

dnsop
Internet-Draft
Intended status: Informational
Expires: January 10, 2014

W. Kumari
Google
O. Gudmundsson
Shinkuro Inc.
G. Barwood

July 09, 2013

Automating DNSSEC delegation trust maintenance
draft-kumari-ogud-dnsop-cds-03

Abstract

This document describes a method to allow DNS operators to more easily update DNSSEC Key Signing Keys using DNS as communication channel. This document does not address the initial configuration of trust anchors for a domain. The technique described is aimed at delegations in which it is currently hard to move information from the child to parent.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

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

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

This Internet-Draft will expire on January 10, 2014.

Copyright Notice

Copyright (c) 2013 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

Internet-Draft

automating delegation maint

July 2013

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	2
1.1.	Terminology	3
1.2.	Requirements notation	4
2.	Background	4
2.1.	DNS delegations	4
2.2.	Relationship between Parent and Child DNS operator . . .	5
2.2.1.	Solution Space	6
2.2.2.	DNSSEC key change process	6
3.	CDS (Child DS) record definition	6
3.1.	CDS Resource Record Format	7
4.	Automating DS maintenance with CDS records	7
4.1.	CDS processing rules	7
5.	Child's CDS Publication	7
6.	Parent side CDS Consumption	8
6.1.	Detecting a changed CDS	8
6.1.1.	CDS Polling	8
6.2.	Usign the new CDS	8
6.2.1.	Parent calculates DS	9
7.	IANA Considerations	10
8.	Security Considerations	10
9.	Acknowledgements	11
10.	References	11
10.1.	Normative References	11
10.2.	Informative References	12
Appendix A.	RRR background	12
Appendix B.	Changes / Author Notes.	13
	Authors' Addresses	14

[1.](#) Introduction

When a DNS operator first signs their zone, they need to communicate their DS record(s) (or DNSKEY(s)) to their parent through some out-of-band method to complete the chain of trust.

Each time the child changes/rolls the key that is represented in the parent, the new and/or deleted key information has to be communicated

to the parent and published there. How this information is sent to the parent depends on the relationship the child has with the parent. In many cases this is a manual process, and not an easy one. For each key roll, there may be two interactions with the parent. Any manual process is susceptible to mistakes and/or errors. In

addition, due to the annoyance factor of the process, operators may avoid performing key rollovers or skip needed steps to publish the new DS at the parent.

DNSSEC provides data integrity to information published in DNS; thus DNS publication can be used to automate maintenance of delegation information. This document describes a method to automate publication of subsequent DS records, after the initial one has been published.

Readers are expected to be familiar with DNSSEC, including [[RFC4033](#)], [[RFC4034](#)], [[RFC4035](#)], [[RFC5011](#)] and [[RFC6781](#)].

This document is a compilation of two earlier drafts: [draft-barwood-dnsop-ds-publish](#)[[I-D.ds-publish](#)] and [draft-wkumari-dnsop-ezkeyroll](#)

This document outlines a technique in which the parent periodically (or upon request) polls its signed children and automatically publish new DS records. To a large extent, the procedures this document follows are as described in [[RFC6781](#)] [section 4.1.2](#)

This technique is in some ways similar to [RFC 5011](#) style rollovers, but for sub-domains DS records, instead of trust anchors

This technique is designed to be friendly both to fully automated tools and humans. Fully automated tools can perform all the actions needed without human intervention, and thus can monitor when it is safe to move to the next step. Humans can look at the CDS and DS records and easily see if there is a difference.

[1.1](#). Terminology

There terminology we use is defined in this section

Highlighted roles

- o Child: "The entity on record that has the delegation of the domain from the parent"
- o Parent: "The domain in which the child is registered"
- o Child DNS operator: "The entity that maintains and publishes the zone information for the child DNS"
- o Parent DNS operator: "The entity that maintains and publishes the zone information for the parent DNS"

- o Parental Agent: "The entity that the child has relationship with, to change its delegation information."
- o Provisioning system: "A system that the operator of the master DNS server operates to maintain the information published in the DNS. This includes the systems that sign the DNS data."

RRR is our shorthand for Registry/Registrar/Registrant model of parent child relationship see [Appendix A](#) for more.

