**The ISP Column**
*A column on things Internet*

Geoff Huston
October 2017

# Not Rolling the KSK

For some years now (and it has definitely been two years, probably three and maybe longer) we have been working on a process of changing the cryptographic key that signs the Root Zone of the DNS. I wrote about this back in March 2016 (http://www.potaroo.net/ispcol/2016-03/rolling.html), describing both the role of this Root Zone Key-Signing Key, and what was at the time the envisaged process that was to be used to roll the key.

If you are at all interested, it might be useful to refresh yourself by glancing through that article, as in this article I will be working from that base to update you on the recent developments, and look at our options from here.

The timetable adopted by ICANN to roll the KSK was not quite according to the original timetable envisaged by the KSK Roll Design team. The steps were the same, but the overall timeline was deferred by a few months. The new KSK, called KSK-2017, was duly generated earlier this year, and stored in both KSK repository sites. On the 11th July KSK-2017 was added to the root zone's DNSKEY record set, duly signed by the old key, KSK-2010. This was intended to allow DNSSEC-validating resolvers a generously large period of time to "learn" the existence of the new key and add it to their locally trusted key set. The minimum introduction period for automated key management, as defined by RFC5011, was 30 days, so by mid-August it was anticipated that those DNSSEC-validating resolvers that used automated key management would've learned the new KSK and added it to their trust set. The intention was to introduce a changed DNSKEY resource record to the root zone on 11 October, where the key used to sign the DNSKEY key set would change from KSK-2010 to KSK-2017. At this point the key would've rolled and we would be then in a period of watching resolvers switch over as KSK-2010 aged out of the local caches, and they would roll over to use KSK-2017 to validate DNSSEC-signed responses.

However, earlier this week, ICANN announced the postponement of this key roll (https://www.icann.org/news/announcement-2017-09-27-en).

## What's going on?

At the recent meeting of the DNS Operations, Analysis, and Research Center (DNS-OARC), Duane Wessels of Verisign, presented on "A Look at RFC 8145 Trust Anchor Signaling for the 2017 KSK Rollover" (https://indico.dns-oarc.net/event/27/session/1/contribution/11/material/slides/0.pdf), and this material, and similar material that has been gathered from other root server operators, was likely the reason why the responsible and safe decision at this point in time is to defer the key roll, allowing time to pause and reflect on this data.

Let's take a step back for a second to look at the issues involved in the KSK roll. This roll is particularly challenging as there is no 'parent' state that guides the imputed trust that needs to be invested into the new key value. The Root KSK simple has no parent key. So the key roll relies only on the old KSK signing over the new KSK, combined with the implicit trust that clients pick up on this and place trust in the new KSK. The areas of uncertainty where we were unable to contain the risk factors in this key roll concern the ability of resolvers to trust the introduction of KSK-2017. There are two primary factors for a DNSSEC-validating resolver which impact its ability to track the introduction of KSK-2017 as a trusted key:

