Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: January 10, 2013

S. Morris
ISC
J. Ihren
Netnod
J. Dickinson
Sinodun

July 9, 2012

DNSSEC Key Timing Considerations
draft-ietf-dnsop-dnssec-key-timing-03.txt

Abstract

This document describes the issues surrounding the timing of events in the rolling of a key in a DNSSEC-secured zone. It presents timelines for the key rollover and explicitly identifies the relationships between the various parameters affecting the process.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 10, 2013.

Table of Contents

1. Introduction

1.1. Key Rolling Considerations

When a zone is secured with DNSSEC, the zone manager must be prepared to replace ("roll") the keys used in the signing process. The rolling of keys may be caused by compromise of one or more of the existing keys, or it may be due to a management policy that demands periodic key replacement for security or operational reasons. In order to implement a key rollover, the keys need to be introduced into and removed from the zone at the appropriate times. Considerations that must be taken into account are:

- DNSKEY records and associated information (such as the associated DS records or RRSIG records created with the key) are not only held at the authoritative nameserver, they are also cached by resolvers. The data on these systems can be interlinked, e.g., a validating resolver may try to validate a signature retrieved from a cache with a key obtained separately.

- Zone "boot-strapping" events, where a zone is signed for the first time, can be common in configurations where a large number of zones are being served. Procedures should be able to cope with the introduction of keys into the zone for the first time as well as "steady-state", where the records are being replaced as part of normal zone maintenance.

- To allow for an emergency re-signing of the zone as soon as possible after a key compromise has been detected, standby keys (additional keys over and above those used to sign the zone) need to be present.

- A query for the DNSKEY RRset returns all DNSKEY records in the zone. As there is limited space in the UDP packet (even with EDNS0 support), key records no longer needed must be periodically removed. (For the same reason, the number of standby keys in the zone should be restricted to the minimum required to support the key management policy.)

Management policy, e.g., how long a key is used for, also needs to be considered. However, the point of key management logic is not to ensure that a rollover is completed at a certain time but rather to ensure that no changes are made to the state of keys published in the zone until it is "safe" to do so ("safe" in this context meaning that at no time during the rollover process does any part of the zone ever go bogus). In other words, although key management logic enforces policy, it may not enforce it strictly.

A high-level overview of key rollover can be found in [I-D.ietf-dnsop-rfc4641bis]. In contrast, this document focuses on the low-level timing detail of two classes of operations described there, the rollover of key-signing keys, and the rollover of zone signing keys.

1.2. Types of Keys

Although DNSSEC validation treats all keys equally, [RFC4033] recognises the broad classification of zone-signing keys (ZSK) and key-signing keys (KSK). A ZSK is used to authenticate information within the zone; a KSK is used to authenticate the zone's DNSKEY RRset. The main implication for this distinction concerns the consistency of information during a rollover.

During operation, a validating resolver must use separate pieces of information to perform an authentication. At the time of authentication, each piece of information may be in its cache or may need to be retrieved from the authoritative server. The rollover process needs to happen in such a way that at all times during the rollover the information is consistent. With a ZSK, the information is the RRSIG (plus associated RRset) and the DNSKEY. These are both obtained from the same zone. In the case of the KSK, the information is the DNSKEY and DS RRset with the latter being obtained from a different zone.

Although there are similarities in the algorithms to roll ZSKs and KSKs, there are a number of differences. For this reason, the two types of rollovers are described separately. It is also possible to use a single key as both the ZSK and KSK. However, the rolling of this type of key is not treated in this document.

1.3. Terminology

The terminology used in this document is as defined in [RFC4033] and [RFC5011].

A number of symbols are used to identify times, intervals, etc. All are listed in Appendix A.

1.4. Requirements Language

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 [RFC2119].

2. Rollover Methods

2.1. ZSK Rollovers

A ZSK can be rolled in one of three ways:

- Pre-Publication: described in [I-D.ietf-dnsop-rfc4641bis], the new key is introduced into the DNSKEY RRset which is then re-signed. This state of affairs remains in place for long enough to ensure that any cached DNSKEY RRsets contain both keys. At that point signatures created with the old key can be replaced by those created with the new key, and the old signatures removed. During the re-signing process (which may or may not be atomic depending on how the zone is managed), it doesn't matter which key an RRSIG record retrieved by a resolver was created with; cached copies of the DNSKEY RRset will contain both the old and new keys.

  Once the zone contains only signatures created with the new key, there is an interval during which RRSIG records created with the old key expire from caches. After this, there will be no signatures anywhere that were created using the old key, and it can can be removed from the DNSKEY RRset.

- Double-Signature: also mentioned in [I-D.ietf-dnsop-rfc4641bis], this involves introducing the new key into the zone and using it to create additional RRSIG records; the old key and existing RRSIG records are retained. During the period in which the zone is being signed (again, the signing process may not be atomic), validating resolvers are always able to validate RRSIGs: any combination of old and new DNSKEY RRset and RRSIG allows at least one signature to be validated.

  Once the signing process is complete and enough time has elapsed to allow all old information to expire from caches, the old key and signatures can be removed from the zone. As before, during this period any combination of DNSKEY RRset and RRSIG will allow validation of at least one signature.

- Double-RRSIG: strictly speaking, the use of the term "Double-Signature" above is a misnomer as the method is not only double signature, it is also double key as well. A true Double-Signature method (here called the Double-RRSIG method) involves introducing new signatures in the zone (while still retaining the old ones) but not introducing the new key.

  Once the signing process is complete and enough time has elapsed to ensure that all caches that may contain an RR and associated RRSIG have a copy of both signatures, the key is changed. After a

further interval during which the old DNSKEY RRset expires from caches, the old signatures are removed from the zone.

Of three methods, Double-Signature is conceptually the simplest - introduce the new key and new signatures, then approximately one TTL later remove the old key and old signatures. Pre-Publication is more complex - introduce the new key, approximately one TTL later sign the records, and approximately one TTL after that remove the old key. Double-RRSIG is essentially the reverse of Pre-Publication - introduce the new signatures, approximately one TTL later change the key, and approximately one TTL after that remove the old signatures.

2.2. KSK Rollovers

For ZSKs, the issue for the validating resolver is to ensure that it has access to the ZSK that corresponds to a particular signature. In the KSK case, this can never be a problem as the KSK is only used for one signature (that over the DNSKEY RRset) and both the key and the signature travel together. Instead, the issue is to ensure that the KSK is trusted.

