Suspenders: Mechanisms to Protect RPKI Resources Holders Against Errors (and Worse)

<insert speaker name here>

BBN Technologies

Reminder: LTAM

- Local Trust Anchor Management (LTAM) was developed to address two primary use cases

1. Allow local use of RFC 1918 address space by an ISP, without breaking RPKI mechanisms in use by the ISP
2. Provide a basis for protecting resource assignments against errors by (or law enforcement actions directed against) CAs in the RPKI, especially by higher tier CA like IANA or the RIRs

Why Suspenders?

- LTAM was presented at SIDR meetings since IETF 75 (July, 2009) but little feedback has been received
- It probably would work well at the IANA & RIR level for the 1st use case (local override of repository system)
- It would not work so well for ISPs that delegate address space, because the “hole punching” it mandated for RPKI certificates conflicts with ISP ROA issuance
- It did not specify details for resource protection (2nd case)
- Suspenders is a replacement for LTAM, for other than the RFC 1918 address space use case

The New Model

- We decided to focus only on ROAs, since changes to the RPKI that adversely affect ROAs are of interest to the associated INR holders (and to RPs)
- We anticipate this model can be extended to cover router certificates too, to support BGPsec
- The model has three elements
  - INR holders detect when their own ROAs no longer validate or are in “competition”
  - INR holders publish external info to “protect” their ROAs
  - RPs detect adverse ROA changes, check external info to decide if the change is OK, or to (optionally) revert to old data

Adverse ROA Changes

- ROA Whacking – A ROA is whacked when it becomes invalid due to any action by a CA (or publication point maintainer) along the path between the ROA EE certificate and a trust anchor. Whacking includes ROA certificate revocation, CA certificate revocation, CA certificate 3779 extension changes, removal of a ROA from the RPKI repository, etc.
- ROA Competition – A new ROA competes with an existing ROA when the new ROA is issued by a different entity, points to a different ASN, and contains the same or a more specific prefix. Competing ROAs are legitimate in some cases, but illegitimate overlaps represent a way to divert traffic.

Self-Monitoring of ROAs

- To first order, every INR holder is also an RP
- Every RP downloads all changed RPKI data at least daily, probably more often
- Each INR holder should be able to configure its own ROA data and use that data to check its ROA status
- RP software should provide info to the INR holder to identify why/where validation of its ROA fails
- The INR holder should contact the indicated party (e.g., using Ghostbusters) to resolve the problem

Publishing External Data

- If an error has caused a ROA to become invalid (or missing), RPs may want to ignore the problem, for a little while, to give the INR holder a chance to fix the problem (before treating the ROA as not valid)
- But, how does an RP know whether an adverse change to ROA data is sanctioned by the INR holder?
- Each INR holder needs a way to signal to RPs when it makes changes to its ROAs
- This signal must be external to the RPKI repository system, an independent assertion about ROAs

The Tough Case

- Today, most INR holders who have certificates and ROAs make use of outsourced CA and publication point management offered by the INR holder's RIR
- If an INR holder fears that the RIR may have been compelled to whack his ROA (e.g., by law enforcement) then he can't rely on his ability to change data within his publication point to fix a problem when it is detected
- An INR holder should publish status data where it is not under the control of the CA/publication point maintainer
- RPs need to be able to verify the authenticity of an external file associated with an INR holder

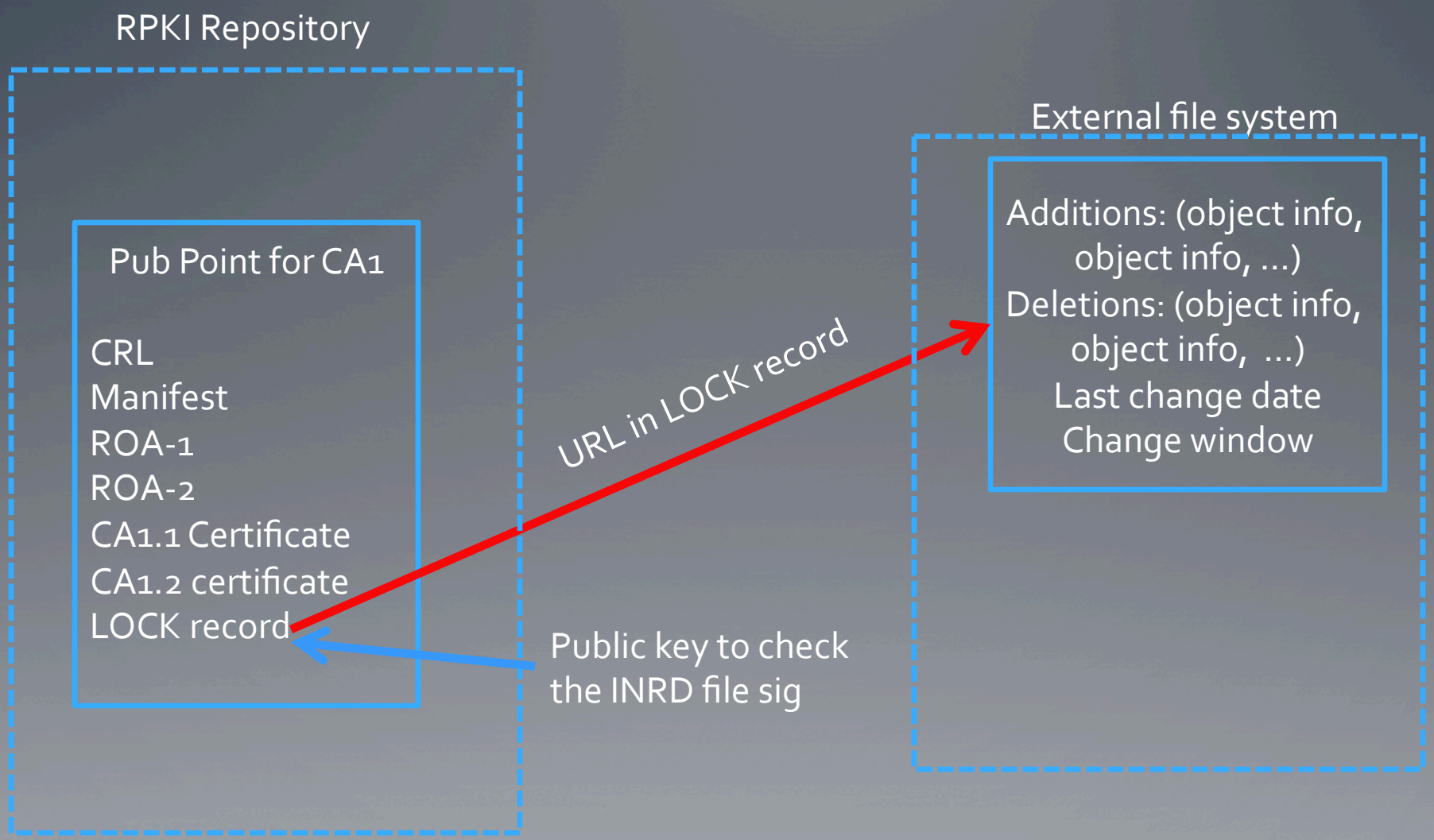
What Could RPs do?

