Intended Status: Informational
DNS Operations
Internet-Draft
Expires: April 26, 2011

M. Larson
VeriSign
O. Gudmundsson
Shinkuro Inc.

October 23, 2010

DNSSEC Trust Anchor Configuration and Maintenance
draft-ietf-dnsop-dnssec-trust-anchor-04

Abstract

This document recommends a preferred format for specifying trust anchors in DNSSEC validating security-aware resolvers and describes how such a resolver should initialize trust anchors for use. This document also describes different mechanisms for keeping trust anchors up to date over time.

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 April 26, 2011.

Copyright Notice

described in the Simplified BSD License.

Table of Contents

1. Introduction

The DNSSEC standards documents ([RFC4033], [RFC4034] and [RFC4035]) describe the need for trust anchors and how they are used. A validating security-aware resolver (subsequently referred to as a "validating resolver") needs to be configured with one or more trust anchors, which specify the public keys of signed zones. To authenticate DNS data, a validating resolver builds a chain of trust from a configured trust anchor to that data.

The DNS root zone is signed and a validating resolver needs to be configured with at least the root zone's trust anchor. A validating resolver might need additional trust anchors configured to accommodate islands of security. (An island of security is a signed, delegated zone that does not have an authentication chain from its delegating parent.) Consider the situation now that the root zone is signed but when a given top-level domain (TLD) zone is not signed. Various second-level zones under this unsigned TLD might be signed and resolver operators might want to validate responses from those zones, requiring a validating resolver to be configured with those zones' trust anchors. Note islands of security can appear at any depth in the DNS tree.

Because many different validating resolvers need be configured there is a benefit to creating a common trust anchor format. A similar situation has occurred with the "root hints", the list of root name server names and IP addresses: this information is distributed in standard master file format and many resolver implementations support this common format.

To simplify this trust anchor configuration process that will occur on a large number of resolvers, this document offers guidance to validating resolver implementers by specifying a standardized format for describing trust anchors. The document also describes how a validating resolver should initialize or "prime" trust anchors before first use. Finally, the document lists options for keeping trust

anchor information current over time.

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 RFC 2119 [RFC2119].

2. Trust Anchor Format and Storage

A trust anchor is a DNSSEC public key configured in a validating resolver. A validating resolver's configuration MUST allow one or more trust anchors to be specified. According to the definition in Section 2 of RFC 4033 [RFC4033], a trust anchor can be specified as either a public key from a DNSKEY resource record (RR) or the hash of a public key as found in a DS RR. (DS records are defined in Section 5 of RFC 4034 [RFC4034].)

This document RECOMMENDS that a trust anchor be specified using the hash of a public key rather than the key itself, i.e., the fields from a DS record rather than from a DNSKEY record. A trust anchor specified in this manner will use all the fields from the corresponding key's DS record, including the owner name to indicate which zone the trust anchor corresponds to as well as the various fields from the DS RDATA. The digest algorithm SHOULD be SHA-256 [RFC4509], which is DS digest type 2. DS records using SHA-1 (DS digest type 1) to specify trust anchors are NOT RECOMMENDED: RFC 4509 encourages the use of DS RRs using SHA-256 over those using SHA-1.

Specifying a trust anchor using a DS format instead of a DNSKEY format offers an advantage because it forces the resolver to make a DNS query to obtain the trust anchor's complete DNSKEY RRSet during a priming operation (described below). If only a DNSKEY record were specified, resolver implementers could conceivably avoid priming the trust anchor. But priming is desirable because it causes the resolver to retrieve an up-to-date version of a zone's DNSKEY RRSet from one of the zone's authoritative servers. It should be noted

that in practice, priming is frequently required, when the data in the trust anchor zone is signed with a different key than the one configured as the trust anchor.

Using a DS format is also recommended because it is smaller than the DNSKEY format and is easier to compare manually, either by typing or cutting and pasting.

2.1. Trust Anchor Storage

For trust anchors to be useful the validating resolver needs to be able to read a file with the trust anchors. This document recommends that all resolvers be able to read trust anchors specified in a file in the following format:

ZoneName [DS] KeyTag DNSKEY-Algorithm Digest-type Digest

Any truncated digest SHOULD be ignored. The text "DS" in input is optional. The input format assumes that the trust anchor is either

in the IN class or is valid in all classes.

