              A Root Key Trust Anchor Sentinel for DNSSEC

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 a mechanism that
   will allow an end user and third parties to determine the trusted key
   state for the root key of the resolvers that handle that user's DNS
   queries. Note that this method is only applicable for determining
   which keys are in the trust store for the root key.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 7841.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   https://www.rfc-editor.org/info/rfc8509.

Copyright Notice

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 of a DNSKEY Resource Record (RR) as described in
   Appendix B of [RFC4034]. RRSIG RRs contain a Key Tag field whose
   value is equal to the Key Tag of the DNSKEY RR that was used to
   generate the corresponding signature.

   This document specifies how security-aware DNS resolvers that perform
   validation of their responses can respond to certain queries in a
   manner that allows an agent performing the queries to deduce whether
   a particular key for the root has been loaded into that resolver's
   trusted-key store. This document also describes a procedure where a
   collection of resolvers can be tested to determine whether at least
   one of these resolvers has loaded a given key into its trusted-key
   store. These tests can be used to determine whether a certain root
   zone Key Signing Key (KSK) is ready to be used as a trusted key,
   within the context of a planned root zone KSK roll.

   There are two primary use cases for this mechanism:

   o Users may wish to ascertain whether their DNS resolution
      environment's resolver is ready for an upcoming root KSK rollover.

   o Researchers want to perform Internet-wide studies about the
      proportion of users who will be negatively impacted by an upcoming
      root KSK rollover.

   The mechanism described in this document satisfies the requirements
   of both these use cases. This mechanism is OPTIONAL to implement and
   use. If implemented, this mechanism SHOULD be enabled by default to
   facilitate Internet-wide measurement. Configuration options MAY be
   provided to disable the mechanism for reasons of local policy.

   The KSK sentinel tests described in this document use a test
   comprising a set of DNS queries to domain names that have special
   values for the leftmost label. The test relies on recursive
   resolvers supporting a mechanism that recognizes this special name
   pattern in queries; under certain defined circumstances, it will
   return a DNS SERVFAIL response code (RCODE 2), mimicking the response
   code that is returned by security-aware resolvers when DNSSEC
   validation fails.

If a browser or operating system is configured with multiple
resolvers, and those resolvers have different properties (for
example, one performs DNSSEC validation and one does not), the
sentinel test described in this document can still be used. The
sentinel test makes a number of assumptions about DNS resolution
behavior that may not necessarily hold in all environments; if these
assumptions do not hold, then this test may produce indeterminate or
inconsistent results. This might occur, for example, if the stub
resolver is required to query the next recursive resolver in the
locally configured set upon receipt of a SERVFAIL response code. In
some cases where these assumptions do not hold, repeating the same
test query set may generate different results.

Note that the measurements facilitated by the mechanism described in
this document are different from those of [RFC8145]. RFC 8145 relies
on resolvers reporting towards the root servers a list of locally
cached trust anchors for the root zone. Those reports can be used to
infer how many resolvers may be impacted by a KSK roll but not what
the user impact of the KSK roll will be.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and
"OPTIONAL" in this document are to be interpreted as described in
BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all
capitals, as shown here.

This document contains a number of terms related to the DNS. The
current definitions of these terms can be found in [RFC7719].

2. Sentinel Mechanism in Resolvers

DNSSEC-validating resolvers that implement this mechanism MUST
perform validation of responses in accordance with the DNSSEC
response validation specification [RFC4035].

This sentinel mechanism makes use of two special labels:

o root-key-sentinel-is-ta-<key-tag>

o root-key-sentinel-not-ta-<key-tag>

These labels trigger special processing in the validating DNS
resolver when responses from authoritative servers are received.
Labels containing "root-key-sentinel-is-ta-<key-tag>" are used to
answer the question, "Is this the Key Tag of a key that the
validating DNS resolver is currently trusting as a trust anchor?"

Labels containing "root-key-sentinel-not-ta-<key-tag>" are used to
answer the question, "Is this the Key Tag of a key that the
validating DNS resolver is *not* currently trusting as a trust
anchor?"

