

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: September 16, 2011

F. Ljunggren  
Kirei AB  
A-M. Eklund-Lowinder  
.SE  
T. Okubo  
ICANN  
March 15, 2011

DNSSEC Policy & Practice Statement Framework  
draft-ietf-dnsop-dnssec-dps-framework-04

Abstract

This document presents a framework to assist writers of DNSSEC Policy and Practice Statements such as Domain Managers and Zone Operators on both the top-level and secondary level, who is managing and operating a DNS zone with Security Extensions (DNSSEC) implemented.

In particular, the framework provides a comprehensive list of topics that should be considered for inclusion into a DNSSEC Policy definition and Practice Statement.

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 September 16, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	4
1.1. Background . . . . .	4
1.2. Purpose . . . . .	5
1.3. Scope . . . . .	5
2. Definitions . . . . .	5
3. Concepts . . . . .	7
3.1. DP . . . . .	7
3.2. DPS . . . . .	7
3.3. Relationship between DNSSEC Policy and Practice Statement . . . . .	8
3.4. Set of Provisions . . . . .	9
4. Contents of a set of provisions . . . . .	10
4.1. Introduction . . . . .	10
4.1.1. Overview . . . . .	10
4.1.2. Document Name and Identification . . . . .	11
4.1.3. Community and Applicability . . . . .	11
4.1.4. Specification Administration . . . . .	11
4.2. Publication and Repositories . . . . .	11
4.3. Operational Requirements . . . . .	12
4.3.1. Meaning of domain names . . . . .	12
4.3.2. Activation of DNSSEC for child zone . . . . .	12
4.3.3. Identification and authentication of child zone manager . . . . .	12
4.3.4. Registration of delegation signer (DS) resource records . . . . .	12
4.3.5. Method to prove possession of private key . . . . .	12
4.3.6. Removal of DS resource records . . . . .	12
4.4. Facility, Management and Operational Controls . . . . .	12
4.4.1. Physical Controls . . . . .	13
4.4.2. Procedural Controls . . . . .	13
4.4.3. Personnel Controls . . . . .	14
4.4.4. Audit Logging Procedures . . . . .	14
4.4.5. Compromise and Disaster Recovery . . . . .	15
4.4.6. Entity termination . . . . .	16
4.5. Technical Security Controls . . . . .	16
4.5.1. Key Pair Generation and Installation . . . . .	16
4.5.2. Private Key Protection and Cryptographic Module Engineering Controls . . . . .	17

4.5.3.	Other Aspects of Key Pair Management . . . . .	18
4.5.4.	Activation data . . . . .	18
4.5.5.	Computer Security Controls . . . . .	18
4.5.6.	Network Security Controls . . . . .	19
4.5.7.	Timestamping . . . . .	19
4.5.8.	Life Cycle Technical Controls . . . . .	19
4.6.	Zone Signing . . . . .	20
4.6.1.	Key lengths and algorithms . . . . .	20
4.6.2.	Authenticated denial of existence . . . . .	20
4.6.3.	Signature format . . . . .	20
4.6.4.	Zone Signing Key Roll-Over . . . . .	20
4.6.5.	Key Signing Key Roll-Over . . . . .	20
4.6.6.	Signature life-time and re-signing frequency . . . . .	20
4.6.7.	Verification of Zone Signing Key Set . . . . .	20
4.6.8.	Verification of resource records . . . . .	21
4.6.9.	Resource records time-to-live . . . . .	21
4.7.	Compliance Audit . . . . .	21
4.7.1.	Frequency of entity compliance audit . . . . .	21
4.7.2.	Identity/qualifications of auditor . . . . .	21
4.7.3.	Auditor's relationship to audited party . . . . .	21
4.7.4.	Topics covered by audit . . . . .	21
4.7.5.	Actions taken as a result of deficiency . . . . .	22
4.7.6.	Communication of results . . . . .	22
4.8.	Legal Matters . . . . .	22
5.	Outline of a set of provisions . . . . .	22
6.	Security Considerations . . . . .	25
7.	Acknowledgements . . . . .	25
8.	References . . . . .	25
8.1.	Normative References . . . . .	25
8.2.	Informative References . . . . .	26
	Authors' Addresses . . . . .	26

## 1. Introduction

### 1.1. Background

The DNS was not originally designed with strong security mechanisms to provide integrity and authenticity of DNS data. Over the years, a number of vulnerabilities have been discovered that threaten the reliability and trustworthiness of the system.

The Domain Name System Security Extensions (DNSSEC, [RFC4033], [RFC4034], [RFC4035]) is a set of IETF specifications which addresses these vulnerabilities by adding data origin authentication, data integrity verification and authenticated denial of existence capabilities to the Domain Name System, using public key cryptography. In short, DNSSEC provides a way for software to validate that Domain Name System (DNS) data has not been modified during transit.

To provide means for the relying parties to evaluate the strength and security of the DNSSEC chain of trust, an entity operating a DNSSEC enabled zone may choose to publish a DNSSEC Practice Statement (DPS), comprising statements of critical security controls and procedures relevant for scrutinizing the trustworthiness of the system. The DPS may also identify one or more DNSSEC Policies which are supported, explaining how it meets the requirements of each Policy.

Even though this document is heavily inspired by the Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework [RFC3647], and large parts drawn from that document, the properties and structure of the DNSSEC PKI is fundamentally different from the X.509 PKI.

In the DNSSEC PKI there is no central control of assurance or trust levels. Each zone manager may select their own way of managing keys and operations, and there is no necessity to perform any coordination of security practices between different zones in the DNS. The degree to which a relying party can trust the binding embodied in the DNSSEC chain of trust is dependent on the weakest link of that chain. This implies that the security of zones is generally more critical higher up in the DNS hierarchy.

Another significant difference is that the DPS is focused only on stating the security posture of a zone, and not the entire domain name system. Moreover, the DNS is of a almost ubiquitous nature and completely open. There exists no agreements with the relying (third) parties, which is all entities relying on signed responses from the DNS.

### 1.2. Purpose

The purpose of this document is twofold. Firstly, the document aims to explain the concept of a DNSSEC Policy (DP) and a DNSSEC Practice Statement (DPS), and describe the relationship between a DP and a DPS. Second, this document aims to present a framework to encourage and assist writers of Policies and Practice Statements in creating heterogenous and comparable documents. In particular, the framework identifies the elements that should be considered in formulating a DP/DPS. It does not, however, define a particular Policy or Practice Statement, nor does it seek to provide legal advice or recommendations as to the contents.

### 1.3. Scope

The scope of this document is limited to discussion of the topics that can be covered in a DP/DPS and does not go into the specific details that could possibly be included in each one. In particular, this document describes the types of information that should be considered for inclusion in a DP/DPS.

The DNSSEC Policy and Practice Statement framework should be viewed and used as a checklist of factors that should be taken in to consideration prior to deploying DNSSEC, and an outline to create a operational practices disclosure document. It is primarily aimed at TLD managers and organizations providing registry services, but may be used by high-value domain holders and serve as a check sheet for DNSSEC readiness at a high level.

This document assumes that the reader is familiar with the general concepts of DNS, DNSSEC and PKI.

## 2. Definitions

This document makes use of the following defined terms:

Audit logs - Control evidence information generated by DNS and DNSSEC-related systems, the surrounding facility or other manually processed, non-electronic documentation to prove the integrity of processes. Audit logs will be examined by the internal and/or external auditors.

Activation data - Data values, other than keys, that are required to operate the cryptographic modules which are usually used to protect the keys from unauthorized use.

Chain of Trust - A hierarchical structure of trust consisting of DNS

keys, signatures, and delegation signer records that, when validated in a series, can provide proof of authenticity of the last element in the chain using the first element in the chain. Usually, the first element is a trust anchor.

Compromise (Key Compromise) - Key Compromise is a situation where the private component of the Key Signing Key or Zone Signing Key is lost, stolen, exposed, modified or used in an unauthorized manner. More strictly, even a suspicion that one of these has occurred will be enough to be considered as key compromise.

DNS - The Domain Name System (DNS) is a hierarchical global naming catalog for computers, services, or any resource connected to the Internet. It associates various information with domain names assigned to each of the participants.

DNS Zone - A portion of the global Domain Name System (DNS) namespace for which administrative responsibility has been delegated.

DNSSEC Policy - A DNSSEC Policy sets forth the requirements and standards to be implemented for a DNSSEC signed zone. A Practice Statement may support a Policy by explaining how it meets the requirements of the Policy.

DNSSEC Practice Statement - A DNSSEC Practices Statement is a practices disclosure document which may be a supplemental document to the DNSSEC Policy (if such exists) and states how the management of a given zone implements procedures and controls at a high level.

Key Roll Over - A operational process of DNSSEC to change either the Key Signing Key or the Zone Signing Key.

Policy Management Authority - A group formed by stake-holders from each group within the organization operating DNSSEC, responsible for managing the DP/DPS.

Public Key Infrastructure - A concept that uses asymmetric cryptography, which may provide integrity, authentication, confidentiality and non-repudiation to a system.

Relying Party - An entity that rely on the signed response from the DNS.

Repository - A location on the Internet to store DP, DPS, Trust Anchors and other related information that should be kept public.

Security Posture - A Security Posture is a barometer that indicates how secure the entity is and how secure the entity should be which is

a result of an adequate threat modelling, vulnerability assessment and risk assessment.

Separation of Duties - A security concept that limits the influence of a single person by segregating the roles and responsibilities.

Trust Anchor - Public portion of the Key Signing Key which is the authoritative entity used to cryptographically validate the chain of trust to the signed resource record.

### 3. Concepts

This section describes the concept of a DNSSEC Policy and of a DNSSEC Practices Statement. Other related concepts are described as well.

#### 3.1. DP

The DP sets forth requirements that are appropriate for a determined level of assurance. For example, a DP may encompass all topics of this framework, each with a certain set of security requirements and possibly grouped them into categories, such as medium impact and high impact. The progression from medium to high levels would correspond to increasing security requirements and corresponding increasing levels of assurance.

A DPS may identify a supported DP, which may subsequently be used by a relying party to evaluate the trustworthiness of any digital signatures verified using the public key of that entity.

DPS also constitute a basis for an audit, accreditation, or another assessment of an entity. Each entity can be assessed against one or more DPS that it is recognized as implementing.

#### 3.2. DPS

Most DNSSEC participants may not have the need to create a thorough and detailed statement of practices. For example, the registrant may itself be the sole relying party of its own zone and would already be aware of the nature and trustworthiness of its services. In other cases, a zone manager may provide registration services providing only a very low level of assurances where the domain names being secured may pose only marginal risks if compromised. Publishing a DPS is most relevant for entities operating a zone which contains a significant number of delegations to other entities.

A DPS should contain information which is relevant to the stakeholders of the relevant zone(s). Since these generally include

the Internet community, it should not contain such information which could be considered to be sensitive details of an entity's operations.

### 3.3. Relationship between DNSSEC Policy and Practice Statement

A DNSSEC Policy and a DNSSEC Practice Statement address the same set of topics that are of interest to the stakeholders in terms of the degree to which a DNSSEC digital signature may be trusted. Their primary difference is in the focus of their provisions. A Policy sets forth the requirements and standards to be implemented for a DNSSEC signed zone. In other words, the purpose of the Policy is to establish what entities must do. A Practice Statement, by contrast, states how a zone operator (and possibly other participants in the management of a given zone) implements procedures and controls to meet the requirements stated in the Policy. To summarise, the purpose of the Practice Statement is to disclose how the participants perform their functions and implement controls.

An additional difference between a Policy and a Practice Statement relates the scope of coverage of the two kinds of documents. Since a Policy is a statement of requirements, it is best used for communicating minimum operating guidelines that must be met by complying parties, but may as such also be used to facilitate interoperation of a level of trust between zones. Thus, a Policy may apply to multiple organizations or multiple zones. By contrast, a Practice Statement would usually apply only to a single zone operator or a single organization.

For example, a TLD Manager or regulatory authority may define requirements in a Policy for operations of one or more zones. The Policy will be a broad statement of the general requirements for managing the zone. A zone operator may be required to write its own Practice Statement to support the Policy by explaining how it meets the requirements of the Policy. Or, a zone operator which is also the manager of that zone and not governed by any external Policy may still choose to disclose operational practices by publishing a DPS, for the purpose of providing transparency and gain community trust in the operations.

A Policy and a Practice Statement also differ in the level of detail of the provisions in each. Although the level of detail may vary, a Practice Statement will generally be more detailed than a Policy. A Practice Statement provides a detailed description of procedures and controls in place to meet the Policy requirements, while a Policy is more general.

The main differences between a Policy and Practice Statement can



therefore be summarized as follows:

- (a) Operation of a DNS zone with DNSSEC may be governed by a Policy, to establish requirements that state what the entity operating that zone must do. An entity can use a Practice Statement to disclose how it meets the requirements of a Policy or how it has implemented critical processes and controls.
- (b) A Policy may facilitate interoperation of level of trust through several parts or levels in the DNS hierarchy. By contrast, a Practice Statement is a statement of a single zone operator or organization.
- (c) A Practice Statement is generally more detailed than a Policy and specifies how the zone operator or organization implements critical processes and controls, and how the entity meets any requirements specified in the one or more Policies under which it operates DNSSEC.

### 3.4. Set of Provisions

A set of provisions is a collection of Policy requirements or Practice statements, which may employ the approach described in this framework by covering the topics appearing in Section 5 below. They are also described in detail in Section 4 below.

A Policy can be expressed as a single set of provisions.

A Practice Statement can also be expressed as a single set of provisions with each component addressing the requirements of one or more Policies. Alternatively, it could be a set of provisions that do not reference any particular policy but instead describe a set of self-imposed provisions to the relying parties. For example, a Practice Statement could be expressed as a combination of the following:

- (a) a list of Policies supported by the DPS;
- (b) for each Policy in (a), a set of provisions that contains statements responding to that Policy by filling in details not stipulated in that policy or expressly left to the discretion of the implementor; such statements serve to show how this particular Practice Statement implements the requirements of the particular Policy; or
- (c) a set of provisions that contains statements regarding the practices of the DNSSEC operations, regardless of any Policy.

The statements provided in (b) may augment or refine the stipulations of an applicable Policy, but generally must not conflict with any of the stipulations of such Policy. In certain cases, however, a Policy Authority may permit exceptions because certain compensating controls

of the entity are disclosed in its Practices Statement, that allow the entity to provide assurances that are equivalent to the assurances provided by entities that are in full compliance with the Policy.

This framework outlines the contents of a set of provisions, in terms of eight primary components, as follows:

1. Introduction
2. Publication and Repositories
3. Operational Requirements
4. Facility, Management, and Operational Controls
5. Technical Security Controls
6. Zone Signing
7. Compliance Audit
8. Legal Matters

This framework can be used by Policy Authorities to write DNSSEC Policies and zone operators to write a DNSSEC Practice Statements. Having a set of documents with the same structure facilitates comparisons and mappings between them with the corresponding documents of other zones.

#### 4. Contents of a set of provisions

This section describes the contents of a set of provisions. Refer to Section 5 for the complete outline.

Drafters of DPSs in conformance with this framework are permitted to add additional levels of subcomponents below the subcomponents described here for the purpose of meeting the needs of the drafter's particular requirements. Drafters may also leave any components without stipulation if so requires, but all components listed in Section 5 should exist.

##### 4.1. Introduction

This component identifies and introduces the set of provisions, and indicates the types of entities and applications for which the document (either the Policy or the Practice Statement) is targeted.

##### 4.1.1. Overview

This subcomponent provides a general introduction to the document being written. It can also be used to provide a description of entities to which the Policy or Practice Statement applies.

#### 4.1.2. Document Name and Identification

This subcomponent provides any applicable names or other identifiers of the document.

#### 4.1.3. Community and Applicability

This subcomponent addresses the stakeholders in DNSSEC along with the expected roles and responsibilities. These include (but are not limited to) an entity signing the zone, an entity that relies on the signed zone, other entities that have operational dependency on the signed zone and an entity that entrusted the zone signing.

#### 4.1.4. Specification Administration

This subcomponent includes the name and contact details of the organization that is responsible for managing the DP/DPS.

Moreover, if a formal or informal Policy Authority is responsible for determining whether a DPS being suitable for the Policy this subcomponent may include the name and contact information of the entity in charge of making such a determination. Finally, in this case, this subcomponent also includes the procedures by which this determination is made.

#### 4.2. Publication and Repositories

This component contains any applicable provisions regarding:

- o The location and method to access the repository;
- o An identification of the entity or entities that operate repositories within the community, such as a zone operator or a TLD Manager;
- o The responsibility of an entity to publish information regarding its practices, public keys, and the current status of such keys, which may include the responsibilities of making the DPS publicly available using various mechanisms and of identifying components and subcomponents. It should also include an indication of the elements of the documents that are not made publicly available owing to their sensitive nature, e.g. security controls, clearance procedures, or business information;
- o When information must be published and the frequency of publication; and
- o Access control on published information objects.

#### 4.3. Operational Requirements

This component describes the operational requirements when operating a DNSSEC signed zone.

##### 4.3.1. Meaning of domain names

This section describes the overall policy of child zone naming, if any.

##### 4.3.2. Activation of DNSSEC for child zone

This section describes the process of establishing the chain-of-trust to the child zone by incorporating DS record(s) into the zone.

##### 4.3.3. Identification and authentication of child zone manager

This section describes how the child zone manager has initially been identified, and how any subsequent change request is authenticated as originating from the manager or its authorized representative.

##### 4.3.4. Registration of delegation signer (DS) resource records

This section describes how the delegation signer resource records are incorporated into the parent zone.

##### 4.3.5. Method to prove possession of private key

This section describes how, if, or under which circumstances the child zone manager is required to provide proof of the possession of the private component of any current or subsequent child zone Key Signing Key that corresponds to a DS record they wish to incorporate into the parent zone.

##### 4.3.6. Removal of DS resource records

This section will explain how, when and under which circumstances the DS records may be removed from the zone.

#### 4.4. Facility, Management and Operational Controls

This component describes non-technical security controls (i.e., physical, procedural, and personnel controls) in use by the entity to securely perform the DNSSEC related functions such as physical access, key management, disaster recovery, auditing and archiving.

These non-technical security controls are critical for trusting the signatures since lack of security may compromise DNSSEC operations

resulting for example, in the creation of signatures with erroneous information or compromising the Key Signing Key and/or Zone Signing Key.

Within each subcomponent, separate consideration will, in general, need to be given to each entity type.

#### 4.4.1. Physical Controls

In this subcomponent, the physical controls on the facility housing the entity systems are described. Topics addressed may include:

- o Site location and construction, such as the construction requirements for high-security areas and the use of locked rooms, cages, safes, and cabinets;
- o Physical access, i.e., mechanisms to control access from one area of the facility to another or access into high-security zones, such as locating DNSSEC operations in a secure computer room monitored by guards, cameras or security alarms and requiring movement from zone to zone to be accomplished using tokens and/or PINs;
- o Power and air conditioning;
- o Water exposures;
- o Fire prevention and protection;
- o Media storage, for example, requiring the storage of backup media in a separate location that is physically secure and protected from fire, smoke, particle and water damage;
- o Waste disposal; and
- o Off-site backup.

#### 4.4.2. Procedural Controls

In this subcomponent, requirements for recognizing trusted roles are described, together with the responsibilities for each role. Examples of trusted roles include system administrators, security officers, and system auditors.

For each task identified, the number of individuals required to perform the task (n of m rule, if applicable) should be stated for each role. Identification and authentication requirements for each role may also be defined.

This component also includes the separation of duties in terms of the roles that cannot be performed by the same individuals.

#### 4.4.3. Personnel Controls

This subcomponent addresses the following:

- o Qualifications, experience, and clearances that personnel must have as a condition of filling trusted roles or other important roles. Examples include credentials, job experiences, and official government clearances that candidates for these positions must have before being hired;
- o Background checks and clearance procedures that are required in connection with the hiring of personnel filling trusted roles or perhaps other important roles; such roles may require a check of their criminal records, financial records, references, and additional clearances that a participant undertakes after a decision has been made to hire a particular person;
- o Training requirements and training procedures for each role following the hiring of personnel;
- o Any retraining period and retraining procedures for each role after completion of initial training;
- o Frequency and sequence for job rotation among various roles;
- o Sanctions against personnel for unauthorized actions, unauthorized use of authority, and unauthorized use of the entity systems;
- o Controls on personnel that are independent contractors rather than employees of the entity; examples include:
  - \* Bonding requirements on contract personnel;
  - \* Contractual requirements including indemnification for damages due to the actions of the contractor personnel;
  - \* Auditing and monitoring of contractor personnel; and
  - \* Other controls on contracting personnel.
- o Documentation to be supplied to personnel during initial training, retraining, or otherwise.

#### 4.4.4. Audit Logging Procedures

This subcomponent is used to describe event logging and audit systems, implemented for the purpose of maintaining an audit trail and provide evidence of processes' integrity. Elements include the following:

- o Types of events recorded, such as attempts to access the system, and requests made to the system;
- o Frequency with which audit logs are processed or archived, for example, weekly, following an alarm or anomalous event, or when ever the audit log is n% full;

- o Period for which audit logs are kept;
- o Protection of audit logs:
  - \* Who can view audit logs, for example only the audit administrator;
  - \* Protection against modification of audit logs, for instance a requirement that no one may modify or delete the audit records or that only an audit administrator may delete an audit file as part of rotating the audit file; and
  - \* Protection against deletion of audit logs.
- o Audit log back up procedures;
- o Whether the audit log collection function is internal or external to the system;
- o Whether the subject who caused an audit event to occur is notified of the audit action; and
- o Vulnerability assessments, for example, where audit data is run through a tool that identifies potential attempts to breach the security of the system.

#### 4.4.5. Compromise and Disaster Recovery

This subcomponent describes requirements relating to notification and recovery procedures in the event of compromise or disaster. Each of the following may need to be addressed separately:

- o Identification or listing of the applicable incident and compromise reporting and handling procedures.
- o The recovery procedures used if computing resources, software, and/or data are corrupted or suspected to be corrupted. These procedures describe how a secure environment is re-established, whether the Key Signing Key or Zone Signing Key requires a roll over, how to assess the damage and carry out the root cause analysis.
- o The recovery procedures used if the Key Signing Key or Zone Signing Key is compromised. These procedures describe how a secure environment is re-established, how the keys are rolled over, how a new Trust Anchor is provided to the users (if applicable) and how new zone is published.
- o The entity's capabilities to ensure business continuity following a natural or other disaster. Such capabilities may include the availability of a disaster recovery site at which operations may be recovered. They may also include procedures for securing its facility during the period of time following a natural or other disaster and before a secure environment is re-established, either at the original site or at a disaster recovery site. For example, procedures to protect against theft of sensitive materials from an

earthquake-damaged site.

#### 4.4.6. Entity termination

This subcomponent describes requirements relating to procedures for entity termination, termination notification and transition of responsibilities of a zone operator or other entity, where the purpose may be to ensure that the transition process will be transparent to the relying parties and will not affect the services.

#### 4.5. Technical Security Controls

This component is used to define the security measures taken to protect the cryptographic keys and activation data (e.g., PINs, passwords, or manually-held key shares) relevant to the DNSSEC operations. Secure key management is critical to ensure that all secret and private keys and activation data are protected and used only by authorized personnel.

Also describes here are other technical security controls used to perform the functions of key generation, authentication, registration, auditing, and archiving. Technical controls include life-cycle security controls (including software development environment security) and operational security controls.

If applicable, other technical security controls on repositories, authoritative name servers or other participants may also be documented here.

##### 4.5.1. Key Pair Generation and Installation

Key pair generation and installation need to be considered, whereas the following questions potentially need to be answered:

1. Who generates the zone's public, private key pairs? Furthermore, how is the key generation performed? Is the key generation performed by hardware or software?
2. How is the private key installed in all parts of the key management system?
3. How are the zones's public keys provided securely to the parent zone and potential relying parties?
4. Who generates the public key parameters, and is the quality of the parameters checked during key generation?