1. Whether a DNSSEC-validating resolver can successfully receive large DNS responses (the current response to a DNSSEC-enabled query to the Root Servers for a DNSKEY query will be 1,414 bytes in size.)

```
$ dig . DNSKEY +dnssec

; <<>> DiG 9.11.2 <<>> . DNSKEY +dnssec
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 35626
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 512
;; QUESTION SECTION:
;.                IN    DNSKEY

;; ANSWER SECTION:
.           117970    IN    DNSKEY    256                   3                       8
AwEAAYvxrQOOujKdZz+37P+oL4l7e35/0diH/mZITGjlp4f81ZGQK42H
NxSfkiSahinPR3t0YQhjC393NX4TorSiTJy76TBWddNOkC/IaGqcb4er
U+nQ75k2Lf0oIpA7qTCk3UkzYBqhKDHHAr2UditE7uFLDcoX4nBLCoaH
5FtfxhUqyTlRu0RBXAEuKO+rORTFP0XgA5vlzVmXtwCkb9G8GknHuO1j
VAwu3syPRVHErIbaXs1+jahvWWL+Do4wd+lA+TL3+pUk+zKTD2ncq7Zb
JBZddo9T7PZjvntWJUzIHIMWZRFAjpi+V7pgh0o1KYXZgDUbiA1s9oLA L1KLSdmoIYM=
.           117970    IN    DNSKEY    256                   3                       8
AwEAAcRIZfxskdElMKgjwvWQO2bQe7EGAvX6zgIaqmbsaMqmMrIpd1+b
P7nyULLuL8jWnKAqcaVfal2yJD50gg5zFl5yW/F9dKNXXEFI7VEcGrPy
G6/OrA9RBU8pGWm0qxpsNm5UIgTU5IX7pb/0rBj67c/R7qln8sjH1yls
r4f1Y3R6p/druiEalKasEjGKA9L2w9jzUQusWxM7fQx/T8c/3x3bsjve
D1dleQ6MJaCx4bpPXYZpqXmSvGn+T2v5350cBVAFqVKhGbjxEyXAweem
8cTU4L1p+DV7Ua11a1tMf0Tlu8pkpLwh7NQIggIEhJwEhPeXE3E4C6Q2 /PFENcoFERc=
.           117970    IN    DNSKEY    257                   3                       8
AwEAAagAIKlVZrpC6Ia7gEzahOR+9W29euxhJhVVLOyQbSEW0O8gcCjF
FVQUTf6v58fLjwBd0YI0EzrAcQqBGCzh/RStIoO8g0NfnfL2MTJRkxoX
bfDaUeVPQuYEhg37NZWAJQ9VnMVDxP/VHL496M/QZxkjf5/Efucp2gaD
X6RS6CXpoY68LsvPVjR0ZSwzz1apAzvN9dlzEheX7ICJBBtuA6G3LQpz
W5hOA2hzCTMjJPJ8LbqF6dsV6DoBQzgul0sGIcGOYl7OyQdXfZ57relS
Qageu+ipAdTTJ25AsRTAoub8ONGcLmqrAmRLKBP1dfwhYB4N7knNnulq QxA+Uk1ihz0=
.           117970    IN    DNSKEY    257                   3                       8
AwEAAaz/tAm8yTn4Mfeh5eyI96WSVexTBAvkMgJzkKTOiW1vkIbzxeF3
+/4RgWOq7HrxRixHlFlExOLAJr5emLvN7SWXgnLh4+B5xQlNVz8Og8kv
ArMtNROxVQuCaSnIDdD5LKyWbRd2n9WGe2R8PzgCmr3EgVLrjyBxWezF
0jLHwVN8efS3rCj/EWgvIWgb9tarpVUDK/b58Da+sqqls3eNbuv7pr+e
oZG+SrDK6nWeL3c6H5Apxz7LjVc1uTIdsIXxuOLYA4/ilBmSVIzuDWfd
RUfhHdY6+cn8HFRm+2hM8AnXGXws9555KrUB5qihylGa8subX2Nn6UwN R1AkUTV74bU=
.           117970    IN    RRSIG    DNSKEY 8 0 172800 20171010000000 20170919000000
19036           .           G1B0YY5YGCRtT3HuZhR6/ivgiiZ5uBSkPri6Mrhz6lZtJeQMeIPiIlAO
+Y8jEkurNYPL4Gk1kaprSKBbKnB3joIeGHGBBRiKYgS0cQk/NWuEX9Jf
LtW0RwZhrXTN7JsH15/WEjFQkH0LnR+R3WUFH8uHR4kxLFKztKDSZoNf
+PR7pa8PK98YcjSW7rZcTV70V3daSwQTeJIpXpUhVUGXXju9WN0cRVVY
Ck7sRteUqKqJQxLBAlzYQX2CgPhZOTypqJxzj12e9Y/9WPGkBLqfxHms
0c/Om+NO5WhNNONLdoXX8Yw4okFCpodGUO/UMrgM4qm7SWxXkjZwedzD ZFJpYA==

;; Query time: 12 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Sat Sep 30 12:15:28 UTC 2017
;; MSG SIZE  rcvd: 1414
```

