Key Rollover for the RPKI

Steve Kent

(Channeling Geoff Huston ☺)

Key Rollover Document

- New doc: draft-huston-sidr-aao-profile-0-keyroll-00.txt
- Formerly a (non-normative) section of the resource profile, now expanded with more details
- Target is a standards track RFC from SIDR
- The document describes
  - what CAs have to do to effect key rollover
  - what RPs can expect during key rollover

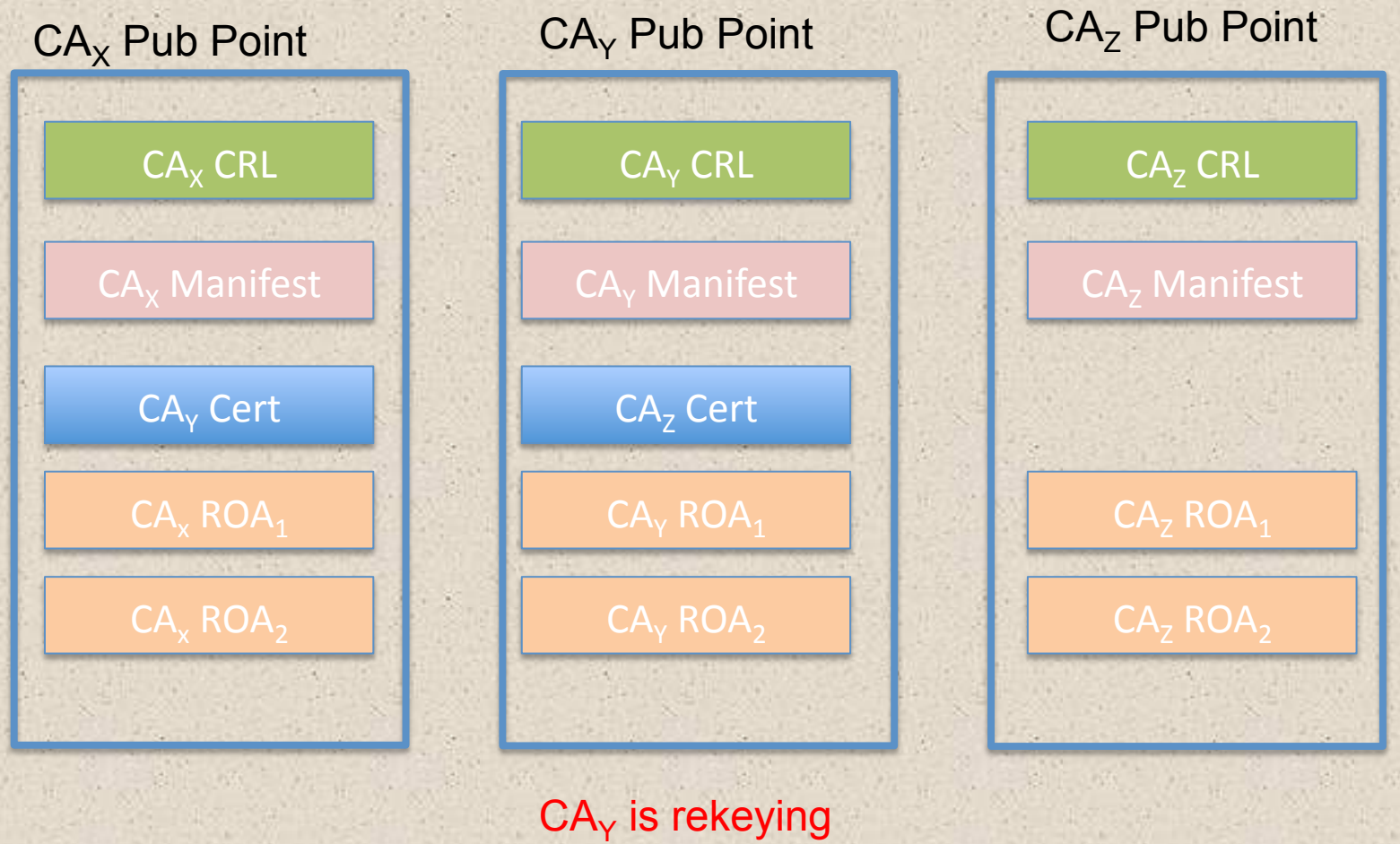
Relation to Repository Structure

- Rekey design relies on some conventions in draft-ietf-sidr-repos-struct-04.txt
  - The file name for a CA certificate is derived from the public key in that certificate
  - The file name for a CRL is derived from public key of the CA that issued the CRL
  - The file name for a manifest is derived from the public key of the CA that issued the manifest
  - The file name for an EE certificate is derived from the public key in that certificate
- These conventions are important as they determine which files are overwritten and which persist, during rekey
- RPs also need the CA to use a read lock on a directory (publication point) while contents are being changed

