DNSSEC @ IANA

RIPE 55 October 2007

Amsterdam

Thanks to Many!!

- IANA's design is built on the trailblazing work by .SE. Without the generous help from Jakob Schlyter and others at .SE, I would still be lost.
- Thanks to nlnetlabs.nl, Olaf, and others for the INVALUABLE “DNSSEC HowTo” and RFC4641 (DNSSEC Operational Practices) documents.

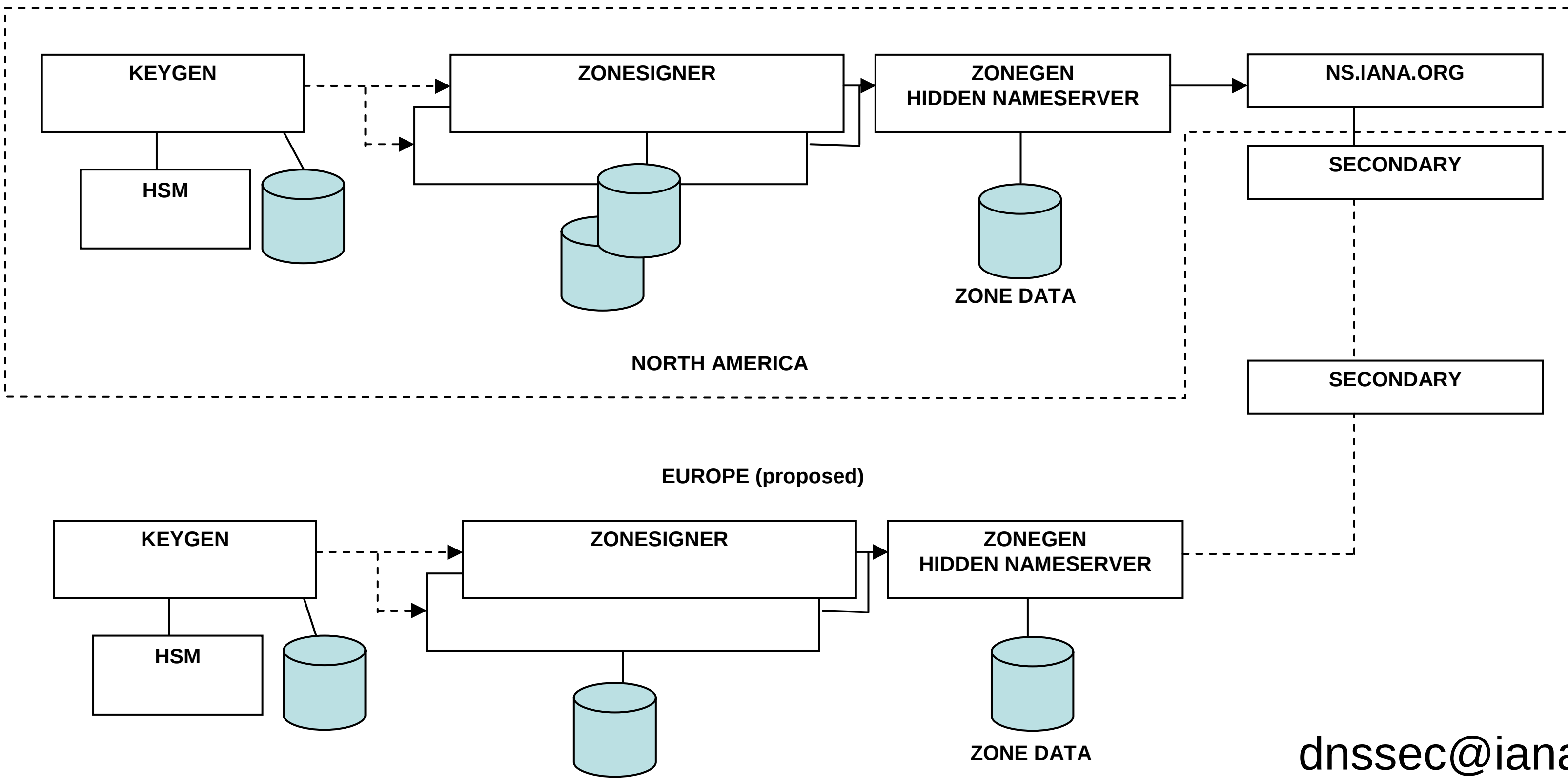
R1

R1 which served as our guide to parameter selection
RL, 24/09/2007

Design Goals

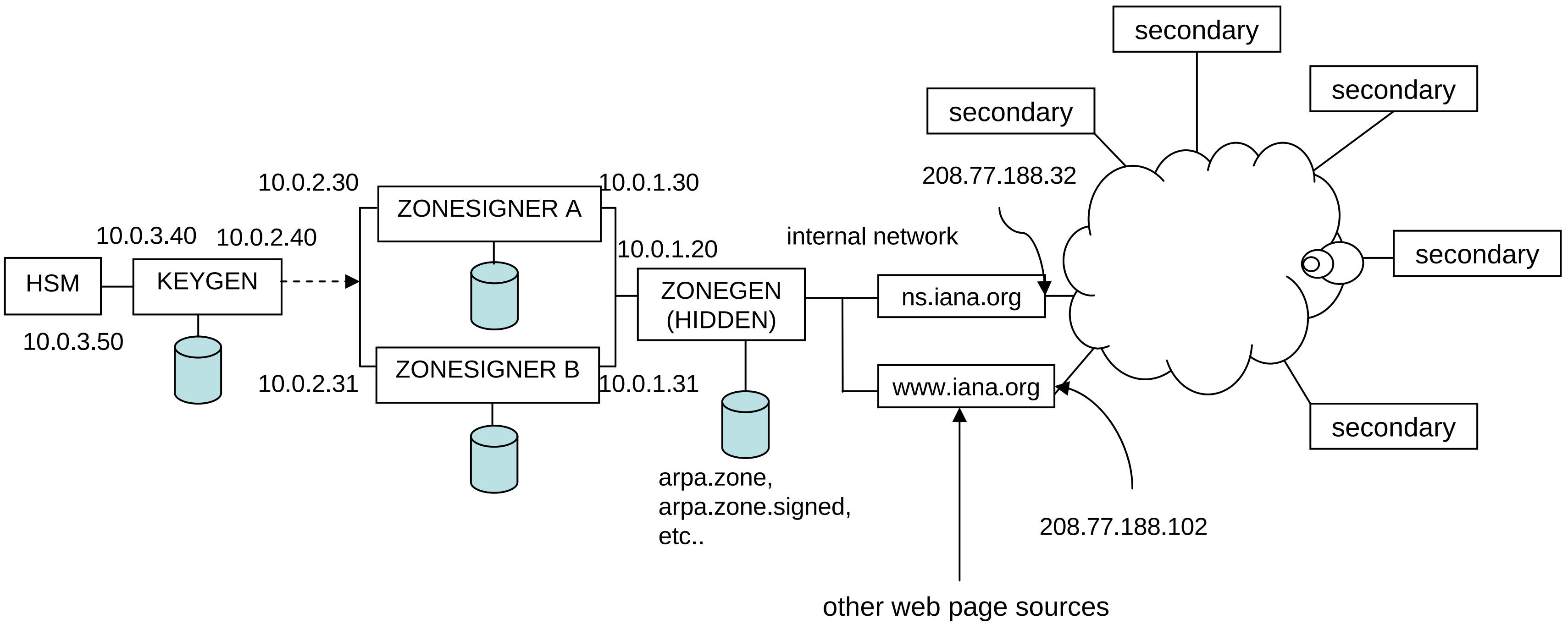
- Maintainability – if its not easy, it will fail
- Reliability – if there is a problem, no one will use it
- Security – it must look and be secure for people to trust it
- Target – arpa, in-addr.arpa, uri.arpa, urn.arpa, iris.arpa, ip6.arpa, int.

"Figure 1"



dnssec@iana

Figure 1 Details



Hardware (per site)

- 4x Dell 1RU 1950 commodity servers
- 1x AEP Keyper Pro (FIPS 140-2 Level 4) external Hardware Security Module (HSM)
- 1x KVM console
- Smart cards, Flash drive
- Locked rack within ICANN cage at secure colo facility

Maintainability - Only Two Scripts

- On ZONESIGNER – **signall**: automatically run daily on multiple machines to pickup zone changes (based on SOA serial, new DS records, or expiring signatures); reload hidden master; check key status; update status web page; and email notifications.
- On KEYGEN – **keyall**: manually run monthly (when notified by email). Generates new keys and signed key bundles for ZONESIGNERs as needed. Also backs up any new keys.

