eppext
Internet-Draft
Intended status: Standards Track
Expires: December 2, 2016

H.W. Ribbers
M.W. Groeneweg
SIDN
R. Gieben

A.L.J Verschuren

May 31, 2016

# Key Relay Mapping for the Extensible Provisioning Protocol
# draft-ietf-eppext-keyrelay-12

## Abstract

This document describes an Extensible Provisioning Protocol (EPP) mapping for a key relay object that relays DNSSEC key material between EPP clients using the poll queue defined in RFC5730.

This key relay mapping will help facilitate changing the DNS operator of a domain while keeping the DNSSEC chain of trust intact.

## 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 December 2, 2016.

## Copyright Notice

Copyright (c) 2016 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. Conventions Used in This Document . . . . . 3
1.2. Secure Transfer of DNSSEC Key Material . . . . . 3
2. Object Attributes . . . . . 4
2.1. DNSSEC Key Material . . . . . 5
2.1.1. <keyRelayData> element . . . . . 5
3. EPP Command Mapping . . . . . 5
3.1. EPP Query Commands . . . . . 5
3.1.1. EPP <check> Command . . . . . 6
3.1.2. EPP <info> Command . . . . . 6
3.1.3. EPP <transfer> Command . . . . . 9
3.2. EPP Transform Commands . . . . . 9
3.2.1. EPP <create> Command . . . . . 9
3.2.2. EPP <delete> Command . . . . . 11
3.2.3. EPP <renew> Command . . . . . 11
3.2.4. EPP <transfer> Command . . . . . 12
3.2.5. EPP <update> Command . . . . . 12
4. Formal Syntax . . . . . 12
5. IANA Considerations . . . . . 13
5.1. XML Namespace . . . . . 13
5.2. XML Schema . . . . . 13
5.3. EPP Extension Registry . . . . . 14
6. Security Considerations . . . . . 14
7. Acknowledgements . . . . . 15
8. References . . . . . 15
8.1. Normative References . . . . . 15
8.2. Informative References . . . . . 15
Appendix A. Changelog . . . . . 16
A.1. draft-gieben-epp-keyrelay-00 . . . . . 16
A.2. draft-gieben-epp-keyrelay-01 . . . . . 16
A.3. draft-gieben-epp-keyrelay-02 . . . . . 16
A.4. draft-gieben-epp-keyrelay-03 . . . . . 16
A.5. draft-ietf-eppext-keyrelay-00 . . . . . 17
A.6. draft-ietf-eppext-keyrelay-01 . . . . . 17
A.7. draft-ietf-eppext-keyrelay-02 . . . . . 17
A.8. draft-ietf-eppext-keyrelay-03 . . . . . 17
A.9. draft-ietf-eppext-keyrelay-04 . . . . . 17
A.10. draft-ietf-eppext-keyrelay-05 . . . . . 18
A.11. draft-ietf-eppext-keyrelay-06 . . . . . 18
A.12. draft-ietf-eppext-keyrelay-07 . . . . . 18

A.13. draft-ietf-eppext-keyrelay-08 . . . . . 18
A.14. draft-ietf-eppext-keyrelay-09 . . . . . 18
A.15. draft-ietf-eppext-keyrelay-10 . . . . . 18
A.16. draft-ietf-eppext-keyrelay-11 . . . . . 18
A.17. draft-ietf-regext-keyrelay-00 . . . . . 18
Authors' Addresses . . . . . 18

## 1. Introduction

There are certain transactions initiated by a DNS-operator that require an authenticated exchange of information between DNS-operators. Often, there is no direct channel between these parties or it is non-scalable and insecure.

One such transaction is the exchange of DNSSEC key material when changing the DNS operator for DNSSEC signed zones. We suggest that DNS-operators use the administrative EPP channel to bootstrap the delegation by relaying DNSSEC key material for the zone.

In this document we define an EPP extension to sent DNSSEC key material between EPP clients. This allows DNS operators to bootstrap automatically, reliable and securely the transfer of a domain name while keeping the DNSSEC chain of trust intact.

### 1.1. Conventions Used in This Document

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 BCP 14, RFC 2119 [RFC2119].

XML is case sensitive. Unless stated otherwise, XML specifications and examples provided in this document MUST be interpreted in the character case presented in order to develop a conforming implementation.

