Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: April 8, 2017

D. Wessels
Verisign
W. Kumari
Google
P. Hoffman
ICANN

October 5, 2016

Signaling Trust Anchor Knowledge in DNS Security Extensions (DNSSEC)
draft-ietf-dnsop-edns-key-tag-03

Abstract

The DNS Security Extensions (DNSSEC) were developed to provide origin authentication and integrity protection for DNS data by using digital signatures. These digital signatures can be verified by building a chain-of-trust starting from a trust anchor and proceeding down to a particular node in the DNS. This document specifies two different ways for validating resolvers to signal to a server which keys are referenced in their chain-of-trust. The data from such signaling allow zone administrators to monitor the progress of rollovers in a DNSSEC-signed zone.

This document describes two independent methods for validating resolvers to publish their referenced keys: an EDNS option and a different DNS query. The reason there are two methods instead of one is some people see significant problems with each method. Having two methods is clearly worse than having just one, but it is probably better for the Internet than having no way to gain the information from the resolvers.

Table of Contents

1. Introduction

The DNS Security Extensions (DNSSEC) [RFC4033], [RFC4034] and [RFC4035] were developed to provide origin authentication and integrity protection for DNS data by using digital signatures. DNSSEC uses Key Tags to efficiently match signatures to the keys from which they are generated. The Key Tag is a 16-bit value computed from the RDATA portion of a DNSKEY RR using a formula not unlike a ones-complement checksum. RRSIG RRs contain a Key Tag field whose value is equal to the Key Tag of the DNSKEY RR that validates the signature.

Likewise, Delegation Signer (DS) RRs also contain a Key Tag field whose value is equal to the Key Tag of the DNSKEY RR to which it refers.

This draft sets out to specify a way for validating resolvers to tell a server in a DNS query which DNSSEC key(s) they would use to validate responses from that zone. This is done in two ways: using an EDNS option for use in the OPT meta-RR [RFC6891] that contains the key tags (described in Section 4), and by periodically sending special "key tag queries" to a server authoritative for the zone (described in Section 5).

Each of these new signaling mechanisms is OPTIONAL to implement and use. These mechanisms serve to measure the acceptance and use of new DNSSEC trust anchors and key signing keys (KSKs). This signaling data can be used by zone administrators as a gauge to measure the successful deployment of new keys. This is of particular interest for the DNS root zone in the event of key and/or algorithm rollovers that rely on [RFC5011] to automatically update a validating DNS resolver's trust anchor.

This document does not introduce new processes for rolling keys or updating trust anchors. Rather, it specifies a means by which a DNS query can signal the set of keys that a client uses for DNSSEC validation.

1.1. Design Evolution

Initially this document was named draft-edns-key-tag and proposed including Key Tag values in a new EDNS(0) option code. It was modeled after [RFC6975], which provides DNSSEC algorithm signaling.

The authors received feedback from Working Group participants that it might be better to convey Key Tags in QNAME of a separate DNS query, rather than as an EDNS(0) option. Mostly this is because forwarding

(e.g. from stub to recursive to authoritative) could be problematic. Reasons include:

1. EDNS(0) is a hop-by-hop protocol. Unknown option codes would not be forwarded by default, as per [RFC6891].

2. Middleboxes might block entire queries containing unknown EDNS(0) option codes.

3. A recursive might need to remember Key Tag values (i.e., keep state) received from its stub clients and then forward them at a later opportunity.

One advantage of the EDNS(0) option code is that it is possible to see that a stub client has a different Key Tag list than its forwarder. In the QNAME-based approach, this is not possible because queries originated by a stub and a forwarder are indistinguishable. The authors feel this advantage is not sufficient to justify the EDNS(0) approach.

One downside to the QNAME approach is that it uses a separate query, whereas with EDNS(0) the Key Tag values are "piggy-backed" on to an existing DNSKEY query. For this reason, this document recommends only sending QNAME-based key tag queries for configured trust anchors, although EDNS-based key tags can be sent with any DNSKEY query.

Another downside to the QNAME-based approach is that since the trust anchor zone might not contain labels matching the QNAME, these queries could be subject to aggressive negative caching features now in development by the Working Group. This could affect the amount of signaling sent by some clients compared to others.

A probably minor downside to the QNAME-based approach is that it cannot be used with extremely long query names if the addition of the prefix would cause the name to be longer than 255 octets.

2. Requirements Terminology

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].

