DNS Extensions
Internet-Draft
Expires: November 15, 2004

R. Arends
Telematica Instituut
R. Austein
ISC
M. Larson
VeriSign
D. Massey
USC/ISI
S. Rose
NIST

May 17, 2004

Resource Records for the DNS Security Extensions
draft-ietf-dnsext-dnssec-records-08

Status of this Memo

Copyright Notice

Abstract

This document is part of a family of documents that describes the DNS Security Extensions (DNSSEC). The DNS Security Extensions are a collection of resource records and protocol modifications that provide source authentication for the DNS. This document defines the public key (DNSKEY), delegation signer (DS), resource record digital

signature (RRSIG), and authenticated denial of existence (NSEC) resource records. The purpose and format of each resource record is described in detail, and an example of each resource record is given.

This document obsoletes RFC 2535 and incorporates changes from all updates to RFC 2535.

Table of Contents

```
1.  Introduction . . . . . . . . . . . . . . . . . . . . . . . . .  4
  1.1   Background and Related Documents . . . . . . . . . . . . .  4
  1.2   Reserved Words . . . . . . . . . . . . . . . . . . . . . .  4
  1.3   Editors' Notes . . . . . . . . . . . . . . . . . . . . . .  4
    1.3.1   Technical Changes or Corrections . . . . . . . . . . .  4
    1.3.2   Typos and Minor Corrections  . . . . . . . . . . . . .  5
2.  The DNSKEY Resource Record . . . . . . . . . . . . . . . . . .  6
  2.1   DNSKEY RDATA Wire Format . . . . . . . . . . . . . . . . .  6
    2.1.1   The Flags Field  . . . . . . . . . . . . . . . . . . .  6
    2.1.2   The Protocol Field . . . . . . . . . . . . . . . . . .  7
    2.1.3   The Algorithm Field  . . . . . . . . . . . . . . . . .  7
    2.1.4   The Public Key Field . . . . . . . . . . . . . . . . .  7
    2.1.5   Notes on DNSKEY RDATA Design . . . . . . . . . . . . .  7
  2.2   The DNSKEY RR Presentation Format  . . . . . . . . . . . .  7
  2.3   DNSKEY RR Example  . . . . . . . . . . . . . . . . . . . .  8
3.  The RRSIG Resource Record  . . . . . . . . . . . . . . . . . .  9
  3.1   RRSIG RDATA Wire Format  . . . . . . . . . . . . . . . . .  9
    3.1.1   The Type Covered Field . . . . . . . . . . . . . . . . 10
    3.1.2   The Algorithm Number Field . . . . . . . . . . . . . . 10
    3.1.3   The Labels Field . . . . . . . . . . . . . . . . . . . 10
    3.1.4   Original TTL Field . . . . . . . . . . . . . . . . . . 11
    3.1.5   Signature Expiration and Inception Fields  . . . . . . 11
    3.1.6   The Key Tag Field  . . . . . . . . . . . . . . . . . . 11
    3.1.7   The Signer's Name Field  . . . . . . . . . . . . . . . 12
    3.1.8   The Signature Field  . . . . . . . . . . . . . . . . . 12
  3.2   The RRSIG RR Presentation Format . . . . . . . . . . . . . 13
  3.3   RRSIG RR Example . . . . . . . . . . . . . . . . . . . . . 13
4.  The NSEC Resource Record . . . . . . . . . . . . . . . . . . . 15
  4.1   NSEC RDATA Wire Format . . . . . . . . . . . . . . . . . . 15
    4.1.1   The Next Domain Name Field . . . . . . . . . . . . . . 15
    4.1.2   The Type Bit Maps Field  . . . . . . . . . . . . . . . 16
    4.1.3   Inclusion of Wildcard Names in NSEC RDATA  . . . . . . 17
  4.2   The NSEC RR Presentation Format  . . . . . . . . . . . . . 17
  4.3   NSEC RR Example  . . . . . . . . . . . . . . . . . . . . . 17
5.  The DS Resource Record . . . . . . . . . . . . . . . . . . . . 19
  5.1   DS RDATA Wire Format . . . . . . . . . . . . . . . . . . . 19
    5.1.1   The Key Tag Field  . . . . . . . . . . . . . . . . . . 20
    5.1.2   The Algorithm Field  . . . . . . . . . . . . . . . . . 20
    5.1.3   The Digest Type Field  . . . . . . . . . . . . . . . . 20
```

```
      5.1.4   The Digest Field . . . . . . . . . . . . . . . . . . . . 20
    5.2   Processing of DS RRs When Validating Responses . . . . . . 20
    5.3   The DS RR Presentation Format  . . . . . . . . . . . . . . 21
    5.4   DS RR Example  . . . . . . . . . . . . . . . . . . . . . . 21
  6.  Canonical Form and Order of Resource Records . . . . . . . . . 22
    6.1   Canonical DNS Name Order . . . . . . . . . . . . . . . . . 22
    6.2   Canonical RR Form  . . . . . . . . . . . . . . . . . . . . 22
    6.3   Canonical RR Ordering Within An RRset  . . . . . . . . . . 23
  7.  IANA Considerations  . . . . . . . . . . . . . . . . . . . . . 24
  8.  Security Considerations  . . . . . . . . . . . . . . . . . . . 25
  9.  Acknowledgments  . . . . . . . . . . . . . . . . . . . . . . . 26
  10.   References . . . . . . . . . . . . . . . . . . . . . . . . . 27
  10.1  Normative References . . . . . . . . . . . . . . . . . . . . 27
  10.2  Informative References . . . . . . . . . . . . . . . . . . . 28
      Authors' Addresses . . . . . . . . . . . . . . . . . . . . . . 28
  A.  DNSSEC Algorithm and Digest Types  . . . . . . . . . . . . . . 30
    A.1   DNSSEC Algorithm Types . . . . . . . . . . . . . . . . . . 30
      A.1.1   Private Algorithm Types  . . . . . . . . . . . . . . . 30
    A.2   DNSSEC Digest Types  . . . . . . . . . . . . . . . . . . . 31
  B.  Key Tag Calculation  . . . . . . . . . . . . . . . . . . . . . 32
    B.1   Key Tag for Algorithm 1 (RSA/MD5)  . . . . . . . . . . . . 33
      Intellectual Property and Copyright Statements . . . . . . . . 34
```

1. Introduction

The DNS Security Extensions (DNSSEC) introduce four new DNS resource record types: DNSKEY, RRSIG, NSEC, and DS. This document defines the purpose of each resource record (RR), the RR's RDATA format, and its presentation format (ASCII representation).

