Internet Engineering Task Force (IETF)
Request for Comments: 7583
Category: Informational
ISSN: 2070-1721

S. Morris
ISC
J. Ihren
Netnod
J. Dickinson
Sinodun
W. Mekking
Dyn

October 2015

DNSSEC Key Rollover Timing Considerations

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 document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7583.

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 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 "bootstrapping" 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 [RFC6781]. In contrast, this document focuses on the low-level timing detail of two classes of operations described there, the rollover of Zone Signing Keys (ZSKs), and the rollover of Key Signing Keys (KSKs).

Note that the process for the introduction of keys into a zone is different from that of rolling a key; see Section 3.3.5 for more information.

1.2. Types of Keys

Although DNSSEC validation treats all keys equally, [RFC4033] recognizes the broad classification of ZSKs and KSKs. 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 an authoritative server. The rollover process needs to happen in such a way that the information is consistent at all times during the rollover. 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.

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. Limitation of Scope

This document represents current thinking at the time of publication. However, the subject matter is evolving and it is not possible for the document to be comprehensive. In particular, it does not cover:

- o Rolling a key that is used as both a ZSK and KSK.
- o Algorithm rollovers. Only the rolling of keys of the same algorithm is described here: not transitions between algorithms.
- o Changing TTLs.

Algorithm rollover is excluded from the document owing to the need for there to be an RRSIG for at least one DNSKEY of each algorithm in the DNSKEY RRset [RFC4035]. This introduces additional constraints on rollovers that are not considered here. Such considerations do not apply where other properties of the key, such as key length, are changed during the rollover: the DNSSEC protocol does not impose any restrictions in these cases.

Also excluded from consideration is the effect of changing the Time to Live (TTL) of records in a zone. TTLs can be changed at any time, but doing so around the time of a key rollover may have an impact on event timings. In the timelines below, it is assumed that TTLs are constant.

2. Rollover Methods

2.1. ZSK Rollovers

For ZSKs, the issue for the zone operator/signer is to ensure that any caching validator that has access to a particular signature also has access to the corresponding valid ZSK.

A ZSK can be rolled in one of three ways:

- o Pre-Publication: described in [RFC6781], 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. During the re-signing process (which may or may not be atomic depending on how the zone is managed), it doesn't matter with which key an RRSIG record retrieved by a resolver was created; 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 be removed from the DNSKEY RRset.

- Double-Signature: also mentioned in [RFC6781], 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 RRSIGs allows at least one signature to be validated.

  Once the signing process is complete and enough time has elapsed to make sure that all validators that have the DNSKEY and signatures in cache have both the old and new information, the old key and signatures can be removed from the zone. As before, during this period any combination of DNSKEY RRset and RRSIGs 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 the 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. It is also the fastest, but suffers from increasing the size of the zone and the size of responses.

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. It does however keep the zone and response sizes to a minimum.

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. However, it has the disadvantage of the Pre-Publication method in terms of time taken to perform the rollover, the disadvantage of the Double-Signature rollover in terms of zone and response sizes, and none of the advantages of either. For these reasons, it is unlikely to be used in any real-world situations and so will not be considered further in this document.

2.2. KSK Rollovers

In the KSK case, there should be no problem with a caching validator not having access to a signature created with a valid KSK. 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 due to either 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 the need to interact with the parent 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-KSK: 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.

o 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.

o 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 is 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 records are removed.

In essence, Double-KSK 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 is removed. It limits interactions with the parent to a minimum but, for the duration of the rollover, the size of the DNSKEY RRset is increased.

With Double-DS, the order of operations is the other way around: introduce the new DS, change the DNSKEY, then remove the old DS. The size of the DNSKEY RRset is kept to a minimum, but two interactions are required with the parent.

Finally, Double-RRset is the fastest way to roll the KSK, but has the drawbacks of both of the other methods: a larger DNSKEY RRset and two interactions with the parent.