3. Terminology

Trust Anchor: A configured DNSKEY RR or DS RR hash of a DNSKEY RR. A validating security-aware resolver uses this public key or hash as a starting point for building the authentication chain to a

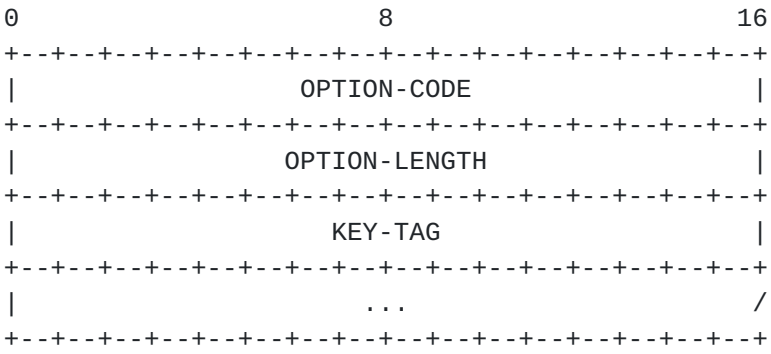
signed DNS response. In general, a validating resolver will have to obtain the initial values of its trust anchors via some secure or trusted means outside the DNS protocol. Presence of a trust anchor also implies that the resolver should expect the zone to which the trust anchor points to be signed. (quoted from [RFC4033] Section 2)

Key Tag: A 16-bit integer that identifies and enables efficient selection of DNSSEC public keys. A Key Tag value can be computed over the RDATA of a DNSKEY RR. The Key Tag field in the RRSIG and DS records can be used to help select the corresponding DNSKEY RR efficiently when more than one candidate DNSKEY RR is available. For most algorithms the Key Tag is a simple 16-bit modular sum of the DNSKEY RDATA. See [RFC4034] Appendix B.

4. Using the edns-key-tag Option

4.1. Option Format

The edns-key-tag option is encoded as follows:



where:

OPTION-CODE: The EDNS0 option code assigned to edns-key-tag, [TBD].

OPTION-LENGTH: The value 2 x number of key-tag values present.

KEY-TAG: One or more 16-bit Key Tag values ([RFC4034], Appendix B).

4.2. Use By Queriers

A validating resolver sets the edns-key-tag option in the OPT meta-RR when sending a DNSKEY query. The validating resolver SHOULD also set the DNSSEC OK bit [RFC4034] to indicate that it wishes to receive DNSSEC RRs in the response.

A DNS client MUST NOT include the edns-key-tag option for non-DNSKEY queries.

The KEY-TAG value(s) included in the edns-key-tag option represent the Key Tag of the Trust Anchor or DNSKEY RR that will be used to validate the expected response. When the client sends a DNSKEY query, the edns-key-tag option represents the Key Tag(s) of the KSK(s) of the zone for which the server is authoritative. A validating resolver learns the Key Tag(s) of the KSK(s) from the zone's DS record(s) (found in the parent), or from a configured trust anchor.

A DNS client SHOULD include the edns-key-tag option when issuing a DNSKEY query for a zone corresponding to a configured Trust Anchor.

A DNS client MAY include the edns-key-tag option when issuing a DNSKEY query for a non-Trust Anchor zone (i.e., Key Tags learned via DS records). Since some DNSSEC validators implement bottom-up validation, non-Trust Anchor Key Tags zone might not be known at the time of the query. Such a validator can include the edns-key-tag option based on previously cached data.

A DNS client MUST NOT include Key Tag(s) for keys which are not learned via either configured Trust Anchor or DS records.

Since the edns-key-tag option is only set in the query, if a client sees these options in the response, no action needs to be taken and the client MUST ignore the option values.

4.2.1. Stub Resolvers

Typically, stub resolvers rely on an upstream recursive server (or cache) to provide a response. Optimal setting of the edns-key-tag option depends on whether the stub resolver elects to perform its own validation.

4.2.1.1. Validating Stub Resolvers

A validating stub resolver sets the DNSSEC OK (DO) bit [RFC4034] to indicate that it wishes to receive additional DNSSEC RRs (i.e., RRSIG RRs) in the response. Such validating resolvers SHOULD include the edns-key-tag option in the OPT RR when sending a DNSKEY query.

4.2.1.2. Non-validating Stub Resolvers

The edns-key-tag option MUST NOT be included by non-validating stub resolvers.