5. For what purposes may the keys be used, and/or for what purposes should usage of the key be restricted?

#### 4.5.2. Private Key Protection and Cryptographic Module Engineering Controls

Requirements for private key protection and cryptographic modules need to be considered for key generation and creation of signatures. The following questions potentially need to be answered:

1. What standards, if any, are required for the cryptographic module used to generate the keys? A cryptographic module can be composed of hardware, software, firmware, or any combination of them. For example, are the zones signatures required to be generated using modules compliant with the US FIPS 140-2 standard? If so, what is the required FIPS 140-2 level of the module? Are there any other engineering or other controls relating to a cryptographic module, such as the identification of the cryptographic module boundary, input/output, roles and services, finite state machine, physical security, software security, operating system security, algorithm compliance, electromagnetic compatibility, and self tests.
2. Is the private key under  $n$  out of  $m$  multi-person control? If yes, provide  $n$  and  $m$  (two person control is a special case of  $n$  out of  $m$ , where  $n = m = 2$ )?
3. Is the private key escrowed? If so, who is the escrow agent, what form is the key escrowed in (examples include plaintext, encrypted, split key), and what are the security controls on the escrow system?
4. Is the private key backed up? If so, who is the backup agent, what form is the key backed up in (examples include plaintext, encrypted, split key), and what are the security controls on the backup system?
5. Is the private key archived? If so, who is the archival agent, what form is the key archived in (examples include plaintext, encrypted, split key), and what are the security controls on the archival system?
6. Under what circumstances, if any, can a private key be transferred into or from a cryptographic module? Who is permitted to perform such a transfer operation? In what form is the private key during the transfer (i.e., plaintext, encrypted, or split key)?
7. How is the private key stored in the module (i.e., plaintext, encrypted, or split key)?

8. Who can activate (use) the private key? What actions must be performed to activate the private key (e.g., login, power on, supply PIN, insert token/key, automatic, etc.)? Once the key is activated, is the key active for an indefinite period, active for one time, or active for a defined time period?
9. Who can deactivate the private key and how? Examples of methods of deactivating private keys include logging out, turning the power off, removing the token/key, automatic deactivation, and time expiration.
10. Who can destroy the private key and how? Examples of methods of destroying private keys include token surrender, token destruction, and zeroizing the key.

#### 4.5.3. Other Aspects of Key Pair Management

Other aspects of key management need to be considered for the zone operator and other participants. For each of these types of entities, the following questions may need to be answered:

1. Is the public key archived? If so, who is the archival agent and what are the security controls on the archival system?
2. What is the operational period of the keys. What are the usage periods, or active lifetimes for the pairs?

#### 4.5.4. Activation data

Activation data refers to data values other than whole private keys that are required to operate private keys or cryptographic modules containing private keys, such as a PIN, passphrase, or portions of a private key used in a key-splitting scheme. Protection of activation data prevents unauthorized use of the private key, and potentially needs to be considered for the zone operator and other participants. Such consideration potentially needs to address the entire life-cycle of the activation data from generation through archival and destruction. For each of the entity types, all of the questions listed in 4.5.1 through 4.5.3 potentially need to be answered with respect to activation data rather than with respect to keys.

#### 4.5.5. Computer Security Controls

This subcomponent is used to describe computer security controls such as:

1. use of the trusted computing base concept or equivalent;
2. discretionary access control, labels, mandatory access controls;
3. object re-use;
4. auditing;
5. identification and authentication;
6. trusted path; and
7. security testing.

Product assurance may also be addressed.

A computer security rating for computer systems may be specified. The rating could be based, for example, on a protection profile (PP) of the Common Criteria for Information Technology Security Evaluation, ISO/IEC 15408:1999. This subcomponent may also address requirements for product evaluation analysis, testing, profiling, product certification, and/or product accreditation related activity undertaken.

#### 4.5.6. Network Security Controls

This subcomponent addresses network security related controls, including firewalls, routers and remote access.

#### 4.5.7. Timestamping

This subcomponent addresses requirements or practices relating to the use of timestamps on various data. It may also discuss whether or not the time-stamping application must use a trusted time source.

#### 4.5.8. Life Cycle Technical Controls

This subcomponent addresses system development controls and security management controls.

System development controls include development environment security, development personnel security, configuration management security during product maintenance, software engineering practices, software development methodology, modularity, layering, use of failsafe design and implementation techniques (e.g., defensive programming) and development facility security.

Security management controls include execution of tools and procedures to ensure that the operational systems and networks adhere to configured security. These tools and procedures include checking the integrity of the security software, firmware, and hardware to ensure their correct operation.

#### 4.6. Zone Signing

This component covers all aspects of zone signing, including the cryptographic specification surrounding the Key Signing Key and Zone Signing Key, signing scheme and methodology for key roll-over and the actual zone signing. Child zones and other relying parties may depend on the information in this section to understand the expected data in the signed zone and determine their own behavior. In addition, this section will be used to state the compliance to the cryptographic and operational requirements pertaining to zone signing, if any.

##### 4.6.1. Key lengths and algorithms

This subcomponent describes the key generation algorithm and the key length used to create the Key Signing Key and the Zone Signing Key.

##### 4.6.2. Authenticated denial of existence

Authenticated denial of existence refers to the usage of NSEC (RFC 4034 [RFC4034]), NSEC3 (RFC 5155 [RFC5155]) or any other record defined in the future that is used to authenticate the denial of existence of the resource record.

##### 4.6.3. Signature format

This subcomponent is used to describe the signing method and algorithms used for the zone signing.

##### 4.6.4. Zone Signing Key Roll-Over

This subcomponent explains the Zone signing key roll-over scheme.

##### 4.6.5. Key Signing Key Roll-Over

This subcomponent addresses the Key signing key roll-over scheme.

##### 4.6.6. Signature life-time and re-signing frequency

This subcomponent describes the life-cycle of the Resource Record Signature (RRSIG) record.

##### 4.6.7. Verification of Zone Signing Key Set

This subsection addresses the controls around the keyset signing process performed by the Key Signing Key. The procedures surrounding KSK management may be different from those of the ZSK, hence it may be necessary to authenticate the data signed by the KSK.

#### 4.6.8. Verification of resource records

This subsection addresses the controls around the verification of the resource records in order to validate and authenticate the data to be signed.

#### 4.6.9. Resource records time-to-live

This subcomponent specifies the time-to-live (TTL) for each DNSSEC related resource record such as DNSKEY, NSEC/NSEC3, DS and RRSIG.

#### 4.7. Compliance Audit

The ideal and the only way to prove the compliance with a Policy or the statements in the Practices Statement is to conduct an audit. This component describes the outline of how the audit is conducted at the zone operator and possibly at other involved entities.

##### 4.7.1. Frequency of entity compliance audit

This subcomponent describes the frequency of the compliance audit.

##### 4.7.2. Identity/qualifications of auditor

This subcomponent addresses what is the qualifications for the auditor. For instance it may be an auditor from a specific association or an auditor that has a certain certifications.

##### 4.7.3. Auditor's relationship to audited party

This subcomponent is used to clarify the relationship between the auditor and the entity being audited. This becomes important if there are any requirements or guidelines for the selection of the auditor.

##### 4.7.4. Topics covered by audit

Topics covered by audit refers to the scope of the audit. Since the DNSSEC Policy and Practices Statement is the document to be audited against, it is ideal to set the scope to the scope of the DP/DPS. However, the scope may be narrowed down or expanded as needed for example in case there are not enough resources to conduct a full audit, or some portion is under development and not ready for the audit.

#### 4.7.5. Actions taken as a result of deficiency

This subcomponent specifies the action taken in order to correct any discrepancy. This could be the remediation process for the audit findings or any other action to correct any discrepancy with the DNSSEC Policy or Practices Statement.

#### 4.7.6. Communication of results

This subcomponent specifies how the results of the audit are communicated to the stakeholders.

#### 4.8. Legal Matters

The introduction of DNSSEC into a zone may have legal implications. Consequently, it may be appropriate to declare the legal status of the binding embodied in the DNSSEC digital signatures and to clarify on any limitations of liability assumed by the Registry Manager.

In most cases, the DPS is not a contract or part of a contract; instead, it is laid out so that its terms and conditions are applied to the parties by separate documents, such as registrar or registrant agreements. In other cases its contents may form part of a legal contract between parties (either directly or via other agreements). In this case legal expertise should be consulted when drawing up sections of the document that may have contractual implications.

At a minimum, the legal matters section should indicate under what jurisdiction the registry is operated, and provide references to any associated agreements which are in force. It may also be appropriate to inform of any identified implications on the protection of personally identifiable private information.

### 5. Outline of a set of provisions

1. INTRODUCTION
  - 1.1. Overview
  - 1.2. Document name and identification
  - 1.3. Community and Applicability
  - 1.4. Specification Administration
    - 1.4.1. Specification administration organization
    - 1.4.2. Contact Information
    - 1.4.3. Specification change procedures
2. PUBLICATION AND REPOSITORIES
  - 2.1. Repositories
  - 2.2. Publication of key signing keys

- 2.3. Access controls on repositories
- 3. OPERATIONAL REQUIREMENTS
  - 3.1. Meaning of domain names
  - 3.2. Activation of DNSSEC for child zone
  - 3.3. Identification and authentication of child zone manager
  - 3.4. Registration of delegation signer (DS) resource records
  - 3.5. Method to prove possession of private key
  - 3.6. Removal of DS record
    - 3.6.1. Who can request removal
    - 3.6.2. Procedure for removal request
    - 3.6.3. Emergency removal request
- 4. FACILITY, MANAGEMENT AND OPERATIONAL CONTROLS
  - 4.1. Physical Controls
    - 4.1.1. Site location and construction
    - 4.1.2. Physical access
    - 4.1.3. Power and air conditioning
    - 4.1.4. Water exposures
    - 4.1.5. Fire prevention and protection
    - 4.1.6. Media storage
    - 4.1.7. Waste disposal
    - 4.1.8. Off-site backup
  - 4.2. Procedural Controls
    - 4.2.1. Trusted roles
    - 4.2.2. Number of persons required per task
    - 4.2.3. Identification and authentication for each role
    - 4.2.4. Tasks requiring separation of duties
  - 4.3. Personnel Controls
    - 4.3.1. Qualifications, experience, and clearance requirements
    - 4.3.2. Background check procedures
    - 4.3.3. Training requirements
    - 4.3.4. Retraining frequency and requirements
    - 4.3.5. Job rotation frequency and sequence
    - 4.3.6. Sanctions for unauthorized actions
    - 4.3.7. Contracting personnel requirements
    - 4.3.8. Documentation supplied to personnel
  - 4.4. Audit Logging Procedures
    - 4.4.1. Types of events recorded
    - 4.4.2. Frequency of processing log
    - 4.4.3. Retention period for audit log information
    - 4.4.4. Protection of audit log
    - 4.4.5. Audit log backup procedures
    - 4.4.6. Audit collection system
    - 4.4.7. Notification to event-causing subject
    - 4.4.8. Vulnerability assessments
  - 4.5. Compromise and Disaster Recovery
    - 4.5.1. Incident and compromise handling procedures
    - 4.5.2. Corrupted computing resources, software, and/or

- data
- 4.5.3. Entity private key compromise procedures
- 4.5.4. Business Continuity and IT Disaster Recovery Capabilities
- 4.6. Entity termination
- 5. TECHNICAL SECURITY CONTROLS
  - 5.1. Key Pair Generation and Installation
    - 5.1.1. Key pair generation
    - 5.1.2. Public key delivery
    - 5.1.3. Public key parameters generation and quality checking
    - 5.1.4. Key usage purposes
  - 5.2. Private key protection and Cryptographic Module Engineering Controls
    - 5.2.1. Cryptographic module standards and controls
    - 5.2.2. Private key (m-of-n) multi-person control
    - 5.2.3. Private key escrow
    - 5.2.4. Private key backup
    - 5.2.5. Private key storage on cryptographic module
    - 5.2.6. Private key archival
    - 5.2.7. Private key transfer into or from a cryptographic module
    - 5.2.8. Method of activating private key
    - 5.2.9. Method of deactivating private key
    - 5.2.10. Method of destroying private key
  - 5.3. Other Aspects of Key Pair Management
    - 5.3.1. Public key archival
    - 5.3.2. Key usage periods
  - 5.4. Activation data
    - 5.4.1. Activation data generation and installation
    - 5.4.2. Activation data protection
    - 5.4.3. Other aspects of activation data
  - 5.5. Computer Security Controls
  - 5.6. Network Security Controls
  - 5.7. Timestamping
  - 5.8. Life Cycle Technical Controls
    - 5.8.1. System development controls
    - 5.8.2. Security management controls
    - 5.8.3. Life cycle security controls
- 6. ZONE SIGNING
  - 6.1. Key lengths and algorithms
  - 6.2. Authenticated denial of existence
  - 6.3. Signature format
  - 6.4. Zone signing key roll-over
  - 6.5. Key signing key roll-over
  - 6.6. Signature life-time and re-signing frequency
  - 6.7. Verification of zone signing key set
  - 6.8. Verification of resource records



- 6.9. Resource records time-to-live
- 7. COMPLIANCE AUDIT
  - 7.1. Frequency of entity compliance audit
  - 7.2. Identity/qualifications of auditor
  - 7.3. Auditor's relationship to audited party
  - 7.4. Topics covered by audit
  - 7.5. Actions taken as a result of deficiency
  - 7.6. Communication of results
- 8. LEGAL MATTERS

## 6. Security Considerations

The sensitivity of the information protected by DNSSEC on all levels in the DNS tree will vary significantly. In addition, there are no restrictions as to what types of information that can be protected using DNSSEC. Entities must evaluate their own environment and the chain of trust originating from their Trust Anchor, the associated threats and vulnerabilities, to determine the level of risk they are willing to accept.

## 7. Acknowledgements

The authors gratefully acknowledges, in no particular order, the contributions of the following persons:

Richard Lamb  
Jakob Schlyter  
Stephen Morris

## 8. References

### 8.1. Normative References

- [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.

## 8.2. Informative References

- [RFC3647] Chokhani, S., Ford, W., Sabett, R., Merrill, C., and S. Wu, "Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework", RFC 3647, November 2003.
- [RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, March 2008.

## Authors' Addresses

Fredrik Ljunggren  
Kirei AB  
P.O. Box 53204  
Goteborg SE-400 16  
Sweden

Email: fredrik@kirei.se

Anne-Marie Eklund-Lowinder  
.SE (The Internet Infrastructure Foundation)  
P.O. Box 7399  
Stockholm SE-103 91  
Sweden

Email: amel@iis.se

Tomofumi Okubo  
Internet Corporation For Assigned Names and Numbers  
4676 Admiralty Way, Suite 330  
Marina del Ray, CA 90292  
USA

Email: tomofumi.okubo@icann.org



Internet Engineering Task Force  
Internet-Draft  
Intended status: Informational  
Expires: September 11, 2011

S. Morris  
ISC  
J. Ihren  
Netnod  
J. Dickinson  
Sinodun  
March 10, 2011

DNSSEC Key Timing Considerations  
draft-ietf-dnsop-dnssec-key-timing-02.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 September 11, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
1.1. Key Rolling Considerations . . . . .	3
1.2. Types of Keys . . . . .	4
1.3. Terminology . . . . .	4
1.4. Requirements Language . . . . .	4
2. Rollover Methods . . . . .	4
2.1. ZSK Rollovers . . . . .	4
2.2. KSK Rollovers . . . . .	6
2.3. Summary . . . . .	7
3. Key Rollover Timelines . . . . .	7
3.1. Key States . . . . .	7
3.2. Zone-Signing Key Timelines . . . . .	9
3.2.1. Pre-Publication Method . . . . .	9
3.2.2. Double-Signature Method . . . . .	11
3.2.3. Double-RRSIG Method . . . . .	13
3.3. Key-Signing Key Rollover Timelines . . . . .	15
3.3.1. Double-Signature Method . . . . .	15
3.3.2. Double-DS Method . . . . .	18
3.3.3. Double-RRset Method . . . . .	21
3.3.4. Interaction with Configured Trust Anchors . . . . .	23
3.3.4.1. Addition of KSK . . . . .	23
3.3.4.2. Removal of KSK . . . . .	24
3.3.5. Introduction of First KSK . . . . .	24
4. Standby Keys . . . . .	24
5. Algorithm Considerations . . . . .	25
6. Limitation of Scope . . . . .	26
7. Summary . . . . .	26
8. IANA Considerations . . . . .	27
9. Security Considerations . . . . .	27
10. Acknowledgements . . . . .	27
11. Change History (To be removed on publication) . . . . .	27
12. References . . . . .	28
12.1. Normative References . . . . .	28
12.2. Informative References . . . . .	29
Appendix A. List of Symbols . . . . .	29
Authors' Addresses . . . . .	32

## 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:

- o 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.
- o 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.
- o 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.
- o 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.

## 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:

- o Pre-Publication: described in [RFC4641], 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 be removed from the DNSKEY RRset.

- o Double-Signature: also mentioned in [RFC4641], 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.

- o 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 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 DS record in the parent zone (which is itself trusted) 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:

- o 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.)

- 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 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 RRSIG.
Double-Signature	Double-Signature	Publish the DNSKEY and RRSIG at same time. (For a KSK, this happens before the DS is published.)
Double-RRSIG	(not applicable)	Publish RRSIG 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	<p>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.</p> <p>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".</p>
Ready	The new key data has been published for long enough to guarantee that any previous versions of it 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	The key is in the zone but a successor key has become active. As there may still be information in caches that that require use of the key, it is being retained until this information expires.
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.

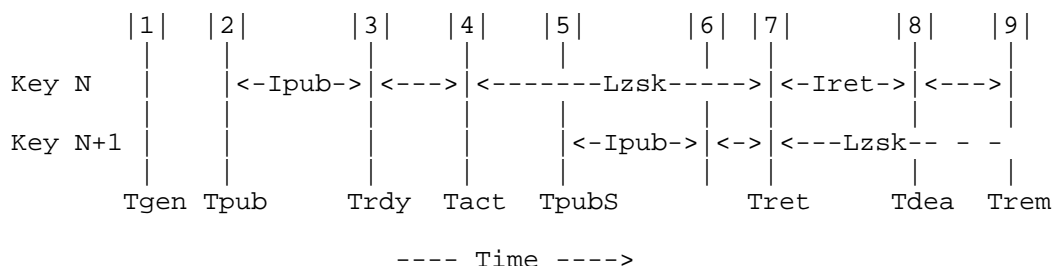


Figure 1: Timeline for a Pre-Publication ZSK rollover.

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:

$$I_{pub} = D_{prp} + TTL_{key}$$

Here,  $D_{prp}$  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.  $TTL_{key}$  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  $I_{pub}$ , 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:

$$Trdy = T_{pub} + I_{pub}$$

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  $I_{pub}$  before it is activated. Denoting the publication time of the successor key by  $T_{pubS}$ , then:

$$T_{pubS} \leq Tact + Lzsk - I_{pub}$$

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 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:

$$T_{ret} = T_{act} + L_{zsk}$$

It is also the activation time of the successor key ( $T_{actS}$ ). Note that operational considerations may cause key N to remain in use for longer than  $L_{zsk}$ ; 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 ( $I_{ret}$ ) given by:

$$I_{ret} = D_{sgn} + D_{prp} + TTL_{sig}$$

$D_{sgn}$  is the delay needed to ensure that all existing RRsets have been re-signed with the new key.  $D_{prp}$  is (as described above) the propagation delay, required to guarantee that the updated zone information has reached all slave servers, and  $TTL_{sig}$  is the maximum TTL of all the RRSIG records in the zone.

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

$$T_{dea} = T_{ret} + I_{ret}$$

...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 and the DNSKEY RRset re-signed with the current key-signing key. This time is the removal time ( $T_{rem}$ ), given by:

$$T_{rem} \geq T_{dea}$$

...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

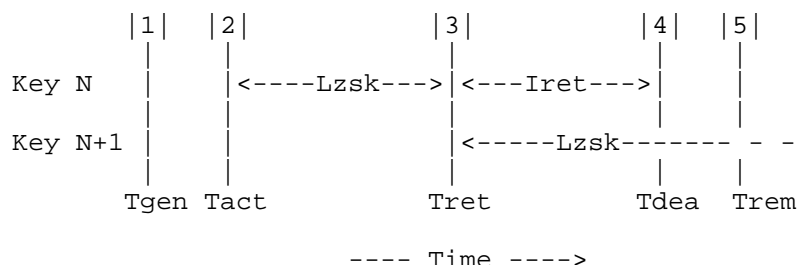


Figure 2: Timeline for a Double-Signature ZSK rollover.

Event 1: key N is generated at the generate time ( $T_{gen}$ ). 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).

$$\text{Tret} = \text{Tact} + \text{Lzsk}$$

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:

$$\text{Iret} = \text{Dsgn} + \text{Dprp} + \max(\text{TTLkey}, \text{TTLsig})$$

As before, Dsgn is the delay needed to ensure that all existing RRsets have been signed with the new key, Dprp is the propagation delay. The final term (the maximum of TTLkey and TTLsig) is the





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 ( $T_{pub}$ ).

Event 3: after the signing interval,  $D_{sgn}$ , 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:

$$D_{prp} + TTL_{sig}$$

As before,  $D_{prp}$  is the propagation delay and  $TTL_{sig}$  is the maximum TTL of all the RRSIG records in the zone.

Again in analogy with other key rollover methods, this is defined as key N's ready time ( $T_{rdy}$ ) 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,  $I_{pubG}$ , i.e.

$$T_{rdy} = T_{pub} + I_{pubG}$$

where

$$I_{pubG} = D_{sgn} + D_{prp} + TTL_{sig}$$

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 ( $T_{act}$ ) 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  $T_{pubS}$  (the publication time of the successor), given by:

$$T_{pubS} \leq T_{act} + L_{zsk} - I_{pubG}$$

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 ( $T_{rdyS}$ ).

$$\text{TrdyS} = \text{TpubS} + \text{IpubG}$$

... where IpubG is as defined above.

Event 8: at some later time key N is removed from the zone and the successor key (key N+1) added. 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. The time at which this as they can not be validated by any key. In analogy with other rollover methods, the time at which this occurs is the dead time (Tdea), given by:

$$\text{Tdea} = \text{Tret} + \text{Iret}$$

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

$$\text{Iret} = \text{Dprp} + \text{TTLkey}$$

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:

$$\text{Trem} \geq \text{Tdea}$$

### 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 its 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



$$I_{pub} = D_{prpC} + TTL_{key}$$

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

$$Trdy = T_{pub} + I_{pub}$$

(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,  $T_{sub}$ .

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 ( $T_{act}$ ):

$$T_{act} = T_{sub} + D_{reg}$$

... where  $D_{reg}$  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,  $T_{pubS}$ , given by:

$$T_{pubS} \leq T_{act} + L_{ksk} - D_{reg} - I_{pub}$$

... where  $L_{ksk}$  is the scheduled lifetime of the KSK.

Event 7: after an interval  $I_{pub}$ , 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 ( $T_{subS}$ ), 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  $T_{ret}$ , given by:

$$T_{ret} = T_{act} + L_{ksk}$$

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:

$$I_{ret} = D_{prpP} + TTL_{ds}$$

... where  $D_{prpP}$  is the propagation delay in the parent zone and  $TTL_{ds}$  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 ( $T_{dea}$ ), given by:

$$T_{dea} = T_{ret} + I_{ret}$$

Event 11: at some later time, key N is removed from the zone (at the remove time  $T_{rem}$ ); the key is now said to be removed.

$$T_{rem} \geq T_{dea}$$

### 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

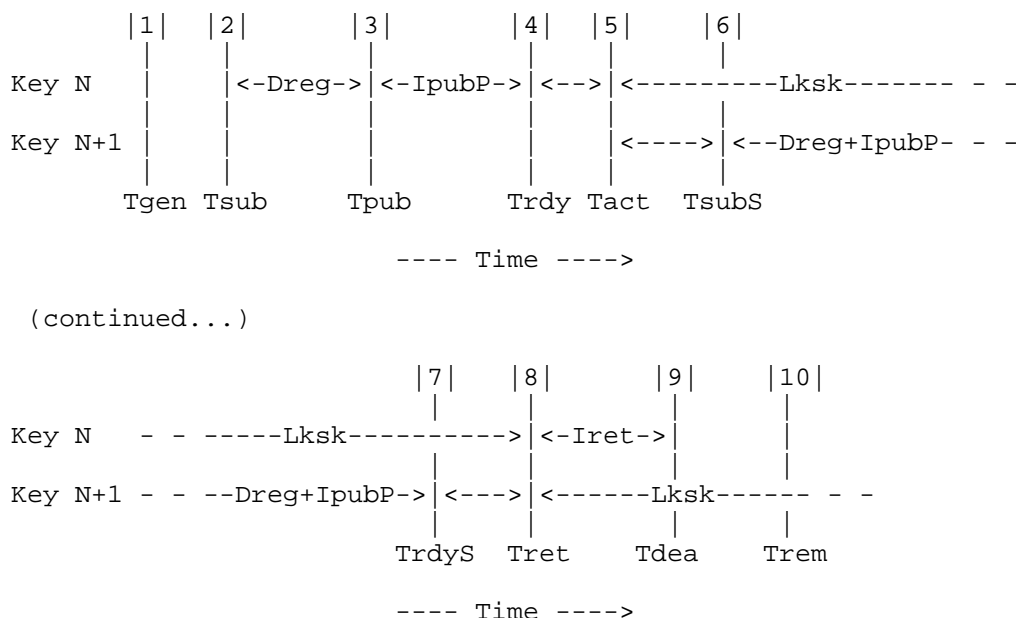


Figure 5: Timeline for a Double-DS KSK rollover.

