draft-spaghetti-sidrops-rpki-ta-tiebreaker

SIDROPS, IETF 120, Vancouver

Job Snijders job@fasty.com
Theo Buehler tb@openbsd.org
Ties de Kock tdekock@ripe.net

Background

Relying Parties periodically fetch Trust Anchor certificates from well-known, remote locations and verify that the key of the self-signed certificate matches the key embedded in its associated Trust Anchor Locator.

This transfer may happen via an unauthenticated channel, and the certificate is verified by checking that it is signed by the public key in the TAL.

*After retrieving a TA certificate, Relying Parties have a **choice** between using a previously retrieved locally cached copy of the TA certificate and the newly-retrieved instance of the TA certificate.*

What if older TA issuances are accidentally re-introduced?

Fun fact: ARIN & AfriNIC TA certificates from 2015/2016 exist which still are valid!

Those ***almost 10 year old*** certificate issuances contain RFC 3779 listings that are of course woefully outdated. Re-introduction of these TA certificates would cause the RP to consider a large number of signed products to be invalid.

A bit of analysis on RIR Trust Anchor certificates:

https://mailarchive.ietf.org/arch/msg/sidrops/NxzvSFH0sPXEmyfOS99cLApFKqM/

Towards shorter lived TA certificates

- Previous slide shows that it might be nice if 10 year old stuff can’t bite us today
- And practically, TA operators critically depend on agility:
  - **Changes in RFC 3779 subordinate resource listings**
    - Recent example is RIRs switching over to 0.0.0.0/0
    - Perhaps one day they’ll switch back? (pending draft-ietf-sidrops-rpki-validation-update)
  - **Changes in Subject Information Access**
    - Add/drop URIs
    - Migrate to RRDP v2 when we get there
  - **Changes to the CPS URI**
    - TA operator might want to update the location of their Certification Practice Statement

Towards RPs protecting themselves against TA replay

Ideally an RP uses a still-valid locally cached version of the TA certificate if it encounters a HTTP or RSYNC error, or incorrect signature.

But if the TA certificate otherwise is valid, how can an RP decide which TA certificate issuance the TA operator intended for the RP to use?

There aren't that many fields! A quick analysis:

- Certificate serial numbers are random values, no monotonicity
- None of the X509v3 extensions provide useful hints: some are static values, some represent expression of the aforementioned agility
- But, *notBefore* and *notAfter* could be used as proxy values for "recency"
- The source of the object can also be taken into account for recency

Proposed TA certificate tiebreaking scheme for RPs

… With manifests this was easier: just prefer the one with the highest manifestNumber! …

For TA certificates the following has been implemented in OpenBSD rpki-client.

Assuming two objects have valid cryptographic signage & ‘now’ is within both their validity periods:

1. Prefer the TA certificate with the most recent *notBefore*
2. In the face of equal notBefore, prefer the TA cert with the soonest *notAfter*
3. In the face of equal validity periods, prefer the newly-fetched TA certificate

Justification for this particular tiebreaking scheme

1. Assuming no or minimal backdating, the *notBefore* ought to be close to the actual issuance moment in time, therefore it is the best indicator of recency. Second benefit: encourages TA operators to *not* recycle notBefore values.

2. Overly long validity periods are unreasonable: it is impossible to argue a given SIA or INR listing will still be correct 50 years from now. Therefore, RPs should prefer shorter validity periods.

3. A locally cached version of the TA cert will have come from a previous fetch, ergo, all previous tiebreakers consider equal, prefer the freshly-fetched version as the final step. The “network” as a source is assumed to be more recent than data read from disk.

The tiebreaking scheme steps are in order of progressive unlikeliness.

Next steps for draft-spaghetti-sidrops-rpki-ta-tiebreaker

- Implementers to assess the proposed approach
- Please review the document and provide feedback
- Discuss how exactly to integrate this concept in the existing body of RFCs
  - -bis the TAL RFC?
  - Minimally update the RFC 8630? (the current approach of the draft)
- Start call for working group adoption?
- Publish as implementation-specific independent submission?