The special labels defined here were chosen after extensive IETF
evaluation of alternative patterns and approaches in light of the
desired behavior (Sections 2.1 and 2.2) within the resolver and the
applied testing methodology (Section 4.3). As one example,
underscore-prefixed names were rejected because some browsers and
operating systems would not fetch them because they are domain names
but not valid hostnames (see [RFC7719] for these definitions).
Consideration was given to local collisions and the reservation of
leftmost labels of a domain name, as well as the impact upon zone
operators who might desire to use a similarly constructed hostname
for a purpose other than those documented here. Therefore, it is
important to note that the reservation of the labels in this manner
is definitely not considered "best practice".

2.1. Preconditions

All of the following conditions must be met to trigger special
processing inside resolver code:

o The DNS response is DNSSEC validated.

o The result of validation is "Secure".

o The Extension Mechanisms for DNS (EDNS(0)) Checking Disabled (CD)
   bit in the query is not set.

o The QTYPE is either A or AAAA (Query Type value 1 or 28).

o The OPCODE is QUERY.

o The leftmost label of the original QNAME (the name sent in the
   Question Section in the original query) is either "root-key-
   sentinel-is-ta-<key-tag>" or "root-key-sentinel-not-ta-<key-tag>".

If any one of the preconditions is not met, the resolver MUST NOT
alter the DNS response based on the mechanism in this document.

Note that the <key-tag> is specified in the DNS label as an unsigned
decimal integer (as described in [RFC4034], Section 5.3) but is zero-
padded to five digits (for example, a Key Tag value of 42 would be
represented in the label as 00042). The precise specification of the
special labels above should be followed exactly. For example, a
label that does not include a Key Tag zero-padded to five digits does

not match this specification and should not be processed as if it did
-- in other words, such queries should be handled as any other label
and not according to Section 2.2.

2.2. Special Processing

Responses that fulfill all of the preconditions in Section 2.1
require special processing, depending on the leftmost label in the
QNAME.

First, the resolver determines if the numerical value of <key-tag> is
equal to any of the Key Tag values of an active root zone KSK that is
currently trusted by the local resolver and stored in its store of
trusted keys. An active root zone KSK is one that could currently be
used for validation (that is, a key that is not in either the AddPend
or Revoked state, as described in [RFC5011]).

Second, the resolver alters the response being sent to the original
query based on both the leftmost label and the presence of a key with
given Key Tag in the trust-anchor store. Two labels and two possible
states of the corresponding key generate four possible combinations,
summarized in the table:

Label	Key is trusted	Key is not trusted
is-ta	return original answer	return SERVFAIL
not-ta	return SERVFAIL	return original answer

The instruction "return SERVFAIL" means that the resolver MUST set
RCODE=SERVFAIL (value 2) and the Answer Section of the DNS response
MUST be empty, ignoring all other documents that specify the content
of the Answer Section.

The instruction "return original answer" means that the resolver MUST
process the query without any further special processing, that is,
exactly as if the mechanism described in this document was not
implemented or was disabled. The answer for the A or AAAA query is
sent on to the client.

3. Sentinel Tests for a Single DNS Resolver

   This section describes the use of the sentinel detection mechanism
   against a single DNS recursive resolver in order to determine whether
   this resolver is using a particular trust anchor to validate DNSSEC-
   signed responses.

   Note that the test in this section applies to a single DNS resolver.
   The test described in Section 4 applies instead to a collection of
   DNS resolvers, as might be found in the DNS configuration of an end-
   user environment.

   The critical aspect of the DNS names used in this mechanism is that
   they contain the specified label for either the positive or negative
   test as the leftmost label in the query name.

   The sentinel detection procedure can test a DNS resolver using three
   queries:

   o A query name containing the leftmost label "root-key-sentinel-is-
      ta-<key-tag>". This corresponds to a validly signed name in the
      parent zone, so that responses associated with this query name can
      be authenticated by a DNSSEC-validating resolver. Any validly
      signed DNS zone can be used as the parent zone for this test.

   o A query name containing the leftmost label "root-key-sentinel-not-
      ta-<key-tag>". This also corresponds to a validly signed name.
      Any validly signed DNS zone can be used as the parent zone for
      this test.

   o A query name that is signed with a DNSSEC signature that cannot be
      validated (described as a "bogus" RRset in Section 5 of [RFC4033]
      when, for example, an RRset is associated with a zone that is not
      signed with a valid RRSIG record).