Validating resolvers ought to be able write out a list of current trust anchors in the format above. Validating resolvers that perform trust anchor maintenance MUST be able to update their trust anchor storage.

Example: (ID width rules force text onto two lines)

```
. 19036 8 2
49AAC11D7B6F6446702E54A1607371607A1A41855200FD2CE1CDDE32F24E8FB5
```

Note: Trust anchor maintenance [RFC5011] and other schemas may require a different format as timers and other meta data is needed.

3. Trust Anchor Priming

A validating resolver needs to obtain and validate the DNSKEY RRSet corresponding to a configured DS for that trust anchor to be usable in DNSSEC validation. This process is called "priming" the trust anchor. Priming can occur when the validating resolver starts, but a validating resolver may want to defer priming of individual trust anchors until each is first needed for verification. This priming on demand is especially important when a validating resolver is configured with a large number of trust anchors to avoid sending a large number of DNS queries on startup. This section adds additional details to the discussion of trust anchors in Section 5 of RFC 4035 [RFC4035].

Following are the steps a validating resolver SHOULD take to prime a

configured trust anchor:

1. Read the trust anchor's information (corresponding to the fields in a DS record as descried above) from the validating resolver's configuration (e.g., a text file).

2. Look up the DNSKEY RRSet corresponding to the owner name of the trust anchor. (The validating resolver can either perform iterative resolution or request recursive service from a recursive name server, depending on its capabilities.)

3. Verify that one of the DNSKEY RR(s) correspond to one the configured trust anchor(s) (i.e., one of the DNSKEY whose hash is configured) appears in the DNSKEY RRSet and that this DNSKEY RR has the Zone Key Flag (DNSKEY RDATA bit 7) set. (This bit only indicates that the DNSKEY is allowed to sign the zone data. This DNSKEY may or may not be a zone signing key (ZSK) as defined in RFC 4641 [RFC4641].)

4. Verify that the DNSKEY RRSet is signed by one of the DNSKEYs found in the previous step, i.e., that there exists a valid RRSIG (cryptographically and temporally) for the DNSKEY RRSet generated with the private key corresponding to the DNSKEY found in the previous step.

If the validating resolver can successfully complete the steps above, all DNSKEY RRs in the RRSet ought to be considered authenticated and can be used to authenticate RRSets at or below the trust anchor. There is one exception: if the revoke bit used by the trust anchor automated update protocol RFC 5011 [RFC5011] is set, the trust anchor MUST be removed and not used.

If any of the steps above result in an error, the validating resolver

SHOULD log them and abort the verification as specified in section 5.5 of RFC 4035 [RFC4035].

If there are multiple trust anchors configured for a zone, any one of them is sufficient to validate data in the zone. For this reason, old trust anchors SHOULD be removed from a validating resolver's trust anchor list soon after the corresponding keys are no longer used by the zone, as described in RFC 5011 [RFC5011]. Even if a

trust anchor is not used in resolution, a validating resolver needs to query for it frequently enough to detect changes as prescribed in RFC5011.

If a validating resolver is unable to retrieve a signed DNSKEY RRSet corresponding to a trust anchor (i.e., prime the trust anchor), it SHOULD log this condition as an error. Inability to prime a zone's trust anchor results in the validating resolver's inability to validate data from the corresponding zone. The validating resolver MUST treat this zone as bogus, until such time it is able to get a DNSKEY set validated by a trust anchor.

4. Trust Anchor Maintenance

Trust anchors usually correspond to zones' key signing keys and these keys do change in the course of normal operation. It is up to validating resolver operators to ensure that configured trust anchor information remains current and does not go stale: each configured trust anchor SHOULD correspond to a DNSKEY RR in the trust anchor zone's apex DNSKEY RRSet. This process is called trust anchor maintenance. (Initial trust anchor configuration requires human intervention to verify the trust anchor's authenticity using out-of-band means and is outside the scope of this document.)