Trust in the KSK is either due to the existence of a signed and validated DS record in the parent zone or an explicitly configured trust anchor. If the former, the rollover algorithm will need to involve the parent zone in the addition and removal of DS records, so timings are not wholly under the control of the zone manager. If the latter, [RFC5011] timings will be needed to roll the keys. (Even in the case where authentication is via a DS record, the zone manager may elect to include [RFC5011] timings in the key rolling process so as to cope with the possibility that the key has also been explicitly configured as a trust anchor.)

It is important to note that this does not preclude the development of key rollover logic; in accordance with the goal of the rollover logic being able to determine when a state change is "safe", the only effect of being dependent on the parent is that there may be a period of waiting for the parent to respond in addition to any delay the key rollover logic requires. Although this introduces additional delays, even with a parent that is less than ideally responsive the only effect will be a slowdown in the rollover state transitions. This may cause a policy violation, but will not cause any operational problems.

Like the ZSK case, there are three methods for rolling a KSK:

- Double-Signature: also known as Double-DNSKEY, the new KSK is added to the DNSKEY RRset which is then signed with both the old and new key. After waiting for the old RRset to expire from

caches, the DS record in the parent zone is changed. After waiting a further interval for this change to be reflected in caches, the old key is removed from the RRset. (The name "Double-Signature" is used because, like the ZSK method of the same name, the new key is introduced and immediately used for signing.)

- Double-DS: the new DS record is published. After waiting for this change to propagate into caches, the KSK is changed. After a further interval during which the old DNSKEY RRset expires from caches, the old DS record is removed.

- Double-RRset: the new KSK is added to the DNSKEY RRset which is then signed with both the old and new key, and the new DS record added to the parent zone. After waiting a suitable interval for the old DS and DNSKEY RRsets to expire from caches, the old DNSKEY and DS record are removed.

In essence, "Double-Signature" means that the new KSK is introduced first and used to sign the DNSKEY RRset. The DS record is changed, and finally the old KSK removed. With "Double-DS" it is the other way around. Finally, Double-RRset does both updates more or less in parallel.

2.3. Summary

The methods can be summarised as follows:

ZSK Method	KSK Method	Description
Pre-Publication	(not applicable)	Publish the DNSKEY before the RRSIGs.
Double-Signature	Double-Signature	Publish the DNSKEY and RRSIGs at same time. For a KSK, this happens before the DS is published.
Double-RRSIG	(not applicable)	Publish RRSIGs before the DNSKEY.
(not applicable)	Double-DS	Publish DS before the DNSKEY.
(not applicable)	Double-RRset	Publish DNSKEY and DS in parallel.

Table 1

3. Key Rollover Timelines

3.1. Key States

During the rolling process, a key moves through different states. The defined states are:

Generated — The key has been created, but has not yet been used for anything.

Published — The DNSKEY record - or information associated with it - is published in the zone, but predecessors of the key (or associated information) may be held in caches.

The idea of "associated information" is used in rollover methods where RRSIG or DS records are published first and the DNSKEY is changed in an atomic operation. It allows the rollover still to be thought of as moving through a set of states. In the rest of this section, the term "key data" should be taken to mean "key or associated information".

Ready — The new key data has been published for long enough to guarantee that any previous versions of the DNSKEY RRset have expired from caches.

Active — The key has started to be used to sign RRsets. Note that when this state is entered, it may not be possible for validating resolvers to use the key for validation in all cases: the zone signing may not have finished, or the data might not have reached the resolver because of propagation delays and/or caching issues. If this is the case, the resolver will have to rely on the key's predecessor instead.

Retired — A successor key has become active and this key is no longer being used to generate RRSIGs. However, as there may still be RRSIGs in caches that were generated using this key, it is being retained in the zone until they have expired.

Dead — The key is published in the zone but there is no longer information anywhere that requires its presence. Hence the key can be removed from the zone at any time.

Removed The key has been removed from the zone.

There is one additional state, used where [RFC5011] considerations are in effect (see Section 3.3.4):

Revoked The key is published for a period with the "revoke" bit set as a way of notifying validating resolvers that have configured it as an [RFC5011] trust anchor that it is about to be removed from the zone.

3.2. Zone-Signing Key Timelines

The following sections describe the rolling of a ZSK. They show the events in the lifetime of a key (referred to as "key N") and cover its replacement by its successor (key N+1).

3.2.1. Pre-Publication Method

The following diagram shows the timeline of a Pre-Publication rollover. Time increases along the horizontal scale from left to right and the vertical lines indicate events in the process. Significant times and time intervals are marked.

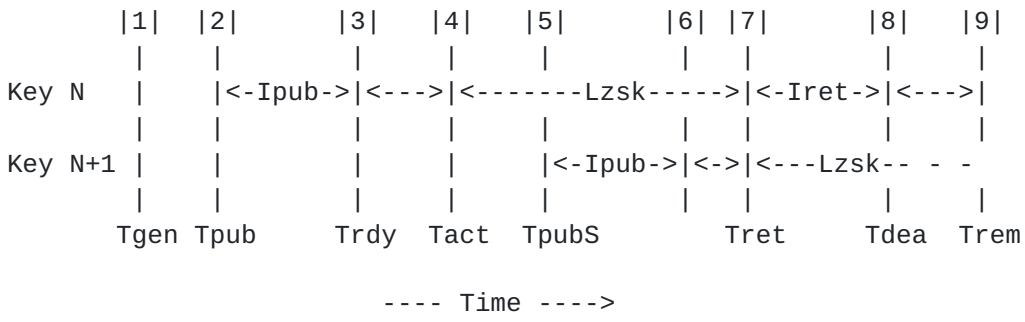

Figure 1: Timeline for a Pre-Publication ZSK rollover.

Event 1: Key N is generated at the generate time (Tgen). Although there is no reason why the key cannot be generated immediately prior to its publication in the zone (Event 2), some implementations may find it convenient to create a pool of keys in one operation and draw from that pool as required. For this reason, it is shown as a separate event. Keys that are available for use but not published are said to be generated.

Event 2: Key N's DNSKEY record is put into the zone, i.e., it is added to the DNSKEY RRset which is then re-signed with the current key-signing key. The time at which this occurs is the key's

publication time (Tpub), and the key is now said to be published. Note that the key is not yet used to sign records.

Event 3: Before it can be used, the key must be published for long enough to guarantee that any cached version of the zone's DNSKEY RRset includes this key.

This interval is the publication interval (Ipub) and, for the second or subsequent keys in the zone, is given by:

Ipub = Dprp + TTLkey

Here, Dprp is the propagation delay - the time taken in the worst-case situation for a change introduced at the master to replicate to all slave servers - which depends on the depth of the master-slave hierarchy. TTLkey is the time-to-live (TTL) for the DNSKEY records in the zone. The sum is therefore the maximum time taken for existing DNSKEY records to expire from caches, regardless of the nameserver from which they were retrieved.