   The responses received from queries to resolve each of these query
   names can be evaluated to infer a trust key state of the DNS
   resolver.

   An essential assumption here is that this technique relies on
   security-aware (DNSSEC-validating) resolvers responding with a
   SERVFAIL response code to queries where DNSSEC checking is requested
   and the response cannot be validated. Note that other issues can
   also cause a resolver to return SERVFAIL responses, and so the
   sentinel processing may sometimes result in incorrect or
   indeterminate conclusions.

To describe this process of classification, DNS resolvers are
classified by five distinct behavior types using the labels: "Vnew",
"Vold", "Vind", "nonV", and "other". These labels correspond to
resolver-system behavior types as follows:

Vnew: A DNS resolver that is configured to implement this mechanism
   and has loaded the nominated key into its local trusted-key stores
   will respond with an A or AAAA RRset response for the associated
   "root-key-sentinel-is-ta" queries, SERVFAIL for "root-key-
   sentinel-not-ta" queries, and SERVFAIL for the signed name queries
   that return "bogus" validation status.

Vold: A DNS resolver that is configured to implement this mechanism
   and has not loaded the nominated key into its local trusted-key
   stores will respond with a SERVFAIL for the associated "root-key-
   sentinel-is-ta" queries, an A or AAAA RRset response for "root-
   key-sentinel-not-ta" queries, and SERVFAIL for the signed name
   queries that return "bogus" validation status.

Vind: A DNS resolver that is not configured to implement this
   mechanism will respond with an A or AAAA RRset response for "root-
   key-sentinel-is-ta", an A or AAAA RRset response for "root-key-
   sentinel-not-ta", and SERVFAIL for the name that returns "bogus"
   validation status. This set of responses does not give any
   information about the trust anchors used by this resolver.

nonV: A non-security-aware DNS resolver will respond with an A or
   AAAA RRset response for "root-key-sentinel-is-ta", an A or AAAA
   RRset response for "root-key-sentinel-not-ta" and an A or AAAA
   RRset response for the name that returns "bogus" validation
   status.

other: There is the potential to admit other combinations of
   responses to these three queries. While this may appear self-
   contradictory, there are cases where such an outcome is possible.
   For example, in DNS resolver farms, what appears to be a single
   DNS resolver that responds to queries passed to a single IP
   address is in fact constructed as a collection of slave resolvers,
   and the query is passed to one of these internal resolver engines.
   If these individual slave resolvers in the farm do not behave
   identically, then other sets of results can be expected from these
   three queries. In such a case, no determination about the
   capabilities of this DNS resolver farm can be made.

Note that SERVFAIL might be cached according to Section 7 of
[RFC2308] for up to 5 minutes and a positive answer for up to its
TTL.

If a client directs these three queries to a single resolver, the
responses should allow the client to determine the capability of the
resolver and, if it supports this sentinel mechanism, whether or not
it has a particular key in its trust-anchor store, as in the
following table:

```
                                  Query
                 +----------+----------+------------+
                 |  is-ta   |  not-ta  |   bogus    |
          +------+----------+----------+------------+
          | Vnew |    Y     | SERVFAIL |  SERVFAIL  |
          | Vold | SERVFAIL |    Y     |  SERVFAIL  |
   Type   | Vind |    Y     |    Y     |  SERVFAIL  |
          | nonV |    Y     |    Y     |     Y      |
          | other|    *     |    *     |     *      |
          +------+----------+----------+------------+
```

In this table, the "Y" response denotes an A or AAAA RRset response
(depending on the query type of A or AAAA records), "SERVFAIL"
denotes a DNS SERVFAIL response code (RCODE 2), and "*" denotes
either response.

Vnew: The nominated key is trusted by the resolver.

Vold: The nominated key is not yet trusted by the resolver.