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:

$$T_{pub} = T_{sub} + D_{reg}$$

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:

$$\text{Trdy} = \text{Tpub} + \text{IpubP}$$

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

$$\text{IpubP} = \text{DprpP} + \text{TTLds}$$

... 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) introduced and used to sign the RRset.

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 - either the resolver 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 have data about the DS record of the successor key. The time at which this occurs is the submission time of the successor, given by:

$$\text{TsubS} \leq \text{Tact} + \text{Lksk} - \text{IpubP} - \text{Dreg}$$

... 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 the interval between events 2 and 4 for key N)

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

$$T_{ret} = T_{act} + L_{ksk}$$

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,  $T_{dea}$ , and is given by:

$$T_{dea} = T_{ret} + I_{ret}$$

... where  $I_{ret}$  is the retire interval, given by:

$$I_{ret} = D_{prpC} + TTL_{key}$$

As before, this term includes  $D_{prpC}$ , the time taken to propagate the RRset change through the master-slave hierarchy of the child zone and  $TTL_{key}$ , 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 ( $T_{rem}$ ), given by:

$$T_{rem} \geq T_{dea}$$

### 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

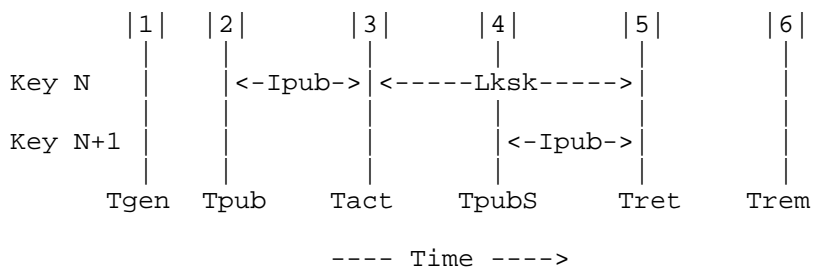


Figure 6: Timeline for a Double-RRset KSK rollover.

Event 1: key N is generated at the generate time ( $T_{gen}$ ). 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 ( $T_{pub}$ ) 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 ( $T_{act}$ ), given by:

$$T_{act} = T_{pub} + I_{pub}$$

... where  $I_{pub}$  is the publication interval, given by:

$$I_{pub} = \max(I_{pubP}, I_{pubC}),$$

$I_{pubP}$  being the publication interval in the parent zone and  $I_{pubC}$  the publication interval in the child zone. The parent zone's publication interval is given by:

$$I_{pubP} = D_{reg} + D_{prpP} + TTL_{ds}$$

where  $D_{reg}$  is the registration delay, the time taken for the DS record to be published in the parent zone.  $D_{prpP}$  is the parent zone's propagation delay and  $TTL_{ds}$  is the TTL of the DS record in that zone.

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

$$I_{pubC} = D_{prpC} + TTL_{key}$$

... where  $D_{prpC}$  is the propagation delay in the child zone and  $TTL_{key}$  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,  $L_{ksk}$ . This time is  $T_{pubS}$ , the publication time of the successor key, and is

given by:

$$T_{pubS} \leq T_{act} + L_{ksk} - I_{pub}$$

... where  $L_{ksk}$  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 ( $T_{actS}$ ) 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:

$$T_{ret} = T_{dea} = T_{actS} = T_{act} + L_{ksk}$$

Event 6: at some later time, the key  $N$ 's DS and DNSKEY records can be removed from their respective zones. In analogy with other rollover methods, this is the removal time ( $T_{rem}$ ), given by:

$$T_{rem} \geq T_{dea}$$

#### 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 ( $I_{pub}$ ) in the Double-Signature and Double-RRset methods should also be subject to the condition:

$$I_{pub} \geq D_{prp} + \max(30 \text{ days}, T_{TLkey})$$

... 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 add hold-down time defined in section 2.4.1 of [RFC5011].

In the Double-DS method, instead of the changing of the KSK RR being

instantaneous, there must now be a period of overlap. In other words, the new KSK must be introduced into the zone at least:

$$D_{prpC} + \max(30 \text{ days}, TTL_{key})$$

... 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 on the DNSKEY RR and is published in (and used to sign) the DNSKEY RRs. The key is maintained in this state for the "revoke" interval,  $I_{rev}$ , given by:

$$I_{rev} \geq 30 \text{ days}$$

... 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 KSK

There is an additional consideration when introducing a KSK into a zone for the first time, and that is that no validating resolver should be in a position where it can access the trust anchor before the KSK appears in the zone. To do so will cause it to declare the zone to be bogus.

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 not be 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 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 considering differ from this those described in this document. Of note 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 transition to/from/ addition/deletion of different algorithms. Algorithm rollovers will require a separate document.

The reader is therefore reminded that DNSSEC is as of publication in 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. Change History (To be removed on publication)

- o 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.
- o draft-ietf-dnsop-dnssec-key-timing-01
  - \* Added section on limitation of scope.
- o draft-ietf-dnsop-dnssec-key-timing-00
  - \* Change to author contact details.
- o 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).
- o 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).
  - o draft-morris-dnsop-dnssec-key-timing-00
    - Initial draft.

## 12. References

### 12.1. Normative References

- [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.

## 12.2. Informative References

[RFC4641] Kolkman, O. and R. Gieben, "DNSSEC Operational Practices", RFC 4641, September 2006.

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

#### Authors' Addresses

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

Phone: +1 650 423 1300  
Email: [stephen@isc.org](mailto: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



DNSOP  
Internet-Draft  
Obsoletes: 2541 (if approved)  
Intended status: Informational  
Expires: September 12, 2011

O. Kolkman  
W. Mekking  
NLnet Labs  
March 11, 2011

DNSSEC Operational Practices, Version 2  
draft-ietf-dnsop-rfc4641bis-06

Abstract

This document describes a set of practices for operating the DNS with security extensions (DNSSEC). The target audience is zone administrators deploying DNSSEC.

The document discusses operational aspects of using keys and signatures in the DNS. It discusses issues of key generation, key storage, signature generation, key rollover, and related policies.

This document obsoletes RFC 4641 as it covers more operational ground and gives more up-to-date requirements with respect to key sizes and the DNSSEC operations.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on September 12, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction . . . . .	5
1.1. The Use of the Term 'key' . . . . .	6
1.2. Time Definitions . . . . .	6
2. Keeping the Chain of Trust Intact . . . . .	7
3. Keys Generation and Storage . . . . .	7
3.1. Operational Motivation for Zone Signing and Key Signing Keys . . . . .	8
3.2. Practical Consequences of KSK and ZSK Separation . . . . .	10
3.2.1. Rolling a KSK that is not a trust-anchor . . . . .	10
3.2.2. Rolling a KSK that is a trust-anchor . . . . .	11
3.2.3. The use of the SEP flag . . . . .	12
3.3. Key Effectivity Period . . . . .	12
3.4. Cryptographic Considerations . . . . .	13
3.4.1. Key Algorithm . . . . .	13
3.4.2. Key Sizes . . . . .	14
3.4.3. Private Key Storage . . . . .	15
3.4.4. Key Generation . . . . .	16
3.4.5. Differentiation for 'High-Level' Zones? . . . . .	17
4. Signature Generation, Key Rollover, and Related Policies . . . . .	17
4.1. Key Rollovers . . . . .	17
4.1.1. Zone Signing Key Rollovers . . . . .	17

4.1.1.1.	Pre-Publish Zone Signing Key Rollover . . . . .	18
4.1.1.2.	Double Signature Zone Signing Key Rollover . . . . .	21
4.1.1.3.	Pros and Cons of the Schemes . . . . .	22
4.1.2.	Key Signing Key Rollovers . . . . .	22
4.1.2.1.	Special Considerations for RFC5011 KSK rollover . . . . .	24
4.1.3.	Difference Between ZSK and KSK Rollovers . . . . .	24
4.1.4.	Rollover for a Single Type Signing Key rollover . . . . .	25
4.1.5.	Algorithm rollovers . . . . .	27
4.1.5.1.	Single Type Signing Scheme Algorithm Rollover . . . . .	31
4.1.5.2.	Algorithm rollover, RFC5011 style . . . . .	31
4.1.5.3.	Single Signing Type Algorithm Rollover, RFC5011 style . . . . .	32
4.1.5.4.	NSEC to NSEC3 algorithm rollover . . . . .	33
4.1.6.	Considerations for Automated Key Rollovers . . . . .	33
4.2.	Planning for Emergency Key Rollover . . . . .	34
4.2.1.	KSK Compromise . . . . .	34
4.2.1.1.	Keeping the Chain of Trust Intact . . . . .	35
4.2.1.2.	Breaking the Chain of Trust . . . . .	36
4.2.2.	ZSK Compromise . . . . .	36
4.2.3.	Compromises of Keys Anchored in Resolvers . . . . .	36
4.2.4.	Stand-by keys . . . . .	37
4.3.	Parent Policies . . . . .	37
4.3.1.	Initial Key Exchanges and Parental Policies Considerations . . . . .	37
4.3.2.	Storing Keys or Hashes? . . . . .	38
4.3.3.	Security Lameness . . . . .	39
4.3.4.	DS Signature Validity Period . . . . .	39
4.3.5.	Changing DNS Operators . . . . .	40
4.3.5.1.	Cooperating DNS operators . . . . .	40
4.3.5.2.	Non Cooperating DNS Operators . . . . .	42
4.4.	Time in DNSSEC . . . . .	44
4.4.1.	Time Considerations . . . . .	44
4.4.2.	Signature Validation Periods . . . . .	46
4.4.2.1.	Maximum Value . . . . .	46
4.4.2.2.	Minimum Value . . . . .	47
4.4.2.3.	Differentiation between RR sets . . . . .	48
5.	Next Record type . . . . .	49
5.1.	Differences between NSEC and NSEC3 . . . . .	49
5.2.	NSEC or NSEC3 . . . . .	50
5.3.	NSEC3 parameters . . . . .	51
5.3.1.	NSEC3 Algorithm . . . . .	51
5.3.2.	NSEC3 Iterations . . . . .	51
5.3.3.	NSEC3 Salt . . . . .	51
5.3.4.	Opt-out . . . . .	52
6.	Security Considerations . . . . .	52
7.	IANA considerations . . . . .	52
8.	Contributors and Acknowledgments . . . . .	53
9.	References . . . . .	54



9.1. Normative References . . . . .	54
9.2. Informative References . . . . .	54
Appendix A. Terminology . . . . .	56
Appendix B. Typographic Conventions . . . . .	57
Appendix C. Transition Figures for Special Case Algorithm Rollovers . . . . .	60
Appendix D. Document Editing History . . . . .	64
D.1. draft-ietf-dnsop-rfc4641-00 . . . . .	64
D.2. version 0->1 . . . . .	65
D.3. version 1->2 . . . . .	65
D.4. version 2->3 . . . . .	66
D.5. version 3->4 . . . . .	67
D.6. version 4->5 . . . . .	67
D.7. version 5->6 . . . . .	67
D.8. Subversion information . . . . .	67

## 1. Introduction

This document describes how to run a DNS Security (DNSSEC)-enabled environment. It is intended for operators who have knowledge of the DNS (see RFC 1034 [1] and RFC 1035 [2]) and want to deploy DNSSEC (RFC 4033 [3], RFC 4034 [4], and RFC 4035 [5]). The focus of the document is on serving authoritative DNS information and is aimed at zone owners, name server operators, registries, registrars and registrants. It assumes that there is no direct relation between those entities and the operators of validating recursive name servers (validators).

During workshops and early operational deployment, operators and system administrators have gained experience about operating the DNS with security extensions (DNSSEC). This document translates these experiences into a set of practices for zone administrators. At the time of writing -the root has just been signed and the first secure delegations are provisioned- there exists relatively little experience with DNSSEC in production environments below the TLD level; this document should therefore explicitly not be seen as representing 'Best Current Practices'. Instead, it describes the decisions that should be made when deploying DNSSEC, gives the choices available for each one, and provides some operational guidelines. The document does not give strong recommendations, that may be subject for a future version of this document.

The procedures herein are focused on the maintenance of signed zones (i.e., signing and publishing zones on authoritative servers). It is intended that maintenance of zones such as re-signing or key rollovers be transparent to any verifying clients.

The structure of this document is as follows. In Section 2, we discuss the importance of keeping the "chain of trust" intact. Aspects of key generation and storage of keys are discussed in Section 3; the focus in this section is mainly on the security of the private part of the key(s). Section 4 describes considerations concerning the public part of the keys. Section 4.1 and Section 4.2 deal with the rollover, or replacement, of keys. Section 4.3 discusses considerations on how parents deal with their children's public keys in order to maintain chains of trust. Section 4.4 covers all kinds of timing issues around keys publication. Section 5 covers the considerations regarding selecting and using NSEC and NSEC3.

The typographic conventions used in this document are explained in Appendix B.

Since this is a document with operational suggestions and there are no protocol specifications, the RFC 2119 [6] language does not apply.

This document obsoletes RFC 4641 [14].

### 1.1. The Use of the Term 'key'

It is assumed that the reader is familiar with the concept of asymmetric keys on which DNSSEC is based (public key cryptography RFC4949 [15]). Therefore, this document will use the term 'key' rather loosely. Where it is written that 'a key is used to sign data' it is assumed that the reader understands that it is the private part of the key pair that is used for signing. It is also assumed that the reader understands that the public part of the key pair is published in the DNSKEY Resource Record and that it is the public part that is used in key exchanges.

### 1.2. Time Definitions

In this document, we will be using a number of time-related terms. The following definitions apply:

- o "Signature validity period" The period that a signature is valid. It starts at the time specified in the signature inception field of the RRSIG RR and ends at the time specified in the expiration field of the RRSIG RR.
- o "Signature publication period" The period that a signature is published. It starts at the time the signature is introduced in the zone for the first time and ends at the time when the signature is removed or replaced with a new signature. After one stops publishing an RRSIG in a zone, it may take a while before the RRSIG has expired from caches and has actually been removed from the DNS.
- o "Key effectivity period" The period during which a key pair is expected to be effective. It is defined as the time between the first inception time stamp and the last expiration date of any signature made with this key, regardless of any discontinuity in the use of the key. The key effectivity period can span multiple signature validity periods.
- o "Maximum/Minimum Zone Time to Live (TTL)" The maximum or minimum value of the TTLs from the complete set of RRs in a zone. Note that the minimum TTL is not the same as the MINIMUM field in the SOA RR. See RFC2308 [9] for more information.

## 2. Keeping the Chain of Trust Intact

Maintaining a valid chain of trust is important because broken chains of trust will result in data being marked as Bogus (as defined in RFC4033 [3] Section 5), which may cause entire (sub)domains to become invisible to verifying clients. The administrators of secured zones need to realize that to verifying clients their zone is, part of a chain of trust.

As mentioned in the introduction, the procedures herein are intended to ensure that maintenance of zones, such as re-signing or key rollovers, will be transparent to the verifying clients on the Internet.

Administrators of secured zones will need to keep in mind that data published on an authoritative primary server will not be immediately seen by verifying clients; it may take some time for the data to be transferred to other (secondary) authoritative nameservers and clients may be fetching data from caching non-authoritative servers. In this light, note that the time for a zonetransfer from master to slave can be negligible when using NOTIFY [8] and incremental transfer (IXFR) [7]. It increases when full zone transfers (AXFR) are used in combination with NOTIFY. It increases even more if you rely on full zone transfers based on only the SOA timing parameters for refresh.

For the verifying clients, it is important that data from secured zones can be used to build chains of trust regardless of whether the data came directly from an authoritative server, a caching nameserver, or some middle box. Only by carefully using the available timing parameters can a zone administrator ensure that the data necessary for verification can be obtained.

The responsibility for maintaining the chain of trust is shared by administrators of secured zones in the chain of trust. This is most obvious in the case of a 'key compromise' when a trade-off must be made between maintaining a valid chain of trust and replacing the compromised keys as soon as possible. Then zone administrators will have to decide, between keeping the chain of trust intact - thereby allowing for attacks with the compromised key - or deliberately breaking the chain of trust and making secured subdomains invisible to security-aware resolvers. (Also see Section 4.2.)

## 3. Keys Generation and Storage

This section describes a number of considerations with respect to the use of keys. For the design of a operational procedure for key generation and storage then a number of decisions need to be made:

- o Does one differentiate between Zone Signing and Key Signing Keys or is the use of one type of key sufficient?
- o Are Key Signing Keys (likely to be) in use as Trust Anchors?
- o What are the timing parameters that are allowed by the operational requirements?
- o What are the cryptographic parameters that fit the operational need?

The following section discusses the considerations that need to be taken into account when making those choices.

### 3.1. Operational Motivation for Zone Signing and Key Signing Keys

The DNSSEC validation protocol does not distinguish between different types of DNSKEYs. The motivations to differentiate between keys are purely operational; validators will not make a distinction.

For operational reasons, described below, it is possible to designate one or more keys to have the role of Key Signing Keys (KSKs). These keys will only sign the apex DNSKEY RRSset in a zone. Other keys can be used to sign all the other RRSets in a zone that require signatures. They are referred to as Zone Signing Keys (ZSKs). In case the differentiation between KSK and ZSK is not made, keys have both the role of KSK and ZSK, we talk about a Single Type signing scheme.

If the two functions are separated then, for almost any method of key management and zone signing, the KSK is used less frequently than the ZSK. Once a key set is signed with the KSK, all the keys in the key set can be used as ZSKs. If there has been an event that increases the risk that a ZSK is compromised it can be simply dropped from the key set. The new key set is then re-signed with the KSK.

Changing a key that is a secure entry point (SEP) for a zone can be relatively expensive as it involves interaction with 3rd parties: When a key is only pointed to by a DS record in the parent zone, one needs to complete the interaction with the responsible registry and wait for the updated DS record to appear in the DNS. In the case where a key is configured as a trust-anchor one has to wait until one has sufficient confidence that all trust anchors have been replaced. In fact, it may be that one is not able to reach the complete user-base with information about the key rollover.

Given the assumption that for KSKs the SEP flag is set, the KSK can be distinguished from a ZSK by examining the flag field in the DNSKEY

RR: If the flag field is an odd number the RR is a KSK; otherwise it is a ZSK.

There is also a risk that keys are compromised through theft or loss. For keys that are installed on file-systems of nameservers that are connected to the network (e.g. for dynamic updates) that risk is relatively high. Where keys are stored on Hardware Security Modules (HSMs) or stored off-line, such risk is relatively low. However, storing keys off-line or with more limitation on access control has a negative effect on the operational flexibility. By separating the KSK and ZSK functionality these risks can be managed while making the tradeoff against the costs involved. For example, a KSK can be stored off-line or with more limitation on access control than ZSKs which need to be readily available for operational purposes such as the addition or deletion of zone data. A KSK stored on a smartcard, that is kept in a safe, combined with a ZSK stored on a filesystem accessible by operators for daily routine may provide a better protection against key compromise, without losing much operational flexibility. It must be said that some HSMs give the option to have your keys online, giving more protection and hardly affecting the the operational flexibility. In those cases, a KSK-ZSK split is not more beneficial than the Single-Type signing scheme.

Finally there is a risk of cryptanalysis of the key material. The costs of such analysis are correlated to the length of the key. However, cryptanalysis arguments provide no strong motivation for a KSK/ZSK split. Suppose one differentiates between a KSK and a ZSK whereby the KSK effectivity period is  $X$  times the ZSK effectivity period. Then, in order for the resistance to cryptanalysis to be the same for the KSK and the ZSK, the KSK needs to be  $X$  times stronger than the ZSK. Since for all practical purposes  $X$  will somewhere of the order of 10 to 100, the associated key sizes will vary only about a byte in size for symmetric keys. When translated to asymmetric keys, is still too insignificant a size difference to warrant a key-split; it only marginally affects the packet size and signing speed.

The arguments for differentiation between the ZSK and KSK are weakest when:

- o the exposure to risk is low (e.g. when keys are stored on HSMs);
- o one can be certain that a key is not used as a trust-anchor;
- o maintenance of the various keys cannot be performed through tools (is prone to human error); and
- o the interaction through the registrar-registry provisioning chain -- in particular the timely appearance of a new DS record in the

parent zone in emergency situations -- is predictable.

If the above holds then the costs of the operational complexity of a KSK-ZSK split may outweigh the costs of operational flexibility and choosing a single type signing scheme is a reasonable option. In other cases we advise that the separation between KSKs and ZSKs is made and that the SEP flag is exclusively set on KSKs.

### 3.2. Practical Consequences of KSK and ZSK Separation

A key that acts only as a Zone Signing Key can be used to sign all the data but the DNSKEY RRset in a zone on a regular basis. When a ZSK is to be rolled, no interaction with the parent is needed. This allows for signature validity periods on the order of days.

A key with only the Key Signing Key role is to be used to sign the DNSKEY RRs in a zone. If a KSK is to be rolled, there may be interactions with other parties. If there is a parent zone, these can include the registry of the parent zone or administrators of verifying resolvers that have the particular key configured as secure entry points. In the latter case, everyone relying on the trust anchor needs to roll over to the new key, a process that may be subject to stability costs if automated trust-anchor rollover mechanisms (such as e.g. RFC5011 [16]) are not in place. Hence, the key effectivity period of these keys can and should be made much longer.

#### 3.2.1. Rolling a KSK that is not a trust-anchor

There are 3 schools of thought on rolling a KSK that is not a trust anchor:

- o It should be done frequently and regularly (possibly every few months) so that a key rollover remains an operational routine.
- o It should be done frequently but irregularly. Frequently meaning every few months, again based on the argument that a rollover is a practiced and common operational routine, and irregular meaning with a large jitter, so that 3rd parties do not start to rely on the key and will not be tempted to configure it as a trust-anchor.
- o It should only be done when it is known or strongly suspected that the key can be or has been compromised.

There is no widespread agreement on which of these three schools of thought is better for different deployments of DNSSEC. There is a stability cost every time a non-anchor KSK is rolled over, but it is possibly low if the communication between the child and the parent is

good. On the other hand, the only completely effective way to tell if the communication is good is to test it periodically. Thus, rolling a KSK with a parent is only done for two reasons: to test and verify the rolling system to prepare for an emergency, and in the case of (preventing) an actual emergency.

Finally, in most cases a zone owner cannot be fully certain that the zone's KSK is not in use as a trust-anchor somewhere. While the configuration of trust-anchors is not the responsibility of the zone owner there may be stability costs for the validator administrator that (wrongfully) configured the trust-anchor when the zone owner roles a KSK.

### 3.2.2. Rolling a KSK that is a trust-anchor

The same operational concerns apply to the rollover of KSKs that are used as trust-anchors: if a trust anchor replacement is done incorrectly, the entire domain that the trust anchor covers will become bogus until the trust anchor is corrected.

In a large number of cases it will be safe to work from the assumption that one's keys are not in use as trust-anchors. If a zone owner publishes a "DNSSEC Signing Policy and Practice Statement" [25] that should be explicit about the fact whether the existence of trust anchors will be taken into account in any way or not. There may be cases where local policies enforce the configuration of trust-anchors on zones which are mission critical (e.g. in enterprises where the trust-anchor for the enterprise domain is configured in the enterprise's validator) It is expected that the zone owners are aware of such circumstances.