(The case of introducing the first ZSK into the zone is discussed in Section 3.3.5.)

After a delay of Ipub, the key is said to be ready and could be used to sign records. The time at which this event occurs is the key's ready time (Trdy), which is given by:

Trdy = Tpub + Ipub

Event 4: At some later time, the key starts being used to sign RRsets. This point is the activation time (Tact) and after this, the key is said to be active.

Event 5: At some point thought must be given to its successor (key N+1). As with the introduction of the currently active key into the zone, the successor key will need to be published at least Ipub before it is activated. Denoting the publication time of the successor key by TpubS, then:

TpubS <= Tact + Lzsk - Ipub

Here, Lzsk is the length of time for which a ZSK will be used (the ZSK lifetime). It should be noted that unlike the publication interval, Lzsk is not determined by timing logic, but by key management policy. Lzsk will be set by the operator according to their assessment of the risks posed by continuing to use a key and the risks associated with key rollover. However, operational considerations may mean a key is active for slightly more or less

than Lzsk.

Event 6: While key N is still active, its successor becomes ready. From this time onwards, key N+1 could be used to sign the zone.

Event 7: When key N has been in use for an interval equal to the ZSK lifetime, it is retired (i.e., it will never again be used to generate new signatures) and key N+1 activated and used to sign the zone. This is the retire time of key N (Tret) and is given by:

Tret = Tact + Lzsk

It is also the activation time of the successor key (TactS). Note that operational considerations may cause key N to remain in use for longer than Lzsk; if so, the retirement actually occurs when the successor key is made active.

Event 8: The retired key needs to be retained in the zone whilst any RRSIG records created using this key are still published in the zone or held in caches. (It is possible that a validating resolver could have an unexpired RRSIG record and an expired DNSKEY RRset in the cache when it is asked to provide both to a client. In this case the DNSKEY RRset would need to be looked up again.) This means that once the key is no longer used to sign records, it should be retained in the zone for at least the retire interval (Iret) given by:

Iret = Dsgn + Dprp + TTLsig

Dsgn is the delay needed to ensure that all existing RRsets have been re-signed with the new key. Dprp is (as described above) the propagation delay, required to guarantee that the updated zone information has reached all slave servers, and TTLsig is the maximum TTL of all the RRSIG records in the zone created with the ZSK.

The time at which all RRSIG records created with this key have expired from resolver caches is the dead time (Tdea), given by:

Tdea = Tret + Iret

... at which point the key is said to be dead.

Event 9: At any time after the key becomes dead, it can be removed from the zone's DNSKEY RRset, which must then be re-signed with the current key-signing key. This time is the removal time (Trem), given by:

Trem >= Tdea

... at which time the key is said to be removed.

3.2.2. Double-Signature Method

The timeline for a double-signature rollover is shown below. The diagram follows the convention described in Section 3.2.1

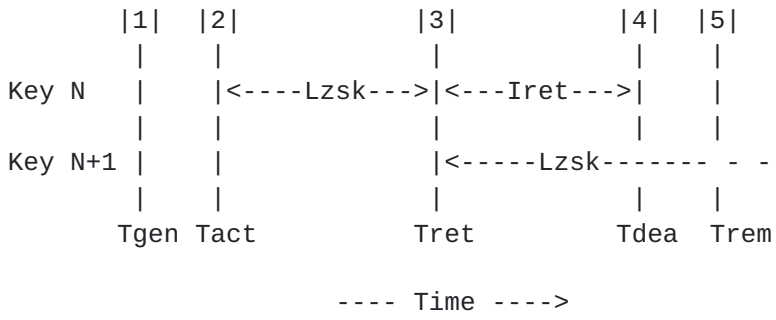

Figure 2: Timeline for a Double-Signature ZSK rollover.

Event 1: Key N is generated at the generate time (Tgen). Although there is no reason why the key cannot be generated immediately prior to its publication in the zone (Event 2), some implementations may find it convenient to create a pool of keys in one operation and draw from that pool as required. For this reason, it is shown as a separate event. Keys that are available for use but not published are said to be generated.

Event 2: Key N is added to the DNSKEY RRset and is then used to sign the zone; existing signatures in the zone are not removed. This is the activation time (Tact), after which the key is said to be active.

Event 3: After the current key (key N) has been in use for its intended lifetime (Lzsk), the successor key (key N+1) is introduced into the zone and starts being used to sign RRsets: neither the current key nor the signatures created with it are removed. The successor is key is now active and the current key is said to be retired. This time is the retire time of the key (Tret); it is also the activation time of the successor key (TactS).

Tret = Tact + Lzsk

Event 4: Before key N can be withdrawn from the zone, all RRsets that need to be signed must have been signed by the successor key (key N+1) and any old RRsets that do not include the new key or new RRSIGs

must have expired from caches. Note that the signatures are not replaced - each RRset is signed by both the old and new key.

This takes Iret, the retire interval, given by the expression:

Iret = Dsgn + Dprp + max(TTLkey, TTLsig)

As before, Dsgn is the delay needed to ensure that all existing RRsets have been signed with the new key and Dprp is the propagation delay. The final term (the maximum of TTLkey and TTLsig) is the period to wait for key and signature data associated with key N to expire from caches. (TTLkey is the TTL of the DNSKEY RRset and TTLsig is the maximum TTL of all the RRSIG records in the zone created with the ZSK. The two may be different: although the TTL of an RRSIG is equal to the TTL of the RRs in the associated RRset [RFC4034], the DNSKEY RRset only needs to be signed with the KSK.)

At the end of this interval, key N is said to be dead. This occurs at the dead time (Tdea) so:

Tdea = Tret + Iret

Event 5: At some later time key N and the signatures generated with it can be removed from the zone. This is the removal time Trem, given by:

Trem >= Tdea

3.2.3. Double-RRSIG Method

The timeline for a double-signature rollover is shown below. The diagram follows the convention described in Section 3.2.1

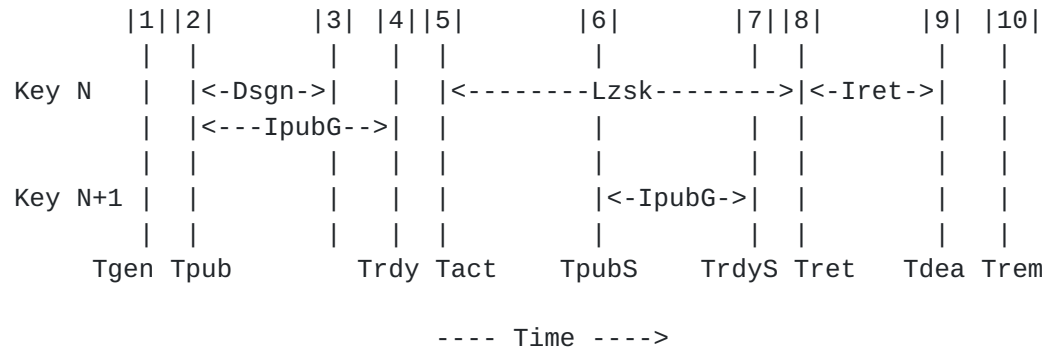

Figure 3: Timeline for a Double-Signature ZSK rollover.

