in
10+0 records out
68f499ee
10 bytes (10 B) copied, 0.00046732 s, 21.4 kB/s

[On utilise ce sel]
% rndc signing -nsec3param 1 0 10  68f499ee auto.rd.nic.fr
```

Et on peut voir que la zone est désormais signée avec NSEC3.

Si on veut utiliser deux clés, une KSK et une ZSK, on doit d'abord les créer (notez la différence, `-f KSK`, ainsi que la taille plus petite de la ZSK) :

```
% dnssec-keygen -a RSASHA256 -3 -f KSK -b 2048 dnssec.fr
Generating key pair....+++ ...................+++
Kdnssec.fr.+008+09634

% dnssec-keygen -a RSASHA256 -3 -b 1024 dnssec.fr
Generating key pair..............++++++ ..............++++++
Kdnssec.fr.+008+29433
```

On copie alors les deux clés dans le répertoire. BIND signera l'enregistrement DNSKEY avec les deux clés et tout le reste avec la ZSK.

Une fois votre zone testée en détail, et après que tous les caches auront possédé les nouvelles informations[23], vous pourrez compléter la chaîne de confiance de DNSSEC en soumettant un enregistrement DS à la zone parente pour publication. Le DS de la zone se calcule avec `dnssec-dsfromkey` :

```
% dnssec-dsfromkey Kdnssec.fr.+008+49734.key
dnssec.fr. IN DS 49734 8 1 3C5BA1C95F17214536AA8E82E9406321A96FBD22
dnssec.fr. IN DS 49734 8 2
A3B4267E78CF26B4442007E14B55B8F7C83A4EB81015122410B9E47E FEA2DBFD
```

Si vous êtes Bureau d'Enregistrement, vous envoyez ces enregistrements DS vous-même, *a priori* en utilisant le protocole EPP. Sinon, vous devrez passer par le mécanisme fourni par votre BE, probablement une interface Web.

---

21 Un message du genre `Key dnssec.fr/RSASHA256/46747 missing or inactive and has no replacement: retaining signatures` apparaîtra dans le journal de temps en temps mais il faut l'ignorer, il est erroné.

22 L'intervalle de re-signature est contrôlable avec le paramètre `sig-validity-interval`.

23 OpenDNSSEC se charge de ces délais automatiquement. Sans lui, il faudra calculer à la main.

En cas de changement du contenu, de la zone, vous devez signaler à BIND de re-signer et de se recharger :

```
% rndc  freeze dnssec.fr

[Éditer le fichier de zone]

% rndc  thaw dnssec.fr
A zone reload and thaw was started.
Check the logs to see the result.
```

On l'a vu, il n'est pas nécessaire de remplacer systématiquement les clés. Toutefois, on peut se retrouver dans l'obligation de le faire, par exemple parce qu'on découvre que la clé a été compromise, ou, tout simplement, pour tester les procédures pour le cas d'une telle compromission.

Le principe avec BIND, pour le cas d'une clé unique, est le suivant : on crée la nouvelle clé, on la met dans le répertoire des clés, on modifie l'ancienne pour indiquer qu'elle n'est plus utilisée et BIND fera le remplacement :

```
[Création de la nouvelle clé]
% dnssec-keygen   -a RSASHA256 -f KSK -b 2048 dnssec.fr
Generating key pair.....+++ ...................................+++
Kdnssec.fr.+008+00847

[Copie dans le répertoire, si nécessaire]
% cp Kdnssec.fr.+008+00847* /where/are/the/keys