One can argue that because of the difficulty of getting all users of a trust anchor to replace an old trust anchor with a new one, a KSK that is a trust anchor should never be rolled unless it is known or strongly suspected that the key has been compromised. In other words the costs of a KSK rollover are prohibitively high because some users cannot be reached.

However, the "operational habit" argument also applies to trust anchor reconfiguration at the clients' validators. If a short key effectivity period is used and the trust anchor configuration has to be revisited on a regular basis, the odds that the configuration tends to be forgotten is smaller. In fact, the costs for those users can be minimized by automating the rollover RFC5011 [16] and by rolling the key regularly (and advertising such) so that the operators of recursive nameservers will put the appropriate mechanism in place to deal with these stability costs, or, in other words, budget for these costs instead of incurring them unexpectedly.



It is therefore recommended, to roll KSKs that are likely to be used as trust-anchors, on a regular basis if and only if those rollovers can be tracked using standardized (e.g. RFC5011) mechanisms.

### 3.2.3. The use of the SEP flag

The so-called Secure Entry Point (SEP) [5] flag can be used to distinguish between keys that are intended to be used as the secure entry point into the zone when building chains of trust, e.g they are (to be) pointed to by parental DS RRs or configured as a trust-anchor.

While the SEP flag does not play any role in the failure it is used in practice for operational purposes such as for the rollover mechanism described in RFC5011 [16]. The common convention is to set the SEP flag on any key that is used for key exchanges with the parent and/or potentially used for configuration as a trust anchor. Therefore it is recommended that the SEP flag is set on keys that are used as KSKs and not on keys that are used as ZSKs, while in those cases where a distinction between KSK and ZSK is not made (i.e. for a Single Type signing scheme) it is recommended that the SEP flag is set on all keys.

Note that signing tools may assume a KSK/ZSK split and use the (non) presence of the SEP flag to determine which key is to be used for signing zone data; these tools may get confused when a single type signing scheme is used.

### 3.3. Key Effectivity Period

In general the available key length sets an upper limit on the Key Effectivity Period. For all practical purposes it is sufficient to define the Key Effectivity Period based on purely operational requirements and match the key length to that value. Ignoring the operational perspective, a reasonable effectivity period for KSKs that have corresponding DS records in the parent zone is of the order of 2 decades or longer. That is, if one does not plan to test the rollover procedure, the key should be effective essentially forever, and only rolled over in case of emergency.

When one chooses for a regular key-rollover, a reasonable key effectivity period for KSKs that have a parent zone is one year, meaning you have the intent to replace them after 12 months. The key effectivity period is merely a policy parameter, and should not be considered a constant value. For example, the real key effectivity period may be a little bit longer than 12 months, because not all actions needed to complete the rollover could be finished in time.

As argued above, this annual rollover gives operational practice of rollovers for both the zone and validator administrators. Besides, in most environments a year is a time-span that is easily planned and communicated.

Where keys are stored on on-line systems and the exposure to various threats of compromise is fairly high, an intended key effectivity period of a month is reasonable for Zone Signing Keys.

Although very short key effectivity periods are theoretically possible, when replacing keys one has to take into account the rollover considerations from Section 4.1 and Section 4.4. Key replacement endures for a couple of Zone TTLs, depending on the rollover scenario. Therefore, a multiple of Zone TTL is a reasonable lower limit on the key effectivity period. Forcing a smaller key effectivity period will result your zone to have a ever-growing keyset.

The motivation for having the ZSK's effectivity period shorter than the KSK's effectivity period is rooted in the operational consideration that it is more likely that operators have more frequent read access to the ZSK than to the KSK. If ZSK's are maintained on cryptographic Hardware Security Modules (HSM) than the motivation to have different key effectivity periods is weakened.

In fact, if the risk of loss, theft or other compromise is the same for a zone and key signing key there is little reason to choose different effectivity periods for ZSKs and KSKs. And when the split between ZSKs and KSKs is not made, the argument is redundant.

There are certainly cases (e.g. where the the costs and risk of compromise, and the costs and risks involved with having to perform an emergency roll are also low) that the use of a single type signing scheme with a long key effectivity period is a good choice.

### 3.4. Cryptographic Considerations

#### 3.4.1. Key Algorithm

At the time of writing, there are three types of signature algorithms that can be used in DNSSEC: RSA, DSA and GOST. Proposals for other algorithms are in the making. All three are fully specified in many freely-available documents, and are widely considered to be patent-free. The creation of signatures with RSA and DSA takes roughly the same time, but DSA is about ten times slower for signature verification. Also, DSA in context of DNSSEC is limited to the maximum of 1024 bit keys.

We suggest the use of RSA/SHA-256 as the preferred signature algorithms and RSA/SHA-1 as an alternative. Both have advantages and disadvantages. RSA/SHA-1 has been deployed for many years, while RSA/SHA-256 has only begun to be deployed. On the other hand, it is expected that if effective attacks on either algorithm appear, they will appear for RSA/SHA-1 first. RSA/MD5 should not be considered for use because RSA/MD5 will very likely be the first common-use signature algorithm to have an effective attack.

At the time of publication, it is known that the SHA-1 hash has cryptanalysis issues and work is in progress on addressing them. We recommend the use of public key algorithms based on hashes stronger than SHA-1 (e.g., SHA-256) as soon as these algorithms are available in implementations (see RFC5702 [23] and RFC4509 [20]).

#### 3.4.2. Key Sizes

This section assumes RSA keys, as suggested in the previous section.

DNSSEC signing keys should be large enough to avoid all known cryptographic attacks during the effectivity period of the key. To date, despite huge efforts, no one has broken a regular 1024-bit key; in fact, the best completed attack is estimated to be the equivalent of a 700-bit key. An attacker breaking a 1024-bit signing key would need to expend phenomenal amounts of networked computing power in a way that would not be detected in order to break a single key. Because of this, it is estimated that most zones can safely use 1024-bit keys for at least the next ten years (A 1024-bit asymmetric key has an approximate equivalent strength of a symmetric 80-bit key).

Depending on local policy (e.g. owners of keys that are used as extremely high value trust anchors, or non-anchor keys that may be difficult to roll over), you may want to use lengths longer than 1024 bits. Typically, the next larger key size used is 2048 bits, which has the approximate equivalent strength of a symmetric 112-bit key (e.g. RFC3766 [12]). Signing and verifying with a 2048-bit key takes of course longer than with a 1024-bit key. The increase depends on software and hardware implementations, but public operations (such as verification) are about four times slower, while private operations (such as signing) slow down about eight times.

Another way to decide on the size of key to use is to remember that the effort it takes for an attacker to break a 1024-bit key is the same regardless of how the key is used. If an attacker has the capability of breaking a 1024-bit DNSSEC key, he also has the capability of breaking one of the many 1024-bit TLS trust anchor keys that are currently installed in web browsers. If the value of a DNSSEC key is lower to the attacker than the value of a TLS trust

anchor, the attacker will use the resources to attack the latter.

It is possible that there will be an unexpected improvement in the ability for attackers to break keys, and that such an attack would make it feasible to break 1024-bit keys but not 2048-bit keys. If such an improvement happens, it is likely that there will be a huge amount of publicity, particularly because of the large number of 1024-bit TLS trust anchors build into popular web browsers. At that time, all 1024-bit keys (both ones with parent zones and ones that are trust anchors) can be rolled over and replaced with larger keys.

Earlier documents (including the previous version of this document) urged the use of longer keys in situations where a particular key was "heavily used". That advice may have been true 15 years ago, but it is not true today when using RSA algorithms and keys of 1024 bits or higher.

#### 3.4.3. Private Key Storage

It is recommended that, where possible, zoneprivate keys and the zone file master copy that is to be signed be kept and used in off-line, non-network-connected, physically secure machines only. Periodically, an application can be run to add authentication to a zone by adding RRSIG and NSEC/NSEC3 RRs. Then the augmented file can be transferred.

When relying on dynamic update [10], or any other update mechanism that runs at a regular interval to manage a signed zone, be aware that at least one private key of the zone will have to reside on the master server (or reside on an HSM to which the server has access). This key is only as secure as the amount of exposure the server receives to unknown clients and the security of the host. Although not mandatory, one could administer a zone using a "hidden master" scheme that minimize the risk. In this arrangement the master that processes the updates is unavailable from general hosts on the Internet; it is not listed in the NS RRSets, although its name appears in the SOA RRs MNAME field. The nameservers in the NS RRSets are able to receive zone updates through IXFR, AXFR, or an out-of-band distribution mechanism, possibly in combination with NOTIFY or another mechanism to trigger zone replication.

The ideal situation is to have a one-way information flow to the network to avoid the possibility of tampering from the network. Keeping the zone master on-line on the network and simply cycling it through an off-line signer does not do this. The on-line version could still be tampered with if the host it resides on is compromised. For maximum security, the master copy of the zone file should be off-net and should not be updated based on an unsecured

network mediated communication.

The ideal situation may not be achievable because of economic tradeoffs between risks and costs. For instance, keeping a zone file off-line is not practical and will increase the costs of operating a DNS zone. So in practice the machines on which zone files are maintained will be connected to a network. Operators are advised to take security measures to shield unauthorized access to the master copy in order to prevent modification of DNS data before its signed.

Similarly the choice for storing a private key in an HSM will be influenced by a tradeoff between various concerns:

- o The risks that an unauthorized person has unnoticed read-access to the private key
- o The remaining window of opportunity for the attacker.
- o The economic impact of the possible attacks (for a TLD that impact will typically be higher than for an individual users).
- o The costs of rolling the (compromised) keys. (The costs of rolling a ZSK is lowest and the costs of rolling a KSK that is in wide use as a trust anchor is highest.)
- o The costs of buying and maintaining an HSM.

For dynamically updated secured zones [10], both the master copy and the private key that is used to update signatures on updated RRs will need to be on-line.

#### 3.4.4. Key Generation

Careful generation of all keys is a sometimes overlooked but is an absolutely essential element in any cryptographically secure system. The strongest algorithms used with the longest keys are still of no use if an adversary can guess enough to lower the size of the likely key space so that it can be exhaustively searched. Technical suggestions for the generation of random keys will be found in RFC 4086 [13] and NIST SP 800-90 [19]. In particular, one should carefully assess whether the random number generator used during key generation adheres to these suggestions. Typically, HSMs tend to provide a good facility for key generation.

Keys with a long effectivity period are particularly sensitive as they will represent a more valuable target and be subject to attack for a longer time than short-period keys. It is strongly recommended that long-term key generation occur off-line in a manner isolated

from the network via an air gap or, at a minimum, high-level secure hardware.

#### 3.4.5. Differentiation for 'High-Level' Zones?

In an earlier version of this document (RFC4641 [14]) we made a differentiation between key lengths for KSKs used for zones that are high in the DNS hierarchy and those for KSKs used low down.

This distinction is now considered not relevant. Longer key lengths for keys higher in the hierarchy are not useful because the cryptographic guidance is that everyone should use keys that no one can break. Also, it is impossible to judge which zones are more or less valuable to an attacker. An attack can only take place if the key compromise goes unnoticed and the attacker can act as a man-in-the-middle (MITM). For example if example.com is compromised and the attacker forges answers for somebank.example.com. and sends them out during an MITM, when the attack is discovered it will be simple to prove that example.com has been compromised and the KSK will be rolled. Designing a long-term successful attack is difficult for keys at any level.

### 4. Signature Generation, Key Rollover, and Related Policies

#### 4.1. Key Rollovers

Regardless of whether a zone uses periodic key rollovers in order to practice for emergencies, or only rolls over keys in an emergency, key rollovers are a fact of life when using DNSSEC. Zone administrators who are in the process of rolling their keys have to take into account that data published in previous versions of their zone still lives in caches. When deploying DNSSEC, this becomes an important consideration; ignoring data that may be in caches may lead to loss of service for clients.

The most pressing example of this occurs when zone material signed with an old key is being validated by a resolver that does not have the old zone key cached. If the old key is no longer present in the current zone, this validation fails, marking the data "Bogus". Alternatively, an attempt could be made to validate data that is signed with a new key against an old key that lives in a local cache, also resulting in data being marked "Bogus".

##### 4.1.1. Zone Signing Key Rollovers

If the choice for splitting zone and key signing keys has been made than those two types of keys can be rolled separately and zone signing keys can be rolled without taking into account DS records

from the parent or the configuration of such a key as trust-anchor.

For "Zone Signing Key rollovers", there are two ways to make sure that during the rollover data still cached can be verified with the new key sets or newly generated signatures can be verified with the keys still in caches. One schema, described in Section 4.1.1.2, uses double signatures; the other uses key pre-publication (Section 4.1.1.1). The pros, cons, and recommendations are described in Section 4.1.1.3.

#### 4.1.1.1. Pre-Publish Zone Signing Key Rollover

This section shows how to perform a ZSK rollover without the need to sign all the data in a zone twice -- the "Pre-Publish key rollover". This method has advantages in the case of a key compromise. If the old key is compromised, the new key has already been distributed in the DNS. The zone administrator is then able to quickly switch to the new key and remove the compromised key from the zone. Another major advantage is that the zone size does not double, as is the case with the Double Signature ZSK rollover.

Pre-Publish key rollover involves four stages as follows:

initial	new DNSKEY	new RRSIGs
SOA_0 RRSIG_Z_10(SOA)	SOA_1 RRSIG_Z_10(SOA)	SOA_2 RRSIG_Z_11(SOA)
DNSKEY_K_1 DNSKEY_Z_10	DNSKEY_K_1 DNSKEY_Z_10 DNSKEY_Z_11	DNSKEY_K_1 DNSKEY_Z_10 DNSKEY_Z_11
RRSIG_K_1(DNSKEY) RRSIG_Z_10(DNSKEY)	RRSIG_K_1(DNSKEY) RRSIG_Z_10(DNSKEY)	RRSIG_K_1(DNSKEY) RRSIG_Z_11(DNSKEY)
-----		
DNSKEY removal		
-----		
SOA_3 RRSIG_Z_11(SOA)		
DNSKEY_K_1 DNSKEY_Z_11		
RRSIG_K_1(DNSKEY) RRSIG_Z_11(DNSKEY)		
-----		

Figure 1: Pre-Publish Key Rollover

initial: Initial version of the zone: DNSKEY\_K\_1 is the Key Signing Key. DNSKEY\_Z\_10 is used to sign all the data of the zone, the Zone Signing Key.

new DNSKEY: DNSKEY\_Z\_11 is introduced into the key set. Note that no signatures are generated with this key yet, but this does not secure against brute force attacks on the public key. The minimum duration of this pre-roll phase is the time it takes for the data to propagate to the authoritative servers plus TTL value of the key set.

new RRSIGs: At the "new RRSIGs" stage (SOA serial 2), DNSKEY\_Z\_11 is used to sign the data in the zone exclusively (i.e., all the signatures from DNSKEY\_Z\_10 are removed from the zone). DNSKEY\_Z\_10 remains published in the key set. This way data that was loaded into caches from version 1 of the zone can still be verified with key sets fetched from version 2 of the zone. The



minimum time that the key set including DNSKEY\_Z\_10 is to be published is the time that it takes for zone data from the previous version of the zone to expire from old caches, i.e., the time it takes for this zone to propagate to all authoritative servers plus the Maximum Zone TTL value of any of the data in the previous version of the zone.

DNSKEY removal: DNSKEY\_Z\_10 is removed from the zone. The key set, now only containing DNSKEY\_K\_1 and DNSKEY\_Z\_11, is re-signed with the DNSKEY\_K\_1 and DNSKEY\_Z\_11.

The above scheme can be simplified by always publishing the "future" key immediately after the rollover. The scheme would look as follows (we show two rollovers); the future key is introduced in "new DNSKEY" as DNSKEY\_Z\_12 and again a newer one, numbered 13, in "new DNSKEY (II)":

initial	new RRSIGs	new DNSKEY
SOA_0	SOA_1	SOA_2
RRSIG_Z_10(SOA)	RRSIG_Z_11(SOA)	RRSIG_Z_11(SOA)
DNSKEY_K_1	DNSKEY_K_1	DNSKEY_K_1
DNSKEY_Z_10	DNSKEY_Z_10	DNSKEY_Z_11
DNSKEY_Z_11	DNSKEY_Z_11	DNSKEY_Z_12
RRSIG_K_1(DNSKEY)	RRSIG_K_1 (DNSKEY)	RRSIG_K_1(DNSKEY)
RRSIG_Z_10(DNSKEY)	RRSIG_Z_11(DNSKEY)	RRSIG_Z_11(DNSKEY)
-----		
-----		
new RRSIGs (II)	new DNSKEY (II)	
SOA_3	SOA_4	
RRSIG_Z_12(SOA)	RRSIG_Z_12(SOA)	
DNSKEY_K_1	DNSKEY_K_1	
DNSKEY_Z_11	DNSKEY_Z_12	
DNSKEY_Z_12	DNSKEY_Z_13	
RRSIG_K_1(DNSKEY)	RRSIG_K_1(DNSKEY)	
RRSIG_Z_12(DNSKEY)	RRSIG_Z_12(DNSKEY)	
-----		

Figure 2: Pre-Publish Zone Signing Key Rollover, Showing Two Rollovers

Note that the key introduced in the "new DNSKEY" phase is not used

for production yet; the private key can thus be stored in a physically secure manner and does not need to be 'fetched' every time a zone needs to be signed.

#### 4.1.1.2. Double Signature Zone Signing Key Rollover

This section shows how to perform a ZSK key rollover using the double zone data signature scheme, aptly named "Double Signature rollover".

During the "new DNSKEY" stage the new version of the zone file will need to propagate to all authoritative servers and the data that exists in (distant) caches will need to expire, requiring at least the Maximum Zone TTL.

Double Signature ZSK rollover involves three stages as follows:

initial	new DNSKEY	DNSKEY removal
SOA_0	SOA_1	SOA_2
RRSIG_Z_10(SOA)	RRSIG_Z_10(SOA)	RRSIG_Z_11(SOA)
	RRSIG_Z_11(SOA)	
DNSKEY_K_1	DNSKEY_K_1	DNSKEY_K_1
DNSKEY_Z_10	DNSKEY_Z_10	
	DNSKEY_Z_11	DNSKEY_Z_11
RRSIG_K_1(DNSKEY)	RRSIG_K_1(DNSKEY)	RRSIG_K_1(DNSKEY)
RRSIG_Z_10(DNSKEY)	RRSIG_Z_10(DNSKEY)	
	RRSIG_Z_11(DNSKEY)	RRSIG_Z_11(DNSKEY)

Figure 3: Double Signature Zone Signing Key Rollover

initial: Initial Version of the zone: DNSKEY\_K\_1 is the Key Signing Key. DNSKEY\_Z\_10 is used to sign all the data of the zone, the Zone Signing Key.

new DNSKEY: At the "New DNSKEY" stage (SOA serial 1) DNSKEY\_Z\_11 is introduced into the key set and all the data in the zone is signed with DNSKEY\_Z\_10 and DNSKEY\_Z\_11. The rollover period will need to continue until all data from version 0 of the zone has expired from remote caches. This will take at least the Maximum Zone TTL of version 0 of the zone.

DNSKEY removal: DNSKEY\_Z\_10 is removed from the zone. All the signatures from DNSKEY\_Z\_10 are removed from the zone. The key set, now only containing DNSKEY\_Z\_11, is re-signed with DNSKEY\_K\_1 and DNSKEY\_Z\_11.

At every instance, RRSIGs from the previous version of the zone can be verified with the DNSKEY RRSIG from the current version and the other way around. The data from the current version can be verified with the data from the previous version of the zone. The duration of the "new DNSKEY" phase and the period between rollovers should be at least the Maximum Zone TTL.

Making sure that the "new DNSKEY" phase lasts until the signature expiration time of the data in the initial version of the zone is recommended. This way all caches are cleared of the old signatures. However, this duration could be considerably longer than the Maximum Zone TTL, making the rollover a lengthy procedure.

Note that in this example we assumed that the zone was not modified during the rollover. New data can be introduced in the zone as long as it is signed with both keys.

#### 4.1.1.3. Pros and Cons of the Schemes

Pre-Publish key rollover: This rollover does not involve signing the zone data twice. Instead, before the actual rollover, the new key is published in the key set and thus is available for cryptanalysis attacks. A small disadvantage is that this process requires four steps. Also the Pre-Publish scheme involves more parental work when used for KSK rollovers as explained in Section 4.1.3.

Double Signature ZSK rollover: The drawback of this signing scheme is that during the rollover the number of signatures in your zone doubles; this may be prohibitive if you have very big zones. An advantage is that it only requires three steps.

#### 4.1.2. Key Signing Key Rollovers

For the rollover of a Key Signing Key, the same considerations as for the rollover of a Zone Signing Key apply. However, we can use a Double Signature scheme to guarantee that old data (only the apex key set) in caches can be verified with a new key set and vice versa. Since only the key set is signed with a KSK, zone size considerations do not apply.

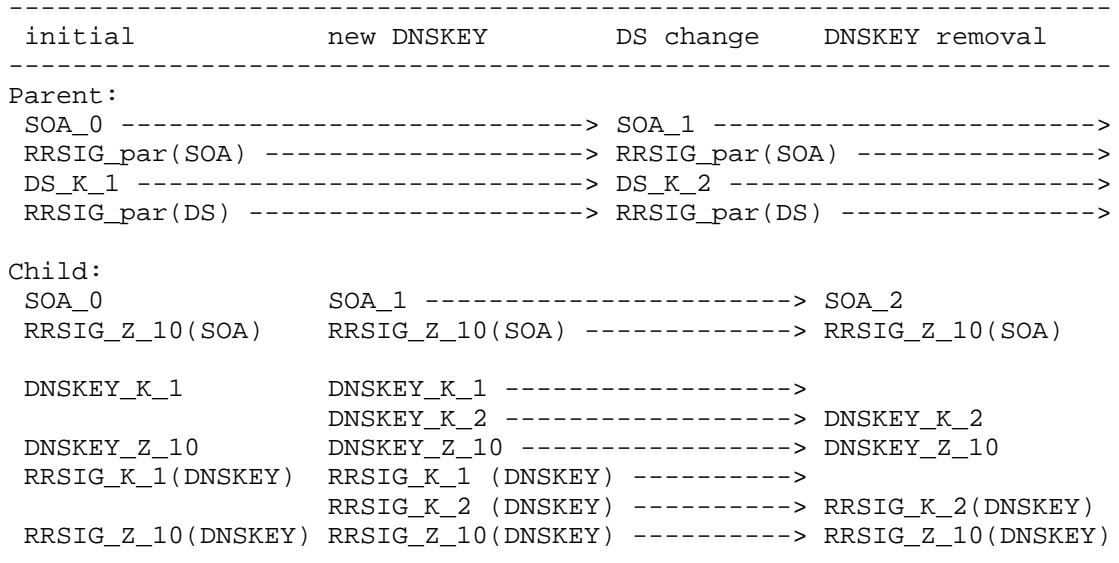


Figure 4: Stages of Deployment for a Double Signature Key Signing Key Rollover

initial: Initial version of the zone. The parental DS points to DNSKEY\_K\_1. Before the rollover starts, the child will have to verify what the TTL is of the DS RR that points to DNSKEY\_K\_1 -- it is needed during the rollover and we refer to the value as TTL\_DS.

new DNSKEY: During the "new DNSKEY" phase, the zone administrator generates a second KSK, DNSKEY\_K\_2. The key is provided to the parent, and the child will have to wait until a new DS RR has been generated that points to DNSKEY\_K\_2. After that DS RR has been published on all servers authoritative for the parent's zone, the zone administrator has to wait at least TTL\_DS to make sure that the old DS RR has expired from caches.

DS change: The parent replaces DS\_K\_1 with DS\_K\_2.

DNSKEY removal: DNSKEY\_K\_1 has been removed.

The scenario above puts the responsibility for maintaining a valid chain of trust with the child. It also is based on the premise that the parent only has one DS RR (per algorithm) per zone. An alternative mechanism has been considered. Using an established trust relation, the interaction can be performed in-band, and the removal of the keys by the child can possibly be signaled by the

parent. In this mechanism, there are periods where there are two DS RRs at the parent. Since at the moment of writing the protocol for this interaction has not been developed, further discussion is out of scope for this document.