Event 1: Key N is generated at the generate time (Tgen). Although there is no reason why the key cannot be generated immediately prior to its publication in the zone (Event 2), some implementations may find it convenient to create a pool of keys in one operation and draw from that pool as required. For this reason, it is shown as a separate event. Keys that are available for use but not published are said to be generated.

Event 2: Key N is used to sign the zone but existing signatures are retained. Although the new ZSK is not published in the zone at this point, for analogy with the other ZSK rollover methods and because this is the first time that key information is visible (albeit indirectly through the created signatures) this time is called the publication time (Tpub).

Event 3: After the signing interval, Dsgn, all RRsets that need to be signed have been signed by the new key. (As a result, all these RRsets are now signed twice, once by the (still-absent) key N and once by its predecessor.

Event 4: There is now a delay while the old signature information expires from caches. This interval is given by the expression:

Dprp + TTLsig

As before, Dprp is the propagation delay and TTLsig is the maximum TTL of all the RRSIG records in the zone created with the ZSK.

Again in analogy with other key rollover methods, this is defined as key N's ready time (Trdy) and the key is said to be in the ready state. (Although the key is not in the zone, it is ready to be used.) The interval between the publication and ready times is the publication interval of the signature, IpubG, i.e.,

Trdy = Tpub + IpubG

where

IpubG = Dsgn + Dprp + TTLsig

Event 5: At some later time the predecessor key is removed and the key N added to the DNSKEY RRset. As all the signed RRs have signatures created by the old and new keys, the records can still be authenticated. This time is the activation time (Tact) and the key is now said to be active.

Event 6: At some point thought must be given to rolling the key. The first step is to publish signatures created by the successor key (key

N+1) early enough for key N to be replaced after it has been active for its scheduled lifetime. This occurs at TpubS (the publication time of the successor), given by:

TpubS <= Tact + Lzsk - IpubG

Event 7: The signatures have propagated and the new key could be added to the zone. This time is the ready time of the successor key (TrdyS).

TrdyS = TpubS + IpubG

... where IpubG is as defined above.

Event 8: At some later time key N is removed from the zone's DNSKEY RRset and the successor key (key N+1) is added to it. This is the retire time of the key (Tret).

Event 9: The signatures must remain in the zone for long enough that the new DNSKEY RRset has had enough time to propagate to all caches. Once caches contain the new DNSKEY, the old signatures are no longer of use and can be considered to be dead as they cannot be validated by any key. In analogy with other rollover methods, the time at which this occurs is the dead time (Tdea), given by:

Tdea = Tret + Iret

... where Iret is the retire interval, given by:

Iret = Dprp + TTLkey

Dprp is as defined earlier and TTLkey is the TTL of the DNSKEY RRset.

Event 10: At some later time the signatures can be removed from the zone. In analogy with other rollover methods, this time is called the remove time (Trem) and is given by:

Trem >= Tdea

3.3. Key-Signing Key Rollover Timelines

The following sections describe the rolling of a KSK. They show the events in the lifetime of a key (referred to as "key N") and cover it replacement by its successor (key N+1).

3.3.1. Double-Signature Method

The timeline for a double-signature rollover is shown below. The diagram follows the convention described in Section 3.2.1

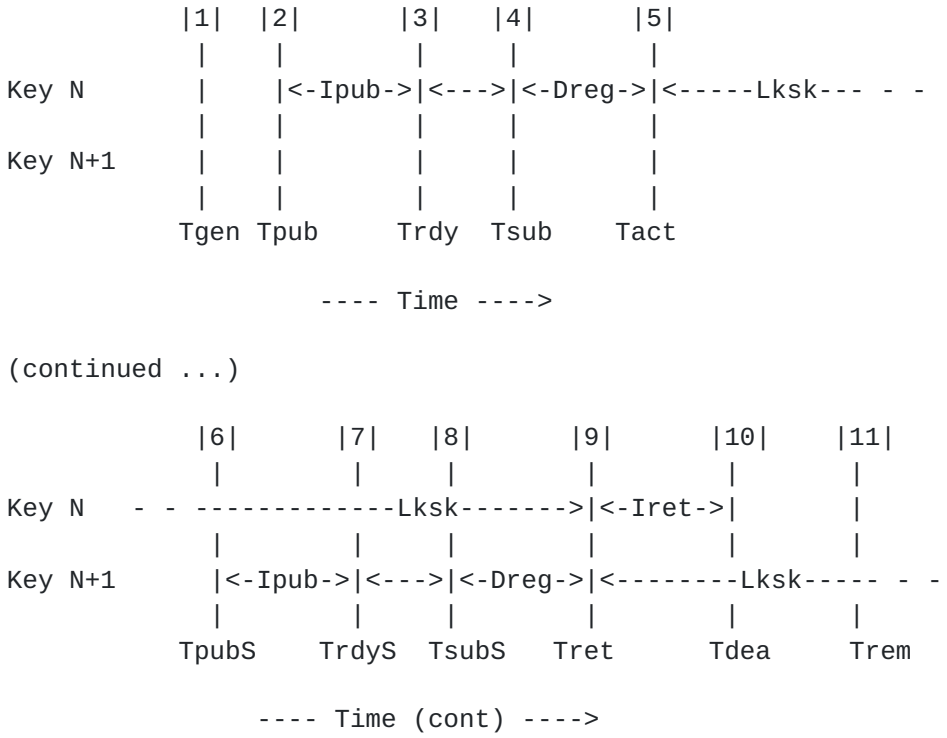

Figure 4: Timeline for a Double-Signature KSK rollover.

Event 1: Key N is generated at the generate time (Tgen). Although there is no reason why the key cannot be generated immediately prior to its publication in the zone (Event 2), some implementations may find it convenient to create a pool of keys in one operation and draw from that pool as required. For this reason, it is shown as a separate event. Keys that are available for use but not published are said to be generated.

Event 2: Key N is introduced into the zone; it is added to the DNSKEY RRset, which is then signed by key N and all currently active KSKs. (So at this point, the DNSKEY RRset is signed by both key N and its predecessor KSK. If other KSKs were active, it is signed by these as well.) This is the publication time (Tpub); after this the key is said to be published.