4.2.2. Recursive Resolvers

4.2.2.1. Validating Recursive Resolvers

A validating recursive resolver is, by definition, configured with at least one trust anchor. Thus, a recursive resolver SHOULD include the edns-key-tag option in its DNSKEY queries as described above.

In addition, the clients of a validating recursive resolver might be configured to do their own validation, with their own trust anchor(s). When a validating recursive resolver receives a query that includes the edns-key-tag option with a Key Tag list that differs from its own, it SHOULD forward both the client's Key Tag list as well as its own. When doing so, the recursive resolver SHOULD transmit the two Key Tag lists using separate instances of the edns-key-tag option code in the OPT meta-RR. For example, if the recursive resolver's Key Tag list is (19036, 12345) and the stub/client's list is (19036, 34567), the recursive would include the edns-key-tag option twice: Once with values (19036, 12345) and once with values (19036, 34567).

A validating recursive resolver MAY combine stub/client Key Tag values from multiple incoming queries into a single outgoing query. It is RECOMMENDED that implementations place reasonable limits on the number of Key Tags to include in the outgoing edns-key-tag option.

If the client included the DO and Checking Disabled (CD) bits, but did not include the edns-key-tag option in the query, the validating recursive resolver MAY include the option with its own Key Tag values in full.

Validating recursive resolvers MUST NOT set the edns-key-tag option in the final response to the stub client.

4.2.2.2. Non-validating Recursive Resolvers

Recursive resolvers that do not validate responses SHOULD copy the edns-key-tag option seen in received queries, as they represent the wishes of the validating downstream resolver that issued the original query.

4.3. Use By Responders

An authoritative name server receiving queries with the edns-key-tag option MAY log or otherwise collect the Key Tag values to provide information to the zone operator.

A responder MUST NOT include the edns-key-tag option in any DNS response.

5. Using the Key Tag Query

5.1. Query Format

A key tag query consists of a standard DNS query of type NULL and of class IN [RFC1035].

The first component of the query name is the string "_ta-" followed by a sorted, hyphen-separated list of hexadecimal-encoded Key Tag values. The zone name corresponding to the trust anchor is appended to this first component.

For example, a validating DNS resolver that has a single root zone trust anchor with key tag 17476 (decimal) would originate a query of the form QTYPE=NULL, QCLASS=IN, QNAME=_ta-4444.

Hexadecimal values MUST be zero-padded. For example, if the key tag is 999 (decimal), it is represented in hexadecimal as 03e7.

When representing multiple key tag values, they MUST be sorted in order from smallest to largest. For example, A validating DNS resolver that has a three trust anchors for the example.com zone with key tags 1589, 43547, 31406 (decimal) would originate a query of the form QTYPE=NULL, QCLASS=IN, QNAME=_ta-0635-7aae-aa1b.example.com.

5.2. Use By Queriers

A validating DNS resolver (stub or recursive) SHOULD originate a key tag query whenever it also originates a DNSKEY query for a configured Trust Anchor zone. In other words, the need to issue a DNSKEY query is also the trigger to issue a key tag query.

The value(s) included in the key tag query represent the Key Tag(s) of the Trust Anchor that will be used to validate the expected DNSKEY response.

A DNS validating resolver SHOULD NOT originate key tag queries when also originating DNSKEY queries for non-Trust Anchor zones.

A non-validating DNS resolver MUST NOT originate key tag queries.

DNS resolvers with caches SHOULD cache and reuse the response to a key tag query just as it would any other response.

5.3. Use By Responders

An authoritative name server receiving key tag queries MAY log or otherwise collect the Key Tag values to provide information to the zone operator.

An authoritative name server MUST generate an appropriate response to the key tag query. A server does not need to have built-in logic that determines the response to key tag queries: the response code is determined by whether the data is in the zone file or covered by wildcard. The zone operator might want to add specific key tag records to its zone, perhaps with specific TTLs, to affect the frequency of key tag queries from clients. [Note RFC1035 says NULL RRs are not allowed in master files, but I believe that to be incorrect]

5.3.1. Interaction With Aggressive Negative Caching

Aggressive NSEC/NSEC3 negative caching [draft-ietf-dnsop-nsec-aggressiveuse] may also affect the quality of key tag signaling. When the response code for a key tag query is NXDOMAIN, DNS resolvers that implement aggressive negative caching will send fewer key tag queries than resolvers that do not implement it.