[1.2.](#) Requirements notation

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.](#) Background

[2.1.](#) DNS delegations

DNS operation consists of delegations of authority. For each delegation there are (most of the time) two parties the parent and child.

In DNS, the parent publishes information about the delegations to the child; for the name-servers it publishes an NS RRset that lists a hint for name-servers that are authoritative for the child. The child also publishes a NS RRset, and this set is the authoritative list of name-servers to the child zone.

The second RRset the parent sometimes publishes is the DS set. The DS RRset provides information about the key(s) that the child has told the parent it will use to sign its DNSKEY RRset. In DNSSEC trust relationship between zones is provided by the following chain:

parent DNSKEY --> DS --> child DNSKEY.

A prior proposal [[I-D.auto-cpsync](#)] suggested that the child send an "update" to the parent via a mechanism similar to Dynamic Update. The main issue became: How does the child find the actual parental agent/server to send the update to? While that could have been solved via technical means, the proposal died.

As the DS record can only be present at the parent [RFC4034](#) [[RFC4034](#)], some other record/method is needed to automate the expression of what the parental zone DS records contents ought to be. One possibility is to use flags in the DNSKEY record. If the SEP bit is set, this indicates that the DNSKEY is intended for use as a secure entry

point. This DNSKEY signs the DNSKEY RRset, and the Parental Agent can calculate DS records based on that. But this fails to meet some operating needs, including the child having no influence what DS digest algorithms are used and DS records can only be published for keys that are in the DNSKEY RRset.

[2.2.](#) Relationship between Parent and Child DNS operator

In the real world, there are many different relationships between the parent and child DNS operators. The type of relationship affects how the child operator communicates with the parent. This section will highlight some of the different situations, but is by no means a complete list.

Different communication paths:

- o Direct/API: The child can change the delegation information via automated/scripted means EPP[RFC5730] used by many TLDs is an example of this. Another example is the web service's programmatic interfaces that Registrars make available to their Reseller's.

- o User Interface: The Child uses a (web) site set up by the Parental Agent for updating delegation information.
- o Indirect: The communication has to be transmitted via out-of-band between two parties, such as email, telephone etc.. This is common when the Child's DNS operator is neither the child itself nor the Registrar for the domain but a third party.
- o Multi-step Combinations: The information flows through an intermediary. It is possible, but unlikely, that all the steps are automated via API's and there are no humans involved.

A domain name holder (Child) may operate its own DNS servers or outsource the operation. While we use the word parent as a singular, parent can consist of single entity or a composite of many discrete parts that have rules and roles. We refer to the entity that the child corresponds with as the Parent.

Another common case is the enterprise case in which an organization may delegate parts of its name-space to be operated by a group that is not the same as that which operates the enterprise's DNS servers. In this case the flow of information is frequently handled in either an ad hoc manner or via some corporate mechanism; this can range from email to fully-automated operation. The word enterprise above covers all organizations where the domains are not sold on the open market and there is some relationship between the entities.

[2.2.1.](#) Solution Space

This document is aimed at the cases in which there is an organizational separation of the child and parent.

A further complication is when the Child DNS Operation is not the Child. There are two common cases of this,

- a) The Parental Agent (e.g. registrar) handles the DNS operation
- b) A third party takes care of the DNS operation.

If the Parental Agent is the DNS operator, life is much easier, as the Parental Agent can inject any delegation changes directly into the Parents Provisioning system. The techniques described below are

not needed in the case when Parental Agent is the DNS operator.

In the case of a third party DNS operator, the Child either needs to relay changes in DNS delegation or give the Child Operator access to its delegation/registration account.

Some parents want the child to express the changes in trust anchors via DS records, while others want to receive DNSKEY records and calculate the DS records themselves. There is no consensus on which method is better; both have good reasons to exist. The proposal below can operate with both models, but the child needs to be aware of the parental policies.

[2.2.2.](#) DNSSEC key change process

After a Child DNS operator first signs the zone, there is a need to interact with the Parent, for example via the delegation account interface, to "upload/paste-in the zone's DS information". The action of logging in through the delegation account user interface authenticates that the user is authorized to change delegation information published in the parent zone. In the case where "Child DNS Operator" does not have access to the registration account, the Child needs to perform the action.