[Modification de l'ancienne clé, la 49734]
[Révocation immédiate]
% dnssec-settime -R +0 Kdnssec.fr.+008+49734.key
[Retrait dans deux jours (le temps que la nouvelle clé soit dans tous
les caches]
% dnssec-settime -I +2d Kdnssec.fr.+008+49734.key
[Suppression dans six jours (le temps que les signatures avec
l'ancienne clé aient disparu des caches)
% dnssec-settime -D +6d Kdnssec.fr.+008+49734.key
```

Attention, les délais doivent être calculés en tenant compte des TTL des enregistrements de la zone. Faites bien attention (ou bien utilisez OpenDNSSEC, qui gère tout cela automatiquement). Un changement de clé (*key rollover*) n'est pas une opération facile.

N'oubliez pas non plus de changer l'enregistrement DS qui se trouve dans la zone parente et de synchroniser ce changement avec les changements faits dans votre zone. Là encore, cela peut s'avérer délicat et il vaut ne pas le réaliser l'opération sans une minutieuse préparation.

Et pour ajouter une nouvelle zone à celles servies ? Les étapes sont les suivantes :

1. générer des clés pour la nouvelle zone,
2. ajouter la zone à la configuration (fichier `named.conf`), et le fichier de zone dans le répertoire de BIND,
3. recharger le serveur (par exemple avec `rndc reload`).

3.4.Débogage

Comme indiqué plus haut, DNSSEC nécessite une approche qualité : il est primordial de tester rigoureusement sa configuration. Avec le DNS ordinaire, c'était déjà souhaitable. Mais, en pratique, cela marchait souvent même si on ne le faisait pas. Avec DNSSEC, on ne peut pas compter sur une telle indulgence. Quels sont les principaux outils de débogage et de tests dont on dispose ? (Une liste plus longue est **maintenue en ligne**[24].)

Pour les problèmes spécifiquement DNSSEC, l'outil le plus souvent utilisé est **DNSviz**[25]. C'est un outil Web qui effectue un certain nombre de tests sur une zone signée et présente les résultats graphiquement, de manière très compréhensible.
Voici un exemple avec une zone correcte :

24 http://www.bortzmeyer.org/tests-dns.html

25 http://dnsviz.net/


DNSKEY
alg=8, id=19036
DNSKEY
alg=8, id=49656
DS
digest alg=2
.
(2013-09-09 12:26:54 UTC)
DNSKEY
alg=8, id=1336
DNSKEY
alg=8, id=62646
DNSKEY
alg=8, id=60274
DNSKEY
alg=8, id=20122
DS
digest alg=2
fr
(2013-09-09 12:53:28 UTC)
DNSKEY
alg=8, id=11627
DNSKEY
alg=8, id=43854
DNSKEY
alg=8, id=17464
DNSKEY
alg=8, id=47911
DS
digest algs=1,2
nic.fr
(2013-09-09 18:57:44 UTC)
DNSKEY
alg=8, id=61807
DNSKEY
alg=8, id=56434
rdnic.fr/SOA
rd.nic.fr
(2013-09-10 01:59:06 UTC)

Figure 1. `rd.nic.fr` vu par DNSviz

Et avec une zone incorrecte (les signatures ont expiré, des détails apparaissent en ligne dans la colonne de gauche) :

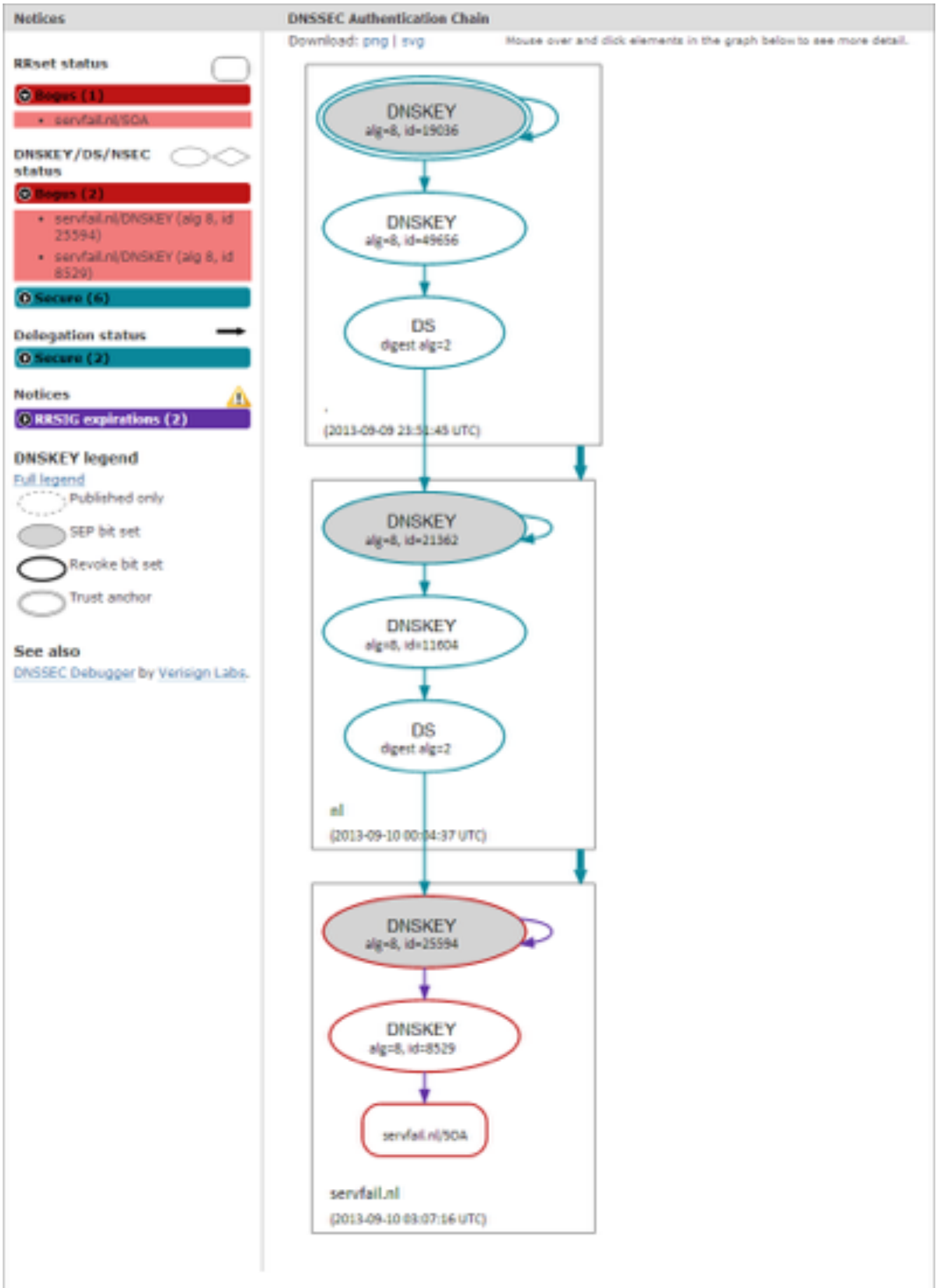


Figure 2. `servfail.nl` vu par DNSviz

Pour tester tous les aspects de la configuration, pas uniquement DNSSEC, on peut utiliser Zonemaster. Ici, en ligne de commandes (l'option -s lui dit de tester aussi DNSSEC) :