Vind: There is no information about the trust anchors of the
       resolver.

nonV: The resolver does not perform DNSSEC validation.

other: The properties of the resolver cannot be analyzed by this
       protocol.

3.1. Forwarders

Some resolvers are configured not to answer queries using the
recursive algorithm first described in [RFC1034], Section 4.3.2 but
instead relay queries to one or more other resolvers. Resolvers
configured in this manner are referred to in this document as
"forwarders".

If the resolver is non-validating and has a single forwarder, then it
will presumably mirror the capabilities of the forwarder's target
resolver.

If the validating resolver has a forwarding configuration, and it
sets the EDNS(0) Checking Disabled (CD) bit as described in
Section 3.2.2 of [RFC4035] on all forwarded queries, then this
resolver is acting in a manner that is identical to a standalone
resolver.

A more complex case is where all of the following conditions hold:

o Both the validating resolver and the forwarder target resolver
   support this trusted key sentinel mechanism.

o The local resolver's queries do not have the EDNS(0) CD bit set.

o The trusted key state differs between the forwarding resolver and
   the forwarder's target resolver.

In such a case, either the outcome is indeterminate validating
("Vind") or there are mixed signals such as SERVFAIL in all three
responses ("other"), which is similarly an indeterminate response
with respect to the trusted key state.

4. Sentinel Tests for Multiple Resolvers

Section 3 describes a trust-anchor test that can be used in the
simple situation where the test queries are being passed to a single
recursive resolver that directly queries authoritative name servers.

However, the common end-user scenario is where a user's local DNS
resolution environment is configured to use more than one recursive
resolver. The single-resolver test technique will not function
reliably in such cases, as a SERVFAIL response from one resolver may
cause the local stub resolver to repeat the query against one of the
other configured resolvers, and the results may be inconclusive.

In describing a test procedure that can be used for a set of DNS
resolvers, there are some necessary changes to the nature of the
question that this test can answer, the assumptions about the
behavior of the DNS resolution environment, and some further
observations about potential variability in the test outcomes.

4.1. Test Scenario and Objective

   This test is not intended to expose which trust anchors are used by
   any single DNS resolver.

   The test scenario is explicitly restricted to that of the KSK
   environment where a current, active KSK (called "KSK-current") is to
   be replaced with a new KSK (called "KSK-new"). The test is designed
   to be run between when KSK-new is introduced into the root zone and
   when the root zone is signed with KSK-new.

   The objective of the test is to determine if the user will be
   negatively impacted by the KSK roll. A "negative impact" for the
   user is defined such that all the configured resolvers are security-
   aware resolvers that perform validation of DNSSEC-signed responses,
   and none of these resolvers have loaded KSK-new into their local
   trust-anchor set. In this situation, it is anticipated that once the
   KSK is rolled, the entire set of the user's resolvers will not be
   able to validate the contents of the root zone, and the user is
   likely to lose DNS service as a result of this inability to perform
   successful DNSSEC validation.

4.2. Test Assumptions

   There are a number of assumptions about the DNS environment used in
   this test. Where these assumptions do not hold, the results of the
   test will be indeterminate.

   o When a recursive resolver returns SERVFAIL to the user's stub
      resolver, the stub resolver will send the same query to the next
      resolver in the locally configured resolver set. It will continue
      to do this until it either gets a non-SERVFAIL response or runs
      out of resolvers to try.

   o When the user's stub resolver passes a query to a resolver in the
      configured resolver set, it will get a consistent answer over the
      time frame of the queries. This assumption implies that if the
      same query is asked by the same stub resolver multiple times in
      succession to the same recursive resolver, the recursive
      resolver's response will be the same for each of these queries.

   o All DNSSEC-validating resolvers have KSK-current in their local
      trust-anchor cache.

   There is no current published measurement data that indicates to what
   extent the first two assumptions listed here are valid or how many
   end users may be impacted by these assumptions. In particular, the
   first assumption, that a consistent SERVFAIL response will cause the

local stub DNS resolution environment to query all of its configured
recursive resolvers before concluding that the name cannot be
resolved, is a critical assumption for this test.

