rfc4034

Enregistrements de ressource pour les extensions de sécurité DNS

Les extensions de sécurité DNS introduisent 4 nouveaux type d'enregistrement de ressource DNS : DNS Public Key (DNSKEY), Resource Record Signature (RRSIG), Next Secure (NSEC), et Delegation Signer (DS). Ce document définis le but de chacun de ces RR, le format RDATA de ces RR, et leur format de présentation (ASCII).

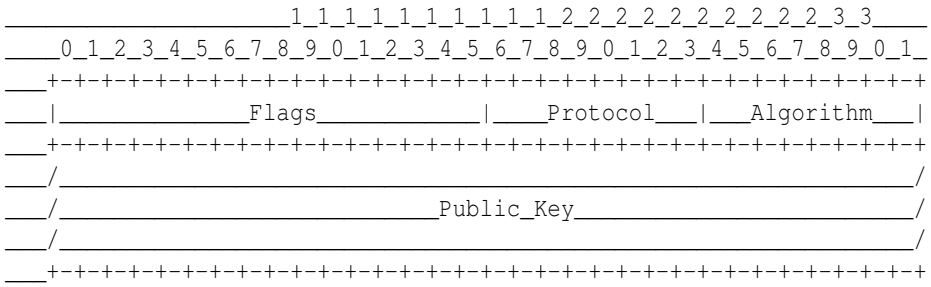
DNSSEC utilise la cryptographie à clé publique pour signer et authentifier les jeus d'enregistrement de ressources (RRsets). Les clé publiques sont stockées dans les RR DNSKEY et sont utilisé dans le processus d'authentification DNSSEC. : Une zone signe son RRset authoritatif en utilisant une clé privée et stocke la clé publique correspondante dans un RR DNSKEY. Un résolveur peut ainsi utiliser la clé publique pour valider les signature couvrant les RRset dans la zone, et donc les authentifier.

Le RR DNSKEY n'est pas prévu comme un enregistrement pour stocker des clé publiques arbitraires et ne doit pas être utilisé pour stocker des certificats ou des clé publiques non liées à l'infrastructure DNS.

La valeur Type pour le RR DNSKEY est 48, le RR DNSKEY est indépendant de la class et n'a pas de requis de TTL spécial.

Format DNSKEY RDATA

Le RDATA pour un RR DNSKEY consiste d'un champ de Flags à 2 octets, un champ Protocole 1 octet, en champ Algorithme 1 octet, et le champ de clé publique :



Champ Flags

Le bit 8 est le flag de clé de zone. Si le bit 7 est mis, l'enregistrement DNSKEY maintient une clé de zone DNS, et le nom propriétaire du RR DNSKEY doit être le nom d'une zone. S'il est non-mis, l'enregistrement DNSKEY maintien un autre type de clé publique DNS et ne doit pas être utilisé pour vérifier les RRSIG qui couvrent les RRsets.

Le bit 15 est le flag de point d'entrée sécurisé, décris dans la rfc3757. Mis, l'enregistrement DNSKEY maintient une clé utilisée comme point d'entrée. Ce flag est seulement prévu pour être un départ de signature de zone, ou pour débugger les logiciel ; les validateurs ne doivent pas altérer leur comportement durant le processus de validation de signature en se basant sur ce bit. Il signifie également qu'un RR DNSKEY avec le bit SEP mis à également besoin du flag Zone Key mis pour être capablede générer des signature légalement. Un RR DNSKEY avec SEP mis et le bit 7 non mis ne doivent pas être utilisés pour vérifier les RRSIG qui couvrent les RRsets.

Champ Protocol

Le champ Protocol doit avoir la valeur 3, et le RR DNSKEY doit être traité comme invalide durant la vérification de la signature si une autre valeur est trouvée.

Champ Algorithm

Identifie l'algorithme cryptographique à clé publique et détermine le format du champ de clé publique.

Champ Public Key

Maintien la clé publique. Le format dépend de l'algorithme de la clé stockée et est décrit dans un document séparé.

Notes sur le design de DNSKEY RDATA

Bient que le chaps Protocol a toujours la valeur 3, il est retenu pour compatibilité avec les premières versions de l'enregistrement KEY

Format de présentation des RR DNSKEY

Le format de présentatin de la portion RDATA est comme suit :

- Le champ Flag doit être représenté comme entier décimal non-signé. Avec les flags actuellement définis, les valeurs possibles sont : 0, 256 et 257.
- Le champ protocol doit être représenté comme entier décimal non-signé avec une valeur de 3
- Le champ algorithm doit être respresenté comme entier décimal non-signé, ou comme mnémonique d'algortihme.
- La clé publique doit être représentée en base64.