```
% zonemaster-cli rd.nic.fr
Seconds Level     Message
======= ========= =======
   2.63 NOTICE    Nameserver ns2.rd.nic.fr has an IP address (192.134.4.81)
with mismatched PTR result (lea.rd.nic.fr.).
   2.67 NOTICE    Nameserver ns2.rd.nic.fr has an IP address (2001:67c:
2218:3::1:7) with mismatched PTR result (dalila.rd.nic.fr.).
  16.77 NOTICE    No target (MX, A or AAAA record) to deliver e-mail for
the domain name.
```

Et sur le domaine qui a un problème :

```
% zonemaster-cli servfail.nl
…
   9.93 ERROR     RRSIG with keytag 25594 and covering type(s) DNSKEY has
already expired (expiration is: 1326490909).
   9.93 ERROR     RRSIG with keytag 8529 and covering type(s) SOA has
already expired (expiration is: 1326490909).
  10.09 ERROR     Signature for DNSKEY with tag 25594 failed to verify with
error 'DNSSEC signature has expired'.
10.09 ERROR     The apex DNSKEY RRset was not correctly signed.
...
```

À noter que si on travaille avec des fichiers de zone[26] qu'on signe soi-même, via un processus dont on n'est pas encore sûr, on peut tester l'intégrité d'un fichier signé avec l'excellent et très rapide outil (même pour des grosses zones) validns[27] :

```
% validns -p all -z dnssec.fr zones/dnssec.fr
%
```

Le traditionnel outil `dig` peut aussi être utilisé, même s'il est plutôt destiné aux gens connaissant bien le DNS. Si votre résolveur local valide DNSSEC, dig vous indiquera les zones correctement signées avec une marque `ad` (pour *Authentic Data*) :