2. Whether a DNSSEC-validating resolver is configured to follow the automated key roll process or not.

   If the resolver is using manually managed trusted keys then it's up to the local system's administrator as to when KSK-2017 is introduced as a trusted key before the time for the KSK roll. If the resolver is using the automated process then then there is the question of whether the resolver is able to correctly follow the roll and trust KSK-2017 before the time when KSK-2010 is removed from the root trust point of the root zone (i.e. when the RRSIG generated by KSK-2010 is removed from the RRSIG RRset of the root zone entry for the DNSKEY record).

The major risk is failure to load the new KSK as a trusted key, and when the design team undertook its work in 2015 it was simply impossible to tell how many resolvers would still be using the old KSK at the time of the KSK roll.

With respect to the first risk factor, that of the use of large responses from root servers, it was possible to undertake experiments to ascertain how well DNSSEC-validating resolvers could cope with the 1,414 byte response, and the experiments did not generate any particular concerns. Within the limits of accuracy of the experimental technique it appeared that the risk factor was essentially negligible, and coupled with the observation that the DNSSEC-signed response from a DNSKEY query for .org was already 1,625 bytes in length and evidently there have been no evident complaints about unreachability of .org names, an observation that appears to validate this experimental result.

With respect to the second risk factor, then this behaviour is simply not possible to quantify using experimental approaches. There is no known way to insert a 'sentinel' into the existing root zone where the trust key state of the resolver can be discerned without changing some aspect of either resolver behaviour, DNSSEC validation behaviour or both. So in this respect we were flying blind. The KSK Design Team report (https://www.iana.org/reports/2016/root-ksk-rollover-design-20160307.pdf) proposed intensive monitoring at the time of the key roll, and if the failure rate in the 72 hours following the roll appeared to affect more than 0.5% of users, then the rollback procedures should be implemented and the old key added back to the root zone.

## Signaling Trust Anchor Information in the DNS

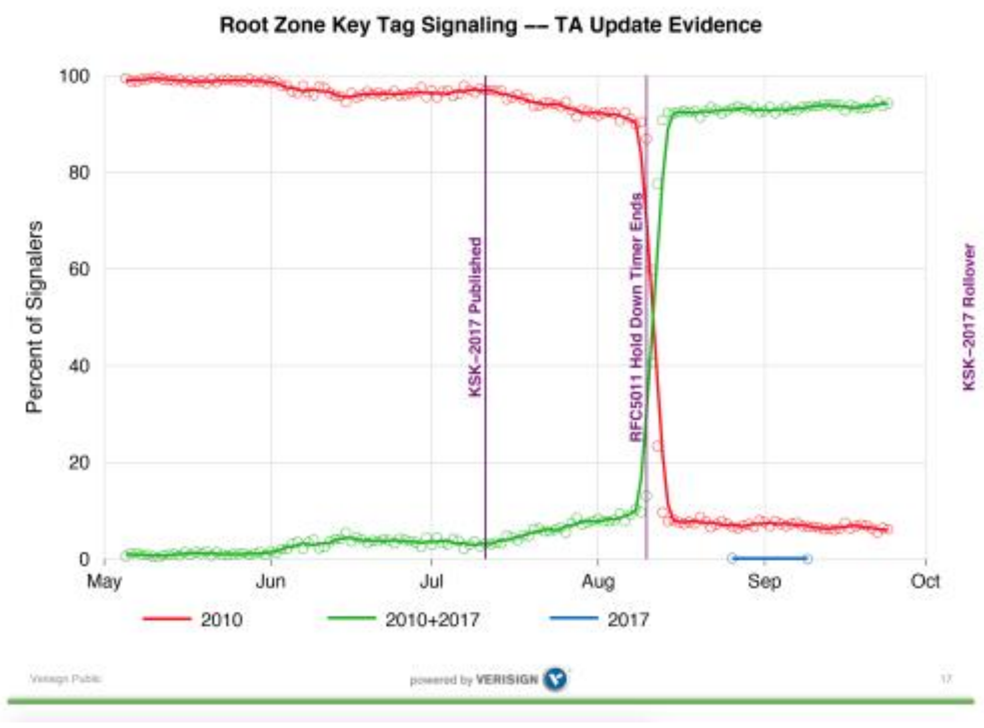
At the time, it was considered to be the best that could be done here. However, in April 2017 the IETF published RFC8145. This document describes a method that enables DNSSEC-validating resolvers to report the trust key(s) that they use as an entry point for a domain to the authoritative servers for that domain. The document describes two independent methods for conveying this Key Tag information. The first is by placing an EDNS option in the OPT RR (RFC6891) that contains the Key Tags, and the second is by periodically sending special "Key Tag queries" to a server authoritative for the zone.