- If an RP wants to give an INR holder an opportunity to fix problems when adverse ROA changes are detected, the RP can use the external INR data to add some inertia to the change process
- An RP that elects to adopt this approach ought not be burdened by this capability
  - It ought to be easy to detect adverse ROA changes
  - Additional data retained by RPs should be minimal
  - External data ought to be fetched ONLY as needed
- It seems possible to meet these criteria by adding one new RPKI object, and a simple external file format

Solution – Data Structures

- An INR holder may optionally add a new RPKI signed object, the LOCK record, to its publication point
- The LOCK object contains a URL pointing to the external data file and a public key (to authenticate the data)
- The external data file (INRD) contains a list of recent changes to protected objects, and the time frame for these changes
- Each RP that elects to make use of this data will maintain a (somewhat more complex) local database of ROA data asserted by each (protected) INR holder

Graphic Details



LOCK Record ASN.1

```
LOCK ::= SEQUENCE {
    version      [0] INTEGER DEFAULT 0,
    outsourced   BOOLEAN,
    uRL          IA5String,
    publicKey    SubjectPublicKeyInfo }
SubjectPublicKeyInfo ::= SEQUENCE {
    algorithm          AlgorithmIdentifier,
    subjectPublicKey   BIT STRING  }
```

The “outsourced” flag tells RPs how to deal with changes to the public key or URL. It is TRUE if the INR holder does not operate its own CA or publication point. When TRUE, changes to the URL or public key are not immediately accepted by RPs

INRD File ASN.1

```
TBSINRD ::= SEQUENCE {
   version          [0] INTEGER DEFAULT 0,
   lastChange           UTCTime,
   changeWindow         ENUMERATED
       {
       1week (7) DEFAULT
       2week (14)
       4week (28)
       },
   additions        [1] SEQUENCE SIZE (1..MAX) OF
                            ProtectedObject OPTIONAL,
   deletions        [2] SEQUENCE SIZE (1..MAX) OF
                            ProtectedObject OPTIONAL,
   keyRollover      [3] OCTET STRING OPTIONAL,
   algRollover      [4] OCTET STRING OPTIONAL
   }
```

Key rollover and algorithm rollover specify the SKIs for the "other" CA(s) when these processes are in progress.

INDR File – More Details

- The additions and deletions SEQUENCEs enumerate all changes to protect objects during the change window
- Objects in these lists are represented in a compact fashion
  - A router certificate is represented by its SKI & ASN
  - A CA certificate is represented by its SKI and its 3779 extensions
  - A “signed object” is represented by the encapsulated content from CMS (no certificate or signature)
- Key rollover and algorithm rollover conditions are signaled by presence of SKIs of the parallel CAs

Solution - Processing

- When an INR holder makes any changes to its ROA data, it must update its external file first
- RPs that choose to support anti-whacking scan changed ROA data for adverse changes
- A potentially adverse change to a protected objects trigger fetching an INRD file, if a LOCK record is present
- A potentially adverse change is accepted by an RP only if the INRD file corroborates the change; otherwise the RP may elect to revert to previously validated data
- During key rollover or algorithm rollover parallel CA(s) are not viewed as competing

What's Next

- SIDR WG action items
  - Revise LTAM to be a tiny document, addressing only the 1st use case
  - Accept Suspenders (draft-ietf-kent-sidr-suspenders-00) as the basis for the 2nd use case
  - Specify updates for RP and CA software to support these new data items and processing (update RFCs 6481, 6483, maybe 6480 too)
- In parallel, confirm that RIRs would be willing to generate and publish LOCK records for their managed CA clients

QUESTIONS ?