3. Key Rollover Timelines

3.1. Key States

DNSSEC validation requires both the DNSKEY and information created from it (referred to as "associated data" in this section). In the case of validation of an RR, the data associated with the key is the corresponding RRSIG. Where there is a need to validate a chain of trust, the associated data is the DS record.

During the rolling process, keys move through different states. The defined states are:

Generated Although keys may be created immediately prior to first use, some implementations may find it convenient to create a pool of keys in one operation and draw from it as required. (Note: such a pre-generated pool must be secured against surreptitious use.) In the timelines below, before the first event, the keys are considered to be created but not yet used: they are said to be in the "Generated" state.

Published   A key enters the published state when either it or its associated data first appears in the appropriate zone.

Ready   The DNSKEY or its associated data have been published for long enough to guarantee that copies of the key(s) it is replacing (or associated data related to that key) have expired from caches.

Active   The data is starting to be used for validation. In the case of a ZSK, it means that the key is now being used to sign RRsets and that both it and the created RRSIGs appear in the zone. In the case of a KSK, it means that it is possible to use it to validate a DNSKEY RRset as both the DNSKEY and DS records are present in their respective zones. Note that when this state is entered, it may not be possible for validating resolvers to use the data 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 predecessor data instead.

Retired   The data has ceased to be used for validation. In the case of a ZSK, it means that the key is no longer used to sign RRsets. In the case of a KSK, it means that the successor DNSKEY and DS records are in place. In both cases, the key (and its associated data) can be removed as soon as it is safe to do so, i.e., when all validating resolvers are able to use the new key and associated data to validate the zone. However, until this happens, the current key and associated data must remain in their respective zones.

Dead   The key and its associated data are present in their respective zones, but there is no longer information anywhere that requires their presence for use in validation. Hence, they can be removed at any time.

Removed   Both the DNSKEY and its associated data have been removed from their respective zones.

Revoked   The DNSKEY is published for a period with the "revoke" bit set as a way of notifying validating resolvers that have configured it as a trust anchor, as used in [RFC5011], that it is about to be removed from the zone. This state is used when [RFC5011] considerations are in effect (see Section 3.3.4).

3.2. ZSK Rollover 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

In this method, the new key is introduced into the DNSKEY RRset. After enough time to ensure that any cached DNSKEY RRsets contain both keys, the zone is signed using the new key and the old signatures are removed. Finally, when all signatures created with the old key have expired from caches, the old key is removed.

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.

```
                |1|      |2|   |3|   |4|      |5|  |6|      |7|   |8|
                 |        |     |     |        |    |        |     |
   Key N         |<-Ipub->|<--->|<-------Lzsk------>|<-Iret->|<--->|
                 |        |     |     |        |    |        |     |
   Key N+1       |        |     |     |<-Ipub->|<-->|<---Lzsk---- - -
                 |        |     |     |        |    |        |     |
   Key N        Tpub     Trdy  Tact                Tret     Tdea  Trem
   Key N+1                           Tpub     Trdy Tact

                                ---- Time ---->
```

Figure 1: Timeline for a Pre-Publication ZSK Rollover

Event 1: 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 currently active KSKs. The time at which this occurs is the publication time (Tpub), and the key is now said to be published. Note that the key is not yet used to sign records.

Event 2: 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 for a change introduced at the master to replicate to all nameservers. TTLkey is the 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 key N's ready time (Trdy), which is given by:

```
Trdy(N) = Tpub(N) + Ipub
```

Event 3: At some later time, the key starts being used to sign RRsets. This point is the activation time (Tact) and after this, key N is said to be active.

```
Tact(N) >= Trdy(N)
```

Event 4: 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. The publication time of key N+1 depends on the activation time of key N:

```
Tpub(N+1) <= Tact(N) + Lzsk - Ipub
```

Here, Lzsk is the length of time for which a ZSK will be used (the ZSK lifetime). It should be noted that in the diagrams, the actual key lifetime is represented; this may differ slightly from the intended lifetime set by key management policy.