1.1 Background and Related Documents

The reader is assumed to be familiar with the basic DNS concepts described in [RFC1034], [RFC1035] and subsequent RFCs that update them: [RFC2136], [RFC2181] and [RFC2308].

This document is part of a family of documents that define the DNS security extensions. The DNS security extensions (DNSSEC) are a collection of resource records and DNS protocol modifications that add source authentication and data integrity to the Domain Name System (DNS). An introduction to DNSSEC and definitions of common terms can be found in [I-D.ietf-dnsext-dnssec-intro]. A description of DNS protocol modifications can be found in [I-D.ietf-dnsext-dnssec-protocol]. This document defines the DNSSEC resource records.

1.2 Reserved Words

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.3 Editors' Notes

1.3.1 Technical Changes or Corrections

Please report technical corrections to dnssec-editors@east.isi.edu. To assist the editors, please indicate the text in error and point out the RFC that defines the correct behavior. For a technical change where no RFC that defines the correct behavior, or if there's more than one applicable RFC and the definitions conflict, please post the issue to namedroppers.

An example correction to dnssec-editors might be: Page X says "DNSSEC RRs SHOULD be automatically returned in responses." This was true in RFC 2535, but RFC 3225 (Section 3, 3rd paragraph) says the DNSSEC RR types MUST NOT be included in responses unless the resolver indicated support for DNSSEC.

1.3.2 Typos and Minor Corrections

Please report any typos corrections to dnssec-editors@east.isi.edu. To assist the editors, please provide enough context for us to find the incorrect text quickly.

An example message to dnssec-editors might be: page X says "the DNSSEC standard has been in development for over 1 years". It should read "over 10 years".

2. The DNSKEY Resource Record

DNSSEC uses public key cryptography to sign and authenticate DNS resource record sets (RRsets). The public keys are stored in DNSKEY resource records and are used in the DNSSEC authentication process described in [I-D.ietf-dnsext-dnssec-protocol]: A zone signs its authoritative RRsets using a private key and stores the corresponding public key in a DNSKEY RR. A resolver can then use the public key to authenticate signatures covering the RRsets in the zone.

The DNSKEY RR is not intended as a record for storing arbitrary public keys and MUST NOT be used to store certificates or public keys that do not directly relate to the DNS infrastructure.

The Type value for the DNSKEY RR type is 48.

The DNSKEY RR is class independent.

The DNSKEY RR has no special TTL requirements.

2.1 DNSKEY RDATA Wire Format

The RDATA for a DNSKEY RR consists of a 2 octet Flags Field, a 1 octet Protocol Field, a 1 octet Algorithm Field, and the Public Key Field.

```
                     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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Flags            |    Protocol   |   Algorithm   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                                                               /
/                            Public Key                         /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.1.1 The Flags Field

Bit 7 of the Flags field is the Zone Key flag. If bit 7 has value 1, then the DNSKEY record holds a DNS zone key and the DNSKEY RR's owner name MUST be the name of a zone. If bit 7 has value 0, then the DNSKEY record holds some other type of DNS public key, such as a public key used by TKEY and MUST NOT be used to verify RRSIGs that cover RRsets.

Bit 15 of the Flags field is the Secure Entry Point flag, described in [RFC3757]. If bit 15 has value 1, then the DNSKEY record holds a

key intended for use as a secure entry point. This flag is only intended to be to a hint to zone signing or debugging software as to the intended use of this DNSKEY record; validators MUST NOT alter their behavior during the signature validation process in any way based on the setting of this bit. This also means a DNSKEY RR with the SEP bit set would also need the Zone Key flag set in order to legally be able to generate signatures. A DNSKEY RR with the SEP set and the Zone Key flag not set is an invalid DNSKEY.

Bits 0-6 and 8-14 are reserved: these bits MUST have value 0 upon creation of the DNSKEY RR, and MUST be ignored upon reception.

2.1.2 The Protocol Field

The Protocol Field MUST have value 3 and the DNSKEY RR MUST be treated as invalid during signature verification if found to be some value other than 3.

2.1.3 The Algorithm Field

The Algorithm field identifies the public key's cryptographic algorithm and determines the format of the Public Key field. A list of DNSSEC algorithm types can be found in Appendix A.1

2.1.4 The Public Key Field

The Public Key Field holds the public key material. The format depends on the algorithm of the key being stored and are described in separate documents.

2.1.5 Notes on DNSKEY RDATA Design

Although the Protocol Field always has value 3, it is retained for backward compatibility with early versions of the KEY record.

2.2 The DNSKEY RR Presentation Format

The presentation format of the RDATA portion is as follows:

The Flag field MUST be represented as an unsigned decimal integer with a value of 0, 256, or 257.

The Protocol Field MUST be represented as an unsigned decimal integer with a value of 3.

The Algorithm field MUST be represented either as an unsigned decimal integer or as an algorithm mnemonic as specified in Appendix A.1.

The Public Key field MUST be represented as a Base64 encoding of the Public Key. Whitespace is allowed within the Base64 text. For a definition of Base64 encoding, see [RFC1521] Section 5.2.

2.3 DNSKEY RR Example

The following DNSKEY RR stores a DNS zone key for example.com.