For this reason, zone operators might choose to create records corresponding to expected key tag queries. During a rollover from key tag 1111 (hex) to key tag 2222 (hex), the zone could include the following records:

```
_ta-1111        IN   NULL   \# 0
_ta-2222        IN   NULL   \# 0
_ta-1111-2222   IN   NULL   \# 0
```

Recall that when multiple key tags are present, the originating client MUST sort them from smallest to largest in the query name.

6. IANA Considerations

The IANA is directed to assign an EDNS0 option code for the edns-key-tag option from the DNS EDNS0 Option Codes (OPT) registry as follows:

Value	Name	Status	Reference
[TBA]	edns-key-tag	Optional	[This document]

7. Security Considerations

This document specifies a way for a client to signal its trust anchor knowledge to a cache or server. The goal of these optional mechanisms is to signal new trust anchor uptake in clients to allow zone administrators to know when it is possible to complete a key rollover in a DNSSEC-signed zone.

There is a possibility that an eavesdropper or server could infer the validator in use by a client by the Key Tag list seen. This may allow an attacker to find validators using old, possibly broken, keys. It could also be used to identify the validator or narrow down the possible validator implementations in use by a client, which could have a known vulnerability that could be exploited by the attacker.

Consumers of data collected from the mechanisms are advised that provided Key Tag values might be "made up" by some DNS clients with malicious or at least mischievous intentions. For example, an attacker with sufficient resources might try to generate large numbers of queries including only old Key Tag values, with the intention of delaying the completion of a key rollover.

DNSSEC does not require keys in a zone to have unique Key Tags. During a rollover there is a small possibility that an old key and a new key will have identical Key Tag values. Zone operators relying on the edns-key-tag mechanism SHOULD take care to ensure that new keys have unique Key Tag values.

8. Privacy Considerations

This proposal adds additional, optional "signaling" to DNS queries in the form of Key Tag values. While Key Tag values themselves are not considered private information, it may be possible for an eavesdropper to use Key Tag values as a fingerprinting technique to identify particular DNS validating clients. This may be especially true if the validator is configured with trust anchor for zones in addition to the root zone.

A validating resolver need not transmit the key tags in every applicable query. Due to privacy concerns, such a resolver MAY choose to transmit the key tags for a subset of queries (e.g., every 25th time), or by random chance with a certain probability (e.g., 5%).

Implementations of this specification MAY be administratively configured to only transmit the key tags for certain zones. For example, the software's configuration file may specify a list of

zones for which use of the mechanisms described here is allowed or denied. Since the primary motivation for this specification is to provide operational measurement data for root zone key rollovers, it is RECOMMENDED that implementations at least include the edns-key-tag option for root zone DNSKEY queries.

9. Acknowledgments

This document was inspired by and borrows heavily from [RFC6975] by Scott Rose and Steve Crocker. The authors would like to thank Mark Andrews, Casey Deccio, Burt Kalisky, Bob Harold, Edward Lewis, Tim Wicinski, Suzanne Woolf, and other members of the dnsop working group for their input.

10. References

10.1. Normative References

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <http://www.rfc-editor.org/info/rfc1035>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

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

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

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

[RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <http://www.rfc-editor.org/info/rfc6891>.

10.2. Informative References

[draft-ietf-dnsop-nsec-aggressiveuse]
Fujiwara, K., "Aggressive use of NSEC/NSEC3", 2016.

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

[RFC6975] Crocker, S. and S. Rose, "Signaling Cryptographic Algorithm Understanding in DNS Security Extensions (DNSSEC)", RFC 6975, DOI 10.17487/RFC6975, July 2013, <http://www.rfc-editor.org/info/rfc6975>.

Appendix A. Changes / Author Notes.

[RFC Editor: Please remove this section before publication]

From -01 to -02:

- Added QNAME-based method of signaling key tags.
- Added Paul Hoffman as coauthor.

From -00 to -01:

- Changed how a recursive should combine a stub's key tag values with its own. Previously it was to be a union of key tag values. Now it is a separate instance of the option code for recursive and stub.
- Added Warren as coauthor.

Authors' Addresses

Duane Wessels
Verisign
12061 Bluemont Way
Reston, VA 20190
United States

Phone: +1 703 948-3200
Email: dwessels@verisign.com
URI: http://verisigninc.com

Warren Kumari
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
United States

Email: warren@kumari.net

Paul Hoffman
ICANN

Email: paul.hoffman@icann.org