Event 3: Before it can be used, the key must be published for long enough to guarantee that any validating resolver that has a copy of

the DNSKEY RRset in its cache will have a copy of the RRset that includes this key: in other words, that any prior cached information about the DNSKEY RRset has expired.

The interval is the publication interval (Ipub) and, for the second or subsequent KSKs in the zone, is given by:

Ipub = DprpC + TTLkey

... where DprpC is the propagation delay for the child zone (the zone containing the KSK being rolled) and TTLkey the TTL for the DNSKEY RRset. The time at which this occurs is the key's ready time, Trdy, given by:

Trdy = Tpub + Ipub

(The case of introducing the first KSK into the zone is discussed in Section 3.3.5.)

Event 4: At some later time, the DS record corresponding to the new KSK is submitted to the parent zone for publication. This time is the submission time, Tsub.

Event 5: The DS record is published in the parent zone. As this is the point at which all information for authentication - both DNSKEY and DS record - is available in the two zones, in analogy with other rollover methods, this is called the activation time of the key (Tact):

Tact = Tsub + Dreg

... where Dreg is the registration delay, the time taken after the DS record has been received by the parent zone manager for it to be placed in the zone. (Parent zones are often managed by different entities, and this term accounts for the organisational overhead of transferring a record.)

Event 6: While key N is active, thought needs to be given to its successor (key N+1). At some time before the scheduled end of the KSK lifetime, the successor KSK is published in the zone. (As before, this means that the DNSKEY RRset is signed by both the current and successor KSK.) This time is the publication time of the successor key, TpubS, given by:

TpubS <= Tact + Lksk - Dreg - Ipub

... where Lksk is the scheduled lifetime of the KSK.

Event 7: After an interval Ipub, key N+1 becomes ready (in that all caches that have a copy of the DNSKEY RRset have a copy of this key). This time is the ready time of the successor (TrdyS).

Event 8: At the submission time of the successor (TsubS), the DS record corresponding to key N+1 is submitted to the parent zone.

Event 9: The successor DS record is published in the parent zone and the current DS record withdrawn. The current key is said to be retired and the time at which this occurs is Tret, given by:

Tret = Tact + Lksk

Event 10: Key N must remain in the zone until any caches that contain a copy of the DS RRset have a copy containing the new DS record. This interval is the retire interval, given by:

Iret = DprpP + TTLds

... where DprpP is the propagation delay in the parent zone and TTLds the TTL of a DS record in the parent zone.

As the key is no longer used for anything, is said to be dead. This point is the dead time (Tdea), given by:

Tdea = Tret + Iret

Event 11: At some later time, key N is removed from the zone's DNSKEY RRset (at the remove time Trem); the key is now said to be removed.

Trem >= Tdea

3.3.2. Double-DS Method

The timeline for a double-DS rollover is shown below. The diagram follows the convention described in Section 3.2.1

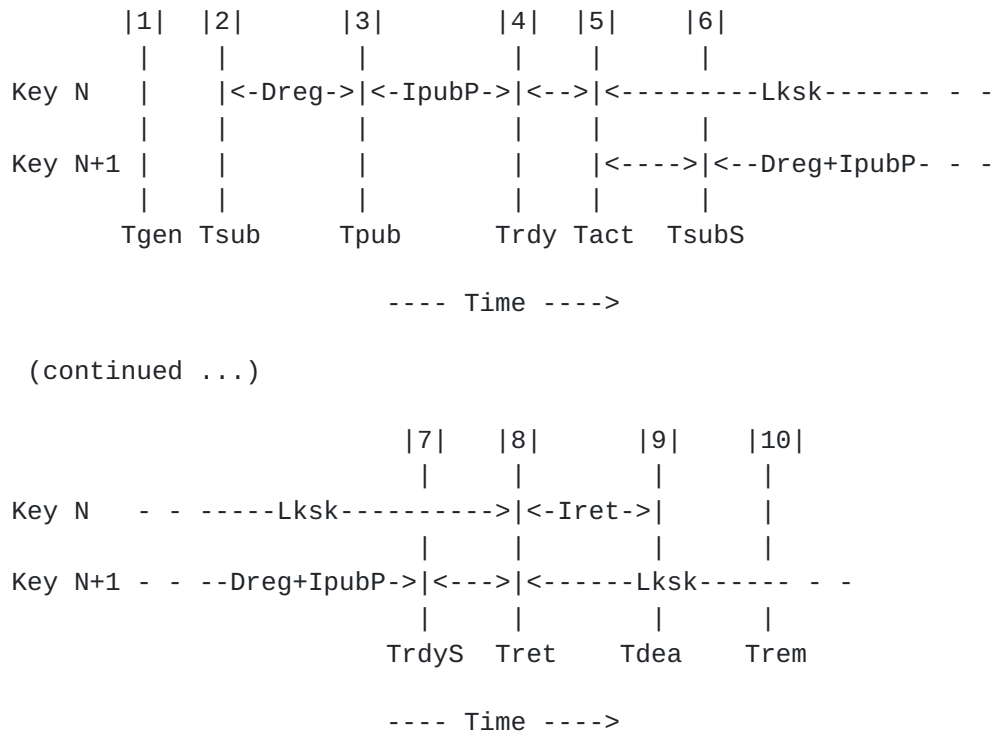

Figure 5: Timeline for a Double-DS KSK rollover.

Event 1: Key N is generated at the generate time (Tgen). Although there is no reason why the key cannot be generated immediately prior to its publication in the zone (Event 2), some implementations may find it convenient to create a pool of keys in one operation and draw from that pool as required. For this reason, it is shown as a separate event. Keys that are available for use but not published are said to be generated.

Event 2: The DS RR is submitted to the parent zone for publication. This time is the submission time, Tsub.

Event 3: After the registration delay, Dreg, the DS record is published in the parent zone. This is the publication time Tpub, given by:

Tpub = Tsub + Dreg

Event 4: At some later time, any cache that has a copy of the DS RRset will have a copy of the DS record for key N. At this point, key N, if introduced into the DNSKEY RRset, could be used to validate the zone. For this reason, this time is known as the key's ready time, Trdy, and is given by:

Trdy = Tpub + IpubP

IpubP is the parent publication interval and is given by the expression:

IpubP = DprpP + TTLds

... where DprpP is the propagation delay for the parent zone and TTLds the TTL assigned to DS records in that zone.

Event 5: At some later time, the key rollover takes place and the new key (key N) is introduced into the DNSKEY RRset and used to sign that.