Event 5: 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 6: 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(N) = Tact(N) + Lzsk
```

It is also the activation time of the successor key N+1. Note that operational considerations may cause key N to remain in use for a longer (or shorter) time than the lifetime set by the key management policy.

Event 7: 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 old RRSIG record in the cache, but the old DNSKEY RRset has expired 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 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 retiring key.

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

Tdea(N) = Tret(N) + Iret

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

Event 8: 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 KSK. This time is the removal time (Trem), given by:

Trem(N) >= Tdea(N)

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

3.2.2. Double-Signature Method

In this rollover, a new key is introduced and used to sign the zone; the old key and signatures are retained. Once all cached DNSKEY and/or RRSIG information contains copies of the new DNSKEY and RRSIGs created with it, the old DNSKEY and RRSIGs can be removed from the zone.

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

```
                          |1|           |2|        |3|   |4|
                           |             |          |     |
             Key N         |<-------Lzsk----------->|<--->|
                           |             |          |     |
                           |             |<--Iret-->|     |
                           |             |          |     |
             Key N+1       |             |<----Lzsk------- - -
                           |             |          |     |
             Key N        Tact                     Tdea  Trem
             Key N+1                    Tact

                                  ---- Time ---->
```

Figure 2: Timeline for a Double-Signature ZSK Rollover

Event 1: Key N is added to the DNSKEY RRset and is then used to sign the zone; existing signatures in the zone are not removed. The key is published and active: this is key N’s activation time (Tact), after which the key is said to be active.

Event 2: As the current key (key N) approaches the end of its actual 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 key is now also active.

Tact(N+1) = Tact(N) + Lzsk - Iret

Event 3: 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, required to guarantee that the updated zone information has reached all slave servers. 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(N) = Tact(N+1) + Iret

Event 4: 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(N) >= Tdea(N)

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

3.3.1. Double-KSK Method

In this rollover, the new DNSKEY is added to the zone. After an interval long enough to guarantee that any cached DNSKEY RRsets contain the new DNSKEY, the DS record in the parent zone is changed. After a further interval to allow the old DS record to expire from caches, the old DNSKEY is removed from the zone.

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

```
                       |1|       |2|   |3|      |4|
                        |         |     |        |
       Key N            |<-IpubC->|<--->|<-Dreg->|<-----Lksk--- - -
                        |         |     |        |
       Key N+1          |         |     |        |
                        |         |     |        |
       Key N           Tpub      Trdy  Tsbm     Tact
       Key N+1

                                   ---- Time ---->

               (continued ...)

                   |5|       |6|   |7|      |8|      |9|    |10|
                    |         |     |        |        |       |
       Key N   - - --------------Lksk------->|<-Iret->|<----->|
                    |         |     |        |        |       |
       Key N+1      |<-IpubC->|<--->|<-Dreg->|<--------Lksk----- - -
                    |         |     |        |        |       |
       Key N                                Tret     Tdea    Trem
       Key N+1     Tpub      Trdy  Tsbm     Tact

                               ---- Time (cont.) ---->
```

Figure 3: Timeline for a Double-KSK Rollover

Event 1: Key N is introduced into the zone; it is added to the DNSKEY RRset, which is then signed by 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 of key N (Tpub); after this, the key is said to be published.

Event 2: 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 in the child zone (IpubC) and is given by:

IpubC = 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 N's ready time, Trdy, given by:

    Trdy(N) = Tpub(N) + IpubC

Event 3: 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, Tsbm:

    Tsbm(N) >= Trdy(N)

Event 4: 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 key N (Tact):

    Tact(N) = Tsbm(N) + Dreg

... where Dreg is the registration delay, the time taken after the DS record has been submitted to 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 organizational overhead of transferring a record. In practice, Dreg will not be a fixed time: instead, the end of Dreg will be signaled by the appearance of the DS record in the parent zone.)

Event 5: 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 all KSKs.) This time is the publication time of the successor key N+1, given by:

    Tpub(N+1) <= Tact(N) + Lksk - Dreg - IpubC

... where Lksk is the actual lifetime of the KSK, and Dreg the registration delay.

Event 6: After an interval IpubC, 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 key N+1 (Trdy).

Event 7: At the submission time of the successor key N+1, Tsbm(N+1), the DS record corresponding to key N+1 is submitted to the parent zone.

Event 8: The successor DS record is published in the parent zone and the current DS record withdrawn. Key N is said to be retired and the time at which this occurs is Tret(N), given by:

Tret(N) = Tsbm(N+1) + Dreg

Event 9: 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, it is said to be dead. This point is the dead time (Tdea), given by:

Tdea(N) = Tret(N) + Iret

Event 10: 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(N) >= Tdea(N)

3.3.2. Double-DS Method

In this rollover, the new DS record is published in the parent zone. When any caches that contain the DS RRset contain a copy of the new record, the KSK in the zone is changed. After a further interval for the old DNSKEY RRset to expire from caches, the old DS record is removed from the parent.

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