Maintainability – Overlapping Keys, Rollover Script

- Multiple overlapping keys (effectivity periods) to simplify rollovers.
- ZSK - three (3), old-active-new, overlapping ZSKs /w staggered effectivity periods. Use currently “active” key to sign records
- KSK - two (2) overlapping KSKs /w staggered effectivity periods. Use both to sign “key bundle” of five (5) keys
- Key generation and rollover automated in **keyall**

A5

```
64000K++++++++++++++++++++++++++++++++++|+++++++++++++++++++++++++++++++++++++++
24000K-------------------------+++++++++|+++++++++++++++++++++++++++++++++++++++
24001-----------------ppppppppp+++++++++|++rrrr---------------------------------
08000Z-------------------------pppppppp+|+++++++++rrrrr-------------------------
92000----------------------------------p|pppppp++++++++++++rrrr-----------------

keyindex file:
dn  type alg tag    publish date   start date     end date       remove date
root KSK 005 64000  19750101000000 19750101000000 19761231235959 19761231235959
root KSK 005 24000  19760101000000 19760101000000 19771231235959 19771231235959
root ZSK 005 24001  19751201000000 19760101000000 19760215000000 19760229235959
root ZSK 005 08000  19760101000000 19760201000000 19760315000000 19760331235959
root ZSK 005 92000  19760201000000 19760301000000 19760415000000 19760430235959
```

A5 From rfc4641:

"Key effectivity period" The period during which a key pair is
expected to be effective. This period is defined as the time
between the first inception time stamp and the last expiration
date of any signature made with this key, regardless of any
discontinuity in the use of the key. The key effectivity period
can span multiple signature validity periods.

"Signature validity period" The period that a signature is valid.
It starts at the time specified in the signature inception field
of the RRSIG RR and ends at the time specified in the expiration
field of the RRSIG RR.

"Signature publication period" Time after which a signature (made
with a specific key) is replaced with a new signature (made with
the same key). This replacement takes place by publishing the
relevant RRSIG in the master zone file. After one stops
publishing an RRSIG in a zone, it may take a while before the
RRSIG has expired from caches and has actually been removed from
the DNS.

Although the "validity period" (RRSIG expiration-inception value) for the ZSK is 6 days with a "publication period" of 1 day (how often I re-compute the RRSIG and shift forward the validity period), the same keys are used for much longer "effectivity period"s. One and a half months for ZSKs with new ZSKs introduced approximately every month (within a 1/2 month window). And two years for KSKs with new KSKs introduced every year (within a 1 year window).
Administrator, 30/07/2007

Maintainability – Compromised Key, Replacement Script

- For bad ZSK (old, active, new keys)
  - old – replace key with newly generated “old” key.
  - active – use old key to sign and generate a replacement. Phase out bad key. A15
  - new – replace key with newly generated “new” key.
  - Normally done in one-step. Two-steps if “close” to a transition to account for DNS propagation delays.
- For bad KSK (2 keys)
  - One - replace key with newly generated KSK with the same effectivity period and immediately publish.
  - Both – generate two keys and phase out bad keys? A6
- Process semi-automated with **badkey** script

A6 Will RFC5011 Revoke bit help here?
Administrator, 24/09/2007

A15 This is the reason for replacing the old key.
Administrator, 11/10/2007

Reliability – Dual Signers

- Signatures on zone records are only valid for six (6) days to limit replay attacks. So an inability to sign for more than 6 days will result in DNSSEC validation to fail.

- Design: Two (2) commodity hardware based ZONESIGNERs periodically executing **signall** to make sure the zone gets signed by one of them.

Reliability - Key Backup

- Must backup even private keys to recover from catastrophe
- Encrypt and propagate new private key material as key operations generate them
- Built into regular key operations script **keyall**

Security – KSK/ZSK Split

A13

- Following .SE's lead, sensitive KSK operations are kept separate from routine ZSK signing operations by only exporting pre-signed public key bundles and a single private ZSK from KEYGEN to ZONESIGNERs.