```
% dig A www.afnic.fr
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53599
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 6, AUTHORITY: 7, ADDITIONAL: 23
```

26 Si on utilise BIND en auto-signature, le fichier de zone sur le disque est sous un format binaire qu'il faut d'abord traduire en texte avec **named-compilezone -j -i none -f raw -F text -o dnssec.fr.signed-text dnssec.fr dnssec.fr.signed**.

27 http://www.validns.net/

Si la zone est mal signée, vous n'aurez qu'un **SERVFAIL** (*SERver FAILure*) :

```
% dig A www.servfail.nl
...
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 52126
```

Est-ce bien un problème DNSSEC ? D'autres raisons peuvent expliquer ce code. Pour le savoir, il faut utiliser **dig** avec l'option **+cd** (pour *Checking Disabled*) :

```
% dig +cd A www.servfail.nl
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53197
;; flags: qr rd ra cd; QUERY: 1, ANSWER: 6, AUTHORITY: 0, ADDITIONAL: 1
...
;; ANSWER SECTION:
www.servfail.nl.        60      IN      CNAME   www.forfun.net.
```

Retenez cette règle simple : si vous obtenez **SERVFAIL** par défaut, et une réponse normale avec l'option **+cd**, c'est que le problème vient de DNSSEC.

Bien sûr, ce test avec un résolveur validant est, d'une certaine façon, le meilleur. Après tout, on utilise DNSSEC pour que les résolveurs validants puissent valider. Mais si vous n'avez pas encore un résolveur validant, ou bien si vous n'êtes pas sûr de sa configuration correcte ? Alors, vous pouvez utiliser un résolveur public. L'OARC en fournit un, l'ODVR[28]. Il consiste en deux logiciels, un BIND et un Unbound. Testons le BIND :

```
% dig +dnssec @2001:4f8:3:2bc:1::64:20 SOA dnssec.fr
...
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18196
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 3, ADDITIONAL: 3
...
```

On voit qu'on a bien récupéré l'option **ad** (*Authentic Data*). Testons un domaine mal signé avec l'autre serveur de l'ODVR, le Unbound :

```
% dig +dnssec @2001:4f8:3:2bc:1::64:21 SOA servfail.nl
...
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 53610
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1
```

Un résolveur validant peut être utilisé pour tester les zones signées depuis un logiciel de supervision comme Nagios. Par exemple, le plugin Nagios officiel **check_dig** peut être configuré ainsi :

```
define service{
        use                             generic-service
        host_name
                        résolveur_validant_1,résolveur_validant_2
        service_description             DNS
```

[28] https://www.dns-oarc.net/oarc/services/odvr

```
        check_command                   check_dig!-H $HOSTADDRESS$ -l
dnssec.fr -T SOA
        }
```

Et on aura alors une alerte Nagios si le domaine renvoie un **SERVFAIL**.

3.5.Supervision

La solution montrée ici a été testée avec le logiciel de supervision Icinga mais elle ne repose que sur des outils compatibles avec l'API Nagios donc elle devrait marcher avec beaucoup d'outils de supervision.

L'outil de base est le script de test de Duane Wessels[29]. Ses spécifications : il se connecte à tous les serveurs DNS d'une zone, demande les signatures, regarde les dates d'expiration et peut signaler un avertissement ou une erreur selon des seuils choisis par l'utilisateur. Un exemple à la main :

```
% perl check_zone_rrsig_expiration -Z nic.fr
ZONE OK: No RRSIGs expiring in the next 3 days; (1.04s) |
time=1.042905s;;;0.000000
```

On peut choisir les seuils, donc mettons qu'on veut un avertissement s'il reste moins d'une semaine :

```
% perl check_zone_rrsig_expiration -Z nic.fr -W 7
ZONE WARNING: MX RRSIG expires in 3.7 days at ns6.ext.nic.fr; (0.28s) |
time=0.281515s;;;0.000000
```

Pour installer et exécuter ce script, il faut Perl et certains modules indiqués dans la documentation. Attention, si vous n'installez pas tous les paquetages indiqués, vous aurez un message pas clair du tout :

```
***  WARNING!!!  The program has attempted to call the method
***  "sigexpiration" for the following RR object:
```

Une fois le programme correctement installé, je vous recommande l'option -d si vous voulez déboguer en détail ce qu'il fait.
On configure ensuite Icinga, par exemple :

```
define command {
        command_name    check-zone-rrsig
        command_line    /usr/local/sbin/check_zone_rrsig_expiration -Z
$HOSTADDRESS$ -W $ARG1$ -C $ARG2$
        }

