CONNOTECH Experts-conseils inc.

'Trust Anchor' Integrity Mechanisms for Internet DNS Security

Thierry Moreau

Document Number C003496

2005/10/17

Table of contents

Document Revision History

C-Number	Date	Explanation
C003471	2005/08/24	Initial release
C003496	2005/10/17	1) Minor change in ASN.1 syntax of section 2.5.4 2) Exchanged meaning of SDDA RR Authentication Mechanism Type values 1 and 2
C003496		Current version

1. Introduction

This document presents a novel security scheme for facilitating the deployment for Internet DNS security protocols in a specific aspect, i.e. trust anchor key renewal operation. The following goals are pursued:

- to *explain* the proposed scheme in the DNS context to an audience already familiar with public key cryptography concepts: the generic TAKREM solution is covered by reference [1] which is a required reading for a complete explanation;
- to *state arguments* for the proposed scheme in light of achievable security levels in realistic operational settings;
- to *specify* additions to the DNS standard documents for to accommodate DNSKEY record authentication, supporting diverse trust anchor key management schemes, including the TAKREM use for DNS security presented here.

Nowadays, digital signatures based on public key cryptography are in common use within schemes such as the X.509 PKI (RFC2459, [2]) and the PGP secure e-mail utility ([3]). In either scheme, digital signatures are chained so that the assurance about an unknown signature is traced to a higher assurance signature. Inescapably, the chain terminates with a last digital signature, using a *root* public key, also called a *trust anchor key*. For instance in the case of an Internet browser, the root public keys are the “subject's public key” in the X.509 security certificates of certification authorities, as can be viewed in the browser's preferences configuration. The present paper addresses trust anchor key distribution issues.

Trust anchor key management lies at a mating point between technology-intensive cryptographic security and labor-intensive security procedures (e.g. manual establishment of a trusted channel). Accordingly, the present document covers both cryptographic aspects and procedural aspects. We became involved with security schemes at the mating point between cryptography and procedures with the SAKEM project ([4]), dealing with the "out-of-band" distribution of symmetric secret key without any prior shared security state. The SAKEM procedure requires a root public key similar to a high level DNS zone trust anchor key. We first thought the TAKREM solution for the SAKEM deployment, and then we realized its applicability to the DNS security deployment. In this area of IT security, we are looking for detailed schemes where cryptographic procedures mate well with procedural steps, in a way that keeps the operational hindrance to a reasonable level.

2. Concepts

2.1 The DNS Use of Digital Signatures

The latest Internet DNS security protocols, DNSSEC ([5], [6], [7]), adapt the digital signature technology in order to provide secure DNS data to name service resolver programs. These resolvers are ultimately responsible for decisions based on DNS data retrieved from an insecure open network (the digital signatures are intrinsically secure, providing end-to-end integrity assurance, both at the individual signature level and an the signature chain level). In other words, the ultimate integrity protection lies with the resolvers as verifiers of digital signature, even if it is the servers incur the burden of publishing the required digital signatures and trust anchor keys. Indeed, the foremost target of trust anchor key distribution is the myriad of resolver programs, just like the target of root X.509 security certificate distribution is the myriad of browsers.

The DNSSEC chains of digital signatures are structured along the DNS tree structured name space. This represents a simplifying assumption within the generalized trust network allowed in the X.509 certification world ([8]). However, the X.509 PKI experience suggests that a single top-level certification authority is hardly conceivable in practice. Likewise, in the DNS hierarchy, a secure root zone is also hardly realistic due to "political" factors, perhaps a consequence of the public relation implications of a single point of failure for overall DNS security. It is an outstanding issue whether the DNSSEC scheme can be deployed with a single root zone public key managed as implied by the DNS security theory of operation.

2.2 Our Perspective on DNSSEC Hierarchical Signature Delegation

For our purpose, it is useful to describe the DNSSEC hierarchical signature delegation scheme as a tree structure with four types of nodes:

- the root DNS zone,
- important DNS zones, i.e. zones controlled by organizations having sufficient interest in DNSSEC for to commit resources to manage the zone key with high operational security, including a wish that their zone KSK is distributed with high assurance,
- ordinary DNS zones, e.g. a zone controlled by managers who just need to keep the zone up and running,
- leaf nodes.

Perhaps the root zone management challenges are stemming from the combined expectations from the important DNS zone managers, who should realize that their zone assurance is critically dependent on higher zones, notably the root. Examples of important DNS zones would be governments and multinational corporations.

Our focus is on the establishment and renewal of trust anchors. For the manager of an important DNS zone, there are two approaches to providing secured DNS data: reliance on signatures and trust anchor distribution by higher zones, or direct distribution of trust anchor keys to resolver programs. We address the later approach.

The notion of "trust points" is different from "important zones". A trust point is any DNS zone that is secured but has an insecure parent. Direct trust anchor key distribution is a requirement for every trust point.

2.3 Trust Anchor Distribution Revisited

Trust anchor key distribution is about sending a digital message from an organization to an end-user, preventing undetected modifications, making creative use of integrity mechanisms other than digital signatures. It remains a end-user system responsibility to store the properly distributed trust anchors differently from ordinary system data so that malicious system configuration is prevented.