- KEYGEN machine is connected to ZONESIGNERs only during key generation and transfer operations

A7

A7 Even a demo signed root might be helpful here as we could immiediately publish the new DS record for arpa
Administrator, 24/09/2007

A13

man in the middle
replay
cache poisoning
Administrator, 24/09/2007

Security – HSM

- To protect against internal as well as external attacks, KSK operations (generation, signing, backup) for critical zones are performed inside the HSM.
- Do this using modified BIND tools with native PKCS11 support
- To minimize HSM operational overhead, child zones falling under .arpa will not use the HSM for KSK operations. Recovery from child zone KSK compromise can be effected quickly

Security – Key Lifetimes

- New ZSK 1024 bit every month to frustrate key guessing
- New KSK 2048 bit every year to frustrate key guessing
- Two KSKs always valid to support orderly replacement of old or compromised KSK
- Three published ZSKs to support orderly replacement and promotion of old or compromised ZSK
- 6 day (short) ZSK signature validity period to limit replay attacks while providing some time to recover from severe signing equipment failure
- 1.5 month key bundle KSK signature validity period to constrain compromised ZSK effects while not requiring daily manual resigning with KSK

Security – Key Backup

- Keys generated inside HSM (KSKs) are encrypted inside HSM before export
- Unencrypted key material (e.g. ZSK), key index, encrypted HSM keys (above), HSM configuration, and any other updated material on KEYGEN's hard drive is further encrypted using internal HSM key before transmission/backup
- Only another HSM with the same internal HSM key can decrypt this material
- Internal HSM key backed up on N of M smartcards

Security - Meatspace

- Key generation operation requires:
  - Access to DNSSEC equipment at a secured colo facility
  - One Security Officer smartcard and PIN to enable the HSM
  - HSM User PIN to generate keys and sign the key bundle
- A minimum of two (2) authorized personnel, controlling different components above, must be present for the entire key generation operation.
- Every access to DNSSEC equipment is logged in a DNSSEC log book A14
- **keyall** propagates its activities to the DNSSEC Administrator via email
- Material used to (re)build KEYGEN and HSM contents will be stored in safety deposit boxes. Each box will contain one of the required 2 out of 4 HSM master key smartcards along with an encrypted backup of current KSKs and miscellaneous configuration files needed for rebuilding

A14 I see at least five (5) people should be authorized at any time with no one person having control of both security officer card card and HSM PIN.

Administrator, 11/10/2007

Software

All software and modifications will be available as open source

KEYGEN

- keyall, kgen, badkey, and support programs
- pkcs11-backup, pkcs11-changepin,pkcs11-encrypt, pkcs11-random
- pkcs11 modified BIND tools: dnssec-signzone and dnssec-keygen

ZONESIGNER

- signall, zsign, and support programs

ZONEGEN

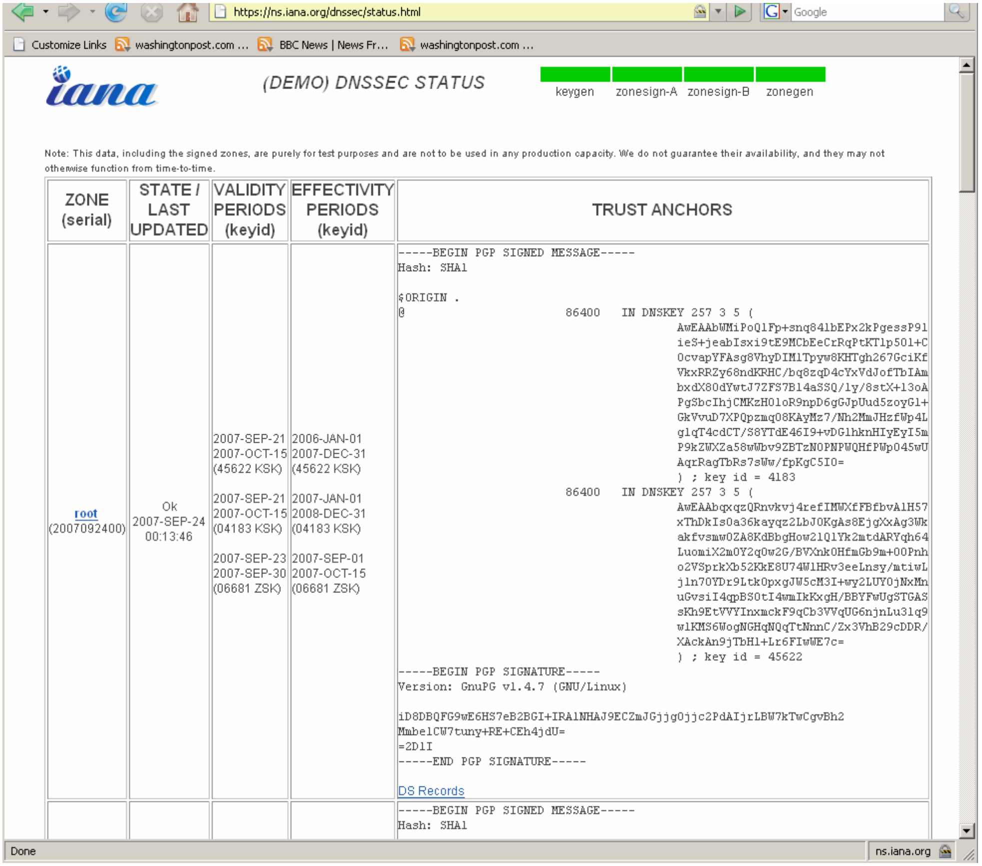
- upsite – DNSSEC status web page generator