```
                 |1|       |2|        |3|  |4|     |5|
                  |         |          |    |       |
       Key N      |<-Dreg->|<-IpubP->|<-->|<-------Lksk---- - -
                  |         |          |    |       |
       Key N+1    |         |          |    |       |<--Dreg-- - -
                  |         |          |    |       |
       Key N     Tsbm      Tpub       Trdy Tact
       Key N+1                                     Tsbm
                                ---- Time ---->

              (continued ...)

                      |6|        |7|  |8|      |9|    |10|
                       |          |    |        |      |
       Key N   - ----------Lksk--------->|<-Iret->|<---->|
                       |          |    |        |      |
       Key N+1 - --Dreg-->|<-IpubP->|<-->|<---Lksk------- - -
                       |          |    |        |      |
       Key N                          Tret     Tdea   Trem
       Key N+1            Tpub      Trdy Tact

                                      ---- Time ---->
```

Figure 4: Timeline for a Double-DS KSK Rollover

Event 1: The DS RR is submitted to the parent zone for publication. This time is the submission time, Tsbm.

Event 2: After the registration delay, Dreg, the DS record is published in the parent zone. This is the publication time (Tpub) of key N, given by:

Tpub(N) = Tsbm(N) + Dreg

As before, in practice, Dreg will not be a fixed time. Instead, the end of Dreg will be signaled by the appearance of the DS record in the parent zone.

Event 3: 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 ready time, Trdy, and is given by:

Trdy(N) = Tpub(N) + IpubP

IpubP is the publication interval of the DS record (in the parent zone) 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 4: 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 it. This time is key N’s activation time (Tact) and at this point key N is said to be active:

Tact(N) >= Trdy(N)

Event 5: 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 key N+1, given by:

Tsbm(N+1) <= Tact(N) + Lksk - IpubP - Dreg

... where Lksk is the actual lifetime of key N (which may differ slightly from the lifetime set in the key management policy) and Dreg is the registration delay.

Event 6. After an interval Dreg, the successor DS record is published in the zone.

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.

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. At this point, 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. A validating resolver can authenticate either the old or new KSK.

This time is the retire time (Tret) of key N, given by:

Tret(N) = Tact(N) + Lksk

This is also the activation time of the successor key N+1.

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(N) = Tret(N) + Iret
```

... where Iret is the retire interval of the key, 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(N) >= Tdea(N)
```

3.3.3. Double-RRset Method

In the Double-RRset rollover, the new DNSKEY and DS records are published simultaneously in the appropriate zones. Once enough time has elapsed for the old DNSKEY and DS RRsets to expire from caches, the old DNSKEY and DS records are removed from their respective zones.

The timeline for this rollover is shown below. The diagram follows the convention described in Section 3.2.1.