Relation to Resource Certificate Profile

- Rekey relies on the AIA extension in a certificate pointing to the parent certificate and CRL files (vs. to the directory in which these files are stored)
- Use of the 3 character file suffixes (for certificates, CRLs, ROAs, and manifests) probably makes life easier for RPs in all cases
- Other??

Key Rollover Example



CA_Y is rekeying

CA Key Rollover States

- **Current** – the active CA used to issue certificates
- **New** – the CA being created, not yet issuing certificates not yet used for validation
- **Pending** – CA is reissuing certificates from Current CA, but is not yet accepting new certificate requests (no revocation requests accepted?)
- **Old** – CA does not accept certificate requests, but does continue to issue CRLs for certificates issued under its key, and generates new manifests (and, hence EE certificates for these manifests), until the old CA certificate is revoked (a short interval)

(These states are internal to the rekeying CA)

Key Rollover Steps 1-4

1. The CA that is rekeying generates a NEW key pair
2. It then generates a certificate request with the NEW key pair and passes the request to the its parent CA
3. The parent CA generates a new name for the CA that is rekeying, signs the certificate and publishes it
4. The "Staging Period" begins (duration is selected by the CA that is rekeying). The period MUST be no less than 24 hours. This interval is intended to allow all RPs to acquire the CA certificate with the new key before any new subordinate certificates are published (the new CA might also publish a CRL during this interval, with no entries, and with a next issue date after the interval ends)

Key Rollover Steps 5 & 6

5. Upon expiration of the Staging Period, the rekeying CA suspends the processing of certificate issuance requests (and revocation requests??). Mark the CURRENT CA as OLD and the NEW CA as PENDING. Halt the operation of the OLD CA for all operations except the issuance of CRLs and EE certificates for manifests. (What about emergency rekey requests during this period, e.g., if it is longer than 1 day?)

6. Use the PENDING CA to generate new certificates for all existing subordinate CA and (multi-use?) EE certificates, and publish them in the same repository publication point and with the same file names as the previous OLD subordinate CA and EE certificates (thus over-writing). The keys in these reissued certificates MUST NOT change.