We find it convenient to isolate the initial trust anchor key distribution from trust anchor key renewal. Key renewal is motivated by cryptographic considerations (i.e. the fear that an old key might have been cracked with extremely compute-intensive attacks) and operational security aspects (e.g. over time, the likelihood of insider deceit is increasing). Trust anchor key renewal is not motivated by the end-user responsibility in maintaining the configuration integrity.

2.3.1 Initial Trust Anchor Distribution

A obvious first scheme for trust anchor distribution is embedding the trust anchor key in a system or software, at system assembly time or software package build time. The integrity of trust anchor key distribution becomes linked to the physical distribution of computer hardware, or to the software distribution mechanism. With the fast obsolescence rate of computer hardware and software, this trust anchor distribution strategy is perhaps covering a significant portion of the needs. In a circular argument reasoning, the need to use trust anchor keys that were issued after a system installation time may contribute to the perceived obsolescence of a given system or software.

Outside of system or software package installation events, the initial distribution of a trust anchor key requires either an out-of-band integrity check or a “leap of faith” copy operation. In practice, very few types of organizations can commit resources to the operational discipline of out-of-band integrity checks. This leaves the “leap of faith” copy operation as the default distribution scheme. For initial trust anchor distribution, this is not as bad as a security-minded reader might think: the first few uses of a trust anchor key, if successful from a end-user perspective, gives a relatively good indication that the trust anchor configuration was right. This is especially so because a) the end-user attention to system operation details is higher with a new kind of service (the end-user is likely to notice an operational glitch triggered by a fraud attempt), and b) a critical use of a system is unlikely to occur at this time. We are not aware of a report of this legitimacy for "leap of faith" public key distribution in the information security literature. Yet the prevailing practice and the fraud experience reports points to this direction.

In summary, initial trust anchor key distribution typically occurs with the installation of a system or software package, or with a "leap of faith" copy operation, which is tolerable in practice, mainly due to the lack of alternatives. Turning to trust anchor key renewal, the picture is somewhat different.

2.3.2 Trust Anchor Renewal

The rationales for scheduled renewals of trust anchor keys ("cryptoperiod") are debatable. If feasible without sacrificing security, trust anchor key renewals would be needed for three reasons:

- an old key might be cracked,
- over time, the uninterrupted confidentiality a production private key becomes less trustworthy,
- a scheduled renewal operation is like a rehearsal for an emergency private key recovery, or an alternative for it (e.g. if the private key breach has no practical impact or it is merely suspected).

For medium and large organizations, public relations plays a significant role in information security. Public image is supported by extremely simple messages, and the subtleties of key management are outright banned from the public relation language. Overall, the public relations factors works against *any* visible sign of trust anchor key management, and thus weights heavily in favor of a very long trust anchor cryptoperiod. In any event, the essence of the data integrity function supports long lasting trust anchors: integrity is related to reliability lasting over time.

Thus, we can assume that a "leap of faith" distribution strategy is not acceptable for trust anchor renewal. The out-of-band integrity check is organizationally even more challenging than in the initial distribution case (it would require a recurrent commitment of labor-intensive resource with little short-term rewards). So the long lasting trust anchor key wins by default. The X.509 experience supports the above understanding of long lasting trust anchor. Mass market web browsers are shipped with pre-configured root CA certificates, most of which having public key validity periods over 10 years.

We are not aware of recommendations for alternate trust anchor management in academic or specialized publications. This is an area where of technological advances are often disclosed in patent documents. In a patent filed in 1995 by Microsoft ([9]), a procedure for root key compromise recovery relies on publication in an out-of-band channel. In 1998, VISA

International filed a patent ([10]) for trust anchor key rollover in which the hash code of the next public key is affixed to the current root security certificate. The technique disclosed by Diversinet in a 1997 patent application ([11]) for trust anchor key rollover is based on a number (e.g. from 3 to 7) of current public keys with a new key being signed by one or more of the existing keys before entering into force.

This last approach is reflected in two DNSSEC Internet drafts, using two different approaches for the resolver procedures aiming at a smooth trust anchor key renewal surviving a single trust anchor key compromise. These resolver procedures drafts are based respectively on a of signature count threshold ([12]) and timing of new public key signature timers ([13]). These procedures require a larger number of KSKs for a trust point zone than for a zone having a parental DS record. The security is based on the expectation that the simultaneous compromise of every private keys in one incident is less likely to occur than a single private key compromise. Such a risk pattern depends on isolation of private keys, which is inducing a labor-intensive operational burden. In any event, the reference [13] is perhaps the most advanced protocol standardization proposal for trust anchor renewal ever since the advent of public key cryptography 25 years ago.

2.4 The TAKREM Proposal

2.4.1 Features

The TAKREM proposal somehow carries the long lasting trust anchor idea but addresses the concerns with old public keys, and insider fraud is countered by precise procedural recommendations. Thus, the arguments in favor of short trust anchor periods are basically satisfied. The TAKREM proposal implies slightly more complex initial trust anchor key generation procedures. Nothing is changed in the trust anchor initial distribution scheme (except for an innocuous addenda to the trust anchor data). A TAKREM trust anchor key rollover message is defined; it requires specific, automated processing by the relying systems (e.g. DNS resolvers). The TAKREM rollover message processing allows a relying system to ignore bogus renewal messages.