At a later date, the Child Operator may want to publish a new DS record in the parent, either because they are rolling keys, or because they want to publish a stand-by key. This involves performing the same process as before. Furthermore when this is a manual process with cut and paste; operational mistakes will happen. Or worse the update action is not performed at all.

[3.](#) CDS (Child DS) record definition

This document specifies a new DNS RRtype CDS that indicates what the Child wants to be in the parents DS RRset.

The CDS record can be published in the child zone and gives the child more control of what is published for it in the parental zone. The CDS RRset expresses what the DS RRset SHOULD look like after the change; it is a "replace" operation, and it is up to the consumer of the records to translate that into the appropriate add/delete

operations in the registration systems.

[3.1.](#) CDS Resource Record Format

The wire and presentation format of the CDS ("Child DS") record is identical to the DS record [[RFC4034](#)]. IANA has allocated RR code 59 for the CDS record via expert review [[I-D.ds-publish](#)].

No special processing is performed by authoritative servers or by resolvers, when serving or resolving. For all practical purposes CDS is a regular RR type.

[4.](#) Automating DS maintenance with CDS records

CDS records are intended to be "consumed" by delegation trust maintainers. The use of CDS is optional.

[4.1.](#) CDS processing rules

Absence of CDS in child signals "No change" to the current DS set. Following acceptance rules are placed on the CDS record as follows:

- o Location: "the CDS record MUST be at the child zone apex". Q: is "_cds.example. a better location for .example?"
- o Signer: "MUST be signed with a key that is represented in both the current DNSKEY and DS RRset's."
- o Continuity: "SHOULD not break the current delegation if applied to DS RRset"

If any these conditions fail the CDS record MUST be ignored.

[5.](#) Child's CDS Publication

Child DNS Operator SHOULD only publish a CDS RRset when it wants to make a change to the DS RRset in the Parent. The CDS RRset SHOULD be compliant with the rules in [Section 4.1](#). When the Parent DS is "in-sync" with the CDS, the Child DNS Operator SHOULD/MUST delete the CDS RRset.

[6.](#) Parent side CDS Consumption

The CDS RRset MAY be used by the Parental Agent to update the DS RRset in the parent zone. The Parental Agent for this uses a tool that understands the CDS signing rules from [Section 4.1](#) so it may not be able to use a standard validator. Parent SHOULD treat the Continuity rule as "MUST".

[6.1.](#) Detecting a changed CDS

How the Parental Agent gets the CDS record may differ, below are two examples as how this can take place.

Polling The Parental Agent operates a tool that periodically checks each of the children that has a DS record to see if there is a CDS record.

Pushing The delegation user interface has a button {Fetch DS} when pushed preforms the CDS processing. If the Parent zone does not contain DS for this delegation then the "push" MUST be ignored.

In either case the Parental Agent MAY apply additional rules that defer the acceptance of a CDS change, these rules may include a condition that the CDS remains in place and valid for some time period before it is accepted. It may be appropriate in the "Pushing" case to assume that the Child is ready and thus accept changes without delay.

[6.1.1.](#) CDS Polling

This is the only defined use of CDS in this document. There are limits to the scalability of polling techniques, thus some other mechanism is likely to be specified later that addresses CDS usage in the situation where polling does not scale to. Having said that Polling will work in many important cases like enterprises, universities, small TLD's etc.

If the CDS RRset does not exist, the Parental Agent MUST take no action. Specifically it MUST NOT delete or alter the existing DS RRset.

[6.2.](#) Usign the new CDS

Regardless of how the Parent DNS operator detected changes to a CDS RR, the Parent DNS operator MUST use a DNSSEC validator to obtain a validated CDS RRset from the Child zone. It would be a good idea if the Parent DNS operator checked all NS RRs listed at the delegation.

However, due to the use of technologies such as load balancing and anycast, this should not be taken as proof that the new CDS is present on all nodes serving the Child zone.

The Parent DNS operator MUST ensure that old versions of the CDS RRset do not overwrite newer versions. This MAY be accomplished by checking that the signature inception in the RRSIG for CDS is newer and/or the serial number on the child's SOA is greater. This may require the Parent DNS operator to maintain some state information.

The Parent DNS operator MAY take extra security measures. For example, to mitigate the possibility that a Child's key signing key has been compromised, the Parent DNS operator may, for example, inform (by email or other methods) the Child DNS operator of the change. However the precise out-of-band measures that a parent zone SHOULD take are outside the scope of this document.