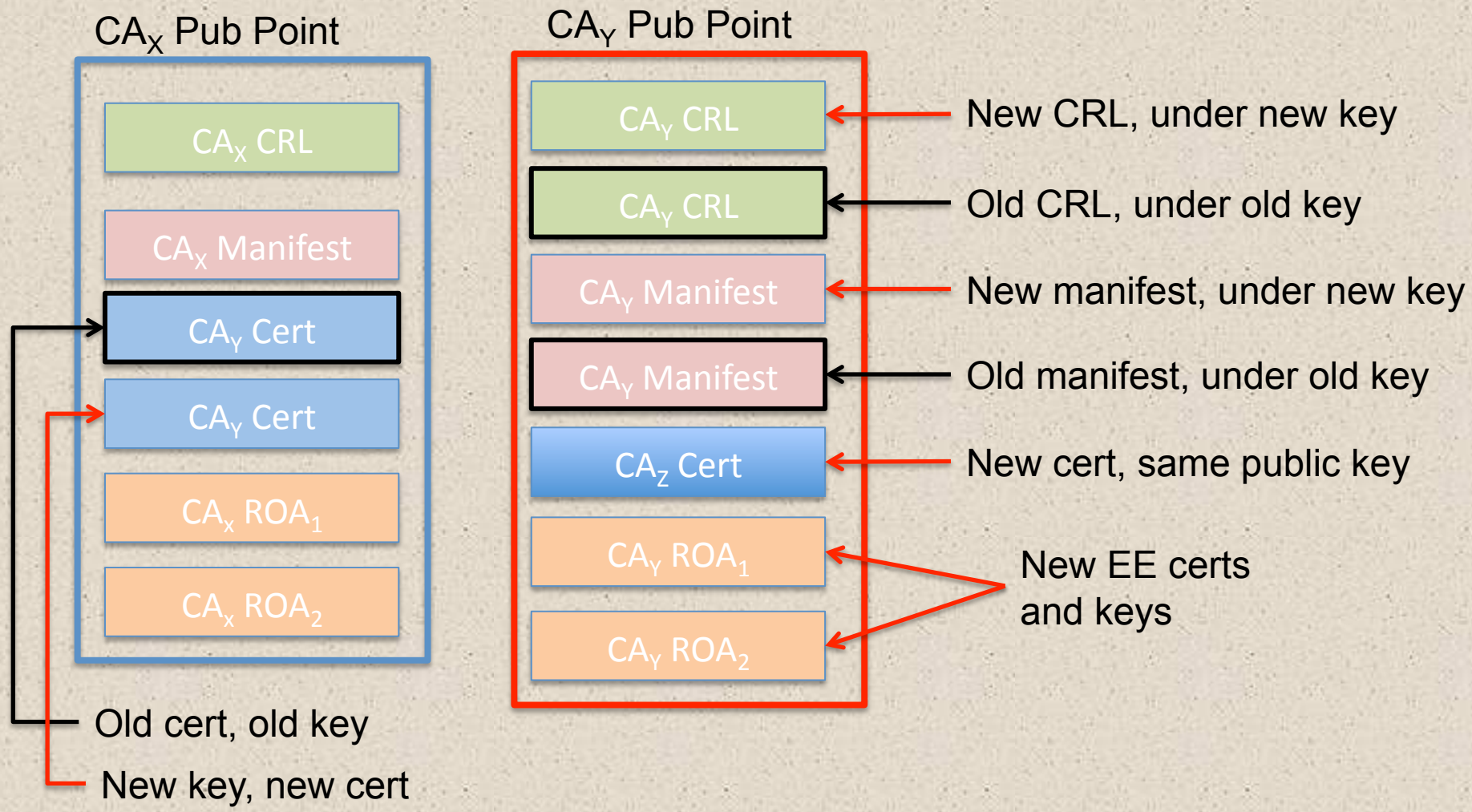
Key Rollover Step 7

7. Each signed object (other than manifests) that includes an OLD CA-issued EE certificate in its signed data (e.g., a ROA) will need to be re-signed using an EE certificate issued by the PENDING CA. For a "single use" EE certificate, if the associated private key has been destroyed, this requires generating a new key pair, re-issuing the EE certificate under the PENDING CA, and signing the data by the newly generated private key. For a "multi-use" EE certificate, the EE certificate is re-issued using the PENDING CA. The object will be signed with the associated private key, and published in the same repository publication point, using the same file name as the previously signed object that it replaces.