```
example.com. 86400 IN DNSKEY 256 3 5 ( AQPSKmynfzW4kyBv015MUG2DeIQ3
                                       Cbl+BBZH4b/0PY1kxkmvHjcZc8no
                                       kfzj31GajIQKY+5CptLr3buXA10h
                                       WqTkF7H6RfoRqXQeogmMHfpftf6z
                                       Mv1LyBUgia7za6ZEzOJBOztyvhjL
                                       742iU/TpPSEDhm2SNKLijfUppn1U
                                       aNvv4w==  )
```

The first four text fields specify the owner name, TTL, Class, and RR type (DNSKEY). Value 256 indicates that the Zone Key bit (bit 7) in the Flags field has value 1. Value 3 is the fixed Protocol value. Value 5 indicates the public key algorithm. Appendix A.1 identifies algorithm type 5 as RSA/SHA1 and indicates that the format of the RSA/SHA1 public key field is defined in [RFC3110]. The remaining text is a Base64 encoding of the public key.

3. The RRSIG Resource Record

DNSSEC uses public key cryptography to sign and authenticate DNS resource record sets (RRsets). Digital signatures are stored in RRSIG resource records and are used in the DNSSEC authentication process described in [I-D.ietf-dnsext-dnssec-protocol]. A validator can use these RRSIG RRs to authenticate RRsets from the zone. The RRSIG RR MUST only be used to carry verification material (digital signatures) used to secure DNS operations.

An RRSIG record contains the signature for an RRset with a particular name, class, and type. The RRSIG RR specifies a validity interval for the signature and uses the Algorithm, the Signer's Name, and the Key Tag to identify the DNSKEY RR containing the public key that a validator can use to verify the signature.

Because every authoritative RRset in a zone must be protected by a digital signature, RRSIG RRs must be present for names containing a CNAME RR. This is a change to the traditional DNS specification [RFC1034] that stated that if a CNAME is present for a name, it is the only type allowed at that name. A RRSIG and NSEC (see Section 4) MUST exist for the same name as a CNAME resource record in a signed zone.

The Type value for the RRSIG RR type is 46.

The RRSIG RR is class independent.

An RRSIG RR MUST have the same class as the RRset it covers.

The TTL value of an RRSIG RR SHOULD match the TTL value of the RRset it covers. This is an exception to the [RFC2181] rules for TTL values of individual RRs within a RRset: individual RRSIG with the same owner name will have different TTL values if the RRsets they cover have different TTL values.

3.1 RRSIG RDATA Wire Format

The RDATA for an RRSIG RR consists of a 2 octet Type Covered field, a 1 octet Algorithm field, a 1 octet Labels field, a 4 octet Original TTL field, a 4 octet Signature Expiration field, a 4 octet Signature Inception field, a 2 octet Key tag, the Signer's Name field, and the Signature field.

```
                        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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Type Covered           |  Algorithm    |     Labels    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         Original TTL                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Signature Expiration                     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Signature Inception                      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Key Tag            |                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+         Signer's Name         /
   /                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   /                                                               /
   /                            Signature                          /
   /                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.1.1 The Type Covered Field

The Type Covered field identifies the type of the RRset that is covered by this RRSIG record.

3.1.2 The Algorithm Number Field

The Algorithm Number field identifies the cryptographic algorithm used to create the signature. A list of DNSSEC algorithm types can be found in Appendix A.1

3.1.3 The Labels Field

The Labels field specifies the number of labels in the original RRSIG RR owner name. The significance of this field is that a validator uses it to determine if the answer was synthesized from a wildcard. If so, it can be used to determine what owner name was used in generating the signature.

To validate a signature, the validator needs the original owner name that was used to create the signature. If the original owner name contains a wildcard label ("*"), the owner name may have been expanded by the server during the response process, in which case the validator will need to reconstruct the original owner name in order to validate the signature. [I-D.ietf-dnsext-dnssec-protocol] describes how to use the Labels field to reconstruct the original owner name.

The value of the Labels field MUST NOT count either the null (root) label that terminates the owner name or the wildcard label (if present). The value of the Labels field MUST be less than or equal to the number of labels in the RRSIG owner name. For example, "www.example.com." has a Labels field value of 3, and "*.example.com." has a Labels field value of 2. Root (".") has a Labels field value of 0.

Although the wildcard label is not included in the count stored in the Labels field of the RRSIG RR, the wildcard label is part of the RRset's owner name when generating or verifying the signature.

3.1.4 Original TTL Field

The Original TTL field specifies the TTL of the covered RRset as it appears in the authoritative zone.

The Original TTL field is necessary because a caching resolver decrements the TTL value of a cached RRset. In order to validate a signature, a validator requires the original TTL. [I-D.ietf-dnsext-dnssec-protocol] describes how to use the Original TTL field value to reconstruct the original TTL.

3.1.5 Signature Expiration and Inception Fields

The Signature Expiration and Inception fields specify a validity period for the signature. The RRSIG record MUST NOT be used for authentication prior to the inception date and MUST NOT be used for authentication after the expiration date.

Signature Expiration and Inception field values are in POSIX.1 time format: a 32-bit unsigned number of seconds elapsed since 1 January 1970 00:00:00 UTC, ignoring leap seconds, in network byte order. The longest interval which can be expressed by this format without wrapping is approximately 136 years. An RRSIG RR can have an Expiration field value which is numerically smaller than the Inception field value if the expiration field value is near the 32-bit wrap-around point or if the signature is long lived. Because of this, all comparisons involving these fields MUST use "Serial number arithmetic" as defined in [RFC1982]. As a direct consequence, the values contained in these fields cannot refer to dates more than 68 years in either the past or the future.