As both the old and new DS records have been in the parent zone long enough to ensure that they are in caches that contain the DS RRset, the zone can be authenticated throughout the rollover - the validating resolver either has a copy of the DNSKEY RRset authenticated by the predecessor key, or it has a copy of the updated RRset authenticated with the new key.

This time is key N's activation time (Tact) and at this point the key is said to be active.

Event 6: At some point thought must be given to key replacement. The DS record for the successor key must be submitted to the parent zone at a time such that when the current key is withdrawn, any cache that contains the zone's DS records has data about the DS record of the successor key. The time at which this occurs is the submission time of the successor, given by:

TsubS <= Tact + Lksk - IpubP - Dreg

... where Lksk is the lifetime of key N according to policy.

Event 7: The successor key (key N+1) enters the ready state, i.e., its DS record is now in caches that contain the parent DS RRset. This is the ready time of the successor key, TrdyS.

(The interval between events 6 and 7 for the key N+1 correspond to the interval between events 2 and 4 for key N)

Event 8: When key N has been active for its lifetime (Lksk), it is replaced in the DNSKEY RRset by key N+1; the RRset is then signed with the new key. This is the retire time (Tret) of key N, given by:

Tret = Tact + Lksk

Event 9: At some later time, all copies of the old DNSKEY RRset have expired from caches and the old DS record is no longer needed. In analogy with other rollover methods, this is called the dead time, Tdea, and is given by:

Tdea = Tret + Iret

... where Iret is the retire interval, given by:

Iret = DprpC + TTLkey

As before, this term includes DprpC, the time taken to propagate the RRset change through the master-slave hierarchy of the child zone and TTLkey, the time taken for the DNSKEY RRset to expire from caches.

Event 10: At some later time, the DS record is removed from the parent zone. In analogy with other rollover methods, this is the removal time (Trem), given by:

Trem >= Tdea

3.3.3. Double-RRset Method

The timeline for a double-RRset rollover is shown below. The diagram follows the convention described in Section 3.2.1

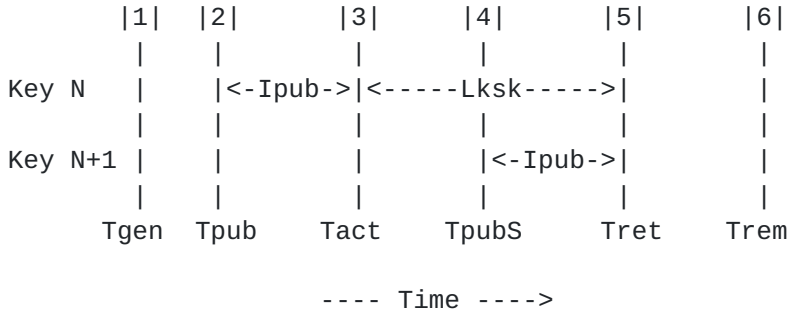

Figure 6: Timeline for a Double-RRset KSK rollover.

Event 1: Key N is generated at the generate time (Tgen). Although there is no reason why the key cannot be generated immediately prior to its publication in the zone (Event 2), some implementations may find it convenient to create a pool of keys in one operation and draw from that pool as required. For this reason, it is shown as a separate event. Keys that are available for use but not published

are said to be generated.

Event 2: The key is added to and used for signing the DNSKEY RRset and is thereby published in the zone. At the same time the corresponding DS record is submitted to the parent zone for publication. This time is the publish time (Tpub) and the key is now said to be published.

Event 3: At some later time, the DS record is published in the parent zone and at some time after that, the updated information has reached all caches: any cache that holds a DNSKEY RRset from the child zone will have a copy that includes the new KSK, and any cache that has a copy of the parent DS RRset will have a copy that includes the new DS record.

The time at which this occurs is called the activation time of the new KSK (Tact), given by:

Tact = Tpub + Ipub

... where Ipub is the publication interval, given by:

Ipub = max(IpubP, IpubC),

IpubP being the publication interval in the parent zone and IpubC the publication interval in the child zone. The parent zone's publication interval is given by:

IpubP = Dreg + DprpP + TTLds

where Dreg is the registration delay, the time taken for the DS record to be published in the parent zone. DprpP is the parent zone's propagation delay and TTLds is the TTL of the DS record in that zone.

The child's publication interval is given by a similar equation:

IpubC = DprpC + TTLkey

... where DprpC is the propagation delay in the child zone and TTLkey the TTL of a DNSKEY record.

Event 4: At some point we need to give thought to key replacement. The successor key (key N+1) must be introduced into the zone (and its DS record submitted to the parent) at a time such that it becomes active when the current key has been active for its lifetime, Lksk. This time is TpubS, the publication time of the successor key, and is given by:

TpubS <= Tact + Lksk - Ipub

... where Lksk is the lifetime of the KSK.

Event 5: Key N+1's DNSKEY and DS records are in any caches that contain the child zone DNSKEY and/or the parent zone DS RR, and so the zone can be validated with the new key. This is the activation time of the successor key (TactS) and by analogy with other rollover methods, it is also the retire time of the current key. Since at this time the zone can be validated by the successor key, there is no reason to keep the current key in the zone and the time can also be regarded as the current key's dead time. Thus:

Tret = Tdea = TactS = Tact + Lksk

Event 6: At some later time, the key N's DS and DNSKEY records are removed from their respective zones. In analogy with other rollover methods, this is the removal time (Trem), given by:

Trem >= Tdea

3.3.4. Interaction with Configured Trust Anchors

Although the preceding sections have been concerned with rolling KSKs where the trust anchor is a DS record in the parent zone, zone managers may want to take account of the possibility that some validating resolvers may have configured trust anchors directly.

Rolling a configured trust anchor is dealt with in [RFC5011]. It requires introducing the KSK to be used as the trust anchor into the zone for a period of time before use, and retaining it (with the "revoke" bit set) for some time after use.

3.3.4.1. Addition of KSK

When the new key is introduced, the publication interval (Ipub) in the Double-Signature and Double-RRset methods should also be subject to the condition:

Ipub >= Dprp + max(Itrp, TTLkey)

... where the right hand side of the expression is the time taken for the change to propagate to all nameservers for the zone plus the "trust point" interval. This latter term is the interval required to guarantee that a resolver configured for the automatic update of keys from a particular trust point will see at least two validated DNSKEY RRsets containing the new key (a requirement from [RFC5011], section 2.4.1). It is defined by the expression:

Itrp >= (2 * queryInterval) + (n * retryTime)

... where queryInterval and retryTime are as defined in section 2.3 of [RFC5011]. "n" is the total number of retries needed by the resolver during the two attempts to get the DNSKEY RRset.

The first term of the expression (2 * queryInterval) represents the time to obtain two validated DNSKEY RRsets. The second term (n * retryTime) is a safety margin, with the value of "n" reflecting the degree of confidence in the communication between a resolver and the trust point.