Once the Parent DNS operator has obtained a valid CDS it MAY double check the publication rules from [section 4.1](#). In particular the Parent DNS operator MUST double check the Continuity rule and do its best not to invalidate the Child zone. Once checked and if the CDS and DS ``differ'' it may apply the changes to the parent zone.

[6.2.1](#). Parent calculates DS

There are cases where the Parent wants to calculate the DS record due to policy reasons. In this case, the Child can still publish a CDS records instructing the parent which DNSKEY's to represent in the DS RRset. This requires publication of future keys in the DNSKEY RRset for the parent to be able to calculate the DS record. The DNS Parent needs to publish guidelines for the children as to what digest algorithms are acceptable in the CDS record.

When a Parent operates in "calculate DS" mode it can operate in one of two sub-modes

full i.e. it only publishes DS records it calculates from DNSKEY records,

augment i.e. it will make sure there are DS records for the digest algorithm(s) it requires(s).

Implications on Parental Agent are that the CDS and DS are not exactly the same after update thus it needs to take that into consideration when checking CDS records. Same goes for the Child Operator, it needs to be able to detect that the new DS RRset is

"equivalent" to the current CDS RRset, thus it can remove the CDS RRset.

[7.](#) IANA Considerations

IANA has assigned RR Type code 59 for CDS. This was done for an earlier version of this document[I-D.ds-publish] This document is to become the reference for CDS RRtype.

[8.](#) Security Considerations

[This needs more work, suggestions welcome.]

This work is for the normal case, when things go wrong there is only so much that automation can fix.

If child breaks DNSSEC validation by removing all the DNSKEYs that are represented in the DS set its only repair actions are to contact the parent or restore the DNSKEYs in the DS set.

In the event of a compromise of the server or system generating signatures for a zone, an attacker might be able to generate and publish new CDS records. The modified CDS records will be picked up by this technique and so may allow the attacker to extend the effective time of his attack. If there a delay in accepting changes to DS, as in [RFC5011](#), then the attacker needs to hope his activity is not detected before the DS in parent is changed. If this type of change takes place, the child need to contact the parent (possibly via a registrar web interface) and remove any compromised DS keys.

A compromise of the account with the parent (e.g. registrar) will not be mitigated by this technique, as the "new registrant" can delete/modify the DS records at will.

While it may be tempting, this SHOULD NOT be used for initial enrollment of keys since there is no way to ensure that the initial key is the correct one. If is used, strict rules for inclusion of keys like hold down times, challenge data inclusion etc., ought to be used, along with some kind of challenge mechanism.

The CDS RR type should allow for enhanced security by simplifying process. Since rollover is automated, updating a DS RRset by other

means may be regarded as unusual and subject to extra security checks.

If there is a failure in applying changes in child zone to all DNS servers listed in either parent or child NS set it is possible that the Parental agent may get confused either not perform action because it gets different answers on different checks or CDS validation fails. In the worst case Parental Agent performs an action reversing a prior action but after the child signing system decides to take the next step in rollover, resulting in a broken delegation.

DNS is a loosely coherent distributed database with local caching; therefore it is important to allow old information to expire from caches before deleting DS or DNSKEY records. Similarly, it is important to allow new records to propagate through the DNS before use, see [[RFC6781](#)] and [[I-D.key-time](#)]

[9.](#) Acknowledgements

This is by no means the invention of the authors. This idea has been floating around for a long time. This simply documents it for discussion.

We would like to thank: Joe Abley, Roy Arends, Jim Galvin, Cricket Liu, Stephan Lagerholm, Matt Larson, Olaf Kolkman, Suzanne Woolf, Paul Wouters, Wes Hardaker, Doug Barton, Brian Dickinson, Marco Sanz, Tony Finch, Antoin Verschuren, Edward Lewis Matthijs Meeking, John Dickinson.

There were a large number of other folk with whom we discussed this, apologies for not remembering everyone.

[10.](#) References

[10.1.](#) 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.

Kumari, et al. Expires January 10, 2014 [Page 11]

Internet-Draft automating delegation maint July 2013

- [RFC5011] StJohns, M., "Automated Updates of DNS Security (DNSSEC) Trust Anchors", STD 74, [RFC 5011](#), September 2007.
- [RFC6781] Kolkman, O., Mekking, W., and R. Gieben, "DNSSEC Operational Practices, Version 2", [RFC 6781](#), December 2012.