...

define service {
       use dns-rrsig-service
       hostgroup_name My-zones
       service_description SIGEXPIRATION
       # Five days left: warning. Two days left: panic.
       check_command   check-zone-rrsig!5!2
}

define host{
        name                            my-zone
        use                         generic-host
   check_command              check-always-up
        check_period                    24x7
        check_interval                  5
```

[29] http://dns.measurement-factory.com/tools/nagios-plugins/check_zone_rrsig_expiration.html

```
        retry_interval                  1
        max_check_attempts              3
        contact_groups                  admins
        notification_period             24x7
        notification_options        u,d,r
        register 0
}

define hostgroup{
       hostgroup_name My-zones
       members dnssec.fr,etc-etc
}

define host{
       use moi-zone
       host_name dnssec.fr
}
```

Une fois que c'est fait, on redémarre Icinga. Ici, voici un test avec une zone délibérement cassée (elle a été signée manuellement en indiquant la date d'expiration, et sans re-signer ensuite). Icinga enverra ce genre d'avertissement :

```
Notification Type: PROBLEM

Service: DNSRRSIG
Host: broken.rd.nic.fr
Address: broken.rd.nic.fr
State: WARNING

Date/Time: Fri Mar 28 06:50:38 CET 2014

Additional Info:

ZONE WARNING: DNSKEY RRSIG expires in 1.2 days at ns2.dnssec.pm:
(1.12s)
```

Puis un CRITICAL puis, lorsque la zone aura vraiment expiré :

```
Notification Type: PROBLEM

Service: DNSRRSIG
Host: broken.rd.nic.fr
Address: broken.rd.nic.fr
State: CRITICAL

Date/Time: Mon Mar 31 09:10:38 CEST 2014

Additional Info:

ZONE CRITICAL: ns2.dnssec.pm has expired RRSIGs: (1.10s)
```

Si on re-signe à ce moment, le problème disparait :

```
Notification Type: RECOVERY

Service: DNSRRSIG
Host: broken.rd.nic.fr
Address: broken.rd.nic.fr
State: OK

Date/Time: Mon Mar 31 09:40:38 CEST 2014

Additional Info:
```

```
ZONE OK: No RRSIGs expiring in the next 3 days: (1.04s)
```

Et, dans le journal d'Icinga, cela apparaitra :

