The ISP Column
A monthly column on things Internet

September 2015
Geoff Huston

Measuring the Root Zone KSK Keyroll

A little over five years ago the root zone of the Domain Name System (DNS) was signed using the DNSSEC name-signing framework. The approach used to sign the root zone is a conventional one, using two keys. The root zone has a "working key", the Zone-Signing Key (ZSK) which is used to sign the all the entries in the root zone with the exception of the DNSKEY entries, and a Key-Signing Key (KSK) used to sign the DNSKEY entries.

Operationally, the ZSK is used to sign each new instance of the root zone, and it is rolled each quarter. The KSK is only used once per quarter, and it is used to generate a signature across the ZSKs. The KSK was generated on the 26th June 2010. The current key management procedures as they relate to the KSK can be found at https://www.iana.org/dnssec.

To engender broad confidence in the trustworthiness of this key, ICANN, the KSK key manager, published a Practice Statement (https://www.iana.org/dnssec/icann-dps.txt) describing its procedures relating to the administration of the KSK. The practice statement includes the commitment:

> 6.5. Key signing key roll-over
>
> Each RZ KSK will be scheduled to be rolled over through a key ceremony as required, or after 5 years of operation.

The proposed approach to this key roll is relatively conventional, and follows that described in RFC_5011. The first step is for the current KSK to "sign over" the incoming new KSK, and then wait for a period of no less than 30 days to allow all resolvers to pick up the new KSK and add it to their local trust anchor set (this is the "holddown" period specified by RFC5011). This is achieved by adding the DNSKEY record for the new KSK to the root zone file, and have the current KSK generate a digital signature (an RRSIG record) across the root zone's DNSKEY collection (which comprises of the current KSK, the current ZSK and the new KSK). The next step is to replace the old KSK with the new KSK. This is achieved by dropping the outgoing KSK's DNSKEY record from the zone file, and signing the DNSKEY set (the current ZSK and the incoming KSK) with the incoming KSK. Then, after some suitable period the old KSK can be re-published, this time with the revoke bit set on the old KSK, and once more using two signatures during this announcement of revocation. Finally, the revoked old KSK can be dropped from the DNSKEY set and the keyroll process is complete.

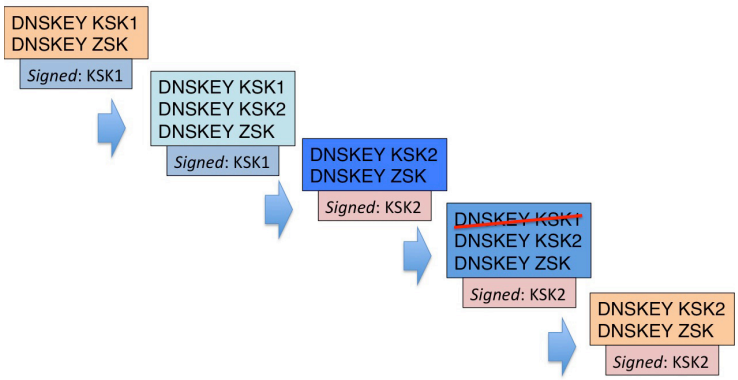

Figure 1 – The KSK Keyroll Process

One of the issues with this procedure is that the number of resolvers who are going to follow the implicit signalling of a new KSK is not necessarily known in advance. Some DNS resolvers may use a code base that does not have RFC 5011 support. Other DNS resolvers may have been configured to use a statically-defined trust anchor value, and by virtue of this configuration setting they will explicitly ignore any form of dynamic signalling of a new key. In both cases these resolvers will fail once the KSK is changed, and they will return SERVFAIL for all queries once the old KSK is withdrawn from the root zone. The number of resolvers that fall into either these categories cannot be measured in advance. The DNS is not a "chatty" protocol and the capability of resolvers to follow RFC5011 key roll procedures is not disclosed in queries to any servers.

All that can be said is that the roll of the KSK will cause some level of disruption to some population of users, but quantification of the number of resolvers that would be affected in this manner, and the population of users that rely on these resolvers is not information that can be clearly established in advance.

This has been a matter of some concern to ICANN's Security and Stability Advisory Committee (SSAC), who released a report on their findings of the risks associated with a KSK roll in their SAC063 report (https://www.icann.org/en/system/files/files/sac-063-en.pdf). In particular, the report recommended that:

> Recommendation 2: ICANN staff should lead, coordinate, or otherwise encourage the creation of a collaborative, representative testbed for the purpose of analyzing behaviors of various validating resolver implementations, their versions, and their network environments (e.g., middle boxes) that may affect or be affected by a root KSK rollover, such that potential problem areas can be identified, communicated, and addressed.
>
> Recommendation 3: ICANN staff should lead, coordinate, or otherwise encourage the creation of clear and objective metrics for acceptable levels of "breakage" resulting from a key rollover.
>
> Recommendation 4: ICANN staff should lead, coordinate, or otherwise encourage the development of rollback procedures to be executed when a rollover has affected operational stability beyond a reasonable boundary.