#### 4.1.2.1. Special Considerations for RFC5011 KSK rollover

The scenario sketched above assumes that the KSK is not in use as a trust-anchor too but that validating nameservers exclusively depend on the parental DS record to establish the zone's security. If it is known that validating nameservers have configured trust-anchors then such needs to be taken into account. Here we assume that operators of zones will deploy RFC5011 [16] style rollovers.

RFC5011 style rollovers increase the duration of key rollovers: the key to be removed must first be revoked. Thus, before the DNSKEY\_K\_1 removal phase, DNSKEY\_K\_1 must be published for one more Maximum Zone TTL with the REVOKE bit set. The revoked key must be self-signed, so in this phase the DNSKEY RRset must also be signed with DNSKEY\_K\_1.

#### 4.1.3. Difference Between ZSK and KSK Rollovers

Note that KSK rollovers and ZSK rollovers are different in the sense that a KSK rollover requires interaction with the parent (and possibly replacing of trust anchors) and the ensuing delay while waiting for it.

A zone key rollover can be handled in two different ways, meaningful: Pre-Publish (Section 4.1.1.1) and Double Signature (Section 4.1.1.2).

As the KSK is used to validate the key set and because the KSK is not changed during a ZSK rollover, a cache is able to validate the new key set of the zone. A Pre-Publish method is also possible for KSKs, known as the Double-DS rollover. The name being a give away, the record that needs to be pre-published is the DS RR at the parent. The Pre-Publish method has some drawbacks for KSKs. We first describe the rollover scheme and then indicate these drawbacks.

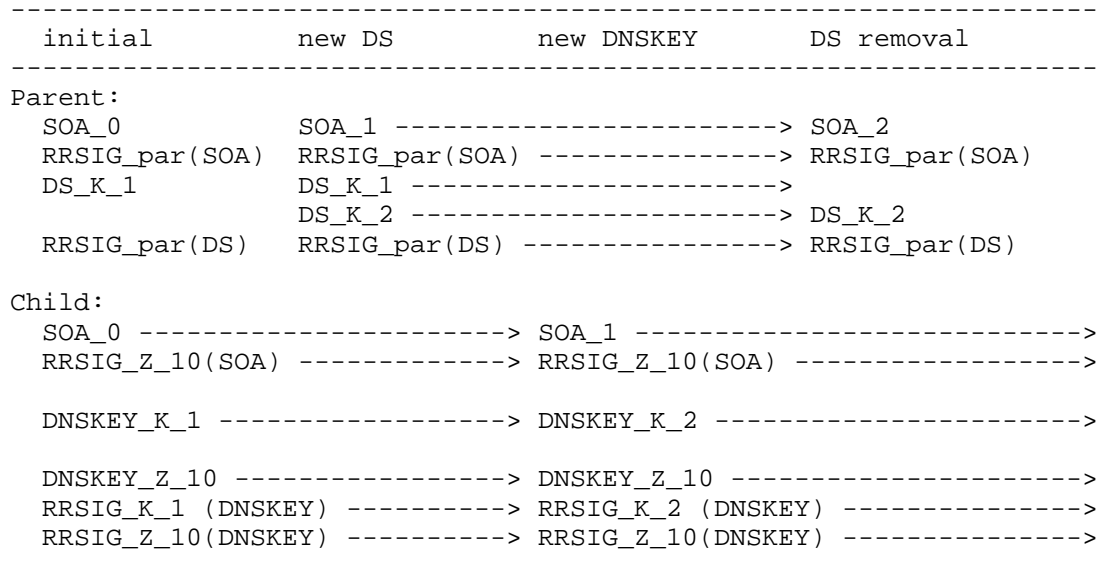


Figure 5: Stages of Deployment for a Double-DS Key Signing Key Rollover

When the child zone wants to roll, it notifies the parent during the "new DS" phase and submits the new key (or the corresponding DS) to the parent. The parent publishes DS\_K\_1 and DS\_K\_2, pointing to DNSKEY\_K\_1 and DNSKEY\_K\_2, respectively. During the rollover ("new DNSKEY" phase), which can take place as soon as the new DS set propagated through the DNS, the child replaces DNSKEY\_K\_1 with DNSKEY\_K\_2. Immediately after that ("DS/DNSKEY removal" phase), it can notify the parent that the old DS record can be deleted.

The drawbacks of this scheme are that during the "new DS" phase the parent cannot verify the match between the DS\_K\_2 RR and DNSKEY\_K\_2 using the DNS -- as DNSKEY\_K\_2 is not yet published. Besides, we introduce a "security lame" key (see Section 4.3.3). Finally, the child-parent interaction consists of two steps. The "Double Signature" method only needs one interaction.

#### 4.1.4. Rollover for a Single Type Signing Key rollover

The rollover of a DNSKEY when a Single Type Signing scheme is used is subject to the same requirement as the rollover of a KSK or ZSK: During any stage of the rollover the chain of trust needs to continue to validate for any combination of data in the zone as well as data that may still live in distant caches.

There are two variants for this rollover. Since the choice for a Single Type Signing scheme is motivated by operational simplicity we first describe the most straightforward rollover scheme first.

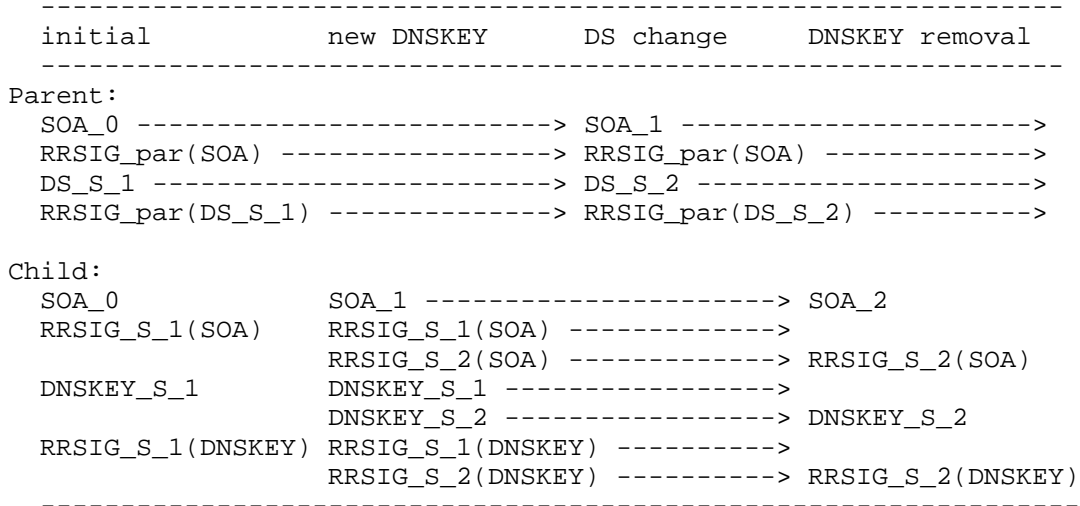


Figure 6: Stages of the Straightforward rollover in a Single Type Signing scheme

initial: Parental DS points to DNSKEY\_K\_1. All RR sets in the zone are signed with DNSKEY\_K\_1.

new DNSKEY: A new key (DNSKEY\_K\_2) is introduced and all the RR sets are signed with both DNSKEY\_K\_1 and DNSKEY\_K\_2.

DS change: After the DNSKEY RRset with the two keys had time to propagate into distant caches (that is the key set exclusively containing DNSKEY\_K\_1 has been expired) the parental DS record can be changed.

DNSKEY removal: After the DS RRset containing DS\_K\_1 has expired from distant caches DNSKEY\_K\_1 can be removed from the DNSKEY RRset .

There is a second variety of this rollover during which one introduces a new DNSKEY into the key set and signs the keyset with both keys while signing the zone data with only the original DNSKEY\_K\_1. One replaces the DNSKEY\_K\_1 signatures with signatures made with DNSKEY\_K\_2 at the moment of DNSKEY\_K\_1 removal.

The second variety of this rollover can be considered when zone size considerations prevent the introduction of double signatures over all of the zone data although in that case choosing for a KSK/ZSK split may be a better option.

A Double-DS rollover scheme is compatible with a rollover using a Single Type signing scheme although in order to maintain a valid chain of trust the zone data would need to be published with a double signatures or a double keyset would need to be published. Since this leads to increase in zone and packet size at both child and parent there are little benefits to a Double-DS rollover with a Single Type signing scheme.

#### 4.1.5. Algorithm rollovers

A special class of key rollover is the one needed for a change of key algorithms (either adding a new algorithm, removing an old algorithm, or both). Additional steps are needed to retain integrity during this rollover. We first describe the generic case, special considerations for rollovers that involve trust-anchors and single type keys are discussed below.

There exist a conservative and a liberal approach for algorithm rollover. This has to do with section 2.2 in RFC4035 [5]:

There MUST be an RRSIG for each RRset using at least one DNSKEY of each algorithm in the zone apex DNSKEY RRset. The apex DNSKEY RRset itself MUST be signed by each algorithm appearing in the DS RRset located at the delegating parent (if any).

The conservative approach interprets this section very strict, meaning that it expects that all RRset has a valid signature for every algorithm signalled by the zone apex DNSKEY RRset, no matter where this RRset is kept. Important to know is that this also includes the resolvers cache. The liberal approach uses a more loose interpretation of the section and limits the rule to RRsets in the zone at the authoritative name servers. There is a reasonable argument for saying that this is valid, because the specific section is a subsection of section 2. in RFC4035: Zone Signing.

When following the more liberal approach, algorithm rollover is just as easy as a regular Double-Signature KSK rollover (Section 4.1.2). Note that the Double-DS rollover method cannot be used, since that would introduce a parental DS of which the apex DNSKEY RRset has not been signed with the introduced algorithm.



However, there are implementations of validators known that follow the more conservative approach. Performing a Double-Signature KSK algorithm rollover will temporarily make your zone appear as Bogus by such validators during the rollover. Therefore, the rollover in this section will explain the stages of deployment assuming the conservative approach.

When adding a new algorithm, the signatures should be added first. After the TTL of RRSIGS has expired, and caches have dropped the old data covered by those signatures, the DNSKEY with the new algorithm can be added.

After the new algorithm has been added, the DS record can be exchanged using Double Signature Key Rollover. You cannot use Pre-Publish key rollover method when you do key algorithm rollover.

When removing an old algorithm, the DNSKEY should be removed first, but only after the DS for the old algorithm was removed from the parent zone.

Figure 7 describes the steps. The underscored number indicates the algorithm and ZSK and KSK indicate the obvious difference in key use. For example `DNSKEY_KSK_1` is a the DNSKEY RR representing the public part of the old key signing key of algorithm type 1 while `RRSIG_ZSK_2(SOA)` is the RRSIG RR made with the private part of the new zone signing key of algorithm type 2 over a SOA RR. It is assumed that the key that signs the SOA RR also signes all other non-DNSKEY RRset data.

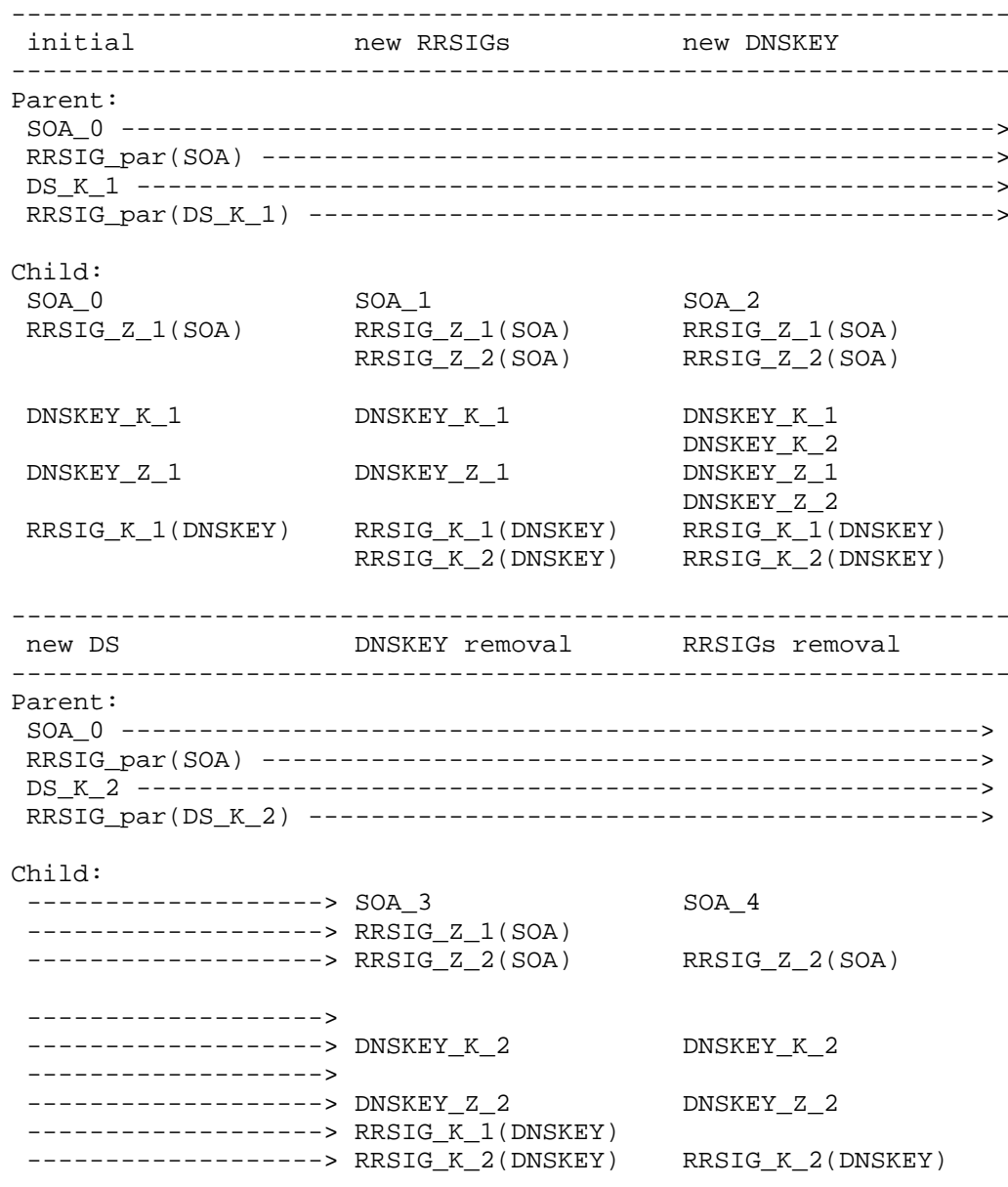


Figure 7: Stages of Deployment during an Algorithm Rollover

initial: Describes state of the zone before any transition is done. Number of the keys may vary, but the algorithm of keys in the zone is same for all DNSKEY records.

new RRSIGs: The signatures made with the new key over all records in the zone are added, but the key itself is not. This includes the signature for the DNSKEY RRset. While in theory, the signatures of the keyset should always be synchronized with the keyset itself, it can be possible that RRSIGs are requested separately, so it is prudent to also sign the DNSKEY set with the new signature.

This step is needed to propagate the signatures created with the new algorithm to the caches. If you do not do that, it might happen that the resolver picks up the new DNSKEY RRset (with the new algorithm included), but still have the old list of signatures stored.

new DNSKEY: After the cache data has expired, the new key can be added to the zone.

new DS: After the cache data for the DNSKEY has expired, the DS record for the new key can be added to the parent zone and the DS record for the old key can be removed in the same step.

DNSKEY removal: After the cache data for the DS has expired, the old algorithm can be removed. This time the key needs to be removed first, before removing the signatures.

RRSIGs removal: After the cache data for the DNSKEY has expired, the signatures can also be removed during this step.

Below we deal with a few special cases of algorithm rollovers.

- 1: Single Type Signing Scheme Algorithm Rollover : when you have chosen not to differentiate between Zone and Key signing keys (Section 4.1.5.1)
- 2: RFC5011 Algorithm Rollover : when trust-anchors can track the roll via RFC5011 style rollover (Section 4.1.5.2)
- 3: 1 and 2 combined : when a Single Type Signing Scheme Algorithm rollover is RFC5011-enabled (Section 4.1.5.3)

In addition to the narrative below these special cases are represented in Figure 11, Figure 12 and Figure 13 in Appendix C.

#### 4.1.5.1. Single Type Signing Scheme Algorithm Rollover

If one key is used that acts both as ZSK and KSK, the same scheme and figure as above applies whereby all DNSKEY\_Z\_\* records from the table are removed and all RRSIG\_Z\_\* are replaced with RRSIG\_K\_\*. The requirement to sign with both algorithms and make sure that old RRSIGS have the opportunity to expire from distant caches before introducing the new algorithm in the DNSKEY RRset is still valid.

Also see Figure 11 in Appendix C.

#### 4.1.5.2. Algorithm rollover, RFC5011 style

Trust anchor algorithm rollover is almost as simple as a regular RFC5011 based rollover. However, the old trust anchor must be revoked before it is removed from the zone.

Take a look at the Figure 7 above. After the "new DS" step, we need an additional step where the DNSKEY is revoked (revoke DNSKEY):

```

-----
revoke DNSKEY
-----
Parent:
----->
----->
----->
----->

Child:
SOA_3
RRSIG_Z_1(SOA)
RRSIG_Z_2(SOA)

DNSKEY_K_1_REVOKED
DNSKEY_K_2
DNSKEY_Z_1
DNSKEY_Z_2
RRSIG_K_1(DNSKEY)
RRSIG_K_2(DNSKEY)
-----

```

Figure 8: The Revoke DNSKEY state that is added to an algorithm rollover when RFC5011 is in use.

There is one exception to the rule above. While all zone data must be signed with an unrevoked key, it is permissible to sign the keyset

with a revoked key. The somewhat esoteric argument follows.

Resolvers that do not understand the RFC5011 Revoke flag will handle DNSKEY\_K\_1\_REVOKED the same as if it was DNSKEY\_K\_1. In other words, they will handle the revoked key as a normal key, and thus RRsets signed with this key will validate. As a result, the signature matches the algorithm listed in the DNSKEY RRset. Resolvers that do implement RFC5011 will remove DNSKEY\_K\_1 from the set of trust anchors. That is okay, since they have already added DNSKEY\_K\_2 as the new trust anchor. Thus, algorithm 2 is the only signaled algorithm by now. That means, we only need RRSIG\_K\_2(DNSKEY) to authenticate the DNSKEY RRset, and we still are compliant with section 2.2 from RFC 4035: There must be a RRSIG for each RRset using at least one DNSKEY of each algorithm in the zone apex DNSKEY RRset.

Also see Figure 12 in Appendix C.

#### 4.1.5.3. Single Signing Type Algorithm Rollover, RFC5011 style

Combining the Single Signing Type Scheme Algorithm Rollover and RFC5011 style rollovers is not trivial.

Should you choose to perform an RFC5011 style rollover with a Single Signing Type key then remember that section 2.1, RFC 5011 states:

Once the resolver sees the REVOKE bit, it MUST NOT use this key as a trust anchor or for any other purpose except to validate the RRSIG it signed over the DNSKEY RRSet specifically for the purpose of validating the revocation.

This means that if you revoke DNSKEY\_KSK\_1, it cannot be used to validate its signatures over non-DNSKEY RRsets. Thus, those RRsets should be signed with a shadow key, DNSKEY\_ZSK\_1, during the algorithm rollover. This shadow key can be introduced at the same time the signatures are pre-published, in step 2 (new RRSIGs). The shadow key must be removed at the same time the revoked KSK\_1 is removed from the zone. De-facto you temporarily falling back to a KSK/ZSK split model.

In other words, the rule that at every RRset there must be at least one signature for each algorithm used in the DNSKEY RRset still applies. This means that a different key with the same algorithm, other than the revoked key, must sign the entire zone. This can be the ZSK. Thus, more operations are needed if the Single Type Signing Scheme is used. Before rolling the algorithm, a new key must be introduced with the same algorithm as the key that is candidate for

revocation. That key can then temporarily act as ZSK during the algorithm rollover.

Just like with algorithm rollover RFC5011 style, while all zone data must be signed with an unrevoked key, it is permissible to sign the keyset with a revoked key, for the same esoteric argument described in Section 4.1.5.2.

The lesson of all of this is that a Single Type Signing scheme algorithm rollover using RFC5011 is as complicated as the name of the rollover implies, one is better off explicitly using a split key temporarily.

Also see Figure 12 in Appendix C.

#### 4.1.5.4. NSEC to NSEC3 algorithm rollover

A special case is the rollover from an NSEC signed zone to an NSEC3 signed zone. In this case algorithm numbers are used to signal support for NSEC3 but they do not mandate the use of NSEC3. Therefore NSEC records should remain in the zone until the rollover to a new algorithm has completed and the new DNSKEY RR set has populated distant caches (at least one TTL into stage 4, or at any time during stage 5). At that point the validators that have not implemented NSEC3 will treat the zone as unsecured as soon as they follow the chain of trust to DS that points to a DNSKEY of the new algorithm while validators that support NSEC3 will happily validate using NSEC. Turning on NSEC3 can then be done when changing from zone serial number, realizing that that involves a resigning of the zone and the introduction of the NSECPARAM record in order to signal authoritative servers to start serving NSEC3 authenticated denial of existence.

Summarizing, an NSEC to NSEC3 rollover is an ordinary algorithm rollover whereby NSEC is used all the time and only after that rollover finished NSEC3 needs to be deployed.

#### 4.1.6. Considerations for Automated Key Rollovers

As keys must be renewed periodically, there is some motivation to automate the rollover process. Consider the following:

- o ZSK rollovers are easy to automate as only the child zone is involved.
- o A KSK rollover needs interaction between parent and child. Data exchange is needed to provide the new keys to the parent; consequently, this data must be authenticated and integrity must

be guaranteed in order to avoid attacks on the rollover.

#### 4.2. Planning for Emergency Key Rollover

This section deals with preparation for a possible key compromise. Our advice is to have a documented procedure ready for when a key compromise is suspected or confirmed.

When the private material of one of your keys is compromised it can be used for as long as a valid trust chain exists. A trust chain remains intact for

- o as long as a signature over the compromised key in the trust chain is valid,
- o as long as the DS RR in the parent zone points to the compromised key,
- o as long as the key is anchored in a resolver and is used as a starting point for validation (this is generally the hardest to update).

While a trust chain to your compromised key exists, your namespace is vulnerable to abuse by anyone who has obtained illegitimate possession of the key. Zone operators have to make a trade-off if the abuse of the compromised key is worse than having data in caches that cannot be validated. If the zone operator chooses to break the trust chain to the compromised key, data in caches signed with this key cannot be validated. However, if the zone administrator chooses to take the path of a regular rollover, during the rollover the malicious key holder can continue to spoof data so that it appears to be valid.

##### 4.2.1. KSK Compromise

A zone containing a DNSKEY RRSset with a compromised KSK is vulnerable as long as the compromised KSK is configured as trust anchor or a DS record in the parent zone points to it.

A compromised KSK can be used to sign the key set of an attacker's zone. That zone could be used to poison the DNS.

Therefore, when the KSK has been compromised, the trust anchor or the parent DS record should be replaced as soon as possible. It is local policy whether to break the trust chain during the emergency rollover. The trust chain would be broken when the compromised KSK is removed from the child's zone while the parent still has a DS record pointing to the compromised KSK (the assumption is that there

is only one DS record at the parent. If there are multiple DS records this does not apply -- however the chain of trust of this particular key is broken).

Note that an attacker's zone still uses the compromised KSK and the presence of the corresponding DS record in the parent would cause the data in this zone to appear as valid. Removing the compromised key would cause the attacker's zone to appear as valid and the child's zone as Bogus. Therefore, we advise not to remove the KSK before the parent has a DS record for the new KSK in place.

#### 4.2.1.1. Keeping the Chain of Trust Intact

If we follow this advice, the timing of the replacement of the KSK is somewhat critical. The goal is to remove the compromised KSK as soon as the new DS RR is available at the parent. We therefore have to make sure that the signature made with a new KSK over the key set that contains the compromised KSK expires just after the new DS appears at the parent. Expiration of that signature will cause expiration of that key set from the caches.