Note that additional precision/determinism may be achievable by
bypassing the normal OS behavior and explicitly testing using each
configured recursive resolver (e.g., using "dig").

4.3. Test Procedure

The sentinel detection process tests a DNS resolution environment
with three query names. Note that these are the same general
categories of query as in Section 3, but the Key Tag used is
different for some queries:

o A query name that is signed with a DNSSEC signature that cannot be
   validated (described as a "bogus" RRset in Section 5 of [RFC4033]
   when, for example, an RRset is not signed with a valid RRSIG
   record).

o A query name containing the leftmost label "root-key-sentinel-not-
   ta-<key-tag-of-KSK-current>". This name MUST be a validly signed
   name. Any validly signed DNS zone can be used for this test.

o A query name containing the leftmost label "root-key-sentinel-is-
   ta-<key-tag-of-KSK-new>". This name MUST be a validly signed
   name. Any validly signed DNS zone can be used for this test.

The responses received from queries to resolve each of these names
can be evaluated to infer a trust key state of the user's DNS
resolution environment.

The responses to these queries are described using a simplified
notation. Each query will result in either a SERVFAIL response
(denoted "S"), indicating that all of the resolvers in the recursive
resolver set returned the SERVFAIL response code, or a response with
the desired RRset value (denoted "A"). The queries are ordered by
the "invalid" name, the "root-key-sentinel-not-ta" label, then the
"root-key-sentinel-is-ta" label, and a triplet notation denotes a
particular response. For example, the triplet "(S S A)" denotes a
SERVFAIL response to the invalid query, a SERVFAIL response to the
"root-key-sentinel-not-ta" query, and an RRset response to the "root-
key-sentinel-is-ta" query.

The set of all possible responses to these three queries are:

(A * *): If any resolver returns an "A" response for the query for
   the invalid name, then the resolver set contains at least one
   non-validating DNS resolver, and the user will not be impacted by
   the KSK roll.

(S A *): If any of the resolvers returns an "A" response for the
   "root-key-sentinel-not-ta" query, then at least one of the
   resolvers does not recognize the sentinel mechanism, and the
   behavior of the collection of resolvers during the KSK roll cannot
   be reliably determined.

(S S A): This case implies that all of the resolvers in the set
   perform DNSSEC validation, all of the resolvers are aware of the
   sentinel mechanism, and at least one resolver has loaded KSK-new
   as a local trust anchor. The user will not be impacted by the KSK
   roll.

(S S S): This case implies that all of the resolvers in the set
   perform DNSSEC validation, all of the resolvers are aware of the
   sentinel mechanism, and none of the resolvers has loaded KSK-new
   as a local trust anchor. The user will be negatively impacted by
   the KSK roll.

5. Security Considerations

   This document describes a mechanism for allowing users to determine
   the trust-anchor state of root zone key signing keys in the DNS
   resolution system that they use. If the user executes third-party
   code, then this information may also be available to the third party.

   The mechanism does not require resolvers to set otherwise-
   unauthenticated responses to be marked as authenticated and does not
   alter the security properties of DNSSEC with respect to the
   interpretation of the authenticity of responses that are so marked.

   The mechanism does not require any further significant processing of
   DNS responses, and queries of the form described in this document do
   not impose any additional load that could be exploited in an attack
   over the normal DNSSEC-validation processing load.

6. Privacy Considerations

   The mechanism in this document enables third parties (with either
   good or bad intentions) to learn something about the security
   configuration of recursive DNS resolvers. That is, someone who can
   cause an Internet user to make specific DNS queries (e.g., via web-
   based advertisements or JavaScript in web pages) can, under certain
   specific circumstances that include additional knowledge of the
   resolvers that are invoked by the user, determine which trust anchors
   are configured in these resolvers. Without this additional
   knowledge, the third party can infer the aggregate capabilities of
   the user's DNS resolution environment but cannot necessarily infer
   the trust configuration of any recursive name server.

7. IANA Considerations

   This document has no IANA actions.

8. References

8.1. Normative References

   [RFC1034] Mockapetris, P., "Domain names - concepts and facilities",
              STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987,
              <https://www.rfc-editor.org/info/rfc1034>.