Exemple de RR DNSKEY

Le RR DNSKEY suivant stocke un clé de zone DNS pour example.com.

```
example.com. 86400 IN DNSKEY 256 3 5 (_AQPSKmynfzW4kyBv015MUG2DeIQ3
_____________________________________Cbl+BBZH4b/0PY1kxkmvHjcZc8no
_____________________________________kfzj31GajIQKY+5CptLr3buXA10h
_____________________________________WqTkF7H6RfoRqXQeogmMHfpftf6z
_____________________________________Mv1LyBUgia7za6ZEzOJBOztyvhjL
_____________________________________742iU/TpPSEDhm2SNKLijfUppn1U
_____________________________________aNvv4w==__)
```

Les 4 premiers champs spécifient le nom propriétaire, le TTL, la Classe , et le type de RR (DNSKEY). Une valeur 256 indique que le bit de clé de zone est mis. La valeur 3 est la valeur de protocole. La valeur 5 indique l'algorithme à clé publique (RSA/SHA1). Le reste est la clé publique encodé en base64

L’enregistrement de ressource RRSIG

DNSSEC utilise la cryptographie à clé publique pour signer et authentifier les RRset. Les signatures numériques sont stockées dans les RRSIG et sont utilisé dans le processus d’authentification DNSSEC décris dans la rfc4035. Un validateur peut utiliser ces RRSIG pour authentifier les RRest de la zone. Le RR RRSIG doit seulement être utilisé pour valider les signatures numérique utilisés pour sécuriser les opérations DNS.

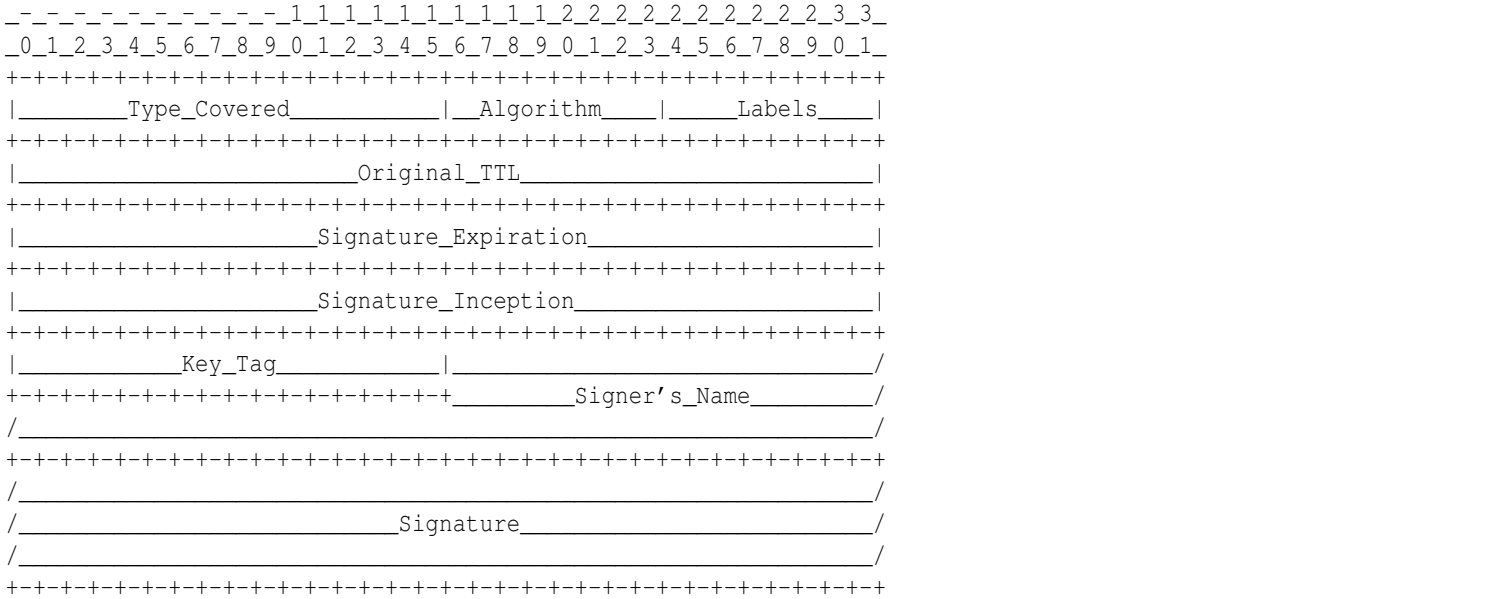
Un enregistrement RRSIG contient la signature pour un RRset avec un nom, classe et type. Le RR RRSIG spécifie un interval de validité pour la signature et utilise Algorithm, nom du signataire, et le Key Tag pour identifier le RR DNSKEY contenant la clé publique qu’un validateur peut utiliser pour vérifier la signature.