3.1.6 The Key Tag Field

The Key Tag field contains the key tag value of the DNSKEY RR that validates this signature. Appendix B explains how to calculate Key Tag values.

3.1.7 The Signer's Name Field

The Signer's Name field value identifies the owner name of the DNSKEY RR which a validator should use to validate this signature. The Signer's Name field MUST contain the name of the zone of the covered RRset. A sender MUST NOT use DNS name compression on the Signer's Name field when transmitting a RRSIG RR.

3.1.8 The Signature Field

The Signature field contains the cryptographic signature that covers the RRSIG RDATA (excluding the Signature field) and the RRset specified by the RRSIG owner name, RRSIG class, and RRSIG Type Covered field. The format of this field depends on the algorithm in use and these formats are described in separate companion documents.

3.1.8.1 Signature Calculation

A signature covers the RRSIG RDATA (excluding the Signature Field) and covers the data RRset specified by the RRSIG owner name, RRSIG class, and RRSIG Type Covered fields. The RRset is in canonical form (see Section 6) and the set RR(1),...RR(n) is signed as follows:

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

         "|" denotes concatenation;

         RRSIG_RDATA is the wire format of the RRSIG RDATA fields
            with the Signer's Name field in canonical form and
            the Signature field excluded;

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

            "owner" is the fully qualified owner name of the RRset in
            canonical form (for RRs with wildcard owner names, the
            wildcard label is included in the owner name);

            Each RR MUST have the same owner name as the RRSIG RR;

            Each RR MUST have the same class as the RRSIG RR;

            Each RR in the RRset MUST have the RR type listed in the
            RRSIG RR's Type Covered field;

            Each RR in the RRset MUST have the TTL listed in the
            RRSIG Original TTL Field;

            Any DNS names in the RDATA field of each RR MUST be in
```

canonical form; and

The RRset MUST be sorted in canonical order.

See Section 6.1 and Section 6.2 for details on canonical name order and canonical RR form.

3.2 The RRSIG RR Presentation Format

The presentation format of the RDATA portion is as follows:

The Type Covered field is represented as a RR type mnemonic. When the mnemonic is not known, the TYPE representation as described in [RFC3597] (section 5) MUST be used.

The Algorithm field value MUST be represented either as an unsigned decimal integer or as an algorithm mnemonic as specified in Appendix A.1.

The Labels field value MUST be represented as an unsigned decimal integer.

The Original TTL field value MUST be represented as an unsigned decimal integer.

The Signature Expiration Time and Inception Time field values MUST be represented either as seconds since 1 January 1970 00:00:00 UTC or in the form YYYYMMDDHHmmSS in UTC, where:
   YYYY is the year (0000-9999, but see Section 3.1.5);
   MM is the month number (01-12);
   DD is the day of the month (01-31);
   HH is the hour in 24 hours notation (00-23);
   mm is the minute (00-59); and
   SS is the second (00-59).

The Key Tag field MUST be represented as an unsigned decimal integer.

The Signer's Name field value MUST be represented as a domain name.

The Signature field is represented as a Base64 encoding of the signature. Whitespace is allowed within the Base64 text. See Section 2.2.

3.3 RRSIG RR Example

The following RRSIG RR stores the signature for the A RRset of 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= )
```

The first four fields specify the owner name, TTL, Class, and RR type (RRSIG). The "A" represents the Type Covered field. The value 5 identifies the algorithm used (RSA/SHA1) to create the signature. The value 3 is the number of Labels in the original owner name. The value 86400 in the RRSIG RDATA is the Original TTL for the covered A RRset. 20030322173103 and 20030220173103 are the expiration and inception dates, respectively. 2642 is the Key Tag, and example.com. is the Signer's Name. The remaining text is a Base64 encoding of the signature.

Note that combination of RRSIG RR owner name, class, and Type Covered indicate that this RRSIG covers the "host.example.com" A RRset. The Label value of 3 indicates that no wildcard expansion was used. The Algorithm, Signer's Name, and Key Tag indicate this signature can be authenticated using an example.com zone DNSKEY RR whose algorithm is 5 and key tag is 2642.

4. The NSEC Resource Record

The NSEC resource record lists two separate things: the owner name of the next authoritative RRset in the canonical ordering of the zone, and the set of RR types present at the NSEC RR's owner name. The complete set of NSEC RRs in a zone both indicate which authoritative RRsets exist in a zone and also form a chain of authoritative owner names in the zone. This information is used to provide authenticated denial of existence for DNS data, as described in [I-D.ietf-dnsext-dnssec-protocol].

Because every authoritative name in a zone must be part of the NSEC chain, NSEC RRs must be present for names containing a CNAME RR. This is a change to the traditional DNS specification [RFC1034] that stated that if a CNAME is present for a name, it is the only type allowed at that name. An RRSIG (see Section 3) and NSEC MUST exist for the same name as a CNAME resource record in a signed zone.

See [I-D.ietf-dnsext-dnssec-protocol] for discussion of how a zone signer determines precisely which NSEC RRs it needs to include in a zone.

The type value for the NSEC RR is 47.

The NSEC RR is class independent.

The NSEC RR SHOULD have the same TTL value as the SOA minimum TTL field. This is in the spirit of negative caching [RFC2308].