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

   [RFC2308] Andrews, M., "Negative Caching of DNS Queries (DNS
              NCACHE)", RFC 2308, DOI 10.17487/RFC2308, March 1998,
              <https://www.rfc-editor.org/info/rfc2308>.

   [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,
              <https://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,
              <https://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,
              <https://www.rfc-editor.org/info/rfc4035>.

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

   [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
              2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
              May 2017, <https://www.rfc-editor.org/info/rfc8174>.

8.2. Informative References

   [RFC7719] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS
              Terminology", RFC 7719, DOI 10.17487/RFC7719, December
              2015, <https://www.rfc-editor.org/info/rfc7719>.

   [RFC8145] Wessels, D., Kumari, W., and P. Hoffman, "Signaling Trust
              Anchor Knowledge in DNS Security Extensions (DNSSEC)",
              RFC 8145, DOI 10.17487/RFC8145, April 2017,
              <https://www.rfc-editor.org/info/rfc8145>.

Appendix A. Protocol Walk-Through Example

   This appendix provides a non-normative example of how the sentinel
   mechanism could be used and what each participant does. It is
   provided in a conversational tone to be easier to follow. The
   examples here all assume that each person has just one resolver or a
   system of resolvers that have the same properties.

   Alice is in charge of the DNS root KSK (Key Signing Key) and would
   like to roll/replace the key with a new one. She publishes the new
   KSK but would like to be able to predict/measure what the impact will
   be before removing/revoking the old key. The current KSK has a Key
   Tag of 11112; the new KSK has a Key Tag of 02323. Users want to
   verify that their resolver will not break after Alice rolls the root
   KSK (that is, starts signing with just the KSK whose Key Tag is
   02323).

   Bob, Charlie, Dave, and Ed are all users. They use the DNS recursive
   resolvers supplied by their ISPs. They would like to confirm that
   their ISPs have picked up the new KSK. Bob's ISP does not perform
   validation. Charlie's ISP does validate, but the resolvers have not
   yet been upgraded to support this mechanism. Dave and Ed's resolvers
   have been upgraded to support this mechanism; Dave's resolver has the
   new KSK, but Ed's resolver hasn't managed to install the 02323 KSK in
   its trust store yet.

   Geoff is a researcher. He would like to both provide a means for
   Bob, Charlie, Dave, and Ed to perform tests and himself be able to
   perform Internet-wide measurements of what the impact will be (and
   report this back to Alice).

   Geoff sets an authoritative DNS server for example.com and also a web
   server (www.example.com). He adds three address records to
   example.com:

      bogus.example.com. IN AAAA 2001:db8::1

      root-key-sentinel-is-ta-02323.example.com. IN AAAA 2001:db8::1

      root-key-sentinel-not-ta-11112.example.com. IN AAAA 2001:db8::1

   Note that the use of "example.com" names and the addresses here are
   examples, and "bogus" intentionally has invalid DNSSEC signatures.
   In a real deployment, the domain names need to be under the control
   of the researcher, and the addresses must be real, reachable
   addresses.

Geoff then DNSSEC signs the example.com zone and intentionally makes
the bogus.example.com record have bogus validation status (for
example, by editing the signed zone and entering garbage for the
signature). Geoff also configures his web server to listen on
2001:db8::1 and serve a resource (for example, a 1x1 GIF, 1x1.gif)
for all of these names. The web server also serves a web page
(www.example.com) that contains links to these three resources
(http://bogus.example.com/1x1.gif, http://root-key-sentinel-is-ta-
02323.example.com/1x1.gif, and http://root-key-sentinel-not-ta-
11112.example.com/1x1.gif).

Geoff then asks Bob, Charlie, Dave, and Ed to browse to
www.example.com. Using the methods described in this document, the
users can figure out what their fate will be when the 11112 KSK is
removed.

Bob is not using a validating resolver. This means that he will be
able to resolve bogus.example.com (and fetch the 1x1 GIF); this tells
him that the KSK roll does not affect him, and so he will be OK.

