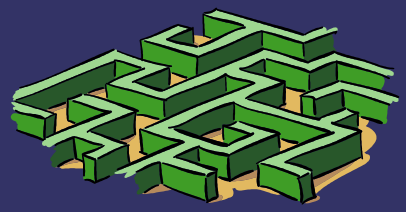
Simple Automated Key Management For DNSSEC-bis

Paul Vixie
ISC, Nov'05

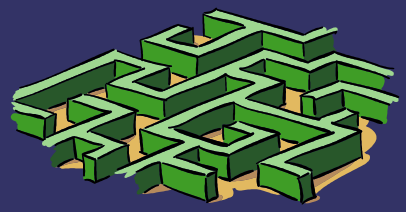


Problem Statement

- Trust anchors have to be:
  - Introduced
  - Updated
  - (Revoked?)
- Trust anchors might exist for:
  - root zone (ultimate key of doom?)
  - private zones
  - private relationships
- DNSSEC-bis assumes:
  - that this problem will be handled “somehow”

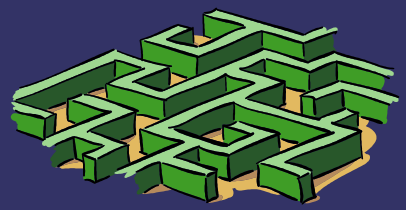
Known Alternatives

- Cut & paste, cron job w/ HTTPS, etc
- Timer based (stjohns)
- Threshold (johani)
- DLV? (vixie)



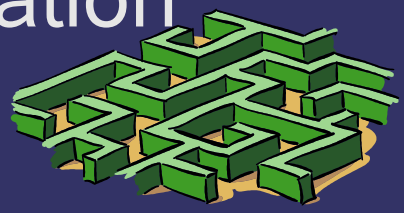
Why Is This So Hard?

- Key management policy is per-nameserver
  - how many keys will be stored? (1? $1x10^6$?)
  - for which zones? (just root? a list? LRU?)
  - how much trust is needed for a rollover?
- Key generation policy is per-zone
  - how many keys are in use?
  - how long do they last?
  - what's the overlap period?



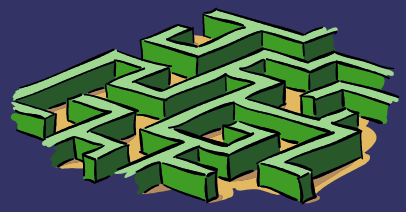
Simplified Key Management

- Add new zone apex RR, H(keyset)
- Include this in authority section of replies
- Interested validators can track it
  - when it changes, fetch new keyset
  - if new keyset validates, it's the new anchor
- Interested validators can also poll
  - on halflife of current keyset lifetime and on halflife of current keyset TTL
  - this is obviated by the new apex RR *if seen*
- New RR is an opportunistic optimization



How This Differs from N-of-M

- For one thing, N is a per-alg constant (“2”)
  - so, there is no policy knob on the client side
- New zone apex RR trumpets new keysets
  - so, most validators mostly won't have to poll
- Revocation is by omission only
  - so, it's always good to have more than one key
  - as well as preannouncing new keys at halflifes
- Puts most of the policy on the server side
  - instantiate, announce, and use new keys
- Validator policy is simple (static?)
  - track configured static trust anchors



Possible Server Side Policy

- Never use a key without also publishing the next one (or the next several)
- Never use a key without also publishing a backup at the same time (for revocation)
- Overlap current/next key lifetimes by 50%
- Start using a new key at the second halflife (25% remaining) of the existing key