RPKI Algorithm Profile

Update on the draft-ietf-sidr-rpki-algs-01 draft

Summary

- Place all the RPKI algorithms and key size specifications into a single document
- Algorithm and key size:
  - MUST: SHA256 with 2048 bit modulus
  - SHOULD: SHA512 with 3072 and 4096 bits
- Transition:
  - SHOULD be able to perform piecemeal transition with multiple algorithms in play
  - HOW requires more thought!

Algorithm Specification

New text (section 2) *(based on WG discussion in September 2009)*:

> "This profile specifies the use of RSASSA-PKCS1-v1_5 [RFC3447] with the SHA-256 hash algorithm to compute the signature of certificates, CRLs, and signed objects in the context of the RPKI. Accordingly, the OID value in the RPKI for such signatures MUST be 1.2.840.113549.1.1.11 (sha256WithRSAEncryption). The RSA key pairs used to compute the signatures MUST have a 2048-bit modulus and a public exponent (e) of 65,537"

Transition

New text *(based on WG discussion in September 2009)*:

> "In order to facilitate a potential need to transition to stronger cryptographic algorithms in the future, Certification Authorities (CAs) and Relying Parties (RPs) SHOULD be able to generate and verify RSASSA-PKCS1-v1_5 signatures using the SHA-512 hash algorithm and RSA key sizes of 3072 and 4096 bits."

Outstanding

- Security ADs Comment:

  "We do have concerns about the algorithm transition strategy for SIDR in general, though. Since there is no negotiation and relying parties will all need to support the new algorithm before the “phased introduction" is completed, there will be **a need to support multiple algorithms** simultaneously. **How is this reflected in the sidr technical specifications?**"

Possible Options -a

- Punt it and do nothing!

Possible Options - b

- Allow multiple signature algorithms in CMS, with multiple EE certificates corresponding to each signature algorithm?
  - this requires a change to the ROA format, and any other RPKI-defined signed object that uses CMS
  - and may not be all that useful in constraining the size of the RPKI in any case as the certificates and CRLs are constrained to a single algorithm, implying that the manifests are also similarly constrained

Possible Options - c

- Support “parallel” certificate issuance
  - Outstanding issues of CA definition, Issuer and Subject Names, CRL management and validation chain construction need to be clearly defined - as with key rollover, a entity’s use of a new algorithm infers that the entity creates a new CA
  - The “overwriting” technique used in key rollover cannot be used here because of the need to maintain “old” and “new” certificate algorithms in parallel
  - Have the CA request certificates against the *Current* and *Sucessor* key values associated with each algorithm and then issue subordinate products in parallel using each algorithm & key combination?
    - There is a potential for recursive duplication in subordinate certificate issuance where there is parallelism introduced at every node – this could become exponentially large if not constrained
    - An alternative appears to be to apply some form of RPKI-wide constraint. e.g.: Force the parallelism into entire parallel trees where this is constrained to a “top down” approach where the issuer’s signing algorithm must match the subject’s algorithm.

Possible Options

- Does “Punt” look like the best of a bad bunch?

  e.g.: “The transition to a successor algorithm MUST be described in standards track documents and associated BCP documents outlining operational deployment for the successor algorithm.”