Charlie's resolvers are validating, but they have not been upgraded
to support the KSK sentinel mechanism. Charlie will not be able to
fetch the http://bogus.example.com/1x1.gif resource (the
bogus.example.com record is bogus, and none of his resolvers will
resolve it). He is able to fetch both of the other resources; from
this, he knows (see the logic in the body of this document) that he
is using validating resolvers but that at least one of these
resolvers is not configured to perform sentinel processing. The KSK
sentinel method cannot provide him with a definitive answer to the
question of whether he will be impacted by the KSK roll.

Dave's resolvers implement the sentinel method and have picked up the
new KSK. For the same reason as Charlie, he cannot fetch the "bogus"
resource. His resolver resolves the root-key-sentinel-is-ta-
02323.example.com name normally (it contacts the example.com
authoritative servers, etc.); as it supports the sentinel mechanism,
just before Dave's recursive resolver sends the reply to Dave's stub,
it performs the KSK sentinel check. The QNAME starts with "root-key-
sentinel-is-ta-", and the recursive resolver does indeed have a key
with the Key Tag of 02323 in its root trust store. This means that
this part of the KSK sentinel check passes (it is true that Key Tag
02323 is in the trust-anchor store), and the recursive resolver
replies normally (with the answer provided by the authoritative
server). Dave's recursive resolver then resolves the root-key-
sentinel-not-ta-11112.example.com name. Once again, it performs the
normal resolution process, but because it implements KSK sentinel
(and the QNAME starts with "root-key-sentinel-not-ta-"), just before
sending the reply, it performs the KSK sentinel check. As it has the

key with key-tag 11112 in its trust-anchor store, the answer to "is
this *not* a trust anchor" is false, and so the recursive resolver
does not reply with the answer from the authoritative server.
Instead, it replies with a SERVFAIL (note that replying with SERVFAIL
instead of the original answer is the only mechanism that KSK
Sentinel uses). This means that Dave cannot fetch "bogus", he can
fetch "root-key-sentinel-is-ta-02323", but he cannot fetch "root-key-
sentinel-not-ta-11112". From this, Dave knows that he is behind a
collection of resolvers that all validate, all have the key with Key
Tag 11112 loaded, and at least one of these resolvers has loaded the
key with Key Tag 02323 into its local trust-anchor cache. Dave will
not be impacted by the KSK roll.

Just like Charlie and Dave, Ed cannot fetch the "bogus" record. This
tells him that his resolvers are validating. When his (sentinel-
aware) resolvers perform the KSK sentinel check for "root-key-
sentinel-is-ta-02323", none of them have loaded the new key with Key
Tag 02323 in their local trust-anchor store. This means the check
fails, and Ed's recursive resolver converts the (valid) answer into a
SERVFAIL error response. It performs the same check for root-key-
sentinel-not-ta-11112.example.com, and as all of Ed's resolvers both
perform DNSSEC validation and recognize the sentinel label, Ed will
be unable to fetch the "root-key-sentinel-not-ta-11112" resource.
This tells Ed that his resolvers have not installed the new KSK and
he will be negatively impacted by the KSK roll.

Geoff would like to do a large-scale test and provide the information
back to Alice. He uses some mechanism such as causing users to go to
a web page to cause a large number of users to attempt to resolve the
three resources, and he then analyzes the results of the tests to
determine what percentage of users will be affected by the KSK
rollover event.

This description is a simplified example. It is not anticipated that
Bob, Charlie, Dave, and Ed will actually look for the absence or
presence of web resources; instead, the web page that they load would
likely contain JavaScript (or similar) that displays the result of
the tests, sends the results to Geoff, or both. This sentinel
mechanism does not rely on the web: it can equally be used by trying
to resolve the names (for example, using the common "dig" command)
and checking which names result in a SERVFAIL.

Acknowledgements

Authors' Addresses

   Geoff Huston
   Email: gih@apnic.net
   URI: http://www.apnic.net

   Joao Silva Damas
   Email: joao@apnic.net
   URI: http://www.apnic.net

   Warren Kumari
   Email: warren@kumari.net