The procedure is as follows:

1. Introduce a new KSK into the key set, keep the compromised KSK in the key set. Lower the TTL for DNSKEYs so that it will expire faster from caches.
2. Sign the key set, with a short validity period. The validity period should expire shortly after the DS is expected to appear in the parent and the old DSes have expired from caches. This provides an upper limit on how long the compromised KSK can be used in a replay attack.
3. Upload the DS for this new key to the parent.
4. Follow the procedure of the regular KSK rollover: Wait for the DS to appear in the authoritative servers and then wait as long as the TTL of the old DS RRs. If necessary re-sign the DNSKEY RRSet and modify/extend the expiration time.
5. Remove the compromised DNSKEY RR from the zone and re-sign the key set using your "normal" TTL and signature validity interval.

An additional danger of a key compromise is that the compromised key could be used to facilitate a legitimate DNSKEY/DS rollover and/or nameserver changes at the parent. When that happens, the domain may be in dispute. An authenticated out-of-band and secure notify mechanism to contact a parent is needed in this case.



Note that this is only a problem when the DNSKEY and or DS records are used for authentication at the parent.

#### 4.2.1.2. Breaking the Chain of Trust

There are two methods to break the chain of trust. The first method causes the child zone to appear 'Bogus' to validating resolvers. The other causes the child zone to appear 'insecure'. These are described below.

In the method that causes the child zone to appear 'Bogus' to validating resolvers, the child zone replaces the current KSK with a new one and re-signs the key set. Next, it sends the DS of the new key to the parent. Only after the parent has placed the new DS in the zone is the child's chain of trust repaired. Note that until that time, the child zone is still vulnerable to spoofing: the attacker is still in possession of the compromised key that the DS points to.

An alternative method of breaking the chain of trust is by removing the DS RRs from the parent zone altogether. As a result, the child zone would become insecure.

#### 4.2.2. ZSK Compromise

Primarily because there is no interaction with the parent required when a ZSK is compromised, the situation is less severe than with a KSK compromise. The zone must still be re-signed with a new ZSK as soon as possible. As this is a local operation and requires no communication between the parent and child, this can be achieved fairly quickly. However, one has to take into account that just as with a normal rollover the immediate disappearance of the old compromised key may lead to verification problems. Also note that until the RRSIG over the compromised ZSK has expired, the zone may be still at risk.

#### 4.2.3. Compromises of Keys Anchored in Resolvers

A key can also be pre-configured in resolvers as a trust-anchor. If trust-anchor keys are compromised, the administrators of resolvers using these keys should be notified of this fact. Zone administrators may consider setting up a mailing list to communicate the fact that a SEP key is about to be rolled over. This communication will of course need to be authenticated by some means, e.g. by using digital signatures.

End-users faced with the task of updating an anchored key should always validate the new key. New keys should be authenticated out-

of-band, for example, through the use of an announcement website that is secured using secure sockets (TLS) [22].

#### 4.2.4. Stand-by keys

Stand-by keys are keys that are published in your zone, but are not used to sign RRsets. There are two reasons why someone would want to use stand-by keys. One is to speed up the emergency key rollover. The other is to recover from a disaster that leaves your production private keys inaccessible.

The way to deal with stand-by keys differs for ZSKs and KSKs. To make a stand-by ZSK, you need to publish its DNSKEY RR. To make a stand-by KSK, you need to get its DS RR published at the parent.

Assuming you have your DNS operation at location A, to prepare stand-by keys you need to:

- o Generate a stand-by ZSK and KSK. Store them safely in a different location (B) than the currently used ZSK and KSK (that are at location A).
- o Pre-publish DNSKEY RR of the stand-by ZSK in the zone.
- o Pre-publish DS of the stand-by KSK in the parent zone.

Now suppose a disaster occurs at location A, that disables the access to your currently used keys. To recover from that situation, follow these procedures:

- o Set up your DNS operations and import the stand-by keys from location B.
- o Post-publish the old ZSK and sign the zone with the stand-by keys.
- o After some time, when the new signatures have been propagated, the old ZSK and DS can be removed.
- o Generate a new stand-by keyset at a different location and continue "normal" operation.

#### 4.3. Parent Policies

##### 4.3.1. Initial Key Exchanges and Parental Policies Considerations

The initial key exchange is always subject to the policies set by the parent. It is specifically important in a registry-registrar model where the key material is to be passed from the DNS operator, to the

(parent) registry via a registrar, where both DNS operator and registrar are selected by the registrant and might be different organisations. When designing a key exchange policy one should take into account that the authentication and authorization mechanisms used during a key exchange should be as strong as the authentication and authorization mechanisms used for the exchange of delegation information between parent and child. That is, there is no implicit need in DNSSEC to make the authentication process stronger than it is for regular DNS.

Using the DNS itself as the source for the actual DNSKEY material has the benefit that it reduces the chances of user error. A DNSKEY query tool can make use of the SEP bit [5] to select the proper key from a DNSSEC key set, thereby reducing the chance that the wrong DNSKEY is sent. It can validate the self-signature over a key; thereby verifying the ownership of the private key material. Fetching the DNSKEY from the DNS ensures that the chain of trust remains intact once the parent publishes the DS RR indicating the child is secure.

Note: the out-of-band verification is still needed when the key material is fetched via the DNS. The parent can never be sure whether or not the DNSKEY RRs have been spoofed.

With some type of key rollovers, the DNSKEY is not pre-published and a DNSKEY query tool is not able to retrieve the successor key. In this case, the out-of-band method is required. This also allows the child to determine the digest algorithm of the DS record.

#### 4.3.2. Storing Keys or Hashes?

When designing a registry system one should consider whether to store the DNSKEYs and/or the corresponding DSes. Since a child zone might wish to have a DS published using a message digest algorithm not yet understood by the registry, the registry can't count on being able to generate the DS record from a raw DNSKEY. Thus, we recommend that registry systems at least support storing DS records (also see draft-ietf-dnsop-dnssec-trust-anchor [26]).

The storage considerations also relate to the design of the customer interface and the method by which data is transferred between registrant and registry; Will the child zone administrator be able to upload DS RRs with unknown hash algorithms or does the interface only allow DNSKEYs? When Registries support the Extensible Provisioning Protocol (EPP) [17], that can be used for registrar-registry interactions since that protocol allows the transfer of both DS and optionally DNSKEY RRs. There is no standardized way for moving the data between the customer and the registrar. Different registrars

have different mechanisms, ranging from simple web interfaces to various APIs. In some cases the use of the DNSSEC extensions to EPP may be applicable.

Having an out-of-band mechanism, such as a registry directory (e.g., Whois), to find out which keys are used to generate DS Resource Records for specific owners and/or zones may also help with troubleshooting.

#### 4.3.3. Security Lameness

Security lameness is defined as the state whereby the parent has a DS RR pointing to a non-existing DNSKEY RR. Security lameness may occur temporarily during a Double-DS rollover scheme. However care should be taken that not all DS RRs are security lame which may cause the child's zone to be marked "Bogus" by verifying DNS clients.

As part of a comprehensive delegation check, the parent could, at key exchange time, verify that the child's key is actually configured in the DNS. However, if a parent does not understand the hashing algorithm used by child, the parental checks are limited to only comparing the key id.

Child zones should be very careful in removing DNSKEY material, specifically SEP keys, for which a DS RR exists.

Once a zone is "security lame", a fix (e.g., removing a DS RR) will take time to propagate through the DNS.

#### 4.3.4. DS Signature Validity Period

Since the DS can be replayed as long as it has a valid signature, a short signature validity period for the DS RRSIG minimizes the time a child is vulnerable in the case of a compromise of the child's KSK(s). A signature validity period that is too short introduces the possibility that a zone is marked "Bogus" in case of a configuration error in the signer. There may not be enough time to fix the problems before signatures expire (this is a generic argument also see Section 4.4.2). Something as mundane as operator unavailability during weekends shows the need for DS signature validity periods longer than two days. We recommend an absolute minimum for a DS signature validity period of a few days.

The maximum signature validity period of the DS record depends on how long child zones are willing to be vulnerable after a key compromise. On the other hand, shortening the DS signature validity interval increases the operational risk for the parent. Therefore, the parent may have policy to use a signature validity interval that is

considerably longer than the child would hope for.

A compromise between the policy/operational constraints of the parent and minimizing damage for the child may result in a DS signature validity period somewhere between a week and months.

In addition to the signature validity period, which sets a lower bound on the number of times the zone owner will need to sign the zone data and which sets an upper bound to the time a child is vulnerable after key compromise, there is the TTL value on the DS RRs. Shortening the TTL reduces the damage of a successful replay attack. It does mean that the authoritative servers will see more queries. But on the other hand, a short TTL lowers the persistence of DS RRsets in caches thereby increasing the speed with which updated DS RRsets propagate through the DNS.

#### 4.3.5. Changing DNS Operators

The parent-child relation is often described in terms of a registry-registrar-registrant model, where a registry maintains the parent zone, and the registrant (the user of the child-domain name) deals with the registry through an intermediary called a registrar. (See [11] for a comprehensive definition). Registrants may out-source the maintenance of their DNS system, including the maintenance of DNSSEC key material, to the registrar or to another third party, which we will call the DNS operator. The DNS operator that has control over the DNS zone and its keys may prevent the registrant to make a timely move to a different DNS operator.

For various reasons, a registrant may want to move between DNS operators. How easy this move will be depends principally on the DNS operator from which the registrant is moving (the losing operator), as they have control over the DNS zone and its keys. The following sections describe the two cases: where the losing operator cooperates with the new operator (the gaining operator), and where the two do not cooperate.

##### 4.3.5.1. Cooperating DNS operators

In this scenario, it is assumed that losing operator will not pass any private key material to the gaining operator (that would constitute a trivial case) but is otherwise fully cooperative.

In this environment one could proceed with a Pre-Publish ZSK rollover whereby the losing operator pre-publishes the ZSK of the gaining operator, combined with a Double Signature KSK rollover where the two registrars exchange public KSKs and independently generate a signature over those keysets that they combine and both publish in

their copy of the zone. Once that is done they can use their own private keys to sign any of their zone content during the transfer.

-----		
initial		pre-publish
-----		
Parent:		
NS_A		NS_A
DS_A		DS_A
-----		
Child at A:	Child at A:	Child at B:
SOA_A0	SOA_A1	SOA_B0
RRSIG_Z_A(SOA)	RRSIG_Z_A(SOA)	RRSIG_Z_B(SOA)
NS_A	NS_A	NS_B
RRSIG_Z_A(NS)	NS_B	RRSIG_Z_B(NS)
	RRSIG_Z_A(NS)	
DNSKEY_Z_A	DNSKEY_Z_A	DNSKEY_Z_A
DNSKEY_K_A	DNSKEY_Z_B	DNSKEY_K_B
RRSIG_Z_A(DNSKEY)	DNSKEY_K_A	DNSKEY_K_A
RRSIG_K_A(DNSKEY)	DNSKEY_K_B	DNSKEY_K_B
	RRSIG_Z_B(DNSKEY)	RRSIG_Z_B(DNSKEY)
	RRSIG_K_B(DNSKEY)	RRSIG_K_B(DNSKEY)
	RRSIG_Z_A(DNSKEY)	RRSIG_Z_A(DNSKEY)
	RRSIG_K_A(DNSKEY)	RRSIG_K_A(DNSKEY)
-----		
-----		
Redelegation		post migration
-----		
Parent:		
	NS_B	NS_B
	DS_B	DS_B
-----		
Child at A:	Child at B:	Child at B:
SOA_A2	SOA_B1	SOA_B2
RRSIG_Z_A(SOA)	RRSIG_Z_B(SOA)	RRSIG_Z_B(SOA)
NS_A	NS_B	NS_B
NS_B	RRSIG_Z_B(NS)	RRSIG_Z_B(NS)
RRSIG_Z_A(NS)		

DNSKEY_Z_A	DNSKEY_Z_A	DNSKEY_Z_B
DNSKEY_Z_B	DNSKEY_Z_B	DNSKEY_K_B
DNSKEY_K_A	DNSKEY_K_A	RRSIG_Z_B(DNSKEY)
DNSKEY_K_B	DNSKEY_K_B	RRSIG_K_B(DNSKEY)
RRSIG_Z_B(DNSKEY)	RRSIG_Z_B(DNSKEY)	
RRSIG_K_B(DNSKEY)	RRSIG_K_B(DNSKEY)	
RRSIG_Z_A(DNSKEY)	RRSIG_Z_A(DNSKEY)	
RRSIG_K_A(DNSKEY)	RRSIG_K_A(DNSKEY)	

-----

Figure 9: Rollover for cooperating operators

In this figure A denotes the losing operator and B the gaining operator. RRSIG\_Z is the RRSIG produced by a ZSK, RRSIG\_K is produced with a KSK, the appended A or B indicates the producers of the key pair. Child at A is how the zone content is represented by the losing DNS operator and Child at B is how the zone content is represented by the gaining DNS operator.

If the registry and registrars allow for DS records to be published, that do not point to a published DNSKEY in the child zone, the Double-DS KSK Rollover is preferred (also known as Pre-Publication KSK Rollover, see Figure 5). This does not require to share the KSK signatures between the operators.

#### 4.3.5.2. Non Cooperating DNS Operators

If the registry and registrars allow for DS records to be published, that do not point to a published DNSKEY in the child zone, the Double-DS KSK Rollover is preferred to resolve the non-cooperative case. The gaining operator publishes a version of the zone, signed with its own key material, and makes a request to the registry to add the corresponding DS record. After the new DS RRset has been propagated to resolver caches, the registrant then asks the registry to remove the DS RR pointing to the losing operator's DNSKEY.

If Double-DS KSK Rollover is not feasible, things are more complicated, assuming that the losing operator will not cooperate and leave the data in the DNS as is. In the extreme case the losing operator may become obstructive and publish a DNSKEY RR with a high TTL and corresponding signature validity so that registrar A's DNSKEY could end up in caches for (in theory at least) tens of years.

The problem arises when a validator tries to validate with the losing operator's key and there is no signature material produced with the losing operator available in the delegation path after redelegation from the losing operator to the gaining operator has taken place.

One could imagine a rollover scenario where the gaining operator pulls all RRSIGs created by the losing operator and publishes those in conjunction with its own signatures, but that would not allow any changes in the zone content. Since a redelegation took place the NS RRset has - by definition - changed so such rollover scenario will not work. Besides if zone transfers are not allowed by the losing operator and NSEC3 is deployed in the losing operator's zone, then the gaining operator's zone will not have certainty that all of A's RRSIGs are transferred.

The only viable option for the registrant is to publish its zone unsigned and ask the registry to remove the DS RR pointing to the losing operator's DNSKEY.

Note that some behavior of resolver implementations may aid in the process of changing DNS operators:

- o TTL sanity checking, as described in RFC2308 [9], will limit the impact the actions of an obstructive, losing operator. Resolvers that implement TTL sanity checking will use an upper limit for TTLs on RRsets in responses.
- o If RRsets at the zone cut (are about to) expire, the resolver restarts its search above the zone cut. Otherwise, the resolver risks to keep using a nameserver that might be undelegated by the parent.
- o Limiting the time DNSKEYS that seem to be unable to validate signatures are cached and/or trying to recover from cases where DNSKEYs do not seem to be able to validate data, also reduces the effects of the problem of non-cooperating registrars.

However, there is no operational methodology to work around this business issue, and proper contractual relationships between all involved parties seems to be the only solution to cope with these problems. It should be noted that in many cases, the problem with temporary broken delegations already exists when a zone changes from one DNS operator to another. Besides, it is often the case that when operators are changed the services that that zone references also change operator, possibly involving some downtime.

In any case, to minimise such problems, the classic recommendation is to have relative short TTL on all involved resource records. That will solve many of the problems regarding changes to a zone regardless of whether DNSSEC is used.



#### 4.4. Time in DNSSEC

Without DNSSEC, all times in the DNS are relative. The SOA fields REFRESH, RETRY, and EXPIRATION are timers used to determine the time elapsed after a slave server synchronized with a master server. The Time to Live (TTL) value and the SOA RR minimum TTL parameter [9] are used to determine how long a forwarder should cache data after it has been fetched from an authoritative server. By using a signature validity period, DNSSEC introduces the notion of an absolute time in the DNS. Signatures in DNSSEC have an expiration date after which the signature is marked as invalid and the signed data is to be considered Bogus.

The considerations in this section are all qualitative and focused on the operational and managerial issues. A more thorough quantitative analysis of rollover timing parameters can be found in draft-ietf-dnsop-dnssec-key-timing [24]

##### 4.4.1. Time Considerations

Because of the expiration of signatures, one should consider the following:

- o We suggest the Maximum Zone TTL of your zone data to be a fraction of your signature validity period.

If the TTL was of similar order as the signature validity period, then all RRsets fetched during the validity period would be cached until the signature expiration time. Section 8.1 of RFC4033 [3] suggests that "the resolver may use the time remaining before expiration of the signature validity period of a signed RRSet as an upper bound for the TTL". As a result, query load on authoritative servers would peak at signature expiration time, as this is also the time at which records simultaneously expire from caches.

To avoid query load peaks, we suggest the TTL on all the RRs in your zone to be at least a few times smaller than your signature validity period.

- o We suggest the signature publication period to end at least one Maximum Zone TTL duration (but preferably a few days) before the end of the signature validity period.

Re-signing a zone shortly before the end of the signature validity period may cause simultaneous expiration of data from caches. This in turn may lead to peaks in the load on authoritative servers. To avoid this schemes are deployed

whereby the zone is periodically visited for a resigning operation and those signatures that are within a so called refresh interval from signature expiration are recreated. Also see Section 4.4.2 below.

In case of an operational error, you would have one Maximum Zone TTL duration to resolve the problem. Re-signing a zone a few days before the end of the signature validity period ensures the signatures will survive a weekend in case of such operational havoc. This is called the Refresh period (see Section 4.4.2).

- o We suggest the Minimum Zone TTL to be long enough to both fetch and verify all the RRs in the trust chain. In workshop environments, it has been demonstrated [18] that a low TTL (under 5 to 10 minutes) caused disruptions because of the following two problems:
  - 1. During validation, some data may expire before the validation is complete. The validator should be able to keep all data until it is completed. This applies to all RRs needed to complete the chain of trust: DS, DNSKEY, RRSIG, and the final answers, i.e., the RRSets that are returned for the initial query.
  - 2. Frequent verification causes load on recursive nameservers. Data at delegation points, DS, DNSKEY, and RRSIG RRs benefit from caching. The TTL on those should be relatively long. Data at the leafs in the DNS tree has less impact on recursive nameservers.
- o Slave servers will need to be able to fetch newly signed zones well before the RRSIGs in the zone served by the slave server pass their signature expiration time.

When a slave server is out of synchronization with its master and data in a zone is signed by expired signatures, it may be better for the slave server not to give out any answer.

Normally, a slave server that is not able to contact a master server for an extended period will expire a zone. When that happens, the server will respond differently to queries for that zone. Some servers issue SERVFAIL, whereas others turn off the 'AA' bit in the answers. The time of expiration is set in the SOA record and is relative to the last successful refresh between the master and the slave servers. There exists no coupling between the signature expiration of RRSIGs in the zone and the expire parameter in the SOA.

If the server serves a DNSSEC zone, then it may well happen that the signatures expire well before the SOA expiration timer counts down to zero. It is not possible to completely prevent this by modifying the SOA parameters.

However, the effects can be minimized where the SOA expiration time is equal to or shorter than the Refresh period (see Section 4.4.2).

The consequence of an authoritative server not being able to update a zone for an extended period of time is that signatures may expire. In this case non-secure resolvers will continue to be able to resolve data served by the particular slave servers while security-aware resolvers will experience problems because of answers being marked as Bogus.

We suggest the SOA expiration timer being approximately one third or a quarter of the signature validity period. It will allow problems with transfers from the master server to be noticed before the actual signature times out.

We also suggest that operators of nameservers that supply secondary services develop systems to identify upcoming signature expirations in zones they slave and take appropriate action where such an event is detected.

When determining the value for the expiration parameter one has to take the following into account: what are the chances that all my secondaries expire the zone? How quickly can I reach an administrator of secondary servers to load a valid zone? These questions are not DNSSEC specific but may influence the choice of your signature validity intervals.

#### 4.4.2. Signature Validation Periods

##### 4.4.2.1. Maximum Value

The first consideration for choosing a maximum signature validity period is the risk of a replay attack. For low-value, long-term stable resources the risks may be minimal and the signature validity period may be several months. Although signature validity periods of many years are allowed the same operational habit arguments as in Section 3.2.2 play a role: when a zone is re-signed with some regularity then operators remain conscious about the operational necessity of re-signing.

#### 4.4.2.2. Minimum Value

The minimum value of the signature validity period is set for the time by which one would like to survive operational failure in provisioning: what is the time that a failure will be noticed, what is the time that action is expected to be taken? By answering these questions availability of operators during (long) weekends or time taken to access to backup media can be taken into account. The result could easily suggest a minimum Signature Validity period of a few days.

Note however, the argument above is assuming that zone data has just been signed and published when the problem occurred. In practice it may be that a zone is signed according to a frequency set by the Re-Sign Period whereby the signer visits the zone content and only refreshes signatures that are close to expiring: the signer will only refresh signatures if they are within the Refresh Period from the signature expiration time. The Re-Sign Period must be smaller than the Refresh Period in order for zone data to be signed in timely fashion.

If an operational problem occurs during resigning then the signatures in the zone to expire first are the ones that have been generated longest ago. In the worst case these signatures are the Refresh Period minus the Re-Sign Period away from signature expiration.

In other words, the minimum Signature Validity interval is set by first choosing the Refresh Period (usually a few days), then defining the Re-Sign period in such a way that the Refresh Period minus the Resign period sets the time in which operational havoc can be resolved.

To make matters slightly more complicated, some signers vary the signature validity period over a small range (the jitter interval) so that not all signatures expire at the same time. The jitter should not influence your calculation as long as it is smaller than the refresh period and the resign period is at least half the refresh period.

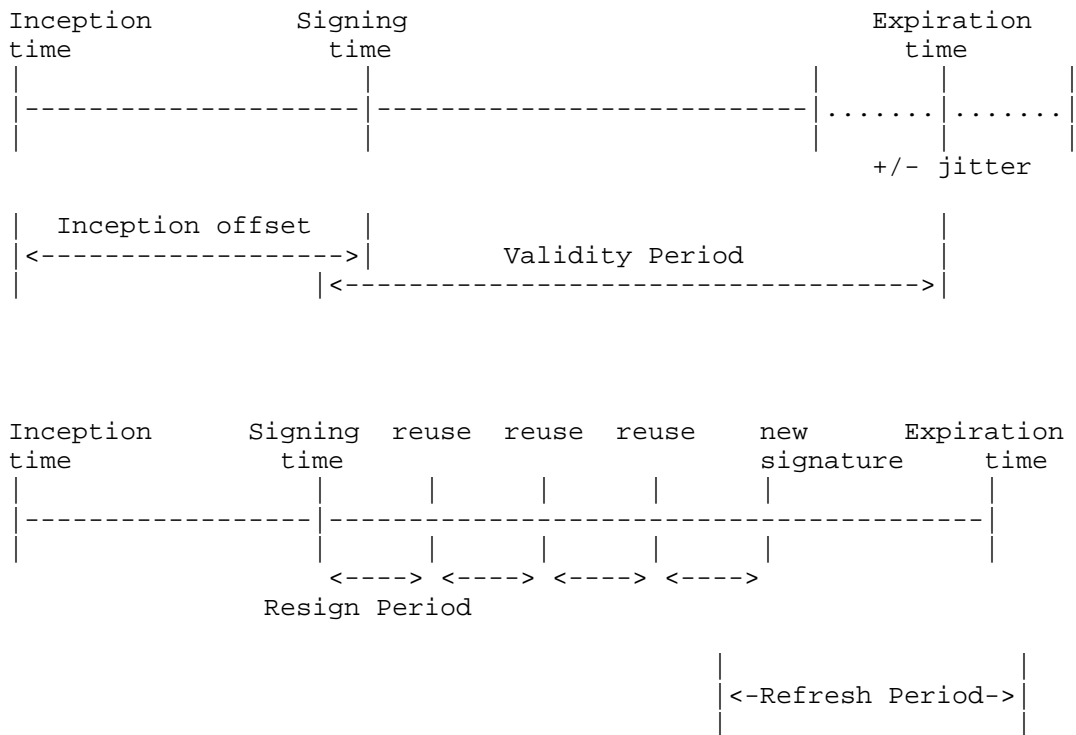


Figure 10: Signature Timing Parameters

Note that in the figure the validity of the signature starts shortly before the inception time. That is done to deal with validators that might have some clock skew. The inception offset should be chosen so that you minimize the false negatives to a reasonable level.