Is it possible to set up a test bed to test the steps involved in a KSK roll?

I'm aware of two efforts to support such a test bed - there may be more. Rick Lamb has a testbed at http://icksk.dnssek.info/fauxroot.html. Warren Kummari has also set up a testbed at http://keyroll.systems. The shortcoming in both of these testbeds is that it is not directly feasible to use these experimental scenarios to estimate the extent of "damage" that may be anticipated in a real KSK roll.

The problem in a nutshell is that those folk who are motivated to direct their resolver to either of both of these test beds are also probably motivated to use some form of key management to track the key roll as it happens. So perhaps there is a slightly different way to formluate this testbed question that can help the process, particularly it is relates to SSAC's recommendations 2 and 3.

A Keyroll Sentinel?

The question is: Is it feasible to instrument the root zone during a KSK roll in such a way to allow the measurement of resolvers and users who are failing to pick up the new KSK before the old KSK is removed?
One approach is to use the concept of a "sentinel" entry in the root zone which is used at the point when the new KSK is introduced into the root zone. The basic keyroll process allows for a period where a new key is signed by the old key, but is not used to sign anything itself. What if the new key is used to directly sign a sentinel record (without the use of a ZSK at all)?

There are two areas of consideration in setting this up: one is the structure of the key signing framework so that resolvers that fail to pick up trust in the new KSK can be clearly identified by their behaviour, and secondly the nature of the name itself used as a sentinel so that behaviours can be clearly identified.

Testing an Incoming KSK - Who Signs What

In the current signed root zone (http://www.internic.net/domain/root.zone) almost all RRSIG entries use key id 1518 (as of the 27th September 2015), which is the current ZSK. There is one exception to this general use of the ZSK, namely the DNSKEY entries in the zone file, where the RRSIG entry has been generated using the current KSK.

If a new KSK were to be introduced, then in the first phase of a key roll this new key would appear in the root zone as an additional DNSKEY entry. There would still be only a single RRSIG record, showing that the current KSK, key id 19036 (as of the 27th September 2015), signed across the set of DNSKEY values.

If we were to add a measurement point then the entry corresponding the name would be signed by the new KSK and not signed by the ZSK. This could be an A record, a TXT record, or the DS records associated with a delegated name. Note that the NSEC records would continue to be signed by the ZSK, so it would only be the A, TXT or DS records of the measurement point that would be signed by the new KSK.

What would be Signed?

One option is to sign a single label:

```
test. 86400 IN TXT   "Signed by KSK2"
test. 86400 IN RRSIG TXT 8 20151007050000 20150927040000 666 . <value>
```

An issue here is that single label processing by resolvers often invokes the use of a local search list, which can confuse the intent of the measurement. Another issue relates to the way in which this label can be used for measurement. The specific problem here is that validation failure is not directly visible to a root server (the resource record value is retrieved by a validating resolver before it attempt to validate the response).

This second issue can be resolved to some extent by using an address Resource Record (RR) for the sentinel label rather than a text RR, and then measuring who would follow the validated DNS resolution outcome to the referenced address. However with a constant value label the issue of DNS caches can confuse the measurement.

The alternative is to use a delegation record:

```
test. 86400 IN NS    ns.xtest.
test. 86400 IN DS    57649 5 2 ( <value> )
test. 86400 IN DS    57649 5 2 ( <value> )
test. 86400 IN RRSIG DS 8 1 86400 20151007050000 20150927040000 666 . <value>
```

This approach allows a sentinel name to contain 2 or more labels, circumventing the typical resolve behaviours of local search list use. This approach also allows multiple delegations at the second level, allowing a number of discrete measurement activities to coexist. The approach also allows the experiment to use dynamically generated wildcard-like name variants that explicitly invoke DNSSEC validation queries for unique names as a means of bypassing local caching, for example.

Would this work?

Yes, and no.

Yes, in that a validating resolver would see these sentinel records as valid.

No, in that a resolver using the old KSK as a manually configured trust anchor would also see these sentinel records as valid.

What is going on here is that there is a validation link from the new KSK to the old KSK because the old KSK has signed over the new KSK (in the same way as the old KSK has signed over the ZSK), and in both cases the validation path relies on the old KSK being a locally configured trust anchor (Figure 2).