```
                       |1|       |2|      |3|      |4|    |5|
                        |         |        |        |      |
          Key N         |<-----------Lksk---------->|<---->|
                        |         |        |        |      |
                        |         |<------Ipub----->|      |
                        |         |        |        |      |
                        |         |<-Dreg->|<-Iret->|      |
                        |         |        |        |      |
          Key N+1       |         |        |<----Lksk-------- - -
                        |         |        |        |      |
          Key N        Tact               Tret     Tdea   Trem
          Key N+1                Tpub     Tact

                           ---- Time ---->
```

Figure 5: Timeline for a Double-RRset KSK Rollover

Event 1: The DS and DNSKEY records have appeared in their respective zones and the latter has been used to sign the DNSKEY RRset. The key is published and active: this is key N's activation time (Tact).

Event 2: As the current key (key N) approaches the end of its actual lifetime (Lksk), the successor key (key N+1) is introduced into the zone and is used to sign the DNSKEY RRset. At the same time, the successor DS record is submitted to the parent zone. This is the publication time of the successor key (Tpub):

    Tpub(N+1) <= Tact(N) + Lksk - Ipub

... where Ipub is defined below.

Event 3: After the registration delay (Dreg), the DS record appears in the parent zone. The DNSKEY record is already in the child zone, so with both the new key and its associated data now visible, this is the key's activation time (Tact) and the key is now said to be active.

    Tact(N+1) = Tpub(N+1) + Dreg

Event 4: Before key N and its associated data can be withdrawn, all RRsets in the caches of validating resolvers must contain the new DS and/or DNSKEY. The time at which this occurs is the dead time of key N (Tdea), given by:

    Tdea(N) = Tpub(N+1) + Ipub

Ipub is the time it takes to guarantee that any prior cached information about the DNSKEY and the DS RRsets have expired. For the DNSKEY, this is the publication interval of the child (IpubC). For the DS, the publication interval (IpubP) starts once the record appears in the parent zone, which is Dreg after it has been submitted. Hence:

    Ipub = max(Dreg + IpubP, IpubC)

The parent zone's publication interval is given by:

    IpubP = DprpP + TTLds

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

The child zone'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.

In analogy with other rollovers, we can also define a retire interval -- the interval between a key becoming active and the time at which its predecessor is considered dead. In this case, Iret is given by:

Iret = Ipub - Dreg

In other words, the retire interval of the predecessor key is the greater of the publication interval of the parent, or the publication interval of the child minus the registration delay.

Event 5: 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(N) >= Tdea(N)

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 expression for the publication interval of the DNSKEY (IpubC) in the Double-KSK and Double-RRset methods is modified to:

IpubC >= DprpC + max(Itrp, TTLkey)

... where the right-hand side of the expression now includes the "trust point" interval. This term is the interval required to guarantee that a resolver configured for the automatic update of keys according to [RFC5011] will accept the new key as a new trust point. That interval is given by:

```
Itrp >= queryInterval + AddHoldDownTime + queryInterval
```

... where queryInterval is as defined in Section 2.3 of [RFC5011] and AddHoldDownTime is the Add Hold-Down Time defined in Section 2.4.1 of the same document.

The first term of the expression (queryInterval) represents the time after which all validating resolvers can be guaranteed to have obtained a copy of the DNSKEY RRset containing the new key. Once retrieved, a validating resolver needs to wait for AddHoldDownTime. Providing it does not see a validly signed DNSKEY RRset without the new key in that period, it will treat it as a trust anchor the next time it retrieves the RRset, a process that can take up to another queryInterval (the third term).

However, the expression for queryInterval given in [RFC5011] contains the DNSKEY’s RRSIG expiration interval, a parameter that only the validating resolver can really calculate. In practice, a modified query interval that depends only on TTLkey can be used:

```
modifiedQueryInterval = MAX(1hr, MIN(15 days, TTLkey / 2))
```

(This is obtained by taking the expression for queryInterval in [RFC5011] and assuming a worst case for RRsigExpirationInterval. It is greater than or equal to queryInterval for all values of the expiration time.) The expression above then becomes (after collecting terms):