[10.2.](#) Informative References

- [I-D.auto-cpsync]
Mekking, W., "Automated (DNSSEC) Child Parent Synchronization using DNS UPDATE", [draft-mekking-dnsop-auto-cpsync-01](#) (work in progress), December 2010.
- [I-D.ds-publish]
Barwood, G., "DNS Transport", [draft-barwood-dnsop-ds-publish-02](#) (work in progress), June 2011.
- [I-D.key-time]
Mekking, W., "DNSSEC Key Timing Considerations", [draft-ietf-dnsop-dnssec-key-timing-03](#) (work in progress), July 2012.
- [RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, [RFC 5730](#), August 2009.

- [RFC5734] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Transport over TCP", STD 69, [RFC 5734](#), 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.

[Appendix A](#). RRR background

In the RRR world, the different parties are frequently from different organizations. In the single enterprise world there are also organizational/geographical/cultural separations that affect how information flows from a Child to the parent.

Due to the complexity of the different roles and interconnections, automation of delegation information has been punted in the past. There have been some proposals to automate this, in order to improve the reliability of the DNS. These proposals have not gained enough traction to become standards.

For example in many of the TLD cases there is the RRR model (Registry, Registrar and Registrant). The Registry operates DNS for

Kumari, et al.	Expires January 10, 2014	[Page 12]
----------------	--------------------------	-----------

Internet-Draft	automating delegation maint	July 2013
----------------	-----------------------------	-----------

the TLD, the Registrars accept registrations and place information into the Registries database. The Registrant only communicates with the Registrar; frequently the Registry is not allowed to communicate with the Registrant. In that case as far as the registrant is concerned the Registrar == Parent.

In many RRR cases the Registrar and Registry communicate via EPP[RFC5730] and use the EPP DNSSEC extension [[RFC5910](#)]. In a number of ccTLDs there are other mechanisms in use as well as EPP, but in general there seems to be a movement towards EPP usage when DNSSEC is enabled in the TLD.

[Appendix B](#). Changes / Author Notes.

[RFC Editor: Please remove this section before publication]

From 02 to 03

- o Applied comments by Matthijs Mekking
- o Incorporated suggestions from Edward Lewis about structure
- o Reworked structure to be easier for implementors to follow
- o Applied many suggestions from a wonderful thorough review by John Dickinson
- o Removed the going Unsigned option

From 01 to 02

- o Major restructuring to facilitate easier discussion
- o Lots of comments from DNSOP mailing list incorporated, including making draft DNSKEY/DS neutral, explain different relationships that exists,
- o added more people to acks.
- o added description of enterprise situations
- o Unified on using Parental Agent over Parental Representative
- o Removed redundant text when possible
- o Added text to explain what can go wrong if not all child DNS servers are in sync.

Kumari, et al.

Expires January 10, 2014

[Page 13]

Internet-Draft

automating delegation maint

July 2013

- o Reference prior work by Matthijs Mekking
- o Added text when parent calculates DS from DNSKEY

From - to -1.

- o Removed from section .1: "If a child zone has gone unsigned, i.e. no DNSKEY and no RRSIG in the zone, the parental representative MAY treat that as intent to go unsigned. (NEEDS DISCUSSION)." Added new text at end. -- suggestion by Scott Rose 20/Feb/13.

- o Added some background on the different DNS Delegation operating situations and how they affect interaction of parties. This moved some blocks of text from later sections into here.
- o Number of textual improvements from Stephan Lagerholm
- o Added motivation why CDS is needed in CDS definition section
- o Unified terminology in the document.
- o Much more background

Authors' Addresses

Warren Kumari
 Google
 1600 Amphitheatre Parkway
 Mountain View, CA 94043
 US

Email: warren@kumari.net

Olafur Gudmundsson
 Shinkuro Inc.
 4922 Fairmont Av, Suite 250
 Bethesda, MD 20814
 USA

Email: ogud@ogud.com

George Barwood
 33 Sandpiper Close
 Gloucester GL2 4LZ
 United Kingdom

Email: george.barwood@blueyonder.co.uk