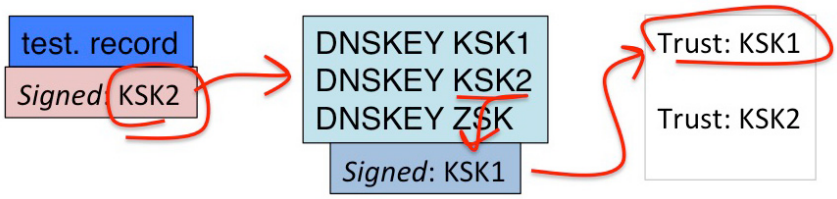

Figure 2 – Validation of a Sentinel Record

Even if the DNSKEY set was signed by both KSKs, then both classes of resolvers would still see the sentinel records as valid. The resolver with managed keys may use either trust anchor (Figure 3), while the resolver with the old KSK as a trust anchor would use the old KSK as its trust point for validation. The result is no discernable difference in resolver behavior between both classes of validating resolvers.

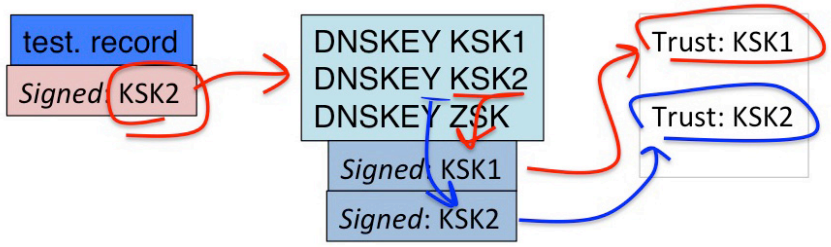

Figure 3 – Validation of a Sentinel Record, dual signed DNSKEY

What we are after is the construction of a condition that generates a different outcome if the resolver uses RFC5011-managed keys as compared to a resolver that uses a static trust configuration that only trusts the old KSK.

A Second Try

Section 2.2 of RFC5011 includes the directive:

> Once the resolver has accepted a key as a trust anchor, the key MUST be considered a valid trust anchor by that resolver until explicitly revoked as described above.

In other words, once a resolver using RFC5011-managed keys recognises a new key as "valid" then it will continue to recognise the key as valid, even when disappears from the DNSKEY RRset of the root zone.

Section 4.2 of RFC5011 describes this situation as:

> Missing: This is an abnormal state. The key remains a valid trust-point key, but was not seen at the resolver in the last validated DNSKEY RRSet. This is an abnormal state because the zone operator should be using the REVOKE bit prior to removal.

While RFC5011 describes this situation as "abnormal", it also notes that the key remains a valid trust point. This could imply that the sentinel records that are signed only by the new KSK will be validated by those resolvers that follow RFC5011, but not by those that do not follow RFC5011.

RFC5011 does not place any time limit on the time a key can stay in a "missing" state, and notes that its re-appearance in the zone would immediately push it to a "valid" state (no second holddown period is required).

If that's the case, then a potentially viable approach to construct a sentinel would be:

1. publish the new KSK for >= 30 days (the RFC5011 holddown period). This will cause a managed-key resolver to add the new KSK to its set of trust points.

2. re-publish the root zone, but in this new zone drop the publication of the new KSK in the DNSKEY RRset (i.e. set it to "missing") and test resolvers by asking them to resolve a sentinel record signed only by the new KSK.

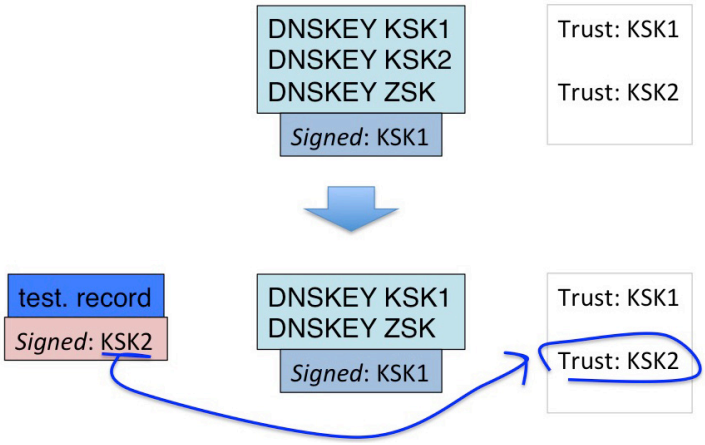

Figure 4 – Validation via a "Missing" Trust Point

The whole idea here is to remove the incoming key from the DNSKEY set in the root zone, but leave it as a "missing" trust point in those resolvers that can follow RFC5011 key roll process. Resolvers who have this as a loaded trust point should validate the signed sentinel, while resolvers with a static key will not accept the new key as a trust point, and once the new key is removed from the DNSKEY set in the root zone, they will be unable to validate any entry that is signed by the new key.

Would this work?

No.