In the Double-DS method, instead of swapping the KSK RRs in a single step, there must now be a period of overlap. In other words, the new KSK must be introduced into the zone at least:

DprpC + max(Itrp, TTLkey)

... before the switch is made.

3.3.4.2. Removal of KSK

The timeline for the removal of the key in all methods is modified by introducing a new state, "revoked". When the key reaches its dead time, instead of being declared "dead", it is revoked; the "revoke" bit is set in the published DNSKEY RR, and the DNSKEY RRset re-signed with the current and revoked keys. The key is maintained in this state for the "revoke" interval, Irev, given by:

Irev >= 30 days

... 30 days being the [RFC5011] remove hold-down time. After this time, the key is dead and can be removed from the zone.

3.3.5. Introduction of First Keys

There are no timing considerations associated with the introduction of the first keys into a zone other that they must be introduced and the zone validly signed before a chain of trust to the zone is created.

This is important: in the case of a secure parent, it means ensuring that the DS record is not published in the parent zone until there is no possibility that a validating resolver can obtain the record yet is not able to obtain the corresponding DNSKEY. In the case of an insecure parent, i.e., the initial creation of a new security apex, it is not possible to guarantee this. It is up to the operator of the validating resolver to wait for the new KSK to appear at all

servers for the zone before configuring the trust anchor.

4. Standby Keys

Although keys will usually be rolled according to some regular schedule, there may be occasions when an emergency rollover is required, e.g., if the active key is suspected of being compromised. The aim of the emergency rollover is to allow the zone to be re-signed with a new key as soon as possible. As a key must be in the ready state to sign the zone, having at least one additional key (a standby key) in this state at all times will minimise delay.

In the case of a ZSK, a standby key only really makes sense with the Pre-Publication method. A permanent standby DNSKEY RR should be included in the zone or successor keys could be introduced as soon as possible after a key becomes active. Either way results in one or more additional ZSKs in the DNSKEY RRset that can immediately be used to sign the zone if the current key is compromised.

(Although in theory the mechanism could be used with both the Double-Signature and Double-RRSIG methods, it would require pre-publication of the signatures. Essentially, the standby key would be permanently active, as it would have to be periodically used to renew signatures. Zones would also permanently require two sets of signatures.)

It is also possible to have a standby KSK. The Double-Signature method requires that the standby KSK be included in the DNSKEY RRset; rolling the key then requires just the introduction of the DS record in the parent. Note that the standby KSK should also be used to sign the DNSKEY RRset. As the RRset and its signatures travel together, merely adding the KSK without using it to sign the DNSKEY RRset does not provide the desired time saving: for a KSK to be used in a rollover the DNSKEY RRset must be signed with it, and this would introduce a delay while the old RRset (not signed with the new key) expires from caches.

The idea of a standby KSK in the Double-RRset rollover method effectively means having two active keys (as the standby KSK and associated DS record would both be published at the same time in their respective zones).

Finally, in the Double-DS method of rolling a KSK, it is not a standby key that is present, it is a standby DS record in the parent zone.

Whatever algorithm is used, the standby item of data can be included in the zone on a permanent basis, or be a successor introduced as

early as possible.

5. Algorithm Considerations

The preceding sections have implicitly assumed that all keys and signatures are created using a single algorithm. However, [RFC4035] (section 2.4) states that "There MUST be an RRSIG for each RRset using at least one DNSKEY of each algorithm in the zone apex DNSKEY RRset".

Except in the case of an algorithm rollover - where the algorithms used to create the signatures are being changed - there is no relationship between the keys of different algorithms. This means that they can be rolled independently of one another. In other words, the key rollover logic described above should be run separately for each algorithm; the union of the results is included in the zone, which is signed using the active key for each algorithm.

6. Limitation of Scope

This document represents current thinking at the time of publication. However, the subject matter is evolving and it is more than likely that this document will need to be revised in the future.

Some of the techniques and ideas that DNSSEC operators are considering differ from this those described in this document. Of particular interest are alternatives to the strict split into KSK and ZSK key roles and the consequences for rollover logic from partial signing (i.e., when the new key initially only signs a fraction of the zone while leaving other signatures generated by the old key in place).

Furthermore, as noted in section 5, this document covers only rolling keys of the same algorithm: it does not cover transitions between algorithms. The timing issues associated with algorithm rollovers will require a separate document.

The reader is therefore reminded that DNSSEC is, as of date of publication, in the early stages of deployment, and best practices may further develop over time.

7. Summary

For ZSKs, "Pre-Publication" is generally considered to be the preferred way of rolling keys. As shown in this document, the time

taken to roll is wholly dependent on parameters under the control of the zone manager.

In contrast, "Double-RRset" is the most efficient method for KSK rollover due to the ability to have new DS records and DNSKEY RRsets propagate in parallel. The time taken to roll KSKs may depend on factors related to the parent zone if the parent is signed. For zones that intend to comply with the recommendations of [RFC5011], in virtually all cases the rollover time will be determined by the RFC5011 "add hold-down" and "remove hold-down" times. It should be emphasized that this delay is a policy choice and not a function of timing values and that it also requires changes to the rollover process due to the need to manage revocation of trust anchors.

Finally, the treatment of emergency key rollover is significantly simplified by the introduction of standby keys as standard practice during all types of rollovers.

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

This document does not introduce any new security issues beyond those already discussed in [RFC4033], [RFC4034], [RFC4035] and [RFC5011].

10. Acknowledgements

The authors gratefully acknowledge help and contributions from Roy Arends, Matthijs Mekking and Wouter Wijngaards.

11. Normative References

[I-D.ietf-dnsop-rfc4641bis]
Kolkman, O. and M. Mekking, "DNSSEC Operational Practices, Version 2", draft-ietf-dnsop-rfc4641bis-11 (work in progress), April 2012.

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

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements",

RFC 4033, March 2005.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.

[RFC5011] StJohns, M., "Automated Updates of DNS Security (DNSSEC) Trust Anchors", RFC 5011, September 2007.

Appendix A. List of Symbols

The document defines a number of symbols, all of which are listed here. All are of the form:

All symbols used in the text are of the form:

<TYPE><id><INST>

where:

<TYPE> is an upper-case character indicating what type the symbol is. Defined types are:

D delay: interval that is a feature of the process

I interval between two events

L lifetime: interval set by the zone manager

T a point in time

TTL TTL of a record

I and T and TTL are self-explanatory. Like I, both D and L are time periods, but whereas I values are intervals between two events (even if the events are defined in terms of the interval, e.g., the dead time occurs "retire interval" after the retire time), D and L are fixed intervals: a "D" interval (delay) is a feature of the process, probably outside control of the zone manager, whereas an "L" interval (lifetime) is chosen by the zone manager and is a feature of policy.