Parce que tout RRset authoritatif dans une zone doit être protégé par une signature numérique, les RR RRSIG doivent être présents pour les noms contenant un RR CNAME. C’est un changement de la spécification DNS (rfc1034), qui status que si un CNAME est présent pour un nom, il est seulement du type permis pour ce nom. Un RRSIG et NSEC doivent exister pour le même nom qu’un enregistrement de ressource CNAME dans une zone signée.

La valeur de Type pour le RR RRSIG est 46. Le RR RRSIG est indépendant de la classe. Un RR RRSIG doit avoir la même classe que le RRset qu’il couvre. La valeur TTL d’un RRSIG doit correspondre au TTL du RRset qu’il couvre. C’est une exception des règles de la rfc2181 pour les valeurs TTL des RR individuels dans un RRset : les RR RRSIG individuels avec le même nom propriétaire ont différentes valeur TTL si le RRset qu’ils couvrent ont différentes valeurs TTL.

Format RRSIG RDATA

Le RDATA pour un RR RRSIG consiste d’un champ Type à 2 octets, un champ Algorithm à 1 octet, un champ Labels 1 octet, un champ TTL 4 octets, un champ Signature Expiration 4 octets, un champ Signature Inception 4 octets, en tag Key 2 octets, le champ du nom du signataire et le champ Signature.



Champ Type Covered

Ce champ identifie le type du RRset couvert pas cet enregistrement RRSIG

Champ Algorithm Number

Ce champ identifie l'algorithme cryptographique utilisé pour créer la signature.

Champ Labels

Spécifie le nombre de labels dans le nom propriétaire du RR RRSIG. La signification de ce champs est qu'un validateur l'utilise pour déterminer si la réponse a été synthétisée depuis un wildcard. Si c'est le cas, il peut être utilisé pour déterminer quel nom propriétaire a été utilisé pour générer la signature.

Pour valider une signature, le validateur a besoin du nom propriétaire original qui a été utilisé pour créer la signature. Si ce nom contient un label wildcard, le nom propriétaire peut avoir été étendu par le serveur durant le processus de réponse, auquel cas le validateur doit reconstruire le nom original pour pouvoir valider la signature. La rfc4035 décris comment utiliser le champ Labels pour reconstruire le nom propriétaire original.

La valeur du champ Label ne doit pas compter le label null (root) qui termine le nom propriétaire ni le label wildcard. La valeur du champ Labels doit être inférieur ou égal au nombre de labels dans le nom propriétaire RRSIG. Par exemple, "www.example.com." a une valeur de champ Labels de 3, et "*.example.com." a une valeur de 2. Root ('.') a une valeur de 0.

Bien que le label wildcard n'est pas inclus dans le compteur stocké dans le champ Labels du RR RRSIG, le label wildcard fait partie du nom propriétaire du RRset quant la signature est générée ou vérifiée.

Champ TTL Original

Spécifie le TTL du RRset couvert tel qu'il apparaît dans le zone authoritative.

Ce champ est nécessaire parce que le cache du résolveur décrémente la valeur TTL d'un RRset en cache. Pour valider une signature, un validateur nécessite le TTL original. la rfc4035 décris comment utiliser ce champ pour reconstruire le TTL original.

Champs Signature Expiration et Inception

Ces champ spécifie la période de validité pour la signature. L'enregistrement RRSIG ne doit pas être utilisé pour authentifier avant la date de début, et ne doit pas être utilisée pour l'authentification après la date d'expiration.

Ce champsspécifie une date et heure sous la forme d'un nombre non-signé 32bits en secondes depuis le 1 Janvier 1970 00 :00 :00 UTC. L'interval le plus long qui peut être exprimé par ce format est approximativement de 136ans. Un RR RRSIG peut avoir un champ Expiration qui est numérique plus petit que le champ Inception si la valeur du champ du champ d'expiration est proche du maximum 32bits ou si la signature a une durée de vie longue.

Champ Key Tag

Contient la valeur de tag de clé du RR DNSKEY qui valide cette signature.

Champs Signer's Name

Ce champ identifie le nom propriétaire du RR DNSKEY qu'un validateur est supposé utiliser pour valider cette signature. Ce champ doit contenir le nom de la zone du RRset couvert. Un émetteur ne doit pas utiliser la compression de nom DNS dans ce champ en transmettant un RR RRSIG

Champ Signature

Le champ signature contient la signature cryptographique qui couvre le RDATA RRSIG (excluant le champ signature) et le RRset spécifié par le nom du RRSIG, la classe RRSIG, et le type RRSIG. Le format de ce champ dépend de l'algorithme utilisé.