#### 4.4.2.3. Differentiation between RR sets

It is possible to vary signature validity periods between signatures over different RR sets in the zone. In practice this could be done when zones contain highly volatile data (which may be the case in dynamic update environments). Note however that the risk of replay (e.g. by stale secondary servers) is what should be leading in determining the signature validity period since the TTL on the data itself still are the primary parameter for cache expiry.

In some cases the risk of replaying existing data might be different from the risk of replaying the denial of data. In those cases the signature validity period on NSEC or NSEC3 records may be tweaked accordingly.

When a zone contains secure delegations then a relatively short signature validity interval protects the child against replay attacks, in the case the child's key is compromised (see Section 4.3.4). Since there is a higher operational risk for the parent registry when choosing a short validity interval and a higher operational risk for the child when choosing a long validity period some (price) differentiation may occur for validity periods between individual DS RRs in a single zone.

There seem to be no other arguments for differentiation in validity periods.

## 5. Next Record type

One of the design tradeoffs made during the development of DNSSEC was to separate the signing and serving operations instead of performing cryptographic operations as DNS requests are being serviced. It is therefore necessary to create records that cover the very large number of non-existent names that lie between the names that do exist.

There are two mechanisms to provide authenticated proof of non-existence of domain names in DNSSEC: a clear text one and an obfuscated-data one. Each mechanism:

- o includes a list of all the RRTYPEs present which can be used to prove the non-existence of RRTYPEs at a certain name;
- o stores only the name for which the zone is authoritative (that is, glue in the zone is omitted); and
- o uses a specific RRTYPE to store information about the RRTYPEs present at the name: the clear-text mechanism uses NSEC, and the obfuscated-data mechanism uses NSEC3.

### 5.1. Differences between NSEC and NSEC3

The clear text mechanism (NSEC) is implemented using a sorted linked list of names in the zone. The obfuscated-data mechanism (NSEC3) is similar but first hashes the names using a one-way hash function, before creating a sorted linked list of the resulting (hashed) strings.

The NSEC record requires no cryptographic operations aside from the validation of its associated signature record. It is human readable and can be used in manual queries to determine correct operation. The disadvantage is that it allows for "zone walking", where one can request all the entries of a zone by following the linked list of

NSEC RRs via the "Next Domain Name" field.

Though all agree DNS data is accessible through query mechanisms, a side effect of NSEC is that it allows the contents of a zone file to be enumerated in full by sequential queries. Whilst for some operators this behavior is acceptable or even desirable, for others it is undesirable for policy, regulatory or other reasons. This is the first difference between NSEC and NSEC3.

The second difference between NSEC and NSEC3 is that NSEC requires a signature over every RR in the zonefile, thereby ensuring that any denial of existence is cryptographically signed. However, in a large zonefile containing many delegations very few of which are to signed zones, this may produce unacceptable additional overhead especially where insecure delegations are subject to frequent update (a typical example might be a TLD operator with few registrants using secure delegations). NSEC3 allows intervals between two such delegations to "Opt-out" in which case they may contain one more more insecure delegations, thus reducing the size and cryptographic complexity of the zone at the expense of the ability to cryptographically deny the existence of names in a specific span.

The NSEC3 record uses a hashing method of the requested RRlabel. To increase the workload required to guess entries in the zone, the number of hashing iteration's can be specified in the NSEC3 record. Additionally, a salt can be specified that also modifies the hashes. Note that NSEC3 does not give full protection against information leakage from the zone.

## 5.2. NSEC or NSEC3

The first motivation to deploy NSEC3, prevention of zone enumeration, only makes sense when zone content is not highly structured or trivially guessable. Highly structured zones such as the in-addr.arpa, ip6.arpa and el64.arpa can be trivially enumerated using ordinary DNS properties while for small zones that only contain contain records in the APEX and a few common RRlabels such as "www" or "mail" guessing zone content and proving completeness is also trivial when using NSEC3.

In those cases the use of NSEC is recommended to ease the work required by signers and validating resolvers.

For large zones where there is an implication of "not readily available" RRlabels, such as those where one has to sign a non-disclosure agreement before obtaining it, NSEC3 is recommended.

The considerations for the second reason to deploy NSEC3 are

discussed below (Section 5.3.4).

### 5.3. NSEC3 parameters

The NSEC3 hashing algorithm is performed on the Fully Qualified Domain Name (FQDN) in its uncompressed form. This ensures brute force work done by an attacker for one (FQDN) RRlabel cannot be re-used for another (FQDN) RRlabel attack, as these entries are, by definition unique.

#### 5.3.1. NSEC3 Algorithm

At the moment of writing there is only one NSEC3 Hashing algorithm defined. [21] specifically calls out that when a new hash algorithm for use with NSEC3 is specified, a transition mechanism **MUST** also be defined. Therefore this document does not consider NSEC3 hash algorithm transition.

#### 5.3.2. NSEC3 Iterations

One of the concerns with NSEC3 is a pre-calculated dictionary attack could be made in order to assess if certain domain names exist within the zones or not. Two mechanisms are introduced in the NSEC3 specification to increase the costs of such dictionary attacks: Iterations and Salt.

RFC5155 Section 10.3 [21] considers the trade-offs between incurring cost during the signing process and imposing costs to the validating nameserver, while still providing a reasonable barrier against dictionary attacks. It provides useful limits of iterations for a given RSA key size. These are 150 iterations for 1024 bit keys, 500 iterations for 2048 bit keys and 2,500 iterations for 4096 bit keys. Choosing a value of 100 iterations is deemed to be a sufficiently costly yet not excessive value: In the worst case scenario, the performance of your nameservers would be halved, regardless of key size [27].

#### 5.3.3. NSEC3 Salt

While the NSEC3 iterations parameter increases the cost of hashing a dictionary word, the NSEC3 salt reduces the lifetime for which that calculated hash can be used. A change of the salt value by the zone owner would cause an attacker to lose all precalculated work for that zone.

The FQDN RRlabel, which is part of the value that is hashed, already ensures that brute force work for one RRlabel can not be re-used to attack other RRlabel (e.g. in other domains) due to their uniqueness.



The salt of all NSEC3 records in a zone needs to be the same. Since changing the salt requires all the NSEC3 records to be regenerated, and thus requires generating new RRSIG's over these NSEC3 records, it is recommended to align the change of the salt with a change of the Zone Signing Key, as that process in itself already usually requires all RRSIG's to be regenerated (you can have a smooth ZSK rollover by honoring the Refresh period). If there is no critical dependency on incremental signing and the zone can be signed with little effort there is no need for such alignment. However, unlike Zone Signing Key changes, NSEC3 salt changes do not need special rollover procedures. It is possible to change the salt each time the zone is updated.

#### 5.3.4. Opt-out

The Opt-Out mechanism was introduced to allow for a gradual introduction of signed records in zones that contain mostly delegation records. The use of the OPT-OUT flag changes the meaning of the NSEC3 span from authoritative denial of the existence of names within the span to a proof that DNSSEC is not available for the delegations within the span. [Editors Note: One could make this construct more correct by talking about the hashed names and the hashed span, but I believe that is overkill]. This allows for the addition or removal of the delegations covered by the span without recalculating or re- signing RRs in the NSEC3 RR chain.

Opt-Out is specified to be used only over delegation points and will therefore only bring relief to zones with a large number of zones and where the number of secure delegations is small. This consideration typically holds for large top-level-domains and similar zones; in most other circumstances Opt-Out should not be deployed. Further considerations can be found in RFC5155 section 12.2 [21].

## 6. Security Considerations

DNSSEC adds data integrity to the DNS. This document tries to assess the operational considerations to maintain a stable and secure DNSSEC service. Not taking into account the 'data propagation' properties in the DNS will cause validation failures and may make secured zones unavailable to security-aware resolvers.

## 7. IANA considerations

There are no IANA considerations with respect to this document

## 8. Contributors and Acknowledgments

Significant parts of the text of this document is copied from RFC4641 [14]. That document was edited by Olaf Kolkman and Miek Gieben. Other people that contributed or where otherwise involved in that work were in random order: Rip Loomis, Olafur Gudmundsson, Wesley Griffin, Michael Richardson, Scott Rose, Rick van Rein, Tim McGinnis, Gilles Guette, Olivier Courtay, Sam Weiler, Jelte Jansen, Niall O'Reilly, Holger Zuleger, Ed Lewis, Hilarie Orman, Marcos Sanz, Peter Koch, Mike StJohns, Emma Bretherick, Adrian Bedford, and Lindy Foster, and O. Courtay.

For this version of the document we would like to acknowledge a few people for significant contributions:

Paul Hoffman    for his contribution on the choice of cryptographic parameters and addressing some of the trust anchor issues;

Jelte Jansen    who provided the initial text in Section 4.1.5;

Paul Wouters    who provided the initial text for Section 5 and Alex Bligh who improved it;

Erik Rescorla    whose blogpost on "the Security of ZSK rollovers" inspired text in Section 3.1;

Stephen Morris    who made a pass on English style and grammar;

Matthijs Mekking    thoroughly reviewed and provided concrete improvements on the specific types of keyrollovers (e.g. he provided the tables in Appendix C); and

Olafur Gudmundsson and Ondrej Sury    who provided input on Section 4.1.5 based on actual operational experience.

Rickard Bellgrim    reviewed the document extensively.

The figure in Section 4.4.2 was adapted from the OpenDNSSEC user documentation.

In addition valuable contributions in the form of text, comments, or review where provided by Mark Andrews, Patrik Faltstrom, Tony Finch, Alfred Hines, Bill Manning, Scott Rose, and Wouter Wijngaards.

## 9. References

## 9.1. Normative References

- [1] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.
- [2] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.
- [3] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.
- [4] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.
- [5] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.

## 9.2. Informative References

- [6] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [7] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, August 1996.
- [8] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, August 1996.
- [9] Andrews, M., "Negative Caching of DNS Queries (DNS NCACHE)", RFC 2308, March 1998.
- [10] Wellington, B., "Secure Domain Name System (DNS) Dynamic Update", RFC 3007, November 2000.
- [11] Hollenbeck, S., "Generic Registry-Registrar Protocol Requirements", RFC 3375, September 2002.
- [12] Orman, H. and P. Hoffman, "Determining Strengths For Public Keys Used For Exchanging Symmetric Keys", BCP 86, RFC 3766, April 2004.
- [13] Eastlake, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.
- [14] Kolkman, O. and R. Gieben, "DNSSEC Operational Practices",

RFC 4641, September 2006.

- [15] Shirey, R., "Internet Security Glossary, Version 2", RFC 4949, August 2007.
- [16] StJohns, M., "Automated Updates of DNS Security (DNSSEC) Trust Anchors", RFC 5011, September 2007.
- [17] Gould, J. and S. Hollenbeck, "Domain Name System (DNS) Security Extensions Mapping for the Extensible Provisioning Protocol (EPP)", RFC 5910, May 2010.
- [18] Rose, S., "NIST DNSSEC workshop notes", , June 2001.
- [19] Barker, E. and J. Kelsey, "Recommendation for Random Number Generation Using Deterministic Random Bit Generators (Revised)", NIST Special Publication 800-90, March 2007.
- [20] Hardaker, W., "Use of SHA-256 in DNSSEC Delegation Signer (DS) Resource Records (RRs)", RFC 4509, May 2006.
- [21] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, March 2008.
- [22] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.
- [23] Jansen, J., "Use of SHA-2 Algorithms with RSA in DNSKEY and RRSIG Resource Records for DNSSEC", RFC 5702, October 2009.
- [24] Morris, S., Ihren, J., and J. Dickinson, "DNSSEC Key Timing Considerations", draft-ietf-dnsop-dnssec-key-timing-01 (work in progress), October 2010.
- [25] Ljunggren, F., Eklund-Lowinder, A., and T. Okubo, "DNSSEC Policy & Practice Statement Framework", draft-ietf-dnsop-dnssec-dps-framework-03 (work in progress), November 2010.
- [26] Larson, M. and O. Gudmundsson, "DNSSEC Trust Anchor Configuration and Maintenance", draft-ietf-dnsop-dnssec-trust-anchor-04 (work in progress), October 2010.
- [27] Schaeffer, Y., "NSEC3 Hash Performance", NLnet Labs document 2010-02, March 2010.

## Appendix A. Terminology

In this document, there is some jargon used that is defined in other documents. In most cases, we have not copied the text from the documents defining the terms but have given a more elaborate explanation of the meaning. Note that these explanations should not be seen as authoritative.

**Anchored key:** A DNSKEY configured in resolvers around the globe. This key is hard to update, hence the term anchored.

**Bogus:** Also see Section 5 of RFC4033 [3]. An RRSset in DNSSEC is marked "Bogus" when a signature of an RRSset does not validate against a DNSKEY.

**Key Signing Key or KSK:** A Key Signing Key (KSK) is a key that is used exclusively for signing the apex key set. The fact that a key is a KSK is only relevant to the signing tool.

**Key size:** The term 'key size' can be substituted by 'modulus size' throughout the document for RSA keys. It is mathematically more correct to use modulus size for RSA keys, but as this is a document directed at operators we feel more at ease with the term key size.

**Private and public keys:** DNSSEC secures the DNS through the use of public key cryptography. Public key cryptography is based on the existence of two (mathematically related) keys, a public key and a private key. The public keys are published in the DNS by use of the DNSKEY Resource Record (DNSKEY RR). Private keys should remain private.

**Key rollover:** A key rollover (also called key supercession in some environments) is the act of replacing one key pair with another at the end of a key effectivity period.

**Refresh Period:** The period before the expiration time of the signature, during which the signature is refreshed by the signer.

**Re-Signing frequency:** Frequency with which a signing pass on the zone is performed. Alternatively expressed as "Re-Signing Period". It defines when the zone is exposed to the signer. During a signing pass not all signatures in the zone may be refreshed, that depend refresh frequency/interval.

Secure Entry Point (SEP) key: A KSK that has a DS record in the parent zone pointing to it or is configured as a trust anchor. Although not required by the protocol, we recommend that the SEP flag [5] is set on these keys.

Self-signature: This only applies to signatures over DNSKEYs; a signature made with DNSKEY x, over DNSKEY x is called a self-signature. Note: without further information, self-signatures convey no trust. They are useful to check the authenticity of the DNSKEY, i.e., they can be used as a hash.

Signing Jitter: Jitter applied to the signature validity interval.

Signer: The system that has access to the private key material and signs the Resource Record sets in a zone. A signer may be configured to sign only parts of the zone, e.g., only those RRSets for which existing signatures are about to expire.

Single Type Signing Scheme: A signing scheme whereby the distinction between Zone Signing Keys and Key SigningKeys is not made.

Zone Signing Key (ZSK): A key that is used for signing all data in a zone (except, perhaps, the DNSKEY RRSets). The fact that a key is a ZSK is only relevant to the signing tool.

Singing the zone file: The term used for the event where an administrator joyfully signs its zone file while producing melodic sound patterns.

Zone administrator: The 'role' that is responsible for signing a zone and publishing it on the primary authoritative server.

## Appendix B. Typographic Conventions

The following typographic conventions are used in this document:

Key notation: A key is denoted by DNSKEY\_x\_y, where x is an identifier for the type of key: K for Keys Signing Key, Z for Zone Signing Key and S when there is no distinction made between KSK and ZSKs but the key is used as a secure entry point. The 'y' denotes a number or an identifier, y could be thought of as the key id.

RRSet notations: RRs are only denoted by the type. All other information -- owner, class, rdata, and TTL -- is left out. Thus: "example.com 3600 IN A 192.0.2.1" is reduced to "A". RRSets are a list of RRs. An example of this would be "A1, A2", specifying the RRSet containing two "A" records. This could again be abbreviated

to just "A".

Signature notation: Signatures are denoted as RRSIG\_x\_y(RRSet), which means that RRSet is signed with DNSKEY\_x\_y.

Zone representation: Using the above notation we have simplified the representation of a signed zone by leaving out all unnecessary details such as the names and by representing all data by "SOA\_x"

SOA representation: SOAs are represented as SOA\_x, where x is the serial number.

RRsets ignored: If the signature of non DNSKEY RRsets have the same parameters as the SOA than those are not mentioned. e.g. In the example below the SOA is signed with the same parameters as the foo.example.com A RRset and the latter is therefore ignored in the abbreviated notation.

Using this notation the following signed zone:

```
example.com. 3600 IN SOA ns1.example.com. olaf.example.net. (
    2005092303 ; serial
    450        ; refresh (7 minutes 30 seconds)
    600        ; retry (10 minutes)
    345600     ; expire (4 days)
    300        ; minimum (5 minutes)
)
3600 RRSIG SOA 5 2 3600 20120824013000 (
    20100424013000 14 example.com.
    NMaFnzmmZ8wevpCOI+/JxqWBzPxrnzPnSXfo
    ...
    OMY3rTMA2qorupQXjQ== )
3600 NS ns1.example.com.
3600 NS ns2.example.com.
3600 NS ns3.example.com.
3600 RRSIG NS 5 2 3600 20120824013000 (
    20100424013000 14 example.com.
    p0Cj3wzGoPFftFZjj3jeKGK6wGWLwY6mCBEz
    ...
    +SqZIoVHpve7YBeH46wuyF8w4XknA4Oeimc4
    zAgaJM/MeG08KpeHhg== )
3600 TXT "Net::DNS domain"
3600 RRSIG TXT 5 2 3600 20120824013000 (
    20100424013000 14 example.com.
    o7eP8LISK2TEutFQRvK/+U3wq7t4X+PQaQkp
    ...
    BcQlo99vwn+IS4+Jlg== )
300 NSEC foo.example.com. NS SOA TXT RRSIG NSEC DNSKEY
```

```

300      RRSIG      NSEC 5 2 300 20120824013000 (
                    20100424013000 14 example.com.
                    JtHm8ta0diCWYGu/TdrE10lsYSHblN2i/IX+
                    ...
                    PkXNI/Vgf4t3xZaIyw== )
3600     DNSKEY    256 3 5 (
                    AQPaoHW/nC0fj9HuCW3hACSGiP0AkPS3dQFX
                    ...
                    sAuryjQ/HFa5r4mrbhkJ
                    ) ; key id = 14
3600     DNSKEY    257 3 5 (
                    AQPuiszMMAi36agx/V+7Tw95l8PYmoVjHWvO
                    ...
                    oy88Nh+u2c9HF1tw0naH
                    ) ; key id = 15
3600     RRSIG      DNSKEY 5 2 3600 20120824013000 (
                    20100424013000 14 example.com.
                    HWj/VEr6p/FiUUil70QQWtk+NBilSJ9mdj5U
                    ...
                    QhhmMwV3tIxJk2eDRQ== )
3600     RRSIG      DNSKEY 5 2 3600 20120824013000 (
                    20100424013000 15 example.com.
                    P47CUy/xPV8qIEuua4tMKG6ei3LQ8RYv3TwE
                    ...
                    JWL70YiUnUG3m9OL9w== )
foo.example.com. 3600 IN A 192.0.2.2
3600     RRSIG      A 5 3 3600 20120824013000 (
                    20100424013000 14 example.com.
                    xHr023P79YrSHHMTSL0alnlfUt4ywn/vWqsO
                    ...
                    JPV/SA4BkoFxIcPrDQ== )
300      NSEC       example.com. A RRSIG NSEC
300      RRSIG      NSEC 5 3 300 20120824013000 (
                    20100424013000 14 example.com.
                    Aaa4kgKhqY7Lzjq3rlPlFidymOeBEK1T6vUF
                    ...
                    Qe000JyzObxx27pY8A== )

```

is reduced to the following representation:

```

SOA_2005092303
RRSIG_Z_14(SOA_2005092303)
DNSKEY_K_14
DNSKEY_Z_15
RRSIG_K_14(DNSKEY)
RRSIG_Z_15(DNSKEY)

```

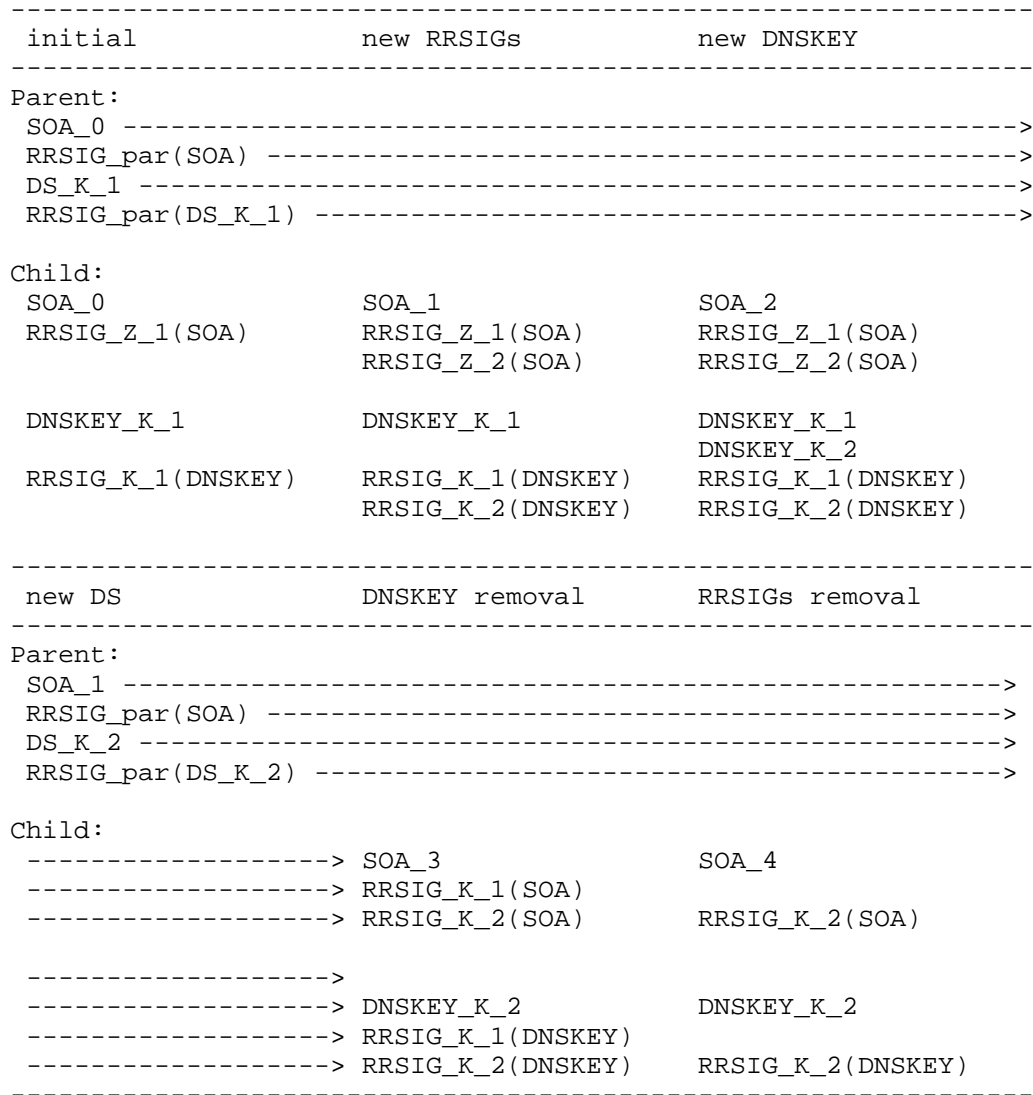
The rest of the zone data has the same signature as the SOA record,



i.e., an RRSIG created with DNSKEY 14.

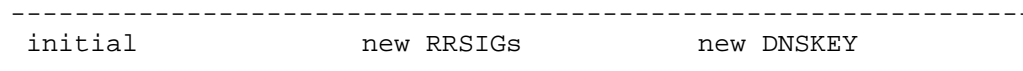
#### Appendix C. Transition Figures for Special Case Algorithm Rollovers

The figures appendix complement and illustrate the special cases of algorithm rollovers as described in Section 4.1.5



Also see Section 4.1.5.1.