```
[Fri Mar 21 21:40:32 2014] SERVICE ALERT:
broken.rd.nic.fr;DNSRRSIG;CRITICAL;SOFT;2;ZONE CRITICAL: DNSKEY RRSIG
expires in 0.6 days at ns2.dnssec.pm: (1.09s)
...
[Fri Mar 21 21:42:32 2014] SERVICE ALERT:
broken.rd.nic.fr;DNSRRSIG;OK;SOFT;3;ZONE OK: No RRSIGs expiring in the
next 7 days: (0.68s)
```

Pour les utilisateurs d'OpenDNSSEC, le paramètre important à régler en même temps que les seuils de la supervision est le paramètre <Refresh>. Comme le dit la documentation : « The signature will be refreshed when the time until the signature expiration is closer than the refresh interval. » Donc, en pratique, c'est la durée qu'il faut indiquer avec l'option -C (seuil critique). Attention, OpenDNSSEC ajoute de légères variations (*jitter*), il faut donc indiquer une durée légèrement supérieure au <Refresh>.

3.6.Autres solutions

Nous vous rappelons qu'il existe d'autres possibilités : il y a évidemment de nombreuses façons de faire du DNSSEC. Nous avons choisi de nous focaliser sur deux méthodes (OpenDNSSEC + NSD ou bien BIND avec auto-dnssec) mais d'autres existent :

- Le logiciel PowerDNS[30] est très populaire, notamment dans les pays du Nord de l'Europe, et sert de nombreuses zones. Il permet de **signer automatiquement**[31].
- OpenDNSSEC peut également être utilisé avec BIND (et pas avec NSD comme dans l'exemple plus haut). Dans ce cas, on utilise typiquement un `<NotifyCommand>rndc reload</NotifyCommand>`.
- Il existe plusieurs « appliances » qui gèrent DNS et DNSSEC comme ceux de **Efficient IP**[32], **Secure64**[33] ou **InfoBlox**[34].

[30] https://www.powerdns.com/

[31] http://doc.powerdns.com/html/dnssec-operational-doctrine.html

[32] http://www.efficientip.com/dnssec

[33] http://www.secure64.com/

[34] http://www.infoblox.com/solutions/best-practices/dns-security-center

4.Conclusion

Des nouvelles techniques permettant d'« améliorer » les attaques DNS par empoisonnement sont publiées régulièrement[35].

Travailler au déploiement de DNSSEC est donc une nécessité et permet de contribuer à la robustesse de cette ressource partagée qu'est le DNS.

Mais DNSSEC est reste une technique nécessitant qualification, méthodologie et précision.

Il est donc nécessaire de bien veiller au bon déroulement de son déploiement.

Outre les lectures déjà mentionnées, nous attirons votre attention sur le guide *nist.dnssec-deployment* (§5.Bibliographie) (guide en anglais mais très complet).

D'autre part, si vous souhaitez une formation pratique, avec travaux sur ordinateur, sur les techniques présentées ici, l'Afnic vous rappelle son partenariat avec HSC et organise des **formations régulières** sur DNSSEC.

35 Au moment de l'écriture de ce texte, la dernière a été présentée à l'IETF à Berlin fin juillet 2013 par Haya Shulman.

5.Bibliographie

- *[afnic.dnssec-thema] Dossier thématique : DNSSEC*. 2010. Afnic. https://www.afnic.fr/medias/documents/afnic-dossier-dnssec-2010-09.pdf

- *[bortzmeyer.dnssec-jres] Sécurité du DNS et DNSSEC*. 2009. Stéphane Bortzmeyer. https://2009.jres.org/planning_files/article/pdf/5.pdf

- *[bortzmeyer.dnssec-satin] Monitoring DNSSEC zones: what, how and when?*. 2011. Stéphane Bortzmeyer. http://conferences.npl.co.uk/satin/papers/satin2011-Bortzmeyer.pdf

- *[nist.dnssec-deployment] Secure Domain Name System (DNS) Deployment Guide*. 2010. Ramaswamy Chandramouli. Scott Rose. http://csrc.nist.gov/publications/nistpubs/800-81r1/sp-800-81r1.pdf

6.Glossaire

BIND : *Berkeley Internet Name Domain* ou *Berkeley Internet Name Daemon*. Nom du logiciel libre le plus couramment utilisé pour la mise en oeuvre des protocoles DNS. BIND se compose de trois parties, un serveur de noms de domaine, un client capable d'interroger d'autres serveurs DNS et des outils techniques de test.

DNS : *Domain Name System*. Service réparti permettant d'associer des ressources Internet (adresses IP, relais de messagerie...) à des noms de domaine.

DNSKEY : Enregistrement DNS servant à stocker une clé publique.

DNSSEC : *Domain Name System Security Extensions*. Ensemble d'extensions de sécurité du protocole DNS.

DS : *Delegation Signer* appelé Délégation du Signataire. Enregistrement DNS qui correspond à l'empreinte de la clé publique.

EDNS : *Extension mechanisms for DNS* est une extension du protocole DNS qui permet d'augmenter la taille de certains paramètres tels que la taille des réponses DNS, limitée autrefois à 512 octets.

KSK : *Key Signing Key*. Clé qui signe les clés ZSK. Son empreinte est publiée dans la zone parente pour créer une chaîne de confiance garantissant son authenticité ainsi que l'authenticité des clés signant la zone.

NS : *Name Server*. Appelé aussi serveur DNS ou serveur de nom. Serveur utilisé pour héberger un nom de domaine.

NSD : *Name Server Daemon*. Serveur DNS faisant autorité, développé par les NLnetLabs et utilisé par de nombreux TLD.

NTP : Appelé aussi *Network Time Protocol* ou Protocole d'Heure Réseau. Protocole qui permet de synchroniser, via un réseau informatique, l'horloge locale d'ordinateurs sur une référence d'heure

OpenDNSSEC : Logiciel de gestion automatique de clés et de signature DNSSEC.

Unbound : Résolveur DNS validant, maintenu par les NLnetLabs.

ZSK : *Zone Signing Key*. Clé qui signe les enregistrements de la zone. La partie publique de la ZSK permet de vérifier les signatures.