Calcul de Signature

Une signature couvre le RDATA RRSIG et couvre les données RRset spécifiées par le nom propriétaire RRSIG, la classe et le type couvert RRSIG. Le RRset est sous la forme canonique et est signé comme suit :

```
signature = sign(RRSIG_RDATA | RR(1) | RR(2)... )
```

ou 'l' dénote une concaténation. RRSIG_RDATA est le format des champs RDATA RRSIG avec le champ Signer's Name sous forme canonique et le champ Signature est exclus.

```
RR(i) = owner | type | class | TTL | RDATA length | RDATA
```

owner est le nom propriétaire pleinement qualifié du RRset sous la forme canonique. Chaque RR doit avoir le même nom propriétaire et la même classe que le RR RRSIG. Chaque RR dans le Rset doit avoir le type RR listé dans le champ Type Covered du RR RRSIG. Tous nom DNS dans le champ RDATA de chaque RR doit être sous la forme canonique, et le RRset doit être trié dans l'ordre canonique.

Format de présentation RR RRSIG

Le format de présentation de la portion RDATA est comme suit :

- Le champ Type Covered est représenté comme un mnémonique du type RR. Quand le mnémonique n'est pas connus, la représentation du TYPE rfc3597 doit être utilisé.
- Le champ Algorithm doit être représenté soit comme entier décimal non-signé ou comme mnémonique.
- Le champ Labels doit être représenté comme entier décimal non-signé
- Le champ Original TTL doit être représenté comme entier décimal non-signé
- Les champs Signature Expiration Time et Inception Time doivent être représenté soit comme entier décimal non-signé indiquant les secondes depuis le 1 Janvier 1970 00 :00 :00 UTC, ou sous la forme YYYMMDDHHmmSS en UTC.

Noter qu'il est toujours possible de faire la distinction entre les 2 format parce que le format YYYYMMDDHHmmSS fait toujours exactement 14 chiffres, alors que la représentation décimale 32bits ne peut jamais dépasser 10 chiffres.

- Le champ Key Tag doit être représenté comme entier décimal non-signé
- Le champ Signer's Name doit être représenté en nom de domaine
- Le champ Signature est représenté en Base64.

Exemple de RR RRSIG

Le RR RRSIG suivant stocke la signature pour un RRset A de host.example.com :

```
host.example.com. 86400 IN RRSIG A 5 3 86400 20030322173103 (
                                    20030220173103 2642 example.com.
                                    oJB1W6WNGv+ldvQ3WDG0MQkg5IEhjRip8WTr
                                    PYGv07h108dUKGMeDPKijVCHX3DDKdfb+v6o
                                    B9wfuh3DTJXUAfI/M0zmO/zz8bW0Rznl8O3t
                                    GNazPwQKkRN20XPXV6nwwfoXmJQbsLNrLfkG
                                    J5D6fwFm8nN+6pBzeDQfsS3Ap3o= )
```

Les 4 premiers champs spécifient le nom propriétaire, le TTL, la classe, et le type RR (RRSIG). Le A represente le type couvert. La valeur 5 identifie l'algorithme (RSA/SHA1), la valeur 3 est le nombre de labels dans le nom propriétaire original. 86400 est le TTL d'origine pour le RRset A couvert. 20030322173103 et 20030220173103 sont les date de début et de fin de validité de la signature. 2642 est le Key Tag, et example.com. est le nom du signataire.

Noter la combinaison du nom, classe et type couvert du RR indiquent que ce RRSIG couvret le RRset A host.example.com. La valeur Labels indique qu'il n'y a pas d'expansion wildcard. Algorithm, Signer's Name et Key Tag indiquent que cette signature peut être authentifiée en utilisant un RR DNSKEY dans la zone example.com dont l'algorithme est 4 et dont le tag de clé est 2642

Enregistrement de ressource NSEC

Le RR NSEC liste 2 choses séparée : le prochain nom propriétaire qui contient les données autoritatives ou un point de délégation RRset NS, et le jeu de type RR présents au nom propriétaire du RR NSEC (rfc3845). Le jeu complet de RR NSEC dans une zone indique quels RRset autoritatif existe dans une zone et forme également une chaîne de noms propriétaires authoritatifs dans la zone. Cette information est utilisée pour fournir un déni d'existance authentifié pour les données DNS, comme décris dans la rfc4035.