Why two approaches?

Well, as the RFC points out, in the situation where a validating client forwards all its queries to another resolver, the EDNS(0) options are not transitive and would not, by default, be passed on in any case. In contrast, one would expect the QNAME queries to be passed on. But perhaps this is not quite the case either. Querying a root server for a name such as `_ta-19036.` will generate an NXDOMAIN response today, and if querier is sitting behind a validating resolver that performs some form of NSEC caching (RFC8192) or local root zone (RFC7706), the queries will not make it to the root servers in any case. It appears that neither approach gives any assurance that all resolvers' trust anchor signals will be seen by a root server, so this form of signalling falls into the category of signals that contain some implicit level of uncertainty.

While this RFC is relatively recent, it has been implemented by a number of DNS resolver vendors in their code. In recent versions of the Bind DNS resolver the configuration parameter "`trust-anchor-telemetry`" is set be default to "yes". In the Unbound resolver, the "`trust-anchor-signaling`" parameter defaults to "`no`", and must be changed to "yes" to enable this reporting.

This gets me back to Duane Wessel's presentation at DNS OARC. It is useful to look at Duane's entire presentation, but I will reproduce here one slide which perhaps is most illustrative of the issue which has led to the postponement of the key roll.



*Trust Anchor Update Evidence – Duane Wessels, DNS OARC 27*

The first observation is that the 30 day 'hold down' timer described in RFC5011 appears to have largely worked as expected. Some 30 days after the introduction of KSK-2017 into the root zone, more than 90% of reporting IP addresses reported that they now had added KSK-2017 to their local trusted key set. A smaller number of reporters had loaded KSK-2017 earlier than the 30 day hold down period, indicating some local configuration, possibly relating to manual key management.

The salient question, however, is in the set of resolvers reporting that they still only trust KSK-2010. The numbers presented at OARC suggest that somewhere between 5% to 10% of reporting IP addresses have not shifted to trust KSK-2017, and in the 40 days since the hold-domain timer expiration this number does not appear to be getting any smaller.

Subsequent discussion at the OARC meeting indicated that this relative level of reporting of continued reliance on KSK-2010 has been seen at other root servers. If this data is illustrative of the general level of takeup of KSK-2017, then the 5% - 10% level of non-readiness for KSK-2017 is a serious concern, even though the number of signalling entities is reported to be some 12,000 resolver IP addresses out of a total resolver pool of some millions of resolvers.

With time running out before the actual key roll, and with some serious concerns about the level of uptake of KSK-2017 from this reported data, then it is to be applauded that the conservative decision was taken to postpone the key roll. We need some further time to understand what is happening here and need to reassess the safety of this action before proceeding further, and if there is a strong signal of issues here with rolling trust anchor keys, then we need to take the time to understand this signal first.

So why are there a set of resolvers reporting the continued reliance on KSK-2017?

## Observations and Interpretations

Discussion in the OARC meeting has pointed to some issues with the implementation of this form of Trust Anchor signal.

The first issue is that there are several false signals intruding into the data set. It is possible, for example, for non-validating Bind resolvers to emit these trusted key state queries even when the resolver is not performing DNSSEC validation, and some further investigation of the collected data reveals that this is indeed the case some of the time. If a Bind resolver has trusted keys, but does not perform DNSSEC validation, it is not going to load and validate a new key. As it is not validating it should not be emitting these trust anchor queries, but under some circumstances this will happen in any case.