DNSEC Status Page

https://ns.iana.org/dnssec/status.html

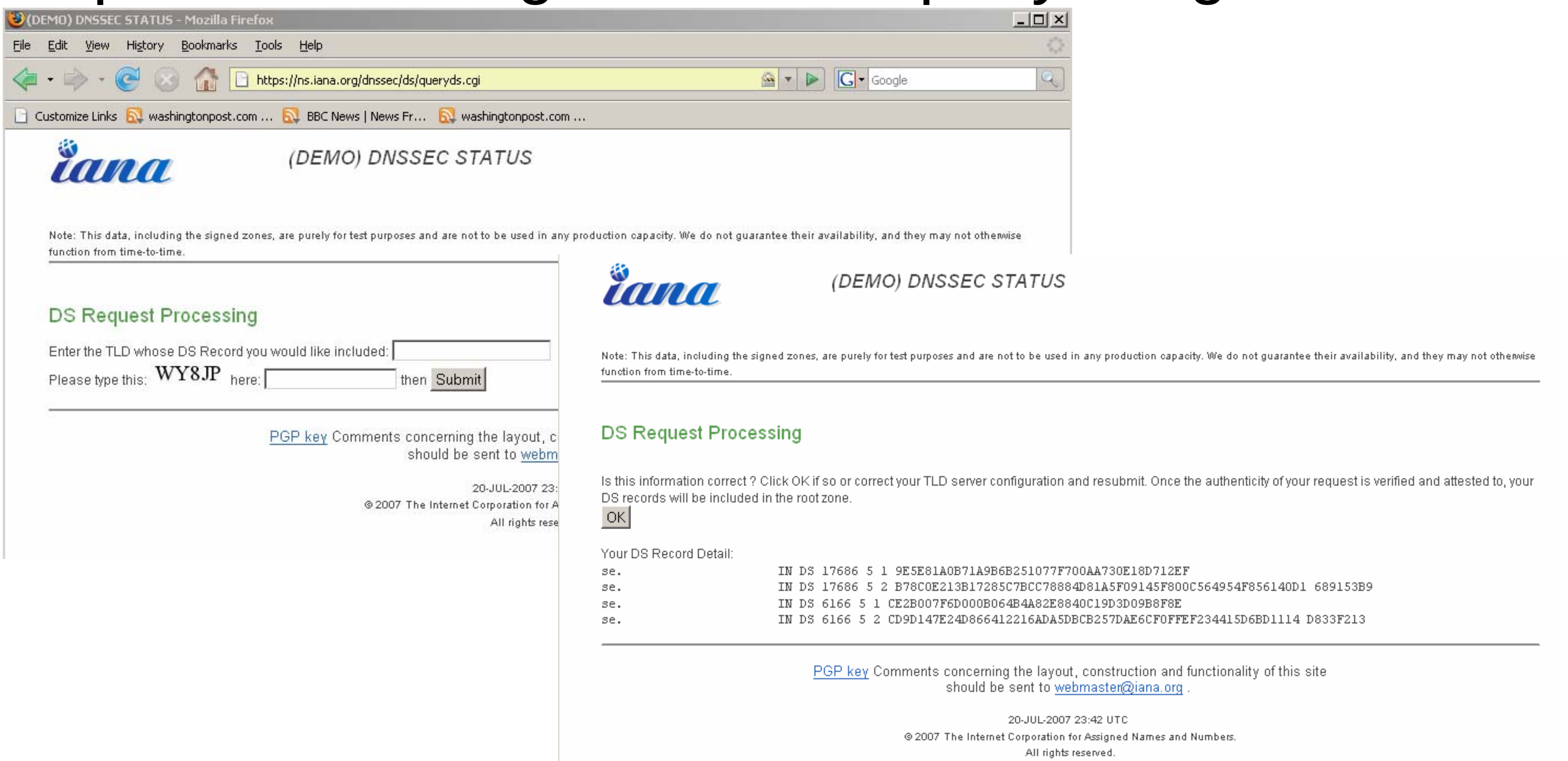
System status and publication of PGP signed trust anchors only on SSL secured site.

