Management of DNSSEC Key Signing Keys

Johan Ihren, Autonomica
johani@autonomica.se

or

How to Sign the DNS root

trust, authentication and distribution

Johan Ihren, Autonomica
johani@autonomica.se

The Basics

- This talk is (somewhat loosely) based upon these papers:

draft-ietf-dnsop-interim-signed-root-01.txt

draft-ihren-dnsext-threshold-validation-00.txt

- Please read (and comment) the drafts.

Protocol + Trust = DNSSEC

- So far we believe that we've managed to get the protocol right.
- Now it's time for the trust part. That requires new players.
  - and a bit of new thinking
  - the new players should have an established "trust base"
- The RIRs may play a crucial role here.

If, what consequences?

- The rest of this talk is an outline of the trust problem that needs to be solved.
- If the RIRs get involved then this
  - has impact on RIR resources
  - affects RIR membership
- The question here is "whether", not the technical details of exactly "how".

What is a "security apex"?

- DNSSEC is based upon the concept of a "chain of trust"
  - this chain is followed from the data that is being "verified" all the way to a "trusted key"
  - the "trusted key" is simply a key, configured in a "resolver" that should perform DNSSEC verification, that the resolver has reason to trust
  - a node in the DNS hierarchy that distributes trusted keys is called a "security apex"

Security apex, cont'd

- At a security apex, like the root, it is possible to have two types of keys, with entirely different uses
  - "operational keys", aka Zone Signing Keys, ZSKs
  - "authenticators", aka Key Signing Keys, KSKs
- The terminology is a bit lacking. Sorry.

Operational keys

- Used for signing the zone data.
- Part of the adminstrative process of maintaining the zone and its contents.

- These are well understood.

"Authenticators"

- Used to authenticate the operational keys. Only.
  - this is achieved by the "trusted key"
  - a trusted key is simply the public part of an Authenticator
  - the trusted key is distributed to and configured in resolvers
- Not used in any operational day-to-day activities.
- These may be less well understood.

The role of the Authenticator

- Authenticators assert the identity of the people that hold operational keys
  - i.e., in the case of the root, they may tell the world that:

    "these are indeed the real official root server operators, we've checked and you may trust us on this"

- The Authenticator function is similar to that of a public notary

The role of the Authenticator

- This is quite similar to how PGP works:
  - You sign someone else's PGP key to help others identify him since they trust you.
  - Signing a PGP key does not involve taking responsibility for what the key is used for (i.e. used to sign).

Proposal:

multiple Authenticators

- Previously the assumption has been that there is one Authenticator
  - controlled by the "zone owner", and
  - possibly multiple operational keys
- Proposal: increase trust in the operational keys by introducing multiple, independent, Authenticators.

Consequences

- With multiple Authenticators, issued by different entities, we get
  - a larger aggregated "trust base", since different "issuers of Authenticators" are trusted by different subsets of the resolver population
  - the possibility of more robust rollovers, since not all trusted keys will or should roll at the same time
  - the option of using local policy to express different security needs

Example #1

- One local policy may be:

> "I require a valid signature by at least one of the following five Authenticators"

- This would improve robustness during rollovers

Example #2

- Another local policy may be:

> "I require valid signatures by at least three of the following five Authenticators"

- This would improve security by drastically lessen the consequences of a (single) key compromise.

Distribution of keys

- A mechanism of distribution of trusted keys for root is needed.
  - it is possible to distribute "new keys" within the DNS protocol (i.e. for key rollovers, etc)
  - out-of-band distribution is also needed and with multiple KSK holders different such mechanisms can be explored
  - eventually it is likely that a major mechanism will be platform specific things like "Windows Update", but that will never by itself be sufficient

Building the "trust base"

- In the end this is all about Trust.
- If the verifying resolvers don't trust the authenticity of the operational keys this will not work
  - and the holders of operational keys cannot do this themselves, because they have no trust base (and that's not their role)

Building the "trust base"

- We need "issuers of Authenticators" that
  - already are trusted by some part of the "resolver population", i.e. have a "trust base"
  - are multiple entities that complement each other (so that the aggregated "trust base" grows)
  - are willing to help work on methods for distributing their trusted keys to the resolvers (hard problem)

Building the "trust base"

- Technical constraints severely limit the number of possible Authenticators for the root
  - not clear where the exact numbers end up (depends on several factors), but somewhere between 4 and 6 is likely
- Important to use the Authenticators wisely to gain a large trust base.

Why use RIRs?

- RIRs already have a relation with a large fraction of the resolver population
  - via their members, LIRs/NIRs, ISPs, etc.
- RIRs are already working on securing this relation
  - establishing their own CA structures, etc.
- Seems to be a very good match for the requirements. Unclear if there is a good alternative.

Thanks for listening.

johani@autonomica.se

This presentation is available at:

http://www.autonomica.se/~johani/talks/arinXI-dnssec-ksk-mgmt.pdf