There are two usage alternatives for TAKREM:

1) as a supplemental authentication mechanism, not altering the circumstances in which a relying system learns the details of a trust anchor key renewal, or
2) as the main authentication mechanism in a trust anchor key renewal scheme.

As a main authentication mechanism, a TAKREM renewal operation is identical and as effective irrespective of whether it is scheduled or triggered by a private key compromise. This is not the case with chained trust anchors as in [10], [11], [12], and [13].

2.4.2 Synopsis and Notation

The purpose of any key management scheme is to preserve the cryptographic properties of algorithms. For TAKREM, this is covered in reference [1] which is a required reading for a comprehensive understanding of the TAKREM proposal for DNS security. In the present subsection 2.4.2, we present a synopsis of the TAKREM processing rules, applied to the DNS security context.

We use the notation R_i for the public trust anchor key R_i, with the private key counterpart r_i.

The zone owner establishes key pairs $<r_0,R_0>$, $<r_1,R_1>$, $<r_2,R_2>$, ..., $<r_n,R_n>$, allocating the pair $<r_0,R_0>$ as the initial trust anchor key pair, and reserving each key pairs $<r_i,R_i>$ for the cryptoperiod starting with the i'th trust anchor renewal, for $1<=i<=n$.

A separate MASH (Modular Arithmetic Secure Hash) instance H_i is created for each R_i. MASH is defined in International standard document ISO/IEC 10118-4:1998 ([14]).

That is, the zone owner selects a large composite modulus number N_i used in the MASH round function and a prime number p_i used in the MASH final reduction function.

Then, the zone owner selects a random salt field s_i.

A hash computation gives a root key digest D_i :

$$D_i=H_i(s_i|R_i|N_i|P_i) .$$

The digest D_i is like an advanced notice of future trust anchor key R_i.

The data tuple $<r_i,R_i,N_i,p_i,s_i>$ is set aside in dead storage.

The trust anchor key initial distribution is

$$R_0,\ D_1,\ D_2,\ ...,\ D_n\ .$$

Security rationale: with data tuple $<r_i,R_i,N_i,p_i,s_i>$ totally concealed until the usage period for key pair $<r_i,R_i>$, an adversary is left with the digest D_i from which it is deemed impossible to mount a brute force attack.

A trust anchor key rollover is triggered by a rollover message carrying the following information:

$$i,<R_i,N_i,p_i,s_i>\ .$$

Upon receipt of this message, the DNS resolver becomes in a position to validate the trust anchor key digest D_i.

2.5 Use of TAKREM in the DNSSEC

The managers of important DNS zones (as defined above) may decide to use TAKREM for providing greater trust anchor assurance to properly configured resolvers. The DNS root zone could do the same. A parent zone may process TAKREM compatible trust anchor data from child zones like a resolver does. However, the parent-to-child key assurance requirements may be fulfilled with other security procedures, so TAKREM adoption by ordinary zones is perhaps harder to justify.

2.5.1 Initial Generation of Trust Anchor Keys

The TAKREM procedure requires the initial generation of a number n of signature key pairs $<r_i,R_i>$, the generation of cryptographic hash code values $D_1, D_2, ..., D_n$ according to subsection 2.5.4, and the independent key pair safeguarding for future use, with the record contents $<r_i,R_i,N_i,p_i,s_i>$ as explained in subsection 2.4.2. Bar code printouts in tamper-evident sealed envelopes appears as a suitable storage media for storage of individual key pairs in separate components. We call this dead storage because the cryptographic key material is remote from any live digital computing device. The key pair count n might be in the order of tens or hundreds of signature key pairs.

For the highest operational security standards, the computation for these initial key parameters should be performed in dedicated systems in complete isolation from computer networks, with a reliable source of entropy. Once the initial setup is complete, the systems should undergo a systematic memory erase operation, including printer buffers. In terms of overall cost structure, the related initial cost offsets future expenses related to key pair generation that would occur periodically in an alternate high security scheme.

2.5.2 Initial Trust Anchor Key Distribution

The public key from one of the key pairs (i.e. $<r_0,R_0>$) is distributed as the initial trust anchor key R_0, and the private counterpart r_0 is put into production (e.g. to sign ZSK DNSKEY resource records). Initial trust anchor distribution mechanisms are outside of the DNSSEC specifications, and would remain so even if the draft [13] was turned into an official standard. The TAKERM proposal does not change this. In any event, the initial trust anchor configuration data is augmented with a reference number f and an innocuous array of cryptographic hash code values $D_1, D_2, ..., D_n$, one for each possible future trust anchor key pair. The reference number f is a 32-bit integer reference value. For a given domain name, the reference range from $f+1$ to

$f+n$ should not be re-used by another initial trust anchor configuration.

Correspondingly, in a resolver the initial trust anchor configuration is received as a 4-tuple comprising:

1) a domain name,
2) an initial trust anchor public key R_0,
3) an array of cryptographic hash codes $D_1, D_2, \ldots, D_n$, and
4) a 32-bit integer reference value f.

No special attention is paid to the hash codes and the 32-bit reference at this initial configuration time: they are simply kept for later reference.