Domains: root, arpa, in-addr.arpa, uri.arpa, urn.arpa, iris.arpa, ip6.arpa, int



DS Record Handling

- Integrate into IANA root zone management
- https://ns.iana.org/dnssec/ds/queryds.cgi



Questions I have

- How’s it look?
- Compromised key recovery in the face of disinterested users. Update vectors: A3
  - Windows update, anti-virus software updates, RFC5011/Revoke bit St. Johns,..?
- How to detect compromised keys? A9
- Other DS record acceptance/derivation mechanisms?
- Other suggestions?

Slide 20

A3 Fix this and oddly enough, for a man-in-the middle attack, KSK recovery might happen faster than ZSK recovery.

Difference is we can push one or all new ZSKs down a no-m-i-t-m channel but w/o rfc5011, we cant push even one new KSK down. Even with 5011, we cannot push 2 new KSKs down.

Administrator 7/12/2007
Analysis

During the validation process a resolver recursively asks for DNSKEY material then verifies it with DS records signed by the parent (I tcpdump/sniffed this). (ugly notation: A b.c = request for A record for domain b.c. K2b.c. = KSK 2 for domain b.c. (K1,K2,Z)Zb.c. = RRSIG of b.c. key bundle with zone key Zb.c.)

A b.c. ->
    <- IP, (IP)Zb.c.
DNSKEY b.c. ->
    <- K1b.c., Zb.c., K2b.c., (K1,K2,Z)Kb.c., (K1,K2,Z)Zb.c.
DS b.c. ->
    <- DS, (DS)Zc.
DNSKEY c. ->
    < K1c., Zc., K2c., (K1,K2,Z)Kc., (K1,K2,Z)Zc.
DS c. ->
    <- DS, (DS)Z.
DNSKEY . ->
    <- K1., Z., K2., (K1,K2,Z)K., (K1,K2,Z)Z.
Validate Z. with K1. and K2.
K1. and K2. match the trust anchors in the end user resolver

If someone compromises K2. (and assuming we notice this), we generate a new K2. which will no longer match the trust anchor in the lazy user's resolver (after caches have expired). So in this case TLDs signed with the old K2. fail. If on the other hand the ISP or another upstream provider intercepts root DNS requests and has the private K2. key, a false sense of security prevails until the trust anchors in the resolver are updated manually on the user machine, at the upstream ISP's DNS server used by its customers, or by Windows Update or other such maintenance mechanism. Although shortening the validity (RRSIG) periods over the KSKs may guard against replay attacks and a ZSK compromise (KSK RRSIG of ZSK good for about a month so for intercepted DNS root server requests, recovery from a ZSK compromise could take this long), it cannot protect against laziness and interception.

Administrator, 24/09/2007

A9 reason to keep distance between KEYGEN and ZONESIGNER: someone who has gained control of ZONESIGNER could instruct it to generate a key bundle with infinite validity period and copy private ZSK. So even w/o having the private KSK, an attacker could cache poison/impersonate for the lifetime of the KSK trust anchors in resolvers: a week to forever but typically a year.

Administrator, 24/09/2007