Figure 11: Single Type Signing Scheme Algorithm Roll



Parent:  
 SOA\_0 ----->  
 RRSIG\_par(SOA) ----->  
 DS\_K\_1 ----->  
 RRSIG\_par(DS\_K\_1) ----->

Child:  
 SOA\_0                      SOA\_1                      SOA\_2  
 RRSIG\_Z\_1(SOA)           RRSIG\_Z\_1(SOA)           RRSIG\_Z\_1(SOA)  
                              RRSIG\_Z\_2(SOA)           RRSIG\_Z\_2(SOA)  
  
 DNSKEY\_K\_1                DNSKEY\_K\_1                DNSKEY\_K\_1  
                              DNSKEY\_K\_2  
 DNSKEY\_Z\_1                DNSKEY\_Z\_1                DNSKEY\_Z\_1  
                              DNSKEY\_Z\_2  
 RRSIG\_K\_1(DNSKEY)        RRSIG\_K\_1(DNSKEY)        RRSIG\_K\_1(DNSKEY)  
                              RRSIG\_K\_2(DNSKEY)        RRSIG\_K\_2(DNSKEY)

-----  
 new DS                      revoke DNSKEY              DNSKEY removal  
 -----

Parent:  
 SOA\_0 ----->  
 RRSIG\_par(SOA) ----->  
 DS\_K\_2 ----->  
 RRSIG\_par(DS\_K\_2) ----->

Child:  
 -----> SOA\_3                      SOA\_4  
 -----> RRSIG\_Z\_1(SOA)            RRSIG\_Z\_1(SOA)  
 -----> RRSIG\_Z\_2(SOA)            RRSIG\_Z\_2(SOA)  
  
 -----> DNSKEY\_K\_1\_REVOKED  
 -----> DNSKEY\_K\_2                      DNSKEY\_K\_2  
 ----->  
 -----> DNSKEY\_Z\_2                      DNSKEY\_Z\_2  
 -----> RRSIG\_K\_1(DNSKEY)           RRSIG\_K\_1(DNSKEY)  
 -----> RRSIG\_K\_2(DNSKEY)           RRSIG\_K\_2(DNSKEY)

RRSIGs removal  
 -----

Parent:  
 ----->  
 ----->  
 ----->  
 ----->

```
Child:
  SOA_5
  RRSIG_Z_2(SOA)

  DNSKEY_K_2
  DNSKEY_Z_2
  RRSIG_K_2(DNSKEY)
```

Also see Section 4.1.5.2.

Figure 12: RFC5011 Style algorithm roll

initial	new RRSIGs	new DNSKEY
-----		
Parent:		
SOA_0 ----->		
RRSIG_par(SOA) ----->		
DS_K_1 ----->		
RRSIG_par(DS_K_1) ----->		
Child:		
SOA_0	SOA_1	SOA_2
RRSIG_K_1(SOA)	RRSIG_Z_1(SOA)	RRSIG_Z_1(SOA)
	RRSIG_Z_2(SOA)	RRSIG_K_2(SOA)
DNSKEY_K_1	DNSKEY_K_1	DNSKEY_K_1
	DNSKEY_Z_1	DNSKEY_K_2
		DNSKEY_Z_1
RRSIG_K_1(DNSKEY)	RRSIG_K_1(DNSKEY)	RRSIG_K_1(DNSKEY)
	RRSIG_K_2(DNSKEY)	RRSIG_K_2(DNSKEY)
-----		
new DS	revoke DNSKEY	DNSKEY removal
-----		
Parent:		
SOA_0 ----->		
RRSIG_par(SOA) ----->		
DS_K_2 ----->		
RRSIG_par(DS_K_2) ----->		
Child:		
-----> SOA_3		SOA_4
-----> RRSIG_Z_1(SOA)		RRSIG_Z_1(SOA)

```

-----> RRSIG_Z_2(SOA)            RRSIG_Z_2(SOA)

-----> DNSKEY_K_1_REVOKED
-----> DNSKEY_K_2                DNSKEY_K_2
----->
-----> DNSKEY_Z_1
-----> RRSIG_K_1(DNSKEY)        RRSIG_K_1(DNSKEY)
-----> RRSIG_K_2(DNSKEY)        RRSIG_K_2(DNSKEY)

-----
RRSIGs removal
-----
Parent:
----->
----->
----->
----->

Child:
SOA_5
RRSIG_K_2(SOA)

DNSKEY_K_2
RRSIG_K_2(DNSKEY)
-----

```

Also see Section 4.1.5.3.

Figure 13: RFC5011 algorithm roll in a Single Type Signing Scheme Environment

#### Appendix D. Document Editing History

[To be removed prior to publication as an RFC]

##### D.1. draft-ietf-dnsop-rfc4641-00

Version 0 was differs from RFC4641 in the following ways.

- o Status of this memo appropriate for I-D
- o TOC formatting differs.
- o Whitespaces, linebreaks, and pagebreaks may be slightly different because of xml2rfc generation.

- o References slightly reordered.
- o Applied the errata from  
[http://www.rfc-editor.org/errata\\_search.php?rfc=4641](http://www.rfc-editor.org/errata_search.php?rfc=4641)
- o Inserted trivial "IANA considerations" section.

In other words it should not contain substantive changes in content as intended by the working group for the original RFC4641.

#### D.2. version 0->1

Cryptography details rewritten. (See [http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/cryptography\\_flawed](http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/cryptography_flawed))

- o Reference to NIST 800-90 added
- o RSA/SHA256 is being recommended in addition to RSA/SHA1.
- o Complete rewrite of Section 3.4.2 removing the table and suggesting a keysize of 1024 for keys in use for less than 8 years, issued up to at least 2015.
- o Replaced the reference to Schneiers' applied cryptography with a reference to RFC4949.
- o Removed the KSK for high level zones consideration

Applied some differentiation with respect of the use of a KSK for parent or trust-anchor relation [http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/differentiation\\_trustanchor\\_parent](http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/differentiation_trustanchor_parent)

[http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/rollover\\_assumptions](http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/rollover_assumptions)

Added Section 4.1.5 as suggested by Jelte Jansen in [http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/Key\\_algorithm\\_roll](http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/Key_algorithm_roll)

Added Section 4.3.5.1 Issue identified by Antoin Verschuren <http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/non-cooperative-registrars>

In Appendix A: ZSK does not necessarily sign the DNSKEY RRset.

#### D.3. version 1->2

- o Significant rewrite of Section 3 whereby the argument is made that the timescales for rollovers are made purely on operational arguments hopefully resolving [http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/discussion\\_of\\_timescales](http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/discussion_of_timescales)
- o Added Section 5 based on <http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/NSEC-NSEC3>
- o Added a reference to draft-morris-dnsop-dnssec-key-timing [24] for the quantitative analysis on keyrolls
- o Updated Section 4.3.5 to reflect that the problem occurs when changing DNS operators, and not DNS registrars, also added the table indicating the redelegation procedure. Added text about the fact that implementations will dismiss keys that fail to validate at some point.
- o Updated a number of references.

#### D.4. version 2->3

- o Added bulleted list to serve as an introduction on the decision tree in Section 3.
- o In section Section 3.1:
  - \* tried to motivate that key length is not a strong motivation for KSK ZSK split (based on [http://www.educatedguesswork.org/2009/10/on\\_the\\_security\\_of\\_zsk\\_rollove.html](http://www.educatedguesswork.org/2009/10/on_the_security_of_zsk_rollove.html))
  - \* Introduced Common Signing Key terminology and made the arguments for the choice of a Common Signing Key more explicit.
  - \* Moved the SEP flag considerations to its own paragraph
- o In a few places in the document, but section Section 4 in particular the comments from Patrik Faltstrom (On Mar 24, 2010) on the clarity on the roles of the registrant, dns operator, registrar and registry was addressed.
- o Added some terms based on [http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/timing\\_terminology](http://www.nlnetlabs.nl/svn/rfc4641bis/trunk/open-issues/timing_terminology)
- o Added paragraph 2 and clarified the second but last paragraph of Section 3.2.2.
- o Clarified the table and some text in Section 4.1.5. Also added some text on what happens when the algorithm rollover also

involves a roll from NSEC to NSEC3.

- o Added a paragraph about rolling KSKs that are also configured as trust-anchors in Section 4.1.2
- o Added Section 4.1.4.
- o Added Section 4.4.2 to address issue "Signature\_validity"

D.5. version 3->4

- o Stephen Morris submitted a large number of language, style and editorial nits.
- o Section 4.1.5 improved based on comments from Olafur Gudmundsson and Ondrej Sury.
- o Tried to improve consistency of notation in the various rollover figures

D.6. version 4->5

- o Improved consistency of notation
- o Matthijs Mekking provided substantive feedback on algorithm rollover and suggested the content of the subsections of Section 4.1.5 and the content of the figures in Appendix C

D.7. version 5->6

- o More improved consistency of notation and some other nits
- o Review of Rickard Bellgrim
- o Review of Sebastian Castro
- o Added a section about Stand-by keys
- o Algorithm rollover: Conservative or Liberal Approach
- o Added a reference to NSEC3 hash performance report

D.8. Subversion information

[www.nlnetlabs.nl/svn/rfc4641bis/](http://www.nlnetlabs.nl/svn/rfc4641bis/)

\$Id: draft-ietf-dnsop-rfc4641bis.xml 93 2011-03-03 15:16:38Z matje \$



Authors' Addresses

Olaf M. Kolkman  
NLnet Labs  
Science Park 140  
Amsterdam 1098 XG  
The Netherlands

EMail: [olaf@nlnetlabs.nl](mailto:olaf@nlnetlabs.nl)  
URI: <http://www.nlnetlabs.nl>

Matthijs Mekking  
NLnet Labs  
Science Park 140  
Amsterdam 1098 XG  
The Netherlands

EMail: [matthijs@nlnetlabs.nl](mailto:matthijs@nlnetlabs.nl)  
URI: <http://www.nlnetlabs.nl>



Individual Submission  
Internet-Draft  
Intended status: BCP  
Expires: September 8, 2011

G. Michaelson  
G. Huston  
APNIC  
March 7, 2011

AS112 Nameserver Delegations for IPv6  
draft-michaelson-as112-ipv6-00

Abstract

To reduce longterm traffic to the DNS root servers and the IP6.ARPA authoritative servers, the IAB is requested to instruct the IANA to delegate a set of sub-domains of IP6.ARPA to the AS112 Project [ID.ietf-dnsop-as112-ops]. These domains represent IPv6 address prefixes that are not conventionally populated in the global reverse-DNS, including IPv6 prefixes that are not globally scoped and certain prefixes used in an anycast context.

The reverse DNS query load associated with these IPv6 address prefixes appear to have unacceptable scaling consequences as IPv6 uptake increases. By delegating these sub-domains to the AS112 project, the DNS query load can be passed to a distributed sink, reducing the query load on the root servers and the IP6.ARPA authoritative servers.

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 September 8, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

#### Table of Contents

1. Reverse DNS Delegation and Local-Use Addresses . . . . .	3
2. IANA Considerations . . . . .	3
3. Security Considerations . . . . .	4
4. Acknowledgments . . . . .	4
5. References . . . . .	5
5.1. Normative References . . . . .	5
5.2. Informative References . . . . .	5
Authors' Addresses . . . . .	5

## 1. Reverse DNS Delegation and Local-Use Addresses

The IPv6 Addressing Architecture [RFC4291] includes certain address prefixes that are not intended to be uniquely used in the global network as globally-scoped unicast addresses. Such addresses include locally-scoped addresses, certain anycast addresses, and loopback addresses.

While such addresses are not intended to be used in the same context as globally-scoped unicast addresses, their use in various local and global contexts is seen to trigger Domain Name System (DNS) [RFC1034] queries (of the form of "reverse lookups") corresponding to these addresses. Since the addresses concerned generally have local rather than global significance, it is good practice for site administrators to ensure that such queries are answered locally [I-D.ietf-dnsop-default-local-zones]. However, it is not uncommon for such queries to follow the normal delegation path in the public DNS instead of being answered within the site. It is not possible for public DNS servers to give useful answers to such queries, and the response to such reverse lookup queries from the global DNS is the "Name Error" RCODE described in [RFC1035], commonly termed "NXDOMAIN".

When the reverse-DNS infrastructure receives a request for undelegated sub-domains, the point of delegation of the last matched label along the name path to the root receives the query. In the case of the IPv6 reverse delegation structure, this implies that the IP6.ARPA authoritative servers will receive the query load. Because the sub-domain is not delegated, the server is obliged to answer with an NXDOMAIN response. Since negative caching is not widely deployed, a large number of these DNS queries are repeated, further increasing the DNS query load imposed on the DNS root servers and the IP6.ARPA authoritative servers.

This query load appears to have unacceptable scaling consequences as IPv6 uptake increases. By delegating these sub-domains to the AS112 project [ID.ietf-dnsop-as112-ops], the DNS query load can be passed off to a distributed dedicated server set, reducing the load on the DNS root and the IP6.ARPA authoritative servers.

## 2. IANA Considerations

As per the provisions of [RFC3152], this document recommends the IAB to direct IANA to delegate the following IP6.ARPA reverse DNS zones to the AS112 project [ID.ietf-dnsop-as112-ops]:

```
0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa (Unspecified)
      f.f.ip6.arpa (Multicast)
        8.e.f.ip6.arpa (Link-Local Scope)
        9.e.f.ip6.arpa (Link-Local Scope)
        a.e.f.ip6.arpa (Link-Local Scope)
        b.e.f.ip6.arpa (Link-Local Scope)
        c.e.f.ip6.arpa (Link-Local Scope)
        d.e.f.ip6.arpa (Link-Local Scope)
        e.e.f.ip6.arpa (Link-Local Scope)
        f.e.f.ip6.arpa (Link-Local Scope)
          0.0.c.f.ip6.arpa (Unique Locally Assigned)
          0.0.d.f.ip6.arpa (Unique Locally Assigned)
            0.0.0.0.1.0.0.2.ip6.arpa (Teredo)
```

AS112 project servers should add these zones to their configuration, and terminate queries efficiently inside their service infrastructure.

This delegation instruction is subject to further direction in the future from the IAB to IANA, as per the provisions of [RFC3152].

### 3. Security Considerations

The Security Considerations described in [ID.ietf-dnsop-as112-ops] also apply to local-use IPv6 addresses, and should be considered in the context of the use of these addresses.

DNS queries may well identify the location of deployment of IPv6 enabled equipment in private contexts, particularly when the reverse queries relate to local-use IPv6 addresses. While operators of the DNS reverse servers should respect the privacy of data relating to individual queries made to these reverse address servers, the unintentional leakage of information beyond its intended scope of use and circulation represents a potential threat to the security of a local private network. This direction to delegate these local-use IPv6 reverse address sub-domains does not substantially change the security risks of information leakage from private environments.

#### 4. Acknowledgments

The authors acknowledge the work of Joe Abley and William Maton and the DNSOPS Working Group in preparing the AS112 framework document for delegation of the private use address blocks in IPv4, and have used parts of their AS112 document as a template for these AS112 delegation instructions in IPv6.

## 5. References

### 5.1. Normative References

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

### 5.2. Informative References

- [I-D.ietf-dnsop-default-local-zones]  
Andrews, M., "Locally-served DNS Zones", Internet Draft draft-ietf-dnsop-default-local-zones, September 2010.
- [ID.ietf-dnsop-as112-ops]  
Abley, J. and W. Maton, "Locally-served DNS Zones", Internet Draft draft-ietf-dnsop-default-local-zones, November 2010.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.
- [RFC3152] Bush, R., "Delegation of IP6.ARPA", BCP 49, RFC 3152, August 2001.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

## Authors' Addresses

George Michaelson  
APNIC

Email: ggm@apnic.net  
URI: <http://www.apnic.net>

Geoff Huston  
APNIC

Email: gih@apnic.net  
URI: <http://www.apnic.net>





Individual Submission  
Internet-Draft  
Intended status: BCP  
Expires: January 5, 2012

W. Sotomayor  
NRC-CNRC  
July 4, 2011

Additional IPv4 Delegations for AS112  
draft-sotomayor-as112-ipv4-cull-00

Abstract

This is a direction to IANA concerning the delegation of certain additional IPv4 zones to the AS112 project.

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 5, 2012.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Reverse DNS Delegation, Local-Use and Test Addresses . . . . .	3
3. Impact of Lame DNS Reverse Delegations by Misconfigured AS112 Nodes . . . . .	3
4. IANA Considerations . . . . .	4
5. Security Considerations . . . . .	5
6. Acknowledgements . . . . .	5
7. References . . . . .	5
7.1. Normative References . . . . .	5
7.2. Informative References . . . . .	6
Appendix A. Change History . . . . .	7
Author's Address . . . . .	7

## 1. Introduction

This is a direction to IANA concerning the delegation of certain additional IPv4 zones to the AS112 Project [I-D.ietf-dnsop-as112-ops] which operates in an anycast cloud [RFC4786].

## 2. Reverse DNS Delegation, Local-Use and Test Addresses

Work documenting special use addresses [RFC5735] have appeared containing good candidates for inclusion with the existing delegations to the AS112 Project. These local and special use addresses have now been captured and identified for Domain Name System (DNS [RFC1034] and [RFC1035]) zone administrators to curtail leaking queries in [I-D.ietf-dnsop-default-local-zones].

Additionally, [RFC5737] identifies a short listing of IPv4 address blocks for use in documentation. These too should be considered for delegation to the AS112 Project.

Thus the purpose of this memo is to effectively signal to the AS112 Project [I-D.ietf-dnsop-as112-ops] that it should reflect the efforts of [I-D.ietf-dnsop-default-local-zones] and follow directives and delegations subsequently issued as described in section 6.

It is interesting to note that while the focus has largely been on the use of [RFC1918] addresses, some addresses (such as those first enumerated within [RFC1912]) have as of late also been the subject of [I-D.ietf-dnsop-default-local-zones] .

## 3. Impact of Lame DNS Reverse Delegations by Misconfigured AS112 Nodes

Although not discussed previously in any related memo before, there exists the issue of AS112 nodes not being current with the published IANA registry or simply missing zones that they should be hosting on behalf of the global DNS system as part of their mandate to participate in the effort.

Lame delegations occur when a DNS server has been identified to not only carry a delegated zone, but to also answer for it authoritatively, yet does neither. The reasons for why this may happen may be varied (though most likely due to a misconfiguration within the DNS server itself), but the point is that lame delegations, even in the reverse zones (such as arpa.) can and do create an impact ultimately to the end-user, along with DNS caches, applications, firewalls, etc.

In the case of AS112 nodes and the zones they hold authority for, if such a node were to be considered lame, then reverse queries to resolve addresses will result in delays created by timeouts. Additionally, these timeouts can be amplified as caching DNS servers no longer hold information about the addresses they may have once cached: They naturally begin seeking to resolve an address by interrogating other DNS servers, ultimately leading to the the root and creating a further load - which is precisely what the AS112 project seeks to prevent.

For the end-user residing on a network that uses addresses which are delegated to a lame AS112 node, the usual result manifests itself in the perception that either the local network is not functioning as usually expected or that the Internet is in some way down.

This in turn may lead to an apparent denial of access to services for the end-user. A domino effect can then be triggered from there, in which case the end-user may believe that a service is unavailable simply because a service administrator is unknowingly depending on a far-off AS112's node to be functioning properly. It would not be long before support personnel would become involved, further committing more resources at the perceived problem.

Finally, the impact to the AS112 Project itself should not be ignored. If too many AS112 nodes are discovered to be lame servers, then it would naturally be in the best interests of the Internet community to remove affected delegations of reverse zones assigned to it as a whole. This in turn lessens the effectiveness of the project and only serves to increase the traffic load to the root servers. On top that, the reputation of the AS112 Project's as providing a service for the common good of the community would be damaged because it would be deemed as unreliable.

#### 4. IANA Considerations

As per the provisions of [RFC3596], this document recommends the IAB to direct IANA to delegate the following IN-ADDR.ARPA reverse DNS zones to the AS112 project [I-D.ietf-dnsop-as112-ops]:

- 0.in-addr.arpa (IPv4 "This" Network)
- 127.in-addr.arpa (IPv4 Loop-Back Network)
- 2.0.192.in-addr.arpa (IPv4 Test Net 1)
- 100.51.198.in-addr.arpa (IPv4 Test Net 2)
- 113.0.203.in-addr.arpa (IPv4 Test Net 3)
- 255.255.255.255.in-addr.arpa (IPv4 Broadcast)

AS112 project servers should add these zones to their configuration,

and terminate queries efficiently inside their service infrastructure.

This delegation instruction is subject to further direction in the future from the IAB to IANA, as per the provisions of [RFC3596].

## 5. Security Considerations

The Security Considerations described in [I-D.ietf-dnsop-as112-ops] also apply to local-use IPv4 addresses, and should be considered in the context of the use of these addresses.

Security administrators as well as general support personnel who are involved in the operations of networked devices may also find the information found in the companion document [I-D.ietf-dnsop-as112-under-attack-help-help] quite helpful.

DNS queries may well identify the location of deployment of IPv4 enabled equipment in private contexts, particularly when the reverse queries relate to local-use IPv4 addresses. While operators of the DNS reverse servers should respect the privacy of data relating to individual queries made to these reverse address servers, the unintentional leakage of information beyond its intended scope of use and circulation represents a potential threat to the security of a local private network. This direction to delegate these local-use IPv4 reverse address sub-domains does not substantially change the security risks of information leakage from private environments.

## 6. Acknowledgements

The author would like to acknowledge the efforts of Mark P. Andrews for preparing the work related to serving certain DNS zones locally and George Michaelson and Geoff Huston for their work [I-D.michaelson-as112-ipv6] in setting the example for a template to be used by the AS112 Project as a direction for the IANA.

The author would also like to thank Joe Abley, Marco d'Itri, Nick Hilliard and Paul Vixie for their feedback.

## 7. References

### 7.1. Normative References

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

## 7.2. Informative References

- [I-D.ietf-dnsop-as112-ops]  
Abley, J. and W. Maton, "AS112 Nameserver Operations",  
draft-ietf-dnsop-as112-ops-09 (work in progress),  
May 2011.
- [I-D.ietf-dnsop-as112-under-attack-help-help]  
Abley, J. and W. Maton, "I'm Being Attacked by  
PRISONER.IANA.ORG!",  
draft-ietf-dnsop-as112-under-attack-help-help-06 (work in  
progress), April 2011.
- [I-D.ietf-dnsop-default-local-zones]  
Andrews, M., "Locally-served DNS Zones",  
draft-ietf-dnsop-default-local-zones-15 (work in  
progress), March 2011.
- [I-D.michaelson-as112-ipv6]  
Michaelson, G. and G. Huston, "AS112 Nameserver  
Delegations for IPv6", draft-michaelson-as112-ipv6-00  
(work in progress), March 2011.
- [RFC1035] Mockapetris, P., "Domain names - implementation and  
specification", STD 13, RFC 1035, November 1987.
- [RFC1912] Barr, D., "Common DNS Operational and Configuration  
Errors", RFC 1912, February 1996.
- [RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and  
E. Lear, "Address Allocation for Private Internets",  
BCP 5, RFC 1918, February 1996.
- [RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi,  
"DNS Extensions to Support IP Version 6", RFC 3596,  
October 2003.
- [RFC4786] Abley, J. and K. Lindqvist, "Operation of Anycast  
Services", BCP 126, RFC 4786, December 2006.
- [RFC5735] Cotton, M. and L. Vegoda, "Special Use IPv4 Addresses",  
BCP 153, RFC 5735, January 2010.
- [RFC5737] Arkko, J., Cotton, M., and L. Vegoda, "IPv4 Address Blocks  
Reserved for Documentation", RFC 5737, January 2010.

## Appendix A. Change History

This section to be removed prior to publication.

00 Initial draft, circulated as draft-sotomayor-as112-ipv4-cull-00.

## Author's Address

William F. Maton Sotomayor  
National Research Council of Canada  
1200 Montreal Road  
Ottawa, ON K1A 0R6  
Canada

Phone: +1 613 993 0880  
Email: wfms@ryouko.imsb.nrc.ca