This section provides a brief overview of some possible mechanisms to keep trust anchor information current:

Manual configuration: The validating resolver operator MAY choose to maintain trust anchor information completely manually. In this case, the operator assumes responsibility for noticing stale trust anchor information (i.e., DS records that no longer point to a corresponding DNSKEY RR in the trust anchor zone's apex DNSKEY RRSet) and updating that information. This process MAY require the operator to use the same out-of-band verification mechanism as used for initial configuration to ensure that the new trust anchor DS record is trustworthy. Because manual maintenance is burdensome and prone to error, and because other automated trust anchor maintenance processes either exist or are in development, manual trust anchor maintenance is NOT RECOMMENDED.

DNSSEC In-band Update: RFC 5011 [RFC5011] defines an automated way keep DNSSEC trust anchors updated. This protocol relies on a small DNSSEC protocol change (an additional flag in the DNSKEY record) and can be implemented either in a validating resolver itself or in an external program with access to the validating resolver's trust anchor configuration data.

Trusted update mechanism: Updated trust anchor information MAY be obtained via a trusted non-DNS update mechanism. One possibility is the operating system update mechanism provided by most software vendors. Operators already place considerable trust in this mechanism, so it is reasonable to extend this trust to allow distribution and update of DNSSEC public key material. Another possibility is to obtain trust anchor configuration directly from the validating resolver software vendor. A possible error condition in this mechanism is that a machine is brought up with an "old" trust configuration, like when a machine is configured from an old media or brought out of storage. The machines ought to be able to detect the fact the list of trust anchors is "out-of-date" and fetch a more recent update. During this process it

may be necessary to disable DNSSEC and only depend on the keys for the update mechanism to authorize the changes to the configuration.

Combination of update mechanisms: It is possible that for a given validating resolver, different trust anchors will be maintained by different mechanisms. For example, some trust anchors might be kept up to date by a trusted update mechanism and others maintained by some site-specific mechanism. In this case, it is important that the mechanisms maintain a mutually exclusive set of trust anchors.

The out-of-sync errors described above in the "Trusted update mechanism" section can occur if the system the validating resolver is offline or in storage for an extend period or reinstalled.

Trust Anchor Repositories (TAR) are sometimes mentioned at the same time as a trust anchor configuration. TARs are in essence an outsourced trust anchor maintenance mechanism, where the user can avoid maintaining a large set of trust anchors by only configuring the root zone's key and the TAR key.

5. Security considerations

This document proposes a standard format for documenting DNSSEC trust anchors. Configuration of trust anchors, especially those obtained from third parties as part of an automated process, is a critical security operation. The procedures listed above describe the minimal checks that should be performed and reporting that should be done when configuring trust anchors.

The root zone is now signed and many TLD's are planning DNSSEC deployment. This state of affairs greatly reduces the number of trust anchors that validating resolvers need to configure and maintain.

If multiple mechanisms are updating the trust anchor list then there is the possibility of conflict, such as one mechanism reinserting an expired trust anchor.

Trust anchors are configuration information. A validating resolver ought to treat this information differently than DNS data obtained over the network and never use the configured trust anchors as part of an answer.

A signed zone that plans to transition to an unsigned state must first give a warning that it is going insecure, such as using the technique described in RFC 5011 [RFC5011]. Failure to do so will cause all validating resolvers that keep a trust anchor for the zone configured to treat responses from the zone as bogus, causing resolution failures.

6. IANA considerations

This document does not have any IANA actions.

7. Acknowledgments

This work was undertaken at the suggestion of the DNSSEC Deployment working group (www.dnssec-deployment.org). The following people are acknowledged for contributing to this document: Alfred Hoenes, Edward Lewis, Wes Hardaker, Geoff Huston, Paul Hoffman, Matthijs Mekking, Scott Rose, Paul Wouters.

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.

[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.

[RFC4509] Hardaker, W., "Use of SHA-256 in DNSSEC Delegation Signer (DS) Resource Records (RRs)", RFC 4509, May 2006.

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

8.2. Informative References

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

Authors' Addresses

Matt Larson
VeriSign, Inc.
21345 Ridgetop Circle
Dulles, VA 20166-6503
USA

Email: mlarson@verisign.com

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

Email: ogud@ogud.com