2.5.3 Trust Anchor Key Rollover

The DNS trust anchor rollover (augmented with TAKREM) can be described as a three phase procedure.

- In the first phase, the zone manager retrieves a coming trust anchor key pair record $<r_i,R_i,N_i,p_i,s_i>$ for some index value i from the secure dead storage and sets
  a) the public key R_i in a DNSKEY resource record,
  b) the private key r_i in the hardware security module used for generating zone digital signatures, and
  c) the TAKREM rollover message elements $f+i$, N_i, p_i, and s_i in an SDDA resource record, where f is the 32-bit reference described in the above subsection 2.5.2 (the SDDA record is described in subsections 3.2 and 3.5).
- The second phase is the new KSK usage in DNS zone signing according to DNSSEC specifications.
- The third phase is run by resolvers having received the TAKREM initial 4-tuple configuration. When they first encounter a DNSKEY RR for a zone name with a remembered 4-tuple, they may query the SDDA RR, and check the associated cryptographic hash.

2.5.4 Hash Function Input Stream Details

The TAKREM rollover message processing requires the unambiguous definition of a cryptographic hash function input stream in a format not necessarily typical of the DNS RR formatting rules. The ASN.1 format is used mainly as an editorial convenience in the first release of the present document. Since the hash code output is computed much in advance of the rollover message verification, it is conceivable that a digital signature algorithm was not even standardized in the DNS standardization activities. The verification processing design is thus based on the X.509 public key encoding standard that is expected to lead the DNS use of the underlying signature algorithm. In addition to providing an unambiguous canonical encoding, this approach supports the inclusion of new digital signature algorithms in the initial 4-tuple configuration. Nonetheless, a more DNS-stylish specification should be provided in a future document release or a follow-up document.

The TAKREM MASH input stream to be (re)constructed is the ASN.1 DER encoding for a value of the ASN.1 type defined as:

```
SEQUENCE {
     OCTET STRING SIZE(20..MAX),
                         -- salt field $s_i$, SDDA RR
                         -- Authentication Mechanism Data field
     [1] IMPLICIT SEQUENCE { -- SubjectPublicKeyInfo $R_i$
                             -- in RFC2459
          SEQUENCE { -- AlgorithmIdentifier in RFC2459
               algorithm  OBJECT IDENTIFIER,
               parameters ANY DEFINED BY algorithm OPTIONAL
          }
          subjectPublicKey BIT STRING
            -- public key encoded as a "subjectPublicKey"
            -- per RFC 2459, or per encoding rules for the
            -- subject public key algorithm
     }
     INTEGER, -- the MASH parameter $N_i$
     INTEGER  -- the MASH parameter $p_i$
}
```

In (re)constructing the MASH input stream, the ASN.1 encoding shall omit the optional fields that are “witness values” that allow verification of either number-theoretic properties intrinsic to the public key value, or the unbiased selection of random public key parameters (e.g. in RC2459 [2] for the Diffie-Hellman cryptosystem). Such fields may occur in the "parameters" ASN.1 field above, and/or in the "subjectPublicKey" ASN.1 field above

Other optional fields should be provided in the ASN.1 encoding of a public key. Some optional fields are required for the complete knowledge of a public key value and must be retained (e.g. the three DSA common parameters are required even if they are encoded in an optional field of an ASN.1 sequence). In the case of RSA public keys, the X.509 mandates a NULL field in the AlgorithmIdentifier ASN.1 structure, which must be retained according to the present specification.

In doing the above encoding conversion from the DNS encoding to ASN.1 encoding inspired from the X.509 specifications, the public key information is stripped of some usage control data. That is, a given type of public key might be allowable with a number of digital signature algorithms. The TAKREM mechanism specified in the subsection 3.5 protects the public key value with the type of public key, but not with a specific digital signature algorithm. Specifically, this allows RSA public keys to be used in the future with hash algorithms different from SHA-1, but also omit to protect against the use of weaker algorithms once stronger algorithms are deployed.

The current digital signature algorithms allowed for DNS zone signing are limited to RSA and DSA, plus the unspecified private algorithms. For RSA (resp. DSA), the ASN.1 encoding specification is in section 7.3.1 (resp. 7.3.3) in RFC2459 ([2]). For private algorithms, a similar public key encoding specification should be relied upon.

2.5.5 Usage Scenarios for DNS Security

The owner of an "important" zone having a DNSSEC-enabled parent may whish to use the TAKREM mechanism for a target resolver population. For instance, a financial institution might whish to offer the TAKREM enhanced authentication strength to its on-line banking users by fear of inadequate parent zone security practices.

If further IETF standardization activities adopt the trust anchor rollover procedure of reference [13], an "important" zone that is also a trust point might make use of TAKREM in addition to the mandatory provisions of [13]. In this perspective, a resolver having received the applicable 4-tuple configuration must wait for the expiry of the "add hold-on timer" if a new trust anchor public key is authenticated by an SDDA record. If the resolver insists on this positive authentication, it prevents a residual vulnerability in the "add hold-on timer" mechanism (see end of section 2.2 in reference [13]).