In examples, "C:" represents lines sent by a protocol client, and "S:" represents lines returned by a protocol server. Indentation and white space in examples is provided only to illustrate element relationships and is not a mandatory feature of this protocol.

### 1.2. Secure Transfer of DNSSEC Key Material

Exchanging DNSSEC key material in preparation of a domain name transfer is one of the phases in the lifecycle of a domain name [I-D.koch-dnsop-dnssec-operator-change].

DNS-operators need to exchange DNSSEC key material before the registration data can be changed to keep the DNSSEC chain of trust intact. This exchange is normally initiated through the gaining registrar.

The gaining and losing DNS operators could talk directly to each other (the ~ arrow in Figure 1) to exchange the DNSKEY, but often there is no trusted path between the two. As both can securely interact with the registry over the administrative channel through the registrar, the registry can act as a relay for the key material exchange.

The registry is merely used as a relay channel. Therefore it is up to the losing DNS-operator to complete the intended transaction. The registry SHOULD have certain policies in place that require the losing DNS operator to cooperate with this transaction, however this is beyond this document. This document focuses on the EPP protocol syntax.

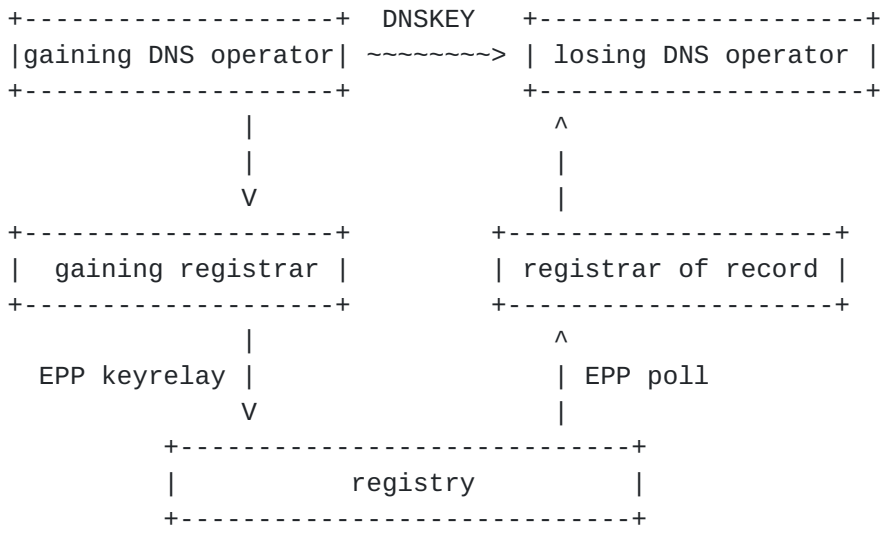



Figure 1: Transfer of DNSSEC key material.

There is no distinction in the EPP protocol between Registrars and DNS-operators, there is only mention of an EPP client and EPP server. Therefore the term EPP client will be used for the interaction with the EPP server for relaying DNSSEC key material.

## 2. Object Attributes

### 2.1. DNSSEC Key Material

The DNSSEC key material is represented in EPP by a <keyRelayData> element.

#### 2.1.1. <keyRelayData> element

The <keyRelayData> contains the following elements:

- One REQUIRED <keyData> element that contains the DNSSEC key material as described in [RFC5910], Section 4

- An OPTIONAL <expiry> element that describes the expected lifetime of the relayed key(s) in the zone. When the <expiry> element is provided the losing DNS operator SHOULD remove the inserted key material from the zone after the expire time. This may be because the transaction that needed the insertion should either be completed or abandoned by that time. If a client receives a key relay object that has been sent previously it MUST update the expire time of the key material. This enables the clients to update the lifetime of the key material when a transfer is delayed.

The <expiry> element MUST contain exactly one of the following child elements:

* <absolute>: The DNSSEC key material is valid from the current date and time until it expires on the specified date and time. If a date in the past is provided this MUST be interpreted as a revocation of a previously sent key relay object.

* <relative>: The DNSSEC key material is valid from the current date and time until the end of the specified duration. If a period of zero is provided this MUST be interpreted as a revocation of a previously sent key relay object.

## 3. EPP Command Mapping

A detailed description of the EPP syntax and semantics can be found in the EPP core protocol specification [RFC5730]. The command mapping described here is specifically for use in this key relay mapping.

