Local Management of Trust Anchors for the RPKI (LTAM)