<id> is lower-case and defines what object or event the variable is related to, e.g.,

act activation

pub publication

ret retire

Finally, <INST> is a capital letter that distinguishes between the same variable applying to different instances of an object and is one of:

C child

G signature

P parent

S successor

The list of variables used in the text is:

Dprp Propagation delay. The amount of time for a change made at a master nameserver to propagate to all the slave nameservers.

DprpC Propagation delay in the child zone.

DprpP Propagation delay in the parent zone.

Dreg Registration delay: the time taken for a DS record submitted to a parent zone to appear in it. As a parent zone is often managed by a different organisation to that managing the child zone, the delays associated with passing data between zones is captured by this term.

Dsgn Signing delay. After the introduction of a new ZSK, the amount of time taken for all the RRs in the zone to be signed with it.

Ipub Publication interval. The amount of time that must elapse after the publication of a key before it can be assumed that any resolvers that have the DNSKEY RRset cached have a copy of this key.

IpubC Publication interval in the child zone.

IpubG Publication interval for the signature created by a ZSK: the amount of time that must elapse after the signature has been created before it can be assumed that any resolves that have the RRset and RRSIG cached have a copy of this signature.

IpubP Publication interval in the parent zone.

Iret Retire interval. The amount of time that must elapse after a key enters the retire state for any signatures created with it to be purged from validating resolver caches.

Irev Revoke interval. The amount of time that a KSK must remain published with the revoke bit set to satisfy [RFC5011] considerations.

Itrp Trust-point interval. The amount of time that a trust anchor must be published for to guarantee that a resolver configured for an automatic update of keys will see the new key at least twice.

Lksk Lifetime of a key-signing key. This is the intended amount of time for which this particular KSK is regarded as the active KSK. Depending on when the key is rolled-over, the actual lifetime may be longer or shorter than this.

Lzsk Lifetime of a zone-signing key. This is the intended amount of time for which the ZSK is used to sign the zone. Depending on when the key is rolled-over, the actual lifetime may be longer or shorter than this.

Tact Activation time of the key; the time at which the key is regarded as the principal key for the zone.

TactS Activation time of the successor key.

Tdea Dead time of a key. Applicable only to ZSKs, this is the time at which any record signatures held in validating resolver caches are guaranteed to be created with the successor key.

Tgen Generate time of a key. The time that a key is created.

Tpub Publication time of a key. The time that a key appears in a zone for the first time.

TpubS Publication time of the successor key.

Trem Removal time of a key. The time at which a key is removed from the zone.

Tret Retire time of a key. The time at which a successor key starts being used to sign the zone.

Trdy Ready time of a key. The time at which it can be guaranteed that validating resolvers that have key information from this zone cached have a copy of this key in their cache. (In the case of KSKs, should the validating resolvers also have DS information from the parent zone cached, the cache must include information about the DS record corresponding to the key.)

TrdyS Ready time of a successor key.

Tsub Submission time - the time at which the DS record of a KSK is submitted to the parent.

TsubS Submission time of the successor key.

TTLds Time to live of a DS record (in the parent zone).

TTLkey Time to live of a DNSKEY record. (By implication, this is also the time to live of the signatures on the DNSKEY RRset.)

TTLsig The maximum time to live of all the RRSIG records in the zone that were created with the ZSK.

Appendix B. Change History (To be removed on publication)

- draft-ietf-dnsop-dnssec-key-timing-03
  * Clarifications of and corrections to wording (Marc Lampo, Alfred Hoenes).
  * Updated timings related to trust anchor interaction (Matthijs Mekking).
  * Updated RFC 4641 reference to 4641bis (Alfred Hoenes).
  * Moved change history to end of document (Alfred Hoenes).

- draft-ietf-dnsop-dnssec-key-timing-02
  * Significant re-wording of some sections.
  * Removal of events noting change of state of predecessor key from ZSK Double-RRSIG and Double-Signature methods.
  * Change order of bullet points (and some wording) in section 1.1.

* Remove discussion of advantages and disadvantages of key roll methods from section 2: draft is informative and does not give recommendations.
* Removal of discussion of upper limit to retire time relationship to signature lifetime.
* Remove timing details of first key in the zone and move discussion of first signing of a zone to later in the document). (Matthijs Mekking)
* Removal of redundant symbols from Appendix A.

- draft-ietf-dnsop-dnssec-key-timing-01
  * Added section on limitation of scope.

- draft-ietf-dnsop-dnssec-key-timing-00
  * Change to author contact details.

- draft-morris-dnsop-dnssec-key-timing-02
  * General restructuring.
  * Added descriptions of more rollovers (IETF-76 meeting).
  * Improved description of key states and removed diagram.
  * Provided simpler description of standby keys.
  * Added section concerning first key in a zone.
  * Moved [RFC5011] to a separate section.
  * Various nits fixed (Alfred Hoenes, Jeremy Reed, Scott Rose, Sion Lloyd, Tony Finch).

- draft-morris-dnsop-dnssec-key-timing-01
  * Use latest boilerplate for IPR text.
  * List different ways to roll a KSK (acknowledgements to Mark Andrews).
  * Restructure to concentrate on key timing, not management procedures.
  * Change symbol notation (Diane Davidowicz and others).
  * Added key state transition diagram (Diane Davidowicz).
  * Corrected spelling, formatting, grammatical and style errors (Diane Davidowicz, Alfred Hoenes and Jinmei Tatuya).
  * Added note that in the case of multiple algorithms, the signatures and rollovers for each algorithm can be considered as more or less independent (Alfred Hoenes).
  * Take account of the fact that signing a zone is not atomic (Chris Thompson).
  * Add section contrasting pre-publication rollover with double signature rollover (Matthijs Mekking).
  * Retained distinction between first and subsequent keys in definition of initial publication interval (Matthijs Mekking).

- draft-morris-dnsop-dnssec-key-timing-00
  Initial draft.

Authors' Addresses

Stephen Morris
Internet Systems Consortium
950 Charter Street
Redwood City, CA 94063
USA

Phone: +1 650 423 1300
Email: stephen@isc.org

Johan Ihren
Netnod
Franzengatan 5
Stockholm, SE-112 51
Sweden

Phone: +46 8615 8573
Email: johani@autonomica.se

John Dickinson
Sinodun Internet Technologies Ltd
Stables 4 Suite 11, Howbery Park
Wallingford, Oxfordshire OX10 8BA
UK

Phone: +44 1491 818120
Email: jad@sinodun.com