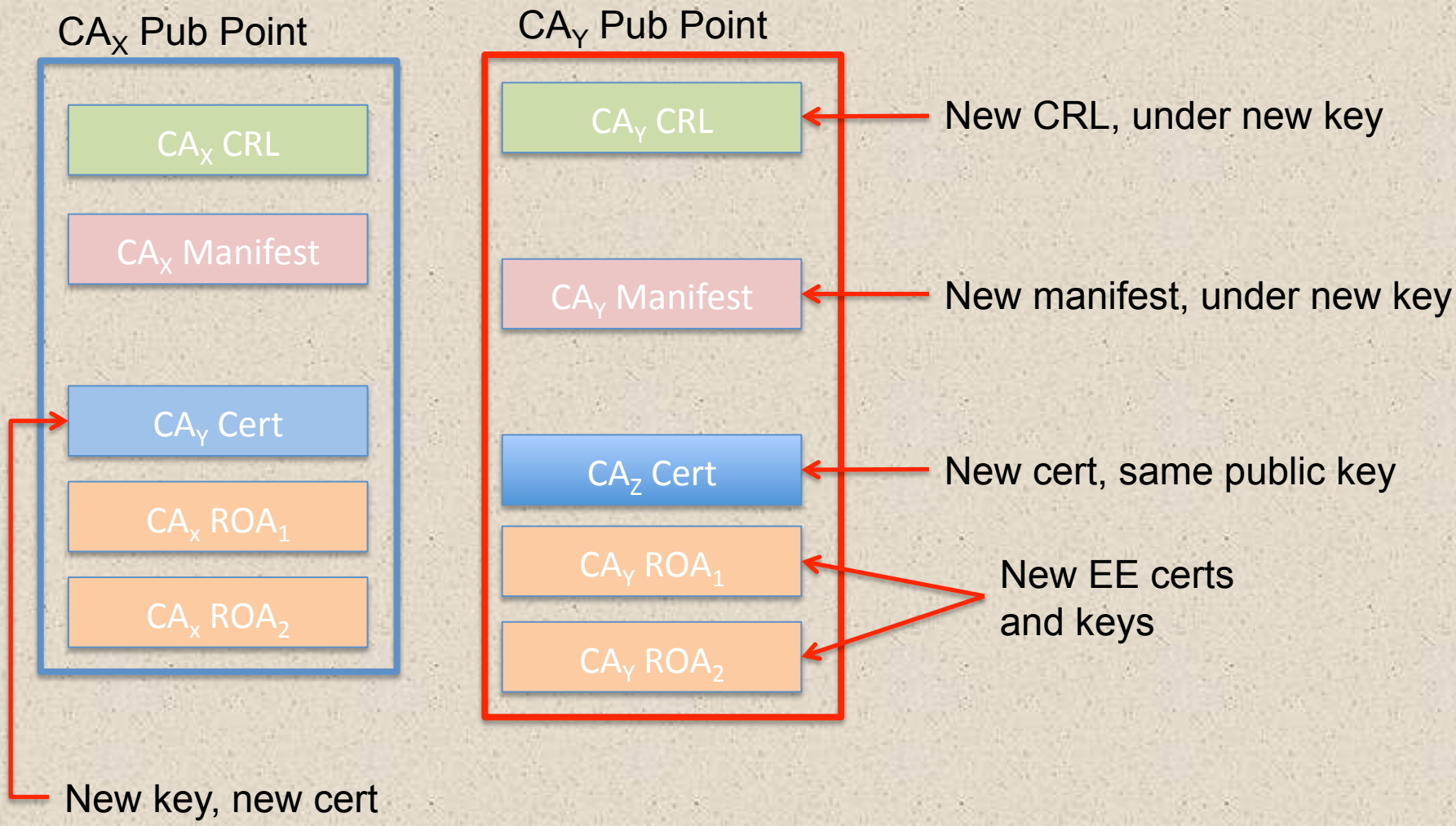
Key Rollover Steps 8-11

8. Use the OLD CA to issue a manifest that lists only the OLD CA's CRL, and use the PENDING CA to issue a manifest that lists all CA and EE certificates and the new CRL issued by the PENDING CA.
9. Mark the PENDING CA as CURRENT and resume processing (CA) certificate issuance requests
10. Generate a certificate revocation request for the OLD CA certificate and pass it to the parent of the CA that is rekeying (what triggers this, e.g., expiration of last certificate issued under the old CA?)
11. Wait for the OLD CA certificate to be revoked then remove the OLD CA's CRL and manifest and destroy the OLD CA's private key.

Example: After Rollover



Example: Rollover Really Complete!



Relying Parties

- The key rollover designs assumes that each RP will sync its local cache within 24 hours or less
- This requirement is tied to the staging period for the key rollover, which is set to no less than 24 hours
- The period allows all RPs to acquire the NEW CA certificate, so that when the CA that is rekeying issues NEW subordinate CA certificates, ROAs, etc., all RPs will be able to validate them
- During the staging period RPs will acquire and validate the OLD products signed by the CA that is rekeying, because the OLD CA certificate is still available, and no new products have yet been published
- The repository system relies on locking during the interval when a CA publishes new certificates, CRLs, and ROAs, so that the manifest associated with a set of signed objects is consistent

Comments

- Note that this is a brand new document, and thus needs review and comments from the WG
- I noted a few issues that need to be addressed in the course of this presentation (in red)
- The biggest question, from my perspective, is whether we should try to adopt a common model for key rollover and algorithm transition (see Roque's presentation)