Parce que tout nom autoritatif dans une zone doit faire partie d'une chaîne NSEC, les RR NSEC doivent être présents pour les noms contenant un RR CNAME. C'est un changement par rapport à la spécification DNS (rfc1034), qui status que si un CNAME est présent pour un nom, il est seulement du type permis pour ce nom. Un RRSIG et NSEC doivent exister pour le même nom comme le fait un RR CNAME dans une zone non-signée.

La valeur du type pour un RR NSEC est 47. Le RR NSEC est indépendant de la classe et devrait avoir le même TTL que le champ SOA minimum TTL.

Format NSEC RDATA

Le RDATA d'un RR NSEC est comme suit :

```
_-_-_-_-_-_-_-_-_-_-_-_1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/_____________________Next_Domain_Name_____________________/
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/_____________________Type_Bit_Maps_____________________/
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Champ Next Domain Name

Ce champ contient le nom propriétaire qui a les données autoritatives ou contient un RRset NS de point de délégation. La valeur de ce

champ dans le dernier enregistrement NSEC dans la zone est le nom de l'apex de zone (le nom propriétaire du RR SOA de la zone). Cela indique que le nom propriétaire du RR NSEC est le dernier nom dans l'ordre canonique de la zone.

Un émetteur ne doit pas utiliser la compression de noms DNS dans ce champ en le transettant. Les noms propriétaires des RRset pour lesquel la zone donnée n'est pas autoritative (comme les enregistrements glue) ne doivent pas être listés dans ce champ sauf si au moins un RRset autoritatif existe avec le même nom propriétaire.

Champ Type Bit Maps

Identifie les types du RRset qui existe au niveau du nom propriétaire du RR NSEC. L'espace du type RR est splitté en 256 blocks, chacun représentant les 8bits LSB de l'espace du type RR 16-bits. Chaque block qui a au moins un type RR actif est encodé en utilisant un simple octet de numéro de fenêtre (de 0 à 255), un simple octet de longueur (de 1 à 32) indiquant le nombre d'octets utilisés pour le bitmap de block, et jusqu'à 32 octets (256bits) de bitmaps. Les blocks sont présent dans le RDATA du RR NSEC en augmentant l'ordre numérique :

```
Type Bit Maps Field = ( Window Block # | Bitmap Length | Bitmap )+
```

où "|" dénote la concaténation

Chaque bitmap encode les 8bits LSB des types RR dans le block, dans l'ordre de bit réseau. Le premier bit est le bit 0. Pour le block de fenêtre 0, le bit 1 correspond au type RR 1 (A), le bit 2 correspond au RR type 2 (NS), etc. pour le block de fenêtre 1, le bit 1 correspond au RR type 257, et le bit 2 correspond au RR type 258. Si un bit est mis, il indique qu'un RRset de ce type est présent pour le nom propriétaire NSEC.

Les bits représentant les pseudo-type doivent être effacés, vu qu'ils n'apparaîssent pas dans la zone de données. Si rencontré, ils doivent être ignorés

Les blocks sans type ne doivent pas être inclus. Les 0 restant dans le bitmap doivent être omis. La longueur de chaque bitmap de block est déterminé par le code de type avec la valeur numérirque la plus grande.

Le bitmap pour le RR NSEC au point de délégation nécessite une attension spéciale. Les bits correspondant au RRset NS de délégation et les type RR pour lequels la zone parent a une donnée autoritative doivent être mis ; les bits correspondants à tout RRset non-NS pour lequel le parent n'est pas autoritatif doivent être effacés.

Une zone ne doit pas inclure le RR NSEC pour un domaine qui ne maintient que des enregistrements glue.

Ajout des noms wildcard dans RDATA NSEC

Il un nom wildcard apparaît dans une zone, le label wildcard est traité comme symbole littéral et est traité de la même manière que pour tout nom propriétaire.

Format de présentation RR NSEC

Le format de présentation de la portion RDATA est comme suit :

- Le champ Next Domain Name est représenté comme un nom de domaine
- Le champs Type Bit Map est resprésenté comme une séquence de mnémoniques de type RR.

Exemple de RR NSEC

L'exemple suivant identifie les RRset associés avec alfa.example.com. et identifie le nom autoritatif suivant après alfa.example.com.