4.1 NSEC RDATA Wire Format

The RDATA of the NSEC RR is as shown below:

```
                     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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                      Next Domain Name                         /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                       Type Bit Maps                           /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

4.1.1 The Next Domain Name Field

The Next Domain Name field contains the owner name of the next authoritative owner name in the canonical ordering of the zone; see Section 6.1 for an explanation of canonical ordering. The value of the Next Domain Name field in the last NSEC record in the zone is the

name of the zone apex (the owner name of the zone's SOA RR).

A sender MUST NOT use DNS name compression on the Next Domain Name field when transmitting an NSEC RR.

Owner names of RRsets not authoritative for the given zone (such as glue records) MUST NOT be listed in the Next Domain Name unless at least one authoritative RRset exists at the same owner name.

4.1.2 The Type Bit Maps Field

The Type Bit Maps field identifies the RRset types which exist at the NSEC RR's owner name.

The RR type space is split into 256 window blocks, each representing the low-order 8 bits of the 16-bit RR type space. Each block that has at least one active RR type is encoded using a single octet window number (from 0 to 255), a single octet bitmap length (from 1 to 32) indicating the number of octets used for the window block's bitmap, and up to 32 octets (256 bits) of bitmap.

Blocks are present in the NSEC RR RDATA in increasing numerical order.

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

where "|" denotes concatenation.
```

Each bitmap encodes the low-order 8 bits of RR types within the window block, in network bit order. The first bit is bit 0. For window block 0, bit 1 corresponds to RR type 1 (A), bit 2 corresponds to RR type 2 (NS), and so forth. For window block 1, bit 1 corresponds to RR type 257, bit 2 to RR type 258. If a bit is set to 1, it indicates that an RRset of that type is present for the NSEC RR's owner name. If a bit is set to 0, it indicates that no RRset of that type is present for the NSEC RR's owner name.

Bits representing pseudo-types MUST be set to 0, since they do not appear in zone data. If encountered, they MUST be ignored upon reading.

Blocks with no types present MUST NOT be included. Trailing zero octets in the bitmap MUST be omitted. The length of each block's bitmap is determined by the type code with the largest numerical value, within that block, among the set of RR types present at the NSEC RR's owner name. Trailing zero octets not specified MUST be interpreted as zero octets.

The bitmap for the NSEC RR at a delegation point requires special attention. Bits corresponding to the delegation NS RRset and the RR types for which the parent zone has authoritative data MUST be set to 1; bits corresponding to any non-NS RRset for which the parent is not authoritative MUST be set to 0.

A zone MUST NOT include an NSEC RR for any domain name that only holds glue records.

4.1.3 Inclusion of Wildcard Names in NSEC RDATA

If a wildcard owner name appears in a zone, the wildcard label ("*") is treated as a literal symbol and is treated the same as any other owner name for purposes of generating NSEC RRs. Wildcard owner names appear in the Next Domain Name field without any wildcard expansion. [I-D.ietf-dnsext-dnssec-protocol] describes the impact of wildcards on authenticated denial of existence.

4.2 The NSEC RR Presentation Format

The presentation format of the RDATA portion is as follows:

The Next Domain Name field is represented as a domain name.

The Type Bit Maps field is represented as a sequence of RR type mnemonics. When the mnemonic is not known, the TYPE representation as described in [RFC3597] (section 5) MUST be used.

4.3 NSEC RR Example

The following NSEC RR identifies the RRsets associated with alfa.example.com. and identifies the next authoritative name after alfa.example.com.

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

The first four text fields specify the name, TTL, Class, and RR type (NSEC). The entry host.example.com. is the next authoritative name after alfa.example.com. in canonical order. The A, MX, RRSIG, NSEC, and TYPE1234 mnemonics indicate there are A, MX, RRSIG, NSEC, and TYPE1234 RRsets associated with the name alfa.example.com.

The RDATA section of the NSEC RR above would be encoded as:

```
            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
```

Assuming that the validator can authenticate this NSEC record, it could be used to prove that beta.example.com does not exist, or could be used to prove there is no AAAA record associated with alfa.example.com. Authenticated denial of existence is discussed in [I-D.ietf-dnsext-dnssec-protocol].

5. The DS Resource Record

The DS Resource Record refers to a DNSKEY RR and is used in the DNS DNSKEY authentication process. A DS RR refers to a DNSKEY RR by storing the key tag, algorithm number, and a digest of the DNSKEY RR. Note that while the digest should be sufficient to identify the public key, storing the key tag and key algorithm helps make the identification process more efficient. By authenticating the DS record, a resolver can authenticate the DNSKEY RR to which the DS record points. The key authentication process is described in [I-D.ietf-dnsext-dnssec-protocol].

The DS RR and its corresponding DNSKEY RR have the same owner name, but they are stored in different locations. The DS RR appears only on the upper (parental) side of a delegation, and is authoritative data in the parent zone. For example, the DS RR for "example.com" is stored in the "com" zone (the parent zone) rather than in the "example.com" zone (the child zone). The corresponding DNSKEY RR is stored in the "example.com" zone (the child zone). This simplifies DNS zone management and zone signing, but introduces special response processing requirements for the DS RR; these are described in [I-D.ietf-dnsext-dnssec-protocol].

The type number for the DS record is 43.

The DS resource record is class independent.

The DS RR has no special TTL requirements.

