# RFC 5155 : DNSSEC Hashed Authenticated Denial of Existence

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 6 mars 2008. Dernière mise à jour le 1 septembre 2010

Date de publication du RFC : Mars 2008

https://www.bortzmeyer.org/5155.html

---

L'enregistrement NSEC3 permet de résoudre un problème agaçant de DNSSEC, le fait que ce dernier permette l'**enumération** de tous les noms de domaine d'une zone signée. NSEC3 permet donc d'établir la non-existence d'un domaine sans rien révéler sur les autres domaines.

DNSSEC (RFC 4033[1]) authentifie les enregistrements DNS en ajoutant un enregistrement RRSIG (RFC 4034) qui contient une signature des enregistrements qu'on veut authentifier. Le problème est que, si le nom de domaine demandé n'existe pas (réponse NXDOMAIN pour "*No Such Domain*"), il n'y a pas d'enregistrement à signer. Ce problème est connu sous le nom de **preuve de non-existence** et c'est un des points les plus difficiles de DNSSEC (section 5.4 du RFC 4035). Le RFC 4034, section 4, le traitait en introduisant un enregistrement NSEC, signé par un RRSIG, qui indiquait le domaine **suivant** celui demandé. En vérifiant que ce domaine suivant était bien situé après celui demandé (et que le domaine demandé était situé après le nom en partie gauche du NSEC), le résolveur pouvait être sûr de la non-existence du nom qu'il avait demandé.

Mais le gros problème de NSEC est qu'il permet l'**énumération**. Une fois qu'on a découvert le nom de domaine suivant, on peut passer au suivant, puis à celui d'après et ainsi de suite, on a récupéré toute la zone, même si son administrateur ne le voulait pas <https://www.bortzmeyer.org/recuperer-zone-dns.html>.

Le registre de .uk, Nominet, a clairement indiqué <http://www.nic.uk/tech/dnssectest/faq/#deploy> que c'était inacceptable pour eux <http://www.ops.ietf.org/lists/namedroppers/

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc4033.txt

namedroppers.2004/msg00780.html> et qu'ils ne déploieraient pas DNSSEC sans une solution à ce problème politico-légal.

Le nouvel enregistrement NSEC3 résout le problème en ne donnant pas le nom du domaine suivant mais son résumé cryptographique, son empreinte.

Le principe de NSEC 3 est le suivant. Les résumés de tous les noms sont classés par ordre alphabétique (c'est arbitraire, l'important est que cela soit reproductible). La partie gauche ("*owner name*") de l'enregistrement NSEC3 n'est pas un nom de domaine mais un résumé. Idem pour la partie droite.

L'enregistrement contient aussi quelques autres paramètres comme l'algorithme de calcul du résumé (section 3.1.1, les sections suivantes décrivant les autres paramètres). Voici un enregistrement NSEC3 au format suggéré section 3.3 :

```
2t7b4g4vsa5smi47k61mv5bv1a22bojr.example.             NSEC3   1 1 12 aabbccdd (
                          2vptu5timamqttgl4luu9kg21e0aor3s A RRSIG )
```

Le résolveur qui reçoit un NXDOMAIN peut calculer le résumé du nom qu'il a demandé et, si l'enregistrement NSEC3 est correct, ce résumé doit tomber entre les deux résumés du NSEC3 (celui de gauche et celui de droite). Ainsi, si la question est le nom D1, et que le résolveur reçoit :

```
... NXDOMAIN
H0     NSEC3   H2 ...
H0     RRSIG   ...
```

il doit calculer H1, le résumé de D1 et vérifier que H0 ¡ H1 ¡ H2 (et naturellement vérifier le RRSIG, la signature). Ne connaissant que H0 et H2, il ne peut pas retrouver D0 et D2. Notez bien que l'ordre ne concerne que H0, H1 et H2, D0, D1 et D2 sont situés dans un ordre complètement différent. Comme avec NSEC, tout repose sur l'existence d'un **ordre** des éléments sauf que, avec NSEC3, ce sont les résumés cryptographiques qui sont ordonnés.

La méthode de calcul du résumé est détaillée en section 5. Par exemple, un sel est ajouté au nom, pour rendre plus difficile les attaques par dictionnaire. Sans ce sel (annexe C.1 pour les détails), un méchant pourrait, hors-ligne, générer tous les résumés pour tous les noms potentiels et vérifier ensuite simplement leur existence. Toujours pour rendre plus difficile les attaques par dictionnaire, l'opération de résume est recommencé plusieurs fois, en suivant un paramètre qui indique le nombre d'itérations (il y en a toujours au moins une, section 10.3). Notez toutefois qu'en 2022, le RFC 9276 a remis en cause ces techniques et que sel et itérations sont désormais très déconseillés.

Ce nombre d'itérations a une influence sur les performances du résolveur et du serveur faisant autorité. Les chiffres exacts peuvent être trouvés dans l'article de Yuri Schaeffer, ≪ "*NSEC3 Hash Performance*" <http://www.nlnetlabs.nl/downloads/publications/nsec3_hash_performance.pdf> ≫. En pratique, les zones actuellement signées ont de une à dix itérations (`dig NSECPARAM $ZONE` pour le voir, il faut un dig ¿= 9.6).