The second is the nature of the query signal. If a resolver uses a forwarder to pass its queries towards the root zone of the DNS then the trusted key queries will still be made, but they will be passed to through the forwarder, and it will appear to the root servers that the forwarding resolver is emitting these queries about its own trust anchor key state. The identity of the original resolver is lost in this case, and the resultant signal is prone to misinterpretation. In addition, this query and its response will be placed in the local cache. This means that there is a potential for multiple resolvers sitting behind a common forwarding resolver to have their trust anchor signals masked, or for the forwarding resolver to make what appears to be inconsistent signal queries to the root servers. In most cases this maksing is not an issue, and the cache lifetime for NXDOMAIN responses is short. However, in the case where the caching resolver is a DNSSEC-validating resolver, and if it is performing RFC8192-style NSEC caching, then the local cached entry of the non-existence of these 'special' domain names will be longer, and much of the original signal from these 'hidden' resolvers will be masked.

And of course there is the mismatch of the signal to the intent. If the original intent was to understand the potential population of affected users, then getting a signal from resolvers is not the same. The distribution of users to the resolvers that they use is heavily skewed, and less than 10,000 individual resolver IP addresses are seen to ask authoritative servers queries on behalf of more than 90% of the entire user population of the Internet. So when a very heavily used resolver, such as Google's Public DNS resolvers for example, reports that it has not loaded KSK-2017 this is a far more impactful observation than if my local DNS resolver that serves just me sends a similar signal. The inference here is that interpreting the signal provided by these trust anchor queries is difficult unless you can also add some element of 'weight' based on served user population data. However here the second issue, that of forwarders, intertwines. The highly used resolvers are far more likely to be forwarders for other resolvers, and if they pass on a `ta-19036.` query to a root server, that should not be interpreted as being a reliable signal that this highly used and very prominent DNS resolver has not loaded KSK-2017 into its trust set, nor even that the reporting resolver even performs DNSSEC validation itself.

The first two cases are these are instances of a 'false positive' signal, where a resolver is incorrectly reporting failure to trust KSK-2017 when in fact it is either not relevant (in the case of non-validating resolvers) or just not true (as the query is actually describing the key state of a hidden resolver), and the third is the problem that there is no clear signal as to its relative 'value' in interpretation.

There are also cases where a resolver will indeed fail to trust KSK-2017, so not all of these queries are in error. As noted already, if a manually managed configuration is waiting right up until October 11, and it is reporting its trust anchor state, then it will correctly report only trust in KSK-2010. On the other hand, there is also the potential for a resolver to be configured in a manner that it cannot follow RFC5011 even though it is configured to do so. Evidently certain configuration combinations require the trusted key state to be written to a file store before it is included into the local trust set. If the resolver is operating in an environment where the file store is not writable for any reason (and such reasons exist) then the resolver will be unable to pick up the new trust anchor even though its configuration directs it to do so.

## Who should be Signalling to Whom?

This is a difficult situation. We now have a signal about the use of trust anchors that is being generated by an extremely small fraction of all the DNSSEC-validating resolvers out there (and some non-validating resolvers, as it turns out). The lack of clarity of the signal means that while it is clear that caution is now advised, it is entirely unclear as to the conditions for this signal stream to reliably provide a 'green light' to proceed.

We need to think about this some more.

One thought here is that perhaps we've been too enthusiastic in trying to implement signalling and remote measurement without considering exactly what problem we are trying to address.

By going down this path of resolvers signalling their status, we have transferred the responsibility of ensuring that the key roll is 'safe' to someone else, namely ICANN in this case. But in transferring this responsibility to the managers of the root zone, we really have not given these people the necessary tools to accurately determine when a 'safe' state exists that could allow them to proceed.