If TAKREM was to becomes an IETF-endorsed solution for trust anchor key renewal, it would be as an alternative to the Internet draft [13] or [12]. Then, any zone that is a trust point would simply roll in a new DNSKEY record with the SEP bit set along with the SDDA record. The TAKREM security effectiveness rests more on end-to-end cryptographic properties than on particulars of DNS protocol operations, perhaps lowering the operational burden for a given zone owner.

In the three above rollover scenarios, the simultaneous publication of a DNSKEY and an SDDA record in a single zone eliminates synchronisation issues that arise in DNSSEC, e,g, in the case of the DS to DNSKEY relationship. Overall, the TAKREM does not change zone management procedures for keeping the DNS chain of trust intact throughout key rollover operations (see reference [15]).

3. Specifications of an Authentication DNS Resource Record

3.1 The SEP DNSKEY Direct Authenticator (SDDA) DNS Resource Record

The SEP DNSSEC Direct Authenticator Resource Record (SDDA RR) provides a generic RR format for authentication mechanisms applicable to DNSKEY records with the SEP bit set. It is intended to fulfill the need of trust anchor key maintenance using cryptographic mechanisms outside of the DNSSEC core specifications. Zero or more SDDA records may occur in a zone for a DNSKEY record with the SEP bit, each using a different combination of algorithms and security context. An SDDA record provides additional information about the DNSKEY. This information may assist resolvers and/or parent zones in authenticating the DNSKEY record contents.

Authentication mechanisms used with SDDA records are expected to require and/or exploit outside arrangements, e.g. authentication key establishment or a pre-existing security association. Two specific mechanisms are initially provided for use in the SDDA record. One is based on a shared secret key MAC and may be useful for the KSK renewal operation between a child zone and its parent. The other mechanism is TAKREM and is used for authentication of "important" DNS zone keys in properly configured resolvers, and perhaps across the child-parent boundary.

The Type value for the SDDA RR type is [[to be determined]].

The DNSKEY RR is class independent.

The DNSKEY RR has no special TTL requirements.

3.2 SDDA RR Wire Format

The RDATA for an SDDA record consists of a 2 octet key tag field, four 1 octet type fields, and a number of other fields as implied, directly or indirectly, by the type indications. The type fields are respectively for

- an algorithm type field,
- a digest type field,
- an authentication mechanism type field,
- an authentication context type field.

Among the other fields, an SDDA record encoding comprises a digest field. The DS RR record specifications from RFC4034 ([6]), section 5.1, mostly apply to the key tag field, the algorithm type field, the digest type field, and the digest field (i.e. if a provision in the present document relates to these fields and may be subject to more than one interpretation, an interpretation compatible with RFC4034 [6] shall prevail).

It is not expected that a decoding software be able to parse the variable-length fields unless it is fully compliant with the set of type indication values.

```
                     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  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Auth Mech Type| Auth Ctx Type |                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               /
/                                                               /
/                  other fields, including Digest               /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The SDDA RR fields are listed in the table below in their order in the SDDA RR wire encoding. Some fields are optional and/or can be repeated. The group of three fields starting with Authentication Algorithm in section 3.2.10 may be repeated more than once. The Authentication Mechanism Data value lies at the end of SDDA RR wire encoding, it may comprise zero, one, or more fields (e.g. a DSA signature value is two integers). An SDDA record points to a DNSKEY record using the fields NAME, Key Tag, Algorithm, Digest Type, optionally Digest Type Extension, and Digest.