```
alfa.example.com. 86400 IN NSEC host.example.com. ( A MX RRSIG NSEC TYPE1234 )
```

Les 4 premiers champs identifient le nom, TTL, class et type RR (NSEC). host.example.com. est le nom autoritatif suivant après alfa.example.com dans l'ordre canonique. Les mnémoniques A, MX, RRSIG, NSEC et TYPE1234 indiquent qu'il y a des RRsets correspondants associés avec le nom alfa.example.com.

La section RDATA du RR NSEC serait encodé :

```
0x04 'h' 'o' 's' 't'
0x07 'e' 'x' 'a' 'm' 'p' 'l' 'e'
0x03 'c' 'o' 'm' 0x00
0x00 0x06 0x40 0x01 0x00 0x00 0x00 0x03
0x04 0x1b 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x20
```

Assumant que le validateur peut authentifier cet enregistrement NSEC, il peut être utilisé pour prouver que beta.example.com n'existe pas, ou pour prouver qu'il n'y a pas d'enregistrement AAAA associé avec alfa.example.com.

Enregistrement de ressource DS

Le RR DS réfère à un RR DNSKEY et est utilisé dans le processus d'authentification DNSKEY. Un RR DS réfèrre à un RR DNSKEY en stockant le tag de clé, le numéro d'algorithme, et un hash du RR DNSKEY. Noter que bien que le hash soit suffisant pour identifier le clé publique, stocker le tag de clé et l'algorithme aide à le processus d'identification plus efficacement. En authentifiant l'enregistrement DS, un résolveur peut authentifier le RR DNSKEY pour lequel le RR DS pointe.

Le RR DS et son RR DNSKEY correspondant ont le même nom propriétaire, mais ils sont stockés à différents emplacements. Le RR DS apparaît seulement côté parental d'une délégation, et est autoritatif dans la zone parent. Par exemple, le DS RR pour example.com est stocké dans la zone com. Le RR DNSKEY est stocké dans la zone example.com. Cela simplifie la gestion de zone DNS et la signature de zone en introduisant une traitement de réponse spécial pour le RR DS.

Le type pour le RR DS est 43, et est indépendant de la classe. Le RR DS n'a pas de prérequis de TTL spécial.

Format DS RDATA

Le RDATA pour un RR DS consiste d'un champ 2 octets Key Tag, un champ algorithme 1 octet Algorithm un champ 1 octet Digest Type, et un champ Digest

```
                    1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Key_Tag              |  Algorithm    |  Digest_Type  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                                                               /
/                            Digest                             /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Champ Key Tag

Le champ Key Tag liste le tag de clé du RR DNSKEY réferré par l'enregistrement DS. Le Key Tag utilisé par le RR DS est identique au Key Tag utilisé pas les RR RRSIG.

Champ Algorithm

Le champ algorithme liste le numéro d'algorithme du RR DNSKEY correspondant à cet enregistrement DS.

Champ Digest Type

Le RR DS réfèrre à un RR DNSKEY en inclant un hash de ce RR DNSKEY. Ce champ identifie l'algorithme utilisé pour construire le hash.

Champ Digest

Contient le hash du RR DNSKEY. Le hash est calculé en concaténant la forme canonique du nom propriétaire pleinement qualifié du RR DNSKEY uavec le RDATA DNSKEY et applique ainsi l'algorithme de hashage :

```
digest = digest_algorithm( DNSKEY owner name | DNSKEY RDATA);
```

"|" dénote la concaténation.

```
DNSKEY RDATA = Flags | Protocol | Algorithm | Public Key.
```

La taille du hash peut varier en fonction de l'algorithme et de la taille du RR DNSKEY.

Traitement des RR DS en validant les réponses

Les RR DS lient la chaîne d'authentification entre les zones, donc le RR DS nécessite une attention particulière dans le traitement. Le RR DNSKEY réferré dans le RR DS doit être une clé de zone DNSSEC. Les flags du RR DNSKEY doivent avoir le bit 7 mis. Si les flags DNSKEY n'indiquent pas une clé de zone DNSSEC, le RR DS (et le RR DNSKEY correspondant) ne doivent pas être utilisés pour le processus de validation.

Format de présentation du RR DS

Le champ key tag doit être représenté comme entier décimal non-signé. Le champ algorithm doit être réprésenté comme entier décimal non-signé ou mnémonique d'algorithme. Le champ Digest Type doit être réprésenté comme entier décimal non-signé. Le champ Digest doit être reprsenté comme une séquence de chiffres hexadécimaux sensible à la casse.