5.1 DS RDATA Wire Format

The RDATA for a DS RR consists of a 2 octet Key Tag field, a one octet Algorithm field, a one octet Digest Type field, and a Digest field.

```
                     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                             /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

5.1.1 The Key Tag Field

The Key Tag field lists the key tag of the DNSKEY RR referred to by the DS record.

The Key Tag used by the DS RR is identical to the Key Tag used by RRSIG RRs. Appendix B describes how to compute a Key Tag.

5.1.2 The Algorithm Field

The Algorithm field lists the algorithm number of the DNSKEY RR referred to by the DS record.

The algorithm number used by the DS RR is identical to the algorithm number used by RRSIG and DNSKEY RRs. Appendix A.1 lists the algorithm number types.

5.1.3 The Digest Type Field

The DS RR refers to a DNSKEY RR by including a digest of that DNSKEY RR. The Digest Type field identifies the algorithm used to construct the digest. Appendix A.2 lists the possible digest algorithm types.

5.1.4 The Digest Field

The DS record refers to a DNSKEY RR by including a digest of that DNSKEY RR.

The digest is calculated by concatenating the canonical form of the fully qualified owner name of the DNSKEY RR with the DNSKEY RDATA, and then applying the digest algorithm.

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

   "|" denotes concatenation

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

The size of the digest may vary depending on the digest algorithm and DNSKEY RR size. As of the time of writing, the only defined digest algorithm is SHA-1, which produces a 20 octet digest.

5.2 Processing of DS RRs When Validating Responses

The DS RR links the authentication chain across zone boundaries, so the DS RR requires extra care in processing. The DNSKEY RR referred to in the DS RR MUST be a DNSSEC zone key. The DNSKEY RR Flags MUST

have Flags bit 7 set to value 1. If the DNSKEY flags do not indicate a DNSSEC zone key, the DS RR (and DNSKEY RR it references) MUST NOT be used in the validation process.

5.3 The DS RR Presentation Format

The presentation format of the RDATA portion is as follows:

The Key Tag field MUST be represented as an unsigned decimal integer.

The Algorithm field MUST be represented either as an unsigned decimal integer or as an algorithm mnemonic specified in Appendix A.1.

The Digest Type field MUST be represented as an unsigned decimal integer.

The Digest MUST be represented as a sequence of case-insensitive hexadecimal digits. Whitespace is allowed within the hexadecimal text.

5.4 DS RR Example

The following example shows a DNSKEY RR and its corresponding DS RR.

```
dskey.example.com. 86400 IN DNSKEY 256 3 5 ( AQOeiiR0GOMYkDshWoSKz9Xz
                                          fwJr1AYtsmx3TGkJaNXVbfi/
                                          2pHm822aJ5iI9BMzNXxeYCmZ
                                          DRD99WYwYqUSdjMmmAphXdvx
                                          egXd/M5+X7OrzKBaMbCVdFLU
                                          Uh6DhweJBjEVv5f2wwjM9Xzc
                                          nOf+EPbtG9DMBmADjFDc2w/r
                                          ljwvFw==
                                          ) ;  key id = 60485