```
Section   Field Description,
                    Occurrence Rule
=======   ==================================
3.2.1     Key Tag,
                    Always present
3.2.2     Algorithm,
                    Always present
3.2.3     Digest Type,
                    Always present
3.2.4     Authentication Mechanism Type,
                    Always present
3.2.5     Authentication Context Type,
                    Always present
3.2.6     Digest Type Extension,
                    Optional based on Digest Type
3.2.7     Authentication Mechanism Type Extension,
                    Optional based on Authentication Mechanism
                    Type
3.2.8     Digest,
                    Always present
3.2.9     Authentication Context Data,
                    Optional based on Authentication Context Type
3.2.10.1  Authentication Algorithm Type,
                    Zero or more occurrences based on
                    Authentication Mechanism Type
3.2.10.2  Authentication Algorithm Type Extension,
                    Optional based on immediately preceding
                    Authentication Algorithm Type
3.2.10.3  Authentication Algorithm Common Parameters,
                    Zero or more occurrences based on immediately
                    preceding Authentication Algorithm Type
3.2.11    Authentication Mechanism Data,
                    Zero or more occurrences based on
                    Authentication Mechanism Type and any
                    Authentication Algorithm Type
```

3.2.1 The Key Tag Field

The Key Tag field lists the key tag of the DNSKEY RR referred to by the SDDA record, in network byte order.

The Key Tag used by the SDDA RR is as specified in Appendix B of RFC4034 ([6]).

3.2.2 The Algorithm Field

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

The algorithm number used by the SDDA RR is identical to the algorithm number used by DNSKEY RRs. Appendix A.1 in RFC4034 ([6]) lists the algorithm number types.

3.2.3 The Digest Type Field

The SDDA 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 in RFC4034 ([6]) lists the possible digest algorithm types.

In the event that private digest types become allowed in DS records using an extension mechanism (e.g. similar to the private algorithm support in DNSKEY private algorithms PRIVATEDNS and PRIVATEOID), the Digest Type Extension optional field is specified in section 3.2.6.

3.2.4 The Authentication Mechanism Type Field

The SDDA RR provides authentication information for a DNSKEY RR according to an authentication mechanism. The Authentication Mechanism type field identifies the specific mechanism applicable to the SDDA record, and partly determines the format of the RDATA variable part after the digest field.

The currently allocated authentication mechanisms are

- 0: reserved,
- 1: TAKREM key rollover message authentication, see section 3.5,
- 2: symmetric-key MAC authentication, see section 3.4,
- 253: Private mechanism identified by a domain-name-encoded extension field, see section 3.2.7,
- 254: Private mechanism identified by an ISO Object Identifier extension field, see section 3.2.7.

Mechanism numbers 253 and 254 are reserved for private use and neither will be assigned to a specific mechanism. Private mechanisms use the Authentication Mechanism Type Extension optional field specified in section 3.2.7.

3.2.5 The Authentication Context Type Field

Some authentication mechanism may need a reference to previously established parameters, e.g. a security association or a simple symmetric secret key. In cases where the Name field of the SDDA record does not provide sufficient context indication, a non-zero value in the Authentication Context Type field tells the format of reference information found in the Authentication Context Data field. The semantic rules of Authentication Context Data field should be specified with any authentication mechanisms that use a given format.

The currently allocated authentication context types are

- 0: none, i.e. the domain name provides adequate context information;
- 1: a DNS name (not compressed) is present in the Authentication Context Data field;
- 2: an opaque 32 bits value (encoded in network order) is present in the Authentication Context Data field.

3.2.6 The Digest Type Extension Field

The Digest Type Extension optional field is specified in anticipation that private digest types may me allowed in the future. In this case, the present extension field should be used as a placeholder for a variable-length private digest type indication. In the meantime, this optional field is not used.

3.2.7 The Authentication Mechanism Type Extension Field

When the Authentication Mechanism Type field value implies the presence of the present extension field, this subsection specification applies.

If the Authentication Mechanism Type field value is 253, the present extension field contains a wire encoded domain name, which must not be compressed. The domain name indicates the private mechanism to use. Entities should only use domain names they control to designate their private mechanisms.

If the Authentication Mechanism Type field value is 254, the present extension field contains an unsigned length byte followed by a BER encoded Object Identifier (ISO OID) of that length. The OID indicates the private mechanism to use. Entities should only use OIDs they control to designate their private mechanisms.

3.2.8 The Digest Field

The SDDA RR refers to a DNSKEY RR by including a digest of that DNSKEY RR, exactly like a DS record does, see section 5.1.4 in RFC4034 ([6]). The Digest Type field (perhaps with the Digest Type Extension field) identifies the algorithm used to construct the digest. The length and contents of this field is dependent on this digest algorithm.

3.2.9 The Authentication Context Data Field

The Authentication Context Data field contains the security context identification data in a format identified by the Authentication Context Type field. This field is present if a non-zero value is present in the Authentication Context Type field. An authentication mechanism may need this information in an SDDA RR instance for a complete reference to previously established security parameters.

3.2.10 Algorithm-Related Fields

A given authentication mechanism may rely on one or more cryptography algorithms, e.g. a digital signature mechanism may use two such algorithms: a cryptographic hash function and a public key primitive. For each cryptographic algorithm, a set of algorithm-related fields may be present. This allows the selection of a specific algorithm among a family of alternatives, and/or the specification of common parameters applicable to an algorithm instance. The presence of one or more sets is implicitly specified by the Authentication Mechanism Type indication in an SDDA RR, as are the algorithm-specific field contents and semantic.