Due à Kim-Minh Kaplan, voici une mise en œuvre en Perl du résumé :

```
use MIME::Base32;
use Digest::SHA1 qw(sha1);

my ($salt, $iteration, $name) = @ARGV;

$salt = pack "H*", $salt;
$name =~ y/A-Z/a-z/;
$name =~ s/\.*$//;
my @labels = split('\.', $name . ".", -1);
@labels = map { pack("C", length($_)) . $_ } @labels;
$name = join '', @labels;
foreach (0..$iteration) {
    $name = sha1("$name$salt");
}
print MIME::Base32::encode($name), "\n";
```

On doit appeler ce programme avec trois paramètres, le sel, le nombre d'itérations supplémentaires et le nom à résumer.

Notez que les dernières versions de BIND (depuis au moins la 9.7) sont livrées avec un programme plus perfectionné, `nsec3hash`, qui comprend en plus d'autres algorithmes que SHA-1. Voici un exemple d'utilisation :

```
[Récupérer les paramètres avec lesquels les noms sont résumés.]
% dig +short NSEC3PARAM pm.
1 0 1 BADFE11A

% nsec3hash  BADFE11A 1 1 pm
P3JUUU69O1AP1AI8DLNVO7L7A0SKT56S (salt=BADFE11A, hash=1, iterations=1)

[Et on retrouve bien ce résumé dans les enregistrements NSEC3 :

% dig +dnssec ANY nimportequoi.pm.
...
P3JUUU69O1AP1AI8DLNVO7L7A0SKT56S.pm. 5400 IN NSEC3 1 1 1 BADFE11A \
       L6U66FHPKD109EODA9R2SV8RGFSUAUQ3 NS SOA RRSIG DNSKEY NSEC3PARAM
```

NSEC3 contient plein d'autres fonctions intéressantes, par exemple la section 6 décrit un mécanisme de retrait (”*opt-out*”) permettant aux enregistrement NSEC3 de ≪ sauter ≫ par dessus les zones non signées (dans un TLD, comme .se, il est clair que la majorité des zones ne seront pas signées, au moins au début).

Les détails sont très complexes (et couverts dans les sections 7 et 8 du RFC) ce qui explique que certains auteurs de logiciels DNS comme Bert Hubert, auteur de PowerDNS, aient estimé qu'il serait difficile de faire du NSEC3 <http://psg.com/lists/namedroppers/namedroppers.2006/msg00176.html> en pratique (voir aussi son bon article sur DNSSEC <http://ds9a.nl/dnssec/index.html#nsec3>).

Mais NSEC3 est aujourd'hui mis en œuvre dans plusieurs logiciels comme BIND, NSD ou le résolveur UNBOUND <https://www.bortzmeyer.org/unbound-dnssec.html>. NSEC3 sera probablement plus complexe et plus lent que NSEC mais c'est le prix à payer pour garder le contrôle de ses données.

Avec BIND 9.6 (ou supérieur), voici comment on génère une clé RSA utilisable pour NSEC 3 :

```
% dnssec-keygen -a NSEC3RSASHA1 -b 1024 -n ZONE
```

et comment on signe :

```
dnssec-signzone -g -t -H 3 -3 babecafe $MYZONEFILE Kmykey.+007+30869
```

(`babecafe` est le sel, en hexadécimal.) Avec le retrait (”*opt-out*”), l'opération de signature prend une option supplémentaire, `-A`. Cette option permet de réduire considérablement le temps de signature (pour une zone d'un million de sous-zones, dont très peu sont signées, le retrait diminue le temps de signature de dix minutes à vingt secondes), au prix de la sécurité (il n'est désormais plus possible, si on utilise le retrait de prouver l'existence ou la non-existence d'une sous-zone non signée).

Ici, si on a une zone `example` comportant trois noms, `example` (l'apex), `preston.example` et `central.example` :

```
@ IN  SOA ns3.bortzmeyer.org. hostmaster.bortzmeyer.org. (
...
  IN  NS  ns3.bortzmeyer.org.
...
  IN DNSKEY 256 3 7 AwEAAa...=

central   IN   A   192.0.2.187

preston IN AAAA  2001:db8:8bd9:8bb0:a00:20ff:fe99:faf4
```

et qu'on la signe avec :

```
% dnssec-signzone -H 3 -3 babecafe -o example db.example Kmykey.+007+30869
```

(trois itérations supplémentaires, un sel qui vaut « babecafe »), on obtient un fichier `db.example.signed` qui comprend trois NSEC3, un par nom :

```
FIDQ6ATJMOUCGLV3PNHS9694C1LFDSDT.example. 43200 IN NSEC3 1 0 3 BABECAFE JKNM5TRUGDISFPUU0CCLEMG2GTGOD2IP AAA
JKNM5TRUGDISFPUU0CCLEMG2GTGOD2IP.example. 43200 IN NSEC3 1 0 3 BABECAFE O7FU7992IPFEUUUVC1A8NAF4255JF7JI NS
O7FU7992IPFEUUUVC1A8NAF4255JF7JI.example. 43200 IN NSEC3 1 0 3 BABECAFE FIDQ6ATJMOUCGLV3PNHS9694C1LFDSDT A R
```

et le programme Perl ci-dessus permet de retrouver quel nom a été résumé pour donner la partie gauche de chacun de ces NSEC3 :

```
% perl hash-nsec3.pl babecafe 3 preston.example
FIDQ6ATJMOUCGLV3PNHS9694C1LFDSDT
```

et on trouve ainsi que le premier NSEC3 de la liste était celui de `preston.example` (mais l'opération inverse est impossible : c'est le principe du résumé).

Merci à Kim Minh Kaplan pour son aide à m'aider à comprendre ce difficile protocole et pour sa relecture.