### 3.1. EPP Query Commands

EPP provides three commands to retrieve object information: <check> to determine if an object is known to the server, <info> to retrieve

detailed information associated with an object, and <transfer> to retrieve object transfer status information.

### 3.1.1. EPP <check> Command

Check semantics do not apply to key relay objects, so there is no mapping defined for the EPP <check> command and the EPP <check> response.

### 3.1.2. EPP <info> Command

Info command semantics do not apply to the key relay objects, so there is no mapping defined for the EPP <info> Command.

The EPP <info> response for key relay objects is used in the EPP poll response, as described in [RFC5730]. The key relay object created with the <create> command, described in Section 3.2.1 is inserted into the receiving client's poll queue. The receiving client will receive the key relay object using the EPP <poll> command, as described in [RFC5730].

When a <poll> command has been processed successfully for a key relay poll message, the EPP <resData> element MUST contain a child <keyrelay:infData> element that is identified by the keyrelay namespace. The <keyrelay:infData> element contains the following child elements:

o A REQUIRED <name> element containing the domain name for which the DNSSEC key material is relayed.

o A REQUIRED <authInfo> element that contains authorization information associated with the domain object ([RFC5731], Section 3.2.1).

o One or more REQUIRED <keyRelayData> elements containing data to be relayed, as defined in Section 2.1. A server MAY apply a server policy that specifies the number of <keyRelayData> elements that can be incorporated. When a server policy is violated, a server MUST respond with an EPP result code 2308 "Data management policy violation".

o An OPTIONAL <crDate> element that contains the date and time of the submitted <create> command.

o An OPTIONAL <reID> element that contains the identifier of the client that requested the key relay.

o An OPTIONAL <acID> element that contains the identifier of the client that SHOULD act upon the key relay.

Example <poll> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
S:    xmlns:keyrelay="urn:ietf:params:xml:ns:keyrelay-1.0"
S:  xmlns:s="urn:ietf:params:xml:ns:secDNS-1.1"
S:  xmlns:d="urn:ietf:params:xml:ns:domain-1.0">
S:  <response>
S:    <result code="1301">
S:      <msg>Command completed successfully; ack to dequeue</msg>
S:    </result>
S:    <msgQ count="5" id="12345">
S:      <qDate>1999-04-04T22:01:00.0Z</qDate>
S:      <msg>Keyrelay action completed successfully.</msg>
S:    </msgQ>
S:    <resData>
S:      <keyrelay:infData>
S:        <keyrelay:name>example.org</keyrelay:name>
S:        <keyrelay:authInfo>
S:          <d:pw>JnSdBAZSxxzJ</d:pw>
S:        </keyrelay:authInfo>
S:        <keyrelay:keyRelayData>
S:          <keyrelay:keyData>
S:            <s:flags>256</s:flags>
S:            <s:protocol>3</s:protocol>
S:            <s:alg>8</s:alg>
S:            <s:pubKey>cmlraXN0aGViZXN0</s:pubKey>
S:          </keyrelay:keyData>
S:          <keyrelay:expiry>
S:            <keyrelay:relative>P1M13D</keyrelay:relative>
S:          </keyrelay:expiry>
S:        </keyrelay:keyRelayData>
S:        <keyrelay:crDate>
S:          1999-04-04T22:01:00.0Z
S:        </keyrelay:crDate>
S:        <keyrelay:reID>
S:          ClientX
S:        </keyrelay:reID>
S:        <keyrelay:acID>
S:          ClientY
S:        </keyrelay:acID>
S:      </keyrelay:infData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54321-ZYX</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

### 3.1.3. EPP <transfer> Command

Transfer semantics do not apply to key relay objects, so there is no mapping defined for the EPP <transfer> command.

## 3.2. EPP Transform Commands

EPP provides five commands to transform objects: <create> to create an instance of an object, <delete> to delete an instance of an object, <renew> to extend the validity period of an object, <transfer> to manage object sponsorship changes, and <update> to change information associated with an object.

### 3.2.1. EPP <create> Command

The EPP <create> command provides a transform operation that allows a client to create a key relay object that includes the domain name and DNSSEC key material to be relayed. When the <create> command is validated, the server MUST insert an EPP <poll> message, using the key relay info response (See Section 3.1.2), in the receiving client's poll queue that belongs to the registrar on record of the provided domain name.