A set of algorithm-related fields comprises at least the Authentication Algorithm Type field described in section 3.2.10.1, optionally the Authentication Algorithm Type Extension field described in section 3.2.10.2, and optionally the Authentication Algorithm Common Parameters field described in section 3.2.10.3. When more than one algorithm deserve some algorithm-related fields, they should appear by groups, e.g. Authentication Algorithm Type, Authentication Algorithm Type Extension, and Authentication Algorithm Common Parameters for a first algorithm, then these three fields for the next algorithm.

3.2.10.1 The Authentication Algorithm Type Field

The Authentication Algorithm Type field is optional and significant in the context of the Authentication Mechanism field value of a given SDDA RR encoding. This should be a one-octet field. The mechanism-specific allocation of algorithm type values may make use of the Authentication Algorithm Type Extension field for private algorithms.

3.2.10.2 The Authentication Algorithm Type Extension Field

The Authentication Algorithm Type Extension field allows private algorithms to be specified in the Authentication Algorithm Type field, as may be specified for a given authentication mechanism.

3.2.10.3 The Authentication Algorithm Common Parameters Field

Some cryptographic algorithm require the specification of common parameters, e.g. the Diffie-Hellman cryptosystem, other algorithms based on the discrete logarithm problem and elliptic curve cryptographic algorithms. The Authentication Algorithm Common Parameters field allows the specification of these parameters, using algorithm-specific rules. These rules may specify the common parameter values directly or by reference to values published elsewhere (e.g the United States NIST organization publishes elliptic curve parameters).

3.2.11 The Authentication Mechanism Data Field

The Authentication Mechanism Data field contains the "payload" data relevant to the authentication mechanism applied to the DNSKEY RR linked to the SDDA RR. The rules applicable to this data field are governed by the authentication mechanism and any authentication algorithm specified in the previous fields of this SDDA RR.

3.3 The SDDA 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 as specified in Appendix A.1 of RFC4034 ([6]).

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

The Authentication Mechanism Type MUST be represented as an unsigned decimal integer.

The Authentication Context Type MUST be represented as an unsigned decimal integer.

The remaining portion of the RDATA field must be represented as a Base64 encoding of its contents. Whitespace is allowed within the Base64 text. For a definition of Base64 encoding, see [16].

3.4 Application of the SDDA RR with Symmetric-Key Authentication

In this subsection, the symmetric-key authentication mechanism defined in RFC2845 [17] is adapted as an application of the SDDA RR. This can be useful for authenticating a KSK renewal to the parent zone, provided adequate key management procedures are in place.

3.4.1 Wire Format Contents

The SDDA RR Authentication Mechanism Type must contain 2 for the present subsection 3.4 to apply.

The SDDA RR Authentication Context Type field should be 1, indicating that the SDDA RR Authentication Context Data field contains a key name encoded as a DNS name (not compressed). Other Authentication Context Type values may be used provided a prior arrangement allowing participating parties to identify the symmetric key to be used in SDDA RR validation.

The SDDA RR must contain one set of Algorithm-Related fields. Within this set, the Authentication Algorithm Type field shall contain the value 1 to indicate the presence of a DNS name (not compressed) for algorithm name in the following Authentication Algorithm Type Extension field. This algorithm name is assigned by IANA for RFC2845 [17] algorithm names. The SDDA RR must contain no Authentication Algorithm Common Parameters field.

The SDDA RR Authentication Mechanism Data field comprises a two-octet integer (in network order) holding the length (octet count) of the MAC output, followed by an octet stream of this length encoding the MAC output.

3.4.2 Processing Rules

When generating or verifying the contents of an SDDA RR compliant to the present subsection 3.4, the symmetric-key based authentication algorithm digests the DNSKEY RR referred to by the SDDA RR, in wire format (i.e. from the DNSKEY RR NAME field to the end of the Public Key field), except that the TTL field is set to zero before the computation. The MAC algorithm must be the symmetric-key based authentication algorithm identified by the Authentication Algorithm Type field value and the Authentication Algorithm Type Extension field value. The secret symmetric key to use is identified by the Authentication Context Type field value and the Authentication Context Data field value, if any.

3.5 Application of the SDDA RR with TAKREM Key Rollover Message Authentication

In this subsection, the TAKREM key rollover message is used to populate an SDDA RR, providing a mechanism allowing DNS resolvers (and parent zones) to authenticate a DNSKEY RR following a KSK renewal.

A zone owner may prepare at once a DNSKEY record and an SDDA record together making up a TAKREM rollover message. Using the notation from subsection 2.4.2, the zone owner retrieves the record contents $<r_i,R_i,N_i,p_i,s_i>$ for some index value i, and the reference number f associated with the initial 4-tuple configuration. From this record, the zone owner prepares the DNSKEY RR holding the public key R_i, and an SDDA RR defined in the present subsection.

A resolver may process an SDDA record by first matching the DNSKEY record pointed by the SDDA record, then reconstructing a TAKREM digest from the DNSKEY and SDDA record, and finally checking the presence of a identical digest value D_i among the initial 4-tuple configurations that the resolver may have accepted as part of trusted configuration.

3.5.1 Wire Format Contents

The SDDA RR Authentication Mechanism Type must contain 1 for the present subsection 3.5 to apply.

The SDDA RR Authentication Context Type field should be 2 or 0.

- When this field value is 2, the 32-bits opaque value in the Authentication Context Data field must hold the value $f+i$. From the value $f+i$, a resolver can readily identify a most probable candidate digest D_i for TAKREM hash code validation.