The issue here lies in section 5 of RFC4035:

> An initial DNSKEY RR can be used to authenticate a zone's apex DNSKEY RRset. To authenticate an apex DNSKEY RRset by using an initial key, the resolver MUST:
>
> 1. verify that the initial DNSKEY RR appears in the apex DNSKEY RRset, and that the DNSKEY RR has the Zone Key Flag (DNSKEY RDATA bit 7) set; and
>
> 2. verify that there is some RRSIG RR that covers the apex DNSKEY RRset, and that the combination of the RRSIG RR and the initial DNSKEY RR authenticates the DNSKEY RRset. The process for using an RRSIG RR to authenticate an RRset is described in Section 5.3.

This means that for the new KSK to be used in validation of a DNS response it must be listed in the zone's DNSKEY RRset. But if that's the case then we are back to the first approach, as this use of a "missing" trust point relies on removing it from the DNSKEY RRset in the root zone.

Can Sentinel Records be used to inform the KSK Roll Process?

Unfortunately it looks like the answer is "no".

The basic precondition is that a key used in validation needs to be included in the DNSKEY RRset, and if it is included in the DNSKEY RRset then as long as the validator trusts the key that signed the DNSKEY RRset, then it trusts the keys described by the DNSKEY RRset.

The implication of this is that is not possible to tell in advance, when the old KSK is still the DNSKEY-signing key, the extent to which resolvers have chosen not to use the managed key framework.

Predicting the Fallout from a KSK Keyroll

This is not the best of outcomes. What it is saying is that there is no clear way to undertake a measurement activity that leads to a reasonable estimate of the number of DNS resolvers that use unmanaged trust keys for the root zone, nor can we reasonably estimate the population of users that would be affected by this planned key roll.

The issue here is that DSNSSEC validation all comes back to the contents of the DNSKEY RRset in the root zone. As long as this DNSKEY RRset is signed by a resolver's trusted key then the resolver is in a position to validate the DNS response.

One implication of this observation is that there is no benefit in dual-signing the DNSKEY RRset for an incoming KSK. As long as the outgoing KSK is not revoked, than as long as the outgoing KSK is part of the DNSKEY RRset then resolvers can use it. The transition from a single-signed DNSKEY RRset to a dual-signed RRset offer no functional change in DNSSEC validation, apart from generating a larger response when querying for the DNSKEYs of the root zone. The transition from a dual-signed DNSKEY RRset (outgoing KSK, incoming KSK) to a single signed RRset (incoming KSK) is precisely the same as a transition with a single signed DNSKEY RRset when swapping the outgoing KSK RRSIG with the incoming KSK RRSIG in a single change in the zone.

So if we can't measure the extent of anticipated impact of the KSK keyroll, is there any benefit in attempting to undo a KSK roll? Certainly once the key roll has been undertaken its possible to observer which resolvers, and how many users, are being affected by the roll. However the problem will not remain static for any extended period. Validating resolvers with out-of-date trust points will be unable to perform any DNS validation and would consistently return SERVFAIL for all queries to signed DNS names, and this would be a strong motivation to urgently install the new KSK or switch off DNSSEC validation, and do so rapidly as a means of restoring the resolver back into a functional state.

Perhaps the best we can do is acknowledge that no key is eternal, and keyroll is just a part of a secure framework, and strongly discourage the use of manually configured keys in validating resolvers.

But also we should consider ways to improve the degree of visibility of key capabilities of resolvers. One approach is to change resolver behaviour by have the resolver disclose its trusted keys to authoritative servers (The EDNS Key Tag Option, https://tools.ietf.org/html/draft-wessels-edns-key-tag-00). Of course in the world of security the disclosure of information is always a risk, and, as this working document points out:

> 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 edns-key-tag option are advised that provided Key Tag values
might be "made up" by some DNS clients with malicious or at least mischievous intentions.
```

More adventurously, we could re-open the specification of DNSSEC and allow a digital signature to be more explicit about which trusted key must be used by a resolver when validating the RRSIG value. Of course heading down that path is one that has downsides in fragility, and imposes an additional maintenance overhead of tracking key rolls in those zones that elect to use such a trust point "pinning" option.

Nothing is a perfect solution here, but perhaps as long as we instrument the key roll as well as is permitted by the technology, and carefully review what we learn from this we should be able improve the situation for the inevitable next roll of the KSK.

Author

Geoff Huston B.Sc., M.Sc., is the Chief Scientist at APNIC, the Regional Internet Registry serving the Asia Pacific region. He has been closely involved with the development of the Internet for many years, particularly within Australia, where he was responsible for building the Internet within the Australian academic and research sector in the early 1990's. He is author of a number of Internet-related books, and was a member of the Internet Architecture Board from 1999 until 2005, and served on the Board of Trustees of the Internet Society from 1992 until 2001. He has worked as a an Internet researcher, as a ISP systems architect and a network operator at various times.

www.potaroo.net

Disclaimer