In addition to the standard EPP command elements, the <create> command MUST contain a <keyrelay:create> element that is identified by the keyrelay namespace. The <keyrelay:create> element contains the following child elements:

o A REQUIRED <keyrelay:name> element containing the domain name for which the DNSSEC key material is relayed.

o A REQUIRED <authInfo> element that contains authorization information associated with the domain object ([RFC5731], Section 3.2.1).

o One or more REQUIRED <keyrelay:keyRelayData> element containing data to be relayed, as defined in Section 2.1

Example <create> commands:

Note that in the provided example the second <keyrelay:keyRelayData> element has a period of zero and thus represents the revocation of a previously sent key relay object (see Section 2.1.1).

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
C:    xmlns:keyrelay="urn:ietf:params:xml:ns:keyrelay-1.0"
C:  xmlns:s="urn:ietf:params:xml:ns:secDNS-1.1"
C:  xmlns:d="urn:ietf:params:xml:ns:domain-1.0">
C:  <command>
C:    <create>
C:      <keyrelay:create>
C:        <keyrelay:name>example.org</keyrelay:name>
C:        <keyrelay:authInfo>
C:          <d:pw>JnSdBAZSxxzJ</d:pw>
C:        </keyrelay:authInfo>
C:        <keyrelay:keyRelayData>
C:          <keyrelay:keyData>
C:            <s:flags>256</s:flags>
C:            <s:protocol>3</s:protocol>
C:            <s:alg>8</s:alg>
C:            <s:pubKey>cmlraXN0aGViZXN0</s:pubKey>
C:          </keyrelay:keyData>
C:          <keyrelay:expiry>
C:            <keyrelay:relative>P1M13D</keyrelay:relative>
C:          </keyrelay:expiry>
C:        </keyrelay:keyRelayData>
C:        <keyrelay:keyRelayData>
C:          <keyrelay:keyData>
C:            <s:flags>256</s:flags>
C:            <s:protocol>3</s:protocol>
C:            <s:alg>8</s:alg>
C:            <s:pubKey>bWFyY2lzdGhlYmVzdA==</s:pubKey>
C:          </keyrelay:keyData>
C:          <keyrelay:expiry>
C:            <keyrelay:relative>P0D</keyrelay:relative>
C:          </keyrelay:expiry>
C:        </keyrelay:keyRelayData>
C:      </keyrelay:create>
C:    </create>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When a server has succesfully processed the <create> command it MUST respond with a standard EPP response. See [RFC5730], Section 2.6.

Example <create> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <trID>
S:       <clTRID>ABC-12345</clTRID>
S:       <svTRID>54321-ZYX</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

When a server cannot process the <create> command due to the server policy it MUST return an EPP 2308 error message. This might be the case when the server knows that the receiving client does not support keyrelay transactions. See [RFC5730], Section 2.6.

Example <create> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="2308">
S:      <msg>Data management policy violation</msg>
S:    </result>
S:    <trID>
S:       <clTRID>ABC-12345</clTRID>
S:       <svTRID>54321-ZYX</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

### 3.2.2. EPP <delete> Command

Delete semantics do not apply to key relay objects, so there is no mapping defined for the EPP <delete> command and the EPP <delete> response.

### 3.2.3. EPP <renew> Command

Renew semantics do not apply to key relay objects, so there is no mapping defined for the EPP <renew> command and the EPP <renew> response.

### 3.2.4. EPP <transfer> Command

Transfer semantics do not apply to key relay objects, so there is no mapping defined for the EPP <transfer> command and the EPP <transfer> response.

### 3.2.5. EPP <update> Command

Update semantics do not apply to key relay objects, so there is no mapping defined for the EPP <update> command and the EPP <update> response.

## 4. Formal Syntax

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="urn:ietf:params:xml:ns:keyrelay-1.0"
  xmlns:keyrelay="urn:ietf:params:xml:ns:keyrelay-1.0"
  xmlns:eppcom="urn:ietf:params:xml:ns:eppcom-1.0"
  xmlns:secDNS="urn:ietf:params:xml:ns:secDNS-1.1"
  xmlns:domain="urn:ietf:params:xml:ns:domain-1.0"
  xmlns="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <annotation>
    <documentation>
      Extensible Provisioning Protocol v1.0 protocol
      extension schema for relaying DNSSEC key material.
    </documentation>
  </annotation>

  <import namespace="urn:ietf:params:xml:ns:eppcom-1.0" />
  <import namespace="urn:ietf:params:xml:ns:secDNS-1.1" />
  <import namespace="urn:ietf:params:xml:ns:domain-1.0" />