- When the Authentication Context Type field is 0, the SDDA RR Authentication Context Data field is absent because the NAME and the Algorithm fields in the SDDA RR should identify the relevant initial trust anchor configuration for which a renewal operation is authenticated by the SDDA RR (e.g. if it is not anticipated that more than one initial 4-tuple configuration is needed for a given zone and digital signature algorithm and indexing techniques are used to search the candidates digests D_i).

- Other Authentication Context Type values may be used provided a prior arrangement allowing relying parties to identify the relevant initial trust anchor configuration.

The SDDA RR must contain one set of Algorithm-Related fields. For the Authentication Algorithm Type field within this set, the currently allocated values are

- 1: MASH-1 as defined in [14], and
- 2: MASH-2 as defined in [14].

Accordingly, the SDDA RR must contain no Authentication Algorithm Type Extension field. Two MASH parameters, respectively N_i, a large composite number of unknown factorization, and p_i, a large prime number, populate the Authentication Algorithm Common Parameters field in the SDDA RR wire format. Each of N_i and p_i is encoded as a two-octet integer (in network order) holding the length (octet count) of the MASH parameter value, followed by an octet stream of this length, encoding the parameter value in network order. The highest significant bit of this large number representation shall be a sign bit in a two complement representation.

The SDDA RR Authentication Mechanism Data field comprises a two-octet integer (in network order) holding the length (octet count) of the TAKREM rollover message salt field s_i, followed by an octet stream of this length, encoding this salt field in binary form.

3.5.2 Processing of TAKREM Rollover Messages Encoded in SDDA RR

As can be inferred from other portions of this document, a resolver should validate the authenticity of a DNS resource record pair SDDA plus linked DNSKEY with the SDDA having an Authentication Mechanism Type set to 2 as if they provide a TAKREM rollover message. This validation is based on a number of data elements:

- the DNSKEY RR linked to the SDDA RR, providing the public key component R_i of the TAKREM MASH input stream,
- the field contents from the SDDA RR Authentication Mechanism Data field, providing the salt component s_i of the TAKREM MASH input stream, and
- the two large integer values from the Authentication Algorithm Common Parameters field, providing the MASH parameters N_i and p_i, thus revealing the specific MASH function used in the computation of the hash code outputs in the initial 4-tuple configuration,

from which a TAKREM MASH input stream must be reconstructed according to subsection 2.5.4, using values s_i, R_i, N_i, p_i. Then the resolver computes a TAKREM rollover message hash code according to the integer value from the Authentication Algorithm Type field (either 1 or 2), providing the selection among the MASH-1 and MASH-2 variants. Finally, the computed message hash code is checked for equality with any of the trusted digests D_i from the initial 4-tuple configurations associated with the relevant domain name. If the SDDA RR Authentication Context Type field holds the value 2, the SDDA RR Authentication Context Data field provides a 32-bit reference value $f+i$ that provides a hint for selecting a candidate digest D_i. If no matching trusted digests D_i can be found, the resolver should deny any trust anchor status to the DNSKEY record.

3.6 IANA Considerations

For the specification of section 3 to become effective, IANA would be requested to assign a DNS type code to the SDDA resource record from the Specification Required portion of the DNS Resource Record Type registry.

If evolved into a recognized standard, the present document creates two "name spaces" (RFC2434, [18]) of relevance to IANA: the SDDA record Authentication Mechanism Type and the SDDA record Authentication Context Type. Other name spaces in the present document are either already covered by provisions of existing standards (Algorithm and Digest Type from RFC4034 [6]) or delegated to the specification for an authentication mechanism (Authentication Algorithm Type).

4. List of Abbreviations

DNS	
DNSSEC	
DNSKEY	
RR	
SDDA	SEP DNSKEY Direct Authenticator
KSK	
SEP	
TAKREM	
SAKEM	
TA	Trust anchor
QTYPE	
IANA	

5. References

[1] Thierry Moreau, "A Note About Trust Anchor Key Distribution", CONNOTECH Experts-conseils inc., Document Number C003444, 2005/07/05, http://www.connotech.com/takrem.pdf

[2] R. Housley, W. Ford, W. Polk, D. Solo, "Internet X.509 Public Key Infrastructure Certificate and CRL Profile", RFC 2459, January 1999

[3] Garfinkel, Simson L., "PGP: Pretty Good Privacy", O'Reilly & Associates, Inc., Sebastopol, Calofirnia, 1995

[4] See http://www.connotech.com/sakem_index.htm

[5] R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005

[6] R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005

[7] R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005

[8] M. Cooper, Y. Dzambasow, P. Hesse, S. Joseph, R. Nicholas, "Internet X.509 Public Key Infrastructure: Certification Path Building", Internet Draft draft-ietf-pkix-certpathbuild-05, January 2005

[9] Spelman, Jeffrey F., Thomlinson, Mattew W., "Root Key Compormise Recovery", US patent document 5,680,458, October 21, 1997 (application number 555,697, November 14, 1995), assigned to Microsoft Corporation

[10] Tony Lewis; "Key replacement in a public key cryptosystem", United States Patent no. 6,240,187, issued on May 29, 2001, assigned to Visa International, filed on February 10, 1998

[11] Barkan, Yuval and Barkan, Mordhai, "System and Method for Reliable Key Transfer", based on PCT application number PCT/IL1998/000381, claiming priority based on Israel patent application 121551, filed Aug. 14, 1997

[12] J. Ihren, O. Kolkman, B. Manning, "An In-Band Rollover Mechanism and an Out-Of-Band Priming Method for DNSSEC Trust Anchors", internet draft draft-ietf-dnsext-trustupdate-threshold-00.txt, October 18, 2004

[13] M. StJohns, "Automated Updates of DNSSEC Trust Anchors", internet draft draft-ietf-dnsext-trustupdate-timers-01.txt, August 15, 2005

[14] International standard document ISO/IEC 10118-4:1998, "Information technology - Security techniques - Hash-functions - Part 4: Hash-functions using modular arithmetic"

[15] O. Kolkman, R. Gieben, "DNSSEC Operational Practices", Internet-Draft draft-ietf-dnsop-dnssec-operational-practices-04.txt, March 2005

[16] S. Josefsson, Ed., "The Base16, Base32, and Base64 Data Encodings", RFC 3548, July 2003

[17] P. Vixie, O. Gudmundsson, D. Eastlake 3rd, B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, May 2000

[18] T. Narten, H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 2434, October 1998