dskey.example.com. 86400 IN DS 60485 5 1 ( 2BB183AF5F22588179A53B0A
                                           98631FAD1A292118 )
```

The first four text fields specify the name, TTL, Class, and RR type (DS). Value 60485 is the key tag for the corresponding "dskey.example.com." DNSKEY RR, and value 5 denotes the algorithm used by this "dskey.example.com." DNSKEY RR. The value 1 is the algorithm used to construct the digest, and the rest of the RDATA text is the digest in hexadecimal.

6. Canonical Form and Order of Resource Records

This section defines a canonical form for resource records, a canonical ordering of DNS names, and a canonical ordering of resource records within an RRset. A canonical name order is required to construct the NSEC name chain. A canonical RR form and ordering within an RRset are required to construct and verify RRSIG RRs.

6.1 Canonical DNS Name Order

For purposes of DNS security, owner names are ordered by treating individual labels as unsigned left-justified octet strings. The absence of a octet sorts before a zero value octet, and upper case US-ASCII letters are treated as if they were lower case US-ASCII letters.

To compute the canonical ordering of a set of DNS names, start by sorting the names according to their most significant (rightmost) labels. For names in which the most significant label is identical, continue sorting according to their next most significant label, and so forth.

For example, the following names are sorted in canonical DNS name order. The most significant label is "example". At this level, "example" sorts first, followed by names ending in "a.example", then names ending "z.example". The names within each level are sorted in the same way.

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

6.2 Canonical RR Form

For purposes of DNS security, the canonical form of an RR is the wire format of the RR where:

1. Every domain name in the RR is fully expanded (no DNS name compression) and fully qualified;
2. All uppercase US-ASCII letters in the owner name of the RR are replaced by the corresponding lowercase US-ASCII letters;

3. If the type of the RR is 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 or NSEC, all uppercase US-ASCII letters in the DNS names contained within the RDATA are replaced by the corresponding lowercase US-ASCII letters;
4. If the owner name of the RR is a wildcard name, the owner name is in its original unexpanded form, including the "*" label (no wildcard substitution); and
5. The RR's TTL is set to its original value as it appears in the originating authoritative zone or the Original TTL field of the covering RRSIG RR.

6.3 Canonical RR Ordering Within An RRset

For purposes of DNS security, RRs with the same owner name, class, and type are sorted by treating the RDATA portion of the canonical form of each RR as a left-justified unsigned octet sequence where the absence of an octet sorts before a zero octet.

[RFC2181] specifies that an RRset is not allowed to contain duplicate records (multiple RRs with the same owner name, class, type, and RDATA). Therefore, if an implementation detects duplicate RRs when putting the RRset in canonical form, the implementation MUST treat this as a protocol error. If the implementation chooses to handle this protocol error in the spirit of the robustness principle (being liberal in what it accepts), the implementation MUST remove all but one of the duplicate RR(s) for purposes of calculating the canonical form of the RRset.

7. IANA Considerations

This document introduces no new IANA considerations, because all of the protocol parameters used in this document have already been assigned by previous specifications. However, since the evolution of DNSSEC has been long and somewhat convoluted, this section attempts to describe the current state of the IANA registries and other protocol parameters which are (or once were) related to DNSSEC.

Please refer to [I-D.ietf-dnsext-dnssec-protocol] for additional IANA considerations.

DNS Resource Record Types: [RFC2535] assigned types 24, 25, and 30 to the SIG, KEY, and NXT RRs, respectively. [RFC3658] assigned DNS Resource Record Type 43 to DS. [RFC3755] assigned types 46, 47, and 48 to the RRSIG, NSEC, and DNSKEY RRs, respectively. [RFC3755] also marked type 30 (NXT) as Obsolete, and restricted use of types 24 (SIG) and 25 (KEY) to the "SIG(0)" transaction security protocol described in [RFC2931] and the transaction KEY Resource Record described in [RFC2930].

DNS Security Algorithm Numbers: [RFC2535] created an IANA registry for DNSSEC Resource Record Algorithm field numbers, and assigned values 1-4 and 252-255. [RFC3110] assigned value 5. [RFC3755] altered this registry to include flags for each entry regarding its use with the DNS security extensions. Each algorithm entry could refer to an algorithm that can be used for zone signing, transaction security (see [RFC2931]) or both. Values 6-251 are available for assignment by IETF standards action. See Appendix A for a full listing of the DNS Security Algorithm Numbers entries at the time of writing and their status of use in DNSSEC.

[RFC3658] created an IANA registry for DNSSEC DS Digest Types, and assigned value 0 to reserved and value 1 to SHA-1.

KEY Protocol Values: [RFC2535] created an IANA Registry for KEY Protocol Values, but [RFC3445] re-assigned all values other than 3 to reserved and closed this IANA registry. The registry remains closed, and all KEY and DNSKEY records are required to have Protocol Octet value of 3.

Flag bits in the KEY and DNSKEY RRs: [RFC3755] created an IANA registry for the DNSSEC KEY and DNSKEY RR flag bits. Initially, this registry only contains an assignment for bit 7 (the ZONE bit) and a reservation for bit 15 for the Secure Entry Point flag (SEP bit) [RFC3757]. Bits 0-6 and 8-14 are available for assignment by IETF Standards Action.

8. Security Considerations

This document describes the format of four DNS resource records used by the DNS security extensions, and presents an algorithm for calculating a key tag for a public key. Other than the items described below, the resource records themselves introduce no security considerations. Please see [I-D.ietf-dnsext-dnssec-intro] and [I-D.ietf-dnsext-dnssec-protocol] for additional security considerations related to the use of these records.

The DS record points to a DNSKEY RR using a cryptographic digest, the key algorithm type and a key tag. The DS record is intended to identify an existing DNSKEY RR, but it is theoretically possible for an attacker to generate a DNSKEY that matches all the DS fields. The probability of constructing such a matching DNSKEY depends on the type of digest algorithm in use. The only currently defined digest algorithm is SHA-1, and the working group believes that constructing a public key which would match the algorithm, key tag, and SHA-1 digest given in a DS record would be a sufficiently difficult problem that such an attack is not a serious threat at this time.

The key tag is used to help select DNSKEY resource records efficiently, but it does not uniquely identify a single DNSKEY resource record. It is possible for two distinct DNSKEY RRs to have the same owner name, the same algorithm type, and the same key tag. An implementation which uses only the key tag to select a DNSKEY RR might select the wrong public key in some circumstances.

9. Acknowledgments

This document was created from the input and ideas of the members of the DNS Extensions Working Group and working group mailing list. The editors would like to express their thanks for the comments and suggestions received during the revision of these security extension specifications. While explicitly listing everyone who has contributed during the decade during which DNSSEC has been under development would be an impossible task, [I-D.ietf-dnsext-dnssec-intro] includes a list of some of the participants who were kind enough to comment on these documents.

10. References

10.1 Normative References

[I-D.ietf-dnsext-dnssec-intro]
Arends, R., Austein, R., Larson, M., Massey, D. and S. Rose, "DNS Security Introduction and Requirements", draft-ietf-dnsext-dnssec-intro-10 (work in progress), May 2004.

[I-D.ietf-dnsext-dnssec-protocol]