  <element name="keyRelayData" type="keyrelay:keyRelayDataType" />
  <element name="infData" type="keyrelay:infDataType" />
  <element name="create" type="keyrelay:createType" />

  <complexType name="createType">
    <sequence>
      <element name="name" type="eppcom:labelType" />
      <element name="authInfo" type="domain:authInfoType" />
      <element name="keyRelayData" type="keyrelay:keyRelayDataType"
          maxOccurs="unbounded"/>
    </sequence>
  </complexType>

 <complexType name="infDataType">
```

```
      <sequence>
        <element name="name" type="eppcom:labelType" />
        <element name="authInfo" type="domain:authInfoType" />
        <element name="keyRelayData" type="keyrelay:keyRelayDataType"
            maxOccurs="unbounded"/>
        <element name="crDate" type="dateTime"/>
        <element name="reID" type="eppcom:clIDType" />
        <element name="acID" type="eppcom:clIDType" />
      </sequence>
    </complexType>

    <complexType name="keyRelayDataType">
      <sequence>
        <element name="keyData" type="secDNS:keyDataType" />
        <element name="expiry" type="keyrelay:keyRelayExpiryType"
            minOccurs="0" />
      </sequence>
    </complexType>

    <complexType name="keyRelayExpiryType">
      <choice>
        <element name="absolute" type="dateTime" />
        <element name="relative" type="duration" />
      </choice>
    </complexType>
  </schema>
```

## 5. IANA Considerations

### 5.1. XML Namespace

This document uses URNs to describe a XML namespace conforming to the registry mechanism described in [RFC3688]. The following URI assignment is requested of IANA:

URI: urn:ietf:params:xml:ns:keyrelay-1.0

Registrant Contact: See the "Author's Address" section of this document.

XML: See the "Formal Syntax" section of this document.

### 5.2. XML Schema

This document uses URNs to describe a XML schema conforming to the registry mechanism described in [RFC3688]. The following URI assignment is requested of IANA:

URI: urn:ietf:params:xml:schema:keyrelay-1.0

XML: See the "Formal Syntax" section of this document.

### 5.3. EPP Extension Registry

The EPP extension described in this document should be registered by the IANA in the EPP Extension Registry described in [RFC7451]. The details of the registration are as follows:

Name of Extension: "Key Relay Mapping for the Extensible Provisioning Protocol"

Document status: Standards Track

Reference: (insert reference to RFC version of this document)

Registrant Name and Email Address: IESG, iesg@ietf.org

TLDs: Any

IPR Disclosure: https://datatracker.ietf.org/ipr/2393/

Status: Active

Notes: None

## 6. Security Considerations

A server SHOULD NOT perform any transformation on data under server management when processing a <keyrelay:create> command. The intent of this command is to put DNSSEC key material on the poll queue of another client. To make sure that this EPP extension is interoperable with the different server policies that already have implemented EPP this extension it is not classified as must not.

Any EPP client can use this mechanism to put data on the message queue of another EPP client, allowing for the potential of a denial of service attack. However this can, and should be detected by the server. A server MAY set a server policy which limits or rejects a <keyrelay:create> command if it detects the mechanism is being abused.

For the <keyrelay:keyRelayData> data a correct <domain:authInfo> element should be used as an indication that putting the key material on the receiving EPP clients poll queue is authorized by the _registrant_ of that domain name. The authorization of EPP clients

to perform DNS changes is not covered in this document as it depends on registry specific policy.

A client that uses this mechanism to send DNSSEC key material to another client could verify through DNS that the DNSSEC key material is added to the authoritive zone of the domain. This check can be used to verify that the DNSSEC key material has traveled end-to-end from the gaining DNS operator to the losing DNS operator. This check does not tell anything about the DNSSEC chain of trust and can merely be used as a verification of a succesful transfer of the DNSSEC key material.

## 7. Acknowledgements

We like to thank the following individuals for their valuable input, review, constructive criticism in earlier revisions or support for the concepts described in this document:

Maarten Wullink, Marco Davids, Ed Lewis, James Mitchell, David Peal, Patrik Faltstrom, Klaus Malorny, James Gould, Patrick Mevzek, Seth Goldman, Maarten Bosteels, Ulrich Wisser, Kees Monshouwer and Scott Hollenbeck.

## 8. References

### 8.1. Normative References

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, RFC 5730, August 2009.

[RFC5731] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Domain Name Mapping", STD 69, RFC 5731, August 2009.

[RFC5910] Gould, J. and S. Hollenbeck, "Domain Name System (DNS) Security Extensions Mapping for the Extensible Provisioning Protocol (EPP)", RFC 5910, May 2010.

### 8.2. Informative References

[I-D.koch-dnsop-dnssec-operator-change]
Koch, P., Sanz, M., and A. Verschuren, "Changing DNS Operators for DNSSEC signed Zones", draft-koch-dnsop-dnssec-operator-change-06 (work in progress), February 2014.

[RFC7451] Hollenbeck, S., "Extension Registry for the Extensible Provisioning Protocol", RFC 7451, February 2015.

## Appendix A. Changelog

[This section should be removed by the RFC editor before publishing]

### A.1. draft-gieben-epp-keyrelay-00

1. Initial document.

### A.2. draft-gieben-epp-keyrelay-01

1. Style and grammar changes;

2. Added an expire element as per suggestion by Klaus Malorny;

3. Make the authInfo element mandatory and make the registry check it as per feedback by Klaus Malorny and James Gould.

### A.3. draft-gieben-epp-keyrelay-02

1. Added element to identify the relaying EPP client as suggested by Klaus Malorny;

2. Corrected XML for missing and excess clTRID as noted by Patrick Mevzek;

3. Added clarifications for the examples based on feedback by Patrick Mevzeck;

4. Reviewed the consistency of using DNS operator versus registrar after review comments by Patrick Faltstrom and Ed Lewis.

### A.4. draft-gieben-epp-keyrelay-03

1. Style and grammar changes

2. Corrected acknowledgement section

3. Corrected XML for Expire element to not be mandatory but only occur once.

### A.5. draft-ietf-eppext-keyrelay-00

1. Added feedback from Seth Goldman and put him in the acknowledgement section.

2. IDnits formatting ajustments

### A.6. draft-ietf-eppext-keyrelay-01

1. Introducing the <relay> command, and thus separating the data and the command.

2. Updated the Introduction, describing the general use of relay vs the intended use-case of relaying DNSSEC key data.

3. Restructuring the document to make it more inline with existing EPP extensions.

### A.7. draft-ietf-eppext-keyrelay-02

1. Updated the XML structure by removing the <relay> command based on WG feedback

2. Updated the wording

### A.8. draft-ietf-eppext-keyrelay-03

1. Updated the document title in the EPP Extension Registry section

2. Restored Acknowledgement section, thanks to Marco Davids

3. Incorperated feedback from Patrick Mevzek

### A.9. draft-ietf-eppext-keyrelay-04

1. Incorperated feedback from James Gould

2. Added additional text when server is aware that receiving clients do not support keyrelay transactions or DNSSEC as suggested by Kees Monshouwer.

3. Added additional text for supporting key revocation as suggested by Kees Monshouwer

4. Updated some of the wording

5. Fix the usage of multiple keys in a create message

### A.10. draft-ietf-eppext-keyrelay-05

1. Review comments after WG last call

### A.11. draft-ietf-eppext-keyrelay-06

1. Review comments by Ulrich Wisser during IESG writeup

### A.12. draft-ietf-eppext-keyrelay-07

1. fixed changelog

### A.13. draft-ietf-eppext-keyrelay-08

1. fixed issue with authinfo
2. fixed issue with relative period in example xml

### A.14. draft-ietf-eppext-keyrelay-09

1. fixed issue with naming

### A.15. draft-ietf-eppext-keyrelay-10

1. removed 4 spaces

### A.16. draft-ietf-eppext-keyrelay-11

1. Processed editorial changes from AD review
2. Processed comments made during IETF last call

### A.17. draft-ietf-regext-keyrelay-00

1. Processed comments made during IESG review

## Authors' Addresses

Rik Ribbers
SIDN
Meander 501
Arnhem 6825 MD
NL

Email: rik.ribbers@sidn.nl
URI: https://www.sidn.nl/

Marc Groeneweg
SIDN
Meander 501
Arnhem 6825 MD
NL

Email: marc.groeneweg@sidn.nl
URI: https://www.sidn.nl/

Miek Gieben

Email: miek@miek.nl

Antoin Verschuren

Email: ietf@antoin.nl