```
Itrp >= AddHoldDownTime + 2 * modifiedQueryInterval
```

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 >= DprpC + modifiedQueryInterval

As before, DprpC is the time taken for the revoked DNSKEY to propagate to all slave zones, and modifiedQueryInterval is the time after which it can be guaranteed that all validating resolvers that adhere to RFC 5011 have retrieved a copy of the DNSKEY RRset containing the revoked key.

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.

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 chain of trust or "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 minimize 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-KSK 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, Section 2.2 of [RFC4035] requires that there 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. Summary

For ZSKs, the Pre-Publication method 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, the Double-RRset method is the most efficient 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 many cases, the rollover time will be determined by the times defined by RFC 5011. 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.

7. Security Considerations

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

8. Normative References

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <http://www.rfc-editor.org/info/rfc4033>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <http://www.rfc-editor.org/info/rfc4034>.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <http://www.rfc-editor.org/info/rfc4035>.

[RFC5011] StJohns, M., "Automated Updates of DNS Security (DNSSEC) Trust Anchors", STD 74, RFC 5011, DOI 10.17487/RFC5011, September 2007, <http://www.rfc-editor.org/info/rfc5011>.

[RFC6781] Kolkman, O., Mekking, W., and R. Gieben, "DNSSEC Operational Practices, Version 2", RFC 6781, DOI 10.17487/RFC6781, December 2012, <http://www.rfc-editor.org/info/rfc6781>.

Appendix A. List of Symbols

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

<TYPE><id><ZONE>

where:

<TYPE> is an uppercase 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, T, and TTL are self-explanatory. Like I, both D and L are time periods, but whereas I values are intervals between two events, a "D" interval (delay) is a feature of the process, probably outside control of the zone manager, and an "L" interval (lifetime) is chosen by the zone manager and is a feature of policy.

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

act activation

pub publication

ret retire

<ZONE> is an optional uppercase letter that distinguishes between the same variable applied to different zones and is one of:

C child

P parent

Within the rollover descriptions, times may have a number in parentheses affixed to their end indicating the instance of the key to which they apply, e.g., Tact(N) is the activation time of key N, Tpub(N+1) the publication time of key N+1 etc.

The list of variables used in the text given below.

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 organization than that managing the child zone, the delays associated with passing data between organizations 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 DNSKEY and/or its associated data before it can be assumed that any resolvers that have the relevant RRset cached have a copy of the new information.

IpubC — Publication interval in the child zone.

IpubP — Publication interval in the parent zone.

Iret — Retire interval. The amount of time that must elapse after a DNSKEY or associated data enters the retire state for any dependent information (e.g., RRSIG for a ZSK) 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 considerations of [RFC5011].

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

Lksk Lifetime of a KSK. This is the actual 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 the intended key lifetime indicated by management policy.

Lzsk Lifetime of a ZSK. This is the actual 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 the intended key lifetime indicated by management policy.

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

Tdea Dead time. The time at which any information held in validating resolver caches is guaranteed to contain information related to the successor key. At this point, the current key and its associated information are not longed required for validation purposes.

Tpub Publication time. The time that the key or associated data appears in the zone for the first time.

Trem Removal time. The time at which the key and its associated information starts being removed from their respective zones.

Tret Retire time. The time at which successor information starts being used.

Trdy Ready time. The time at which it can be guaranteed that validating resolvers that have information about the key and/or associated data cached have a copy of the new information.

Tsbm Submission time. The time at which the DS record of a KSK is submitted to the parent zone.

TTLds Time to live of a DS record.

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.

Acknowledgements

The authors gratefully acknowledge help and contributions from Roy Arends, Tim Wicinski, and Wouter Wijngaards.

Authors' Addresses

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

Email: stephen@isc.org
URI: http://www.isc.org

Johan Ihren
Netnod
Franzengatan 5
Stockholm SE-112 51
Sweden

Email: johani@netnod.se
URI: http://www.netnod.se

John Dickinson
Sinodun Internet Technologies Ltd
Magdalen Centre
Oxford Science Park
Robert Robertson Avenue
Oxford, Oxfordshire OX4 4GA
United Kingdom

Email: jad@sinodun.com
URI: http://www.sinodun.com

W. (Matthijs) Mekking
Dyn, Inc.
150 Dow St
Manchester NH 03101
United States

Email: mmekking@dyn.com
URI: https://www.dyn.com