Arends, R., Austein, R., Larson, M., Massey, D. and S. Rose, "Protocol Modifications for the DNS Security Extensions", draft-ietf-dnsext-dnssec-protocol-06 (work in progress), May 2004.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC1521] Borenstein, N. and N. Freed, "MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1521, September 1993.

[RFC1982] Elz, R. and R. Bush, "Serial Number Arithmetic", RFC 1982, August 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2136] Vixie, P., Thomson, S., Rekhter, Y. and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.

[RFC2308] Andrews, M., "Negative Caching of DNS Queries (DNS NCACHE)", RFC 2308, March 1998.

[RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, August 1999.

[RFC2931] Eastlake, D., "DNS Request and Transaction Signatures (SIG(0)s)", RFC 2931, September 2000.

[RFC3110] Eastlake, D., "RSA/SHA-1 SIGs and RSA KEYs in the Domain Name System (DNS)", RFC 3110, May 2001.

[RFC3445] Massey, D. and S. Rose, "Limiting the Scope of the KEY Resource Record (RR)", RFC 3445, December 2002.

[RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record (RR) Types", RFC 3597, September 2003.

[RFC3658] Gudmundsson, O., "Delegation Signer (DS) Resource Record (RR)", RFC 3658, December 2003.

[RFC3755] Weiler, S., "Legacy Resolver Compatibility for Delegation Signer", RFC 3755, April 2004.

[RFC3757] Kolkman, O., Schlyter, J. and E. Lewis, "KEY RR Secure Entry Point Flag", RFC 3757, April 2004.

10.2 Informative References

[I-D.ietf-dnsext-nsec-rdata] Schlyter, J., "KEY RR Secure Entry Point Flag", draft-ietf-dnsext-nsec-rdata-05 (work in progress), March 2004.

[RFC2535] Eastlake, D., "Domain Name System Security Extensions", RFC 2535, March 1999.

[RFC2930] Eastlake, D., "Secret Key Establishment for DNS (TKEY RR)", RFC 2930, September 2000.

Authors' Addresses

Roy Arends
Telematica Instituut
Drienerlolaan 5
7522 NB Enschede
NL

EMail: roy.arends@telin.nl

Rob Austein
Internet Systems Consortium
950 Charter Street
Redwood City, CA 94063
USA

EMail: sra@isc.org

Matt Larson
VeriSign, Inc.
21345 Ridgetop Circle
Dulles, VA 20166-6503
USA

EMail: mlarson@verisign.com

Dan Massey
USC Information Sciences Institute
3811 N. Fairfax Drive
Arlington, VA 22203
USA

EMail: masseyd@isi.edu

Scott Rose
National Institute for Standards and Technology
100 Bureau Drive
Gaithersburg, MD 20899-8920
USA

EMail: scott.rose@nist.gov

Appendix A. DNSSEC Algorithm and Digest Types

The DNS security extensions are designed to be independent of the underlying cryptographic algorithms. The DNSKEY, RRSIG, and DS resource records all use a DNSSEC Algorithm Number to identify the cryptographic algorithm in use by the resource record.   The DS resource record also specifies a Digest Algorithm Number to identify the digest algorithm used to construct the DS record. The currently defined Algorithm and Digest Types are listed below. Additional Algorithm or Digest Types could be added as advances in cryptography warrant.

A DNSSEC aware resolver or name server MUST implement all MANDATORY algorithms.

A.1 DNSSEC Algorithm Types

The DNSKEY, RRSIG, and DS RRs use an 8-bit number used to identify the security algorithm being used. These values are stored in the "Algorithm number" field in the resource record RDATA.

Some algorithms are usable only for zone signing (DNSSEC), some only for transaction security mechanisms (SIG(0) and TSIG), and some for both. Those usable for zone signing may appear in DNSKEY, RRSIG, and DS RRs. Those usable for transaction security would be present in SIG(0) and KEY RRs as described in [RFC2931]

Value	Algorithm [Mnemonic]	Zone Signing	References	Status
0	reserved			
1	RSA/MD5 [RSAMD5]	n	RFC 2537	NOT RECOMMENDED
2	Diffie-Hellman [DH]	n	RFC 2539	-
3	DSA/SHA-1 [DSA]	y	RFC 2536	OPTIONAL
4	Elliptic Curve [ECC]		TBA	-
5	RSA/SHA-1 [RSASHA1]	y	RFC 3110	MANDATORY
252	Indirect [INDIRECT]	n		-
253	Private [PRIVATEDNS]	y	see below	OPTIONAL
254	Private [PRIVATEOID]	y	see below	OPTIONAL
255	reserved			

6 - 251 Available for assignment by IETF Standards Action.

A.1.1 Private Algorithm Types

Algorithm number 253 is reserved for private use and will never be assigned to a specific algorithm. The public key area in the DNSKEY RR and the signature area in the RRSIG RR begin with a wire encoded

domain name, which MUST NOT be compressed. The domain name indicates the private algorithm to use and the remainder of the public key area is determined by that algorithm. Entities should only use domain names they control to designate their private algorithms.

Algorithm number 254 is reserved for private use and will never be assigned to a specific algorithm. The public key area in the DNSKEY RR and the signature area in the RRSIG RR begin with an unsigned length byte followed by a BER encoded Object Identifier (ISO OID) of that length. The OID indicates the private algorithm in use and the remainder of the area is whatever is required by that algorithm. Entities should only use OIDs they control to designate their private algorithms.

A.2 DNSSEC Digest Types

A "Digest Type" field in the DS resource record types identifies the cryptographic digest algorithm used by the resource record.   The following table lists the currently defined digest algorithm types.

VALUE	Algorithm	STATUS
0	Reserved	-
1	SHA-1	MANDATORY
2-255	Unassigned	-

Appendix B. Key Tag Calculation

The Key Tag field in the RRSIG and DS resource record types provides a mechanism for selecting a public key efficiently. In most cases, a combination of owner name, algorithm, and key tag can efficiently identify a DNSKEY record. Both the RRSIG and DS resource records have corresponding DNSKEY records. The Key Tag field in the RRSIG and DS records can be used to help select the corresponding DNSKEY RR efficiently when more than one candidate DNSKEY RR is available.

However, it is essential to note that the key tag is not a unique identifier. It is theoretically possible for two distinct DNSKEY RRs to have the same owner name, the same algorithm, and the same key tag. The key tag is used to limit the possible candidate keys, but it does not uniquely identify a DNSKEY record. Implementations MUST NOT assume that the key tag uniquely identifies a DNSKEY RR.

The key tag is the same for all DNSKEY algorithm types except algorithm 1 (please see Appendix B.1 for the definition of the key tag for algorithm 1). The key tag algorithm is the sum of the wire format of the DNSKEY RDATA broken into 2 octet groups. First the RDATA (in wire format) is treated as a series of 2 octet groups, these groups are then added together ignoring any carry bits.

A reference implementation of the key tag algorithm is as an ANSI C function is given below with the RDATA portion of the DNSKEY RR is used as input. It is not necessary to use the following reference code verbatim, but the numerical value of the Key Tag MUST be identical to what the reference implementation would generate for the same input.

Please note that the algorithm for calculating the Key Tag is almost but not completely identical to the familiar ones complement checksum used in many other Internet protocols. Key Tags MUST be calculated using the algorithm described here rather than the ones complement checksum.

The following ANSI C reference implementation calculates the value of a Key Tag. This reference implementation applies to all algorithm types except algorithm 1 (see Appendix B.1). The input is the wire format of the RDATA portion of the DNSKEY RR. The code is written for clarity, not efficiency.

```
   /*
    * Assumes that int is at least 16 bits.
    * First octet of the key tag is the most significant 8 bits of the
    * return value;
    * Second octet of the key tag is the least significant 8 bits of the
    * return value.
    */

   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;
   }
```

B.1 Key Tag for Algorithm 1 (RSA/MD5)

The key tag for algorithm 1 (RSA/MD5) is defined differently than the key tag for all other algorithms, for historical reasons. For a DNSKEY RR with algorithm 1, the key tag is defined to be the most significant 16 bits of the least significant 24 bits in the public key modulus (in other words, the 4th to last and 3rd to last octets of the public key modulus).

Please note that Algorithm 1 is NOT RECOMMENDED.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.