It is extremely challenging to understand what circumstances would need to exist in order to allow the interpretation of this query stream to indicate that the roll is adequately 'safe'. Even if the nature of the signal was altered and we revised RFC8145 to add information or alter the behaviour of the signal, this form of measurement is starting from a poor position in the first place.

At the same time, we have disenfranchised the consumers of the DNS outcomes. Is there any way a user can test their resolution environment to see if it will function correctly across the key roll? In all this effort to make DNS secure have we built a system where, oddly enough, the only message that a consumer is given is "trust us!" Far be it for me to accuse the DNS of being dismissive of the very clients who rely on the DNS, but in some of these aspects of protocol design the end user of the service is often ignored and we concentrate instead on the inner machinery of the DNS and the inner relationships between the various components of the service infrastructure.

In so many ways this outcome, where the user is forced to trust the infrastructure without any ability to test that trust, seems to be profoundly unsatisfying. Clients of resolver services, be they end users or networks themselves, should be able to ascertain the properties of the DNSSEC trust environment. Sending a signal which is ultimately directed into the effectively opaque space of the root servers and having a reverse feedback of a vague report quoting some very obtuse summarised numbers whose interpretation is inevitably dubious seems at best to be extremely unsatisfying. There must be a better way of doing this!

Why isn't there a simple query mechanism that informs the queries as to the trusted key status of the DNSSEC validation being performed on their behalf? Why should this information be signalled further into the DNS abyss and why shouldn't we be signalling back to the consumer of this service?

As an example, of what I'm thinking here, we can do this with DNSSEC validation by setting up a badly signed name. If you can resolve this name then your resolvers are not all performing DNSSEC validation. For example, if you try to resolve the domain name www.dnssec-failed.org then as a consumer of a DNS resolution service you can see if your service actually implements DNSSEC validation or not (the name resolution service should fail to resolve this particular name if all the resolvers you use perform DNSSEC validation).

If we can do this for DNSSEC validation, then why can't we have a 'golden' top level domain that would allow users to test their resolvers for the trusted key state? Why can't we set up some test label that is only resolvable if you have loaded a certain key into your locally trusted set of keys?

In this respect the DNSSEC test is easy – it simply requires an incorrectly signed DNS name. The comparable model for trusted key sets is far more complex, and so far it appears that we cannot achieve a useful sentinel test for trusted keys without at the very least changing the behaviour of DNS resolvers, the algorithm used to perform DNSSEC validation, or perhaps even both. And even then, it's not clear we can achieve this. But the effort to design such a test could be worthwhile. If we could define such an approach, then rather than signalling into the root of the DNS, we would be reversing the direction of the signal and sending the capability signal back to the actual consumer of the DNS service.

Looking at this more generally, if we really have a choice of resolvers here, and we can choose to use DNS resolution services that we are prepared to trust, then we should make explicit aspects of the nature of that trust. Maybe, instead of asking the folk at ICANN to make decisions on our behalf and stand ready to castigate them whenever they make a call that we might disagree with, we should take some greater level of responsibility ourselves. Maybe it is a reasonable step to find more ways to empower the users and consumers of DNS services and allow them ready access to tools that can show them the capabilities of various available DNS resolution services. It always seems to me to be a better outcome if one can empower users instead of making them dependant on others.

## Not is Good Here

In closing, getting back to the KSK roll, I believe that the folk at ICANN have made the only responsible call that they could under the circumstances. Postponing the KSK roll was a safe and conservative decision in my view, and I applaud them for this.

But I wonder if the very nature of the trusted keys signal we are using is such that its interpretation will always be uncertain and I worry that we may not achieve a state where this signal feed will give a clear and unambiguous interpretation that a KSK roll is 'safe' to proceed. But the time to pause, consider and regroup should not be wasted time, and we can work to improve this picture and provide some further necessary clarity on the state of these DNSSEC-validating resolvers as we roll the trust keys.

## Author

*Geoff Huston* B.Sc., M.Sc., is the Chief Scientist at APNIC, the Regional Internet Registry serving the Asia Pacific region.

*www.potaroo.net*

## Disclaimer