Exemple de RR DS

L'exemple suivant montre un RR DNSKEY et son RR DS correspondant :

```
dskey.example.com. 86400 IN DNSKEY 256 3 5 ( AQOeiiR0GOMYkDshW
_____________________________________________fwJr1AYtsmx3TGkJaNXV
_____________________________________________2pHm822aJ5iI9BMzNXxe
_____________________________________________DRD99WYwYqUSdjMmmAph
_____________________________________________egXd/M5+X7OrzKBaMbCV
_____________________________________________Uh6DhweJBjEVv5f2wwjM
_____________________________________________nOf+EPbtG9DMBmADjFDc
_____________________________________________ljwvFw==
_____________________________________________) ; key id = 60485

dskey.example.com. 86400 IN DS 60485 5 1 ( 2BB183AF5F22588179A
_____________________________________________98631FAD1A292118 )
```

Le 4 premiers champs spécifient le nom, TTL, classe et type RR (DS). La valeur 60485 est le tag de clé pour dskey.example.com. Les RR DNSKEY, et la valeur 5 dénotent l'algorithme utilisé par ce RR DNSKEY. La valeur 1 est l'algorithme utilisé pour construire le hash, et le reste est le hash.

Forme canonique et ordre d'enregistrement de ressource

Cette section définis une forme canonique pour les enregistrements de ressource, un ordre canonique des noms DNS, et un ordre canonique d'enrgistrements de ressource dans un RRset. Un ordre de nom canonique est requis pour construire la chaîne de nom NSEC. Une forme RR canonique est l'ordre dans un RRset sont requis pour construire et vérifier les RR RRSIG.

Ordre de noms DNS canonique

Pour la sécurité DNS, les noms propriétaires sont ordonnés en traitant les labels individuels comme chaînes d'octets non-signés justifiés à gauche. L'absence d'un trie d'octet avant un octet zéro, et les lettre US-ASCII majuscule sont traités comme s'ils étaient en minuscule.

Pour calculer l'ordre canonique d'un jeu de noms DNS, on commence par trier les noms en accord avec leur labels les plus signifiant. Pour les noms dans lequel le label le plus signifiant est identique, on continue à trier en accord avec le label suivant, et ainsi de suite.

Par exemple, les noms suivants sont triés par ordre de nom DNS canoniques. Le lable le plus signifiant est "example". À ce niveau, "example" est en premier, suivis par les noms se terminant dans "a.example", puis par les noms se terminant par "z.example". Les noms dans chaque niveau sont traités de la même manière :

```
example
a.example
yljkjljk.a.example
Z.a.example
zABC.a.EXAMPLE
z.example
\001.z.example
.z.example
\200.z.example
```

Forme RR canonique

Pour la sécurité DNS, la forme canonique d'un RR est le format des RR où :

1. Tout nom de domaine dans le RR est étendu et pleinement qualifié)
2. Toute lettre US-ASCII majuscule dans le nom propriétaire du RR sont remplacés par les lettres minuscules.
3. Si le type du RR est NS, MD, MF, CNAME, SOA, MB, MG, MR, PTR, HINFO, MINFO, MX, HINFO, RP, AFSDB, RT, SIG, PX, NXT, NAPTR, KX, SRV, DNAME, A6, RRSIG, ou NSEC, toutes les lettres US-ASCII majuscule dans les noms DNS contenur dans le RDATA sont remplacés par des minuscules.
4. Si le nom propriétaire du RR est un nom wildard, le nom propriétaire est dans sa forme non-étendue originale, inclunat le label '*'
5. Le TTL du RR est mis à sa valeur d'origine telle qu'elle apparaît dans la zone autoritative d'origine ou le champ Original TTL du RR RRSIG.

Ordre RR canonique dans un RRset

Pour la sécurité DNS, les RR avec le même nom propriétaire, class et type sont triés en traitant la portion RDATA de la forme canonique de chaque RR comme séquence d'octet non-signé justifié à gauche dans laquelle l'absence d'un octet trie avant un octet zéro.

La rfc2181 spécifie qu'un RRset n'est pas autorisé à contenir des enregistrements dupliqués (plusieurs RR avec le même nom propriétaire, class, type, et RDATA). Cependant, si une implémentation détecte des RR dupliqués en plaçant le RRset dans une forme canonique, il doit traite comme une erreur de protocole. Si l'implémentation choisis de gérer cette erreur de protocole dans l'esprit du principe de robustesse, il doit supprimer les RR dupliqués pour calculer la forme canonique du RRset.

Considérations de sécurité

L'enregistrement DS pointe vers un RR DNSKEY en utilisant un hash cryptographique, le type d'algorithme de clé, et un tag de clé. L'enregistrement DS est prévu pour identifier un RR DNSKEY existant, mais il est théoriquement possible pour un attaquant de générer un DNSKEY qui corresponde à ces champs. La probabilité de construire un DNSKEY qui correspond dépend de l'algorithme de hasha utilisé. Actuellement, seul SHA-1 est définis.

Le tag de clé est utilisé pour sélectionner le RR DNSKEY efficacement, mais il n'identifie pas de manière unique un simple RR DNSKEY. Il est possible que 2 RR DNSKEY distincts aient le même nom propriétaire, le même type d'algorithme, et le même tag de clé. Une implémentation qui utiliser seulement le tag de clé pour sélectionner un RR DNSKEY peut sélectionner la mauvaise clé dans certaines circonstances.

Algorithme et types de hashage

Les extensions de sécurité DNS sont conçus pour être indépendants des algorithmes cryptographique. Les enregistrements de ressource DNSKEY, RRSIG, et DS utilisent tous un numéro d'algorithme DNSSEC pour identifier l'algorithme utilisé par le RR. le RR DS spécifie également un numéro d'algorithme de hashage.

```
Value_Algorithm_[Mnemonic]__Signing__References___Status
__---_-------------_------_-------__------
____0___reserved
____1___RSA/MD5_[RSAMD5]_________n______[RFC2537]__NOT_RECOMMENDED
____2___Diffie-Hellman_[DH]______n______[RFC2539]___-
____3___DSA/SHA-1_[DSA]__________y______[RFC2536]__OPTIONAL
____4___Elliptic_Curve_[ECC]_____________TBA_______-
____5___RSA/SHA-1_[RSASHA1]______y______[RFC3110]__MANDATORY
__252___Indirect_[INDIRECT]______n__________________-
__253___Private_[PRIVATEDNS]_____y______see_below__OPTIONAL
__254___Private_[PRIVATEOID]_____y______see_below__OPTIONAL
```

__255___reserved

Le numéro d'algorithme 253 est réservé pour une utilisation privée et n'est jamais assigné à un algorithme spécifique. La zone de clé publique dans le RR DNSKEY et la zone de signature dans le RR RRSIG commence avec un nom de domaine encodé, qui ne doit pas être compressé. Le nom de domaine indique l'algorithme utilisé, et le reste de la zone de clé publique est déterminé par cet algorithme.

L'algorithme 254 est réservé pour une utilisation privée et n'est jamais assigné à un algorithme spécifique. La zone de clé publique dans le RR DNSKEY et la zone de signature dans le RR RRSIG commencent avec un octet de longueur non-signé suivi par un OID encodé BER du cette longueur. L'OID indique l'algorithme de clé privée utilisé.

Types de hash DNSSEC

Un champ Digest Type dans un RR DS identifie l'algorithme de hasha utilisé par le RR. La table suivante liste les types actuellement définis :

0 Réservé
1 SHA-1
2-255 non-assigné

Calcule du tag de clé

Le champ key tag dans les RR RRSIG et DS fournissent un mécanisme pour sélectionner une clé publique efficacement. Dans la plupars des cas, une combinaison de nom propriétaire, algorithme, et tag de clé peut efficacement identifier un enregistrement DNSKEY. Il est essentiel de noter que le Key Tag n'est pas unique.

Le tag de clé est le même pour tous les types d'algorithmes DNSKEY excpté l'algorithme 1. L'algorithme de tag de clé est la somme du format du RDATA DNSKEY séparé en 2 groupes d'octets. D'abord, le RDATA est traité comme séries de groupes de 2 octets. Ces groupes sont ainsi ajoutés ensemble, en ignorant les bits de retenu.

L'implémentation de référence ANSI C suivante calcule la valeur d'un Key Tag. Cette implémentation de référence s'applique à tous les algorithmes excepté l'algorithme 1. Le code est écrit pour la clarté, non l'efficacité :

```
/*
 On assure que l'entier fait au moins 16 bits
 Le premier octet du tag de clé sont les 8 bits MSB de la valeur retournée
 Le second octet du tag de clé sont les 8 bits LSB de la valeur retournée
/
unsigned int
keytag (
 unsigned char key[], /*the RDATA part of the DNSKEY RR*/
 unsigned int keysize /*the RDLENGTH*/
)
{
 unsigned long ac; /*assumed to be 32 bits or larger*/
 int i; /*loop index*/

 for ( ac = 0, i = 0; i < keysize; ++i )
  ac += (i & 1) ? key[i] : key[i] « 8;
 ac += (ac » 16) & 0xFFFF;
 return ac & 0xFFFF;
}
```

Algorithme 1 pour le tag de clé

Le type de clé pour l'algorithme 1 (RSA/MD5) est définis différemment du tag de clé pour tous les autres algorithmes, pour des raisons historiques. Pour un RR DNSKEY avec l'algorithme 1, le tag de clé est les 16bits les plus signifiant des 24bits les moins signifiants dans le modulo de la clé publique. Noter que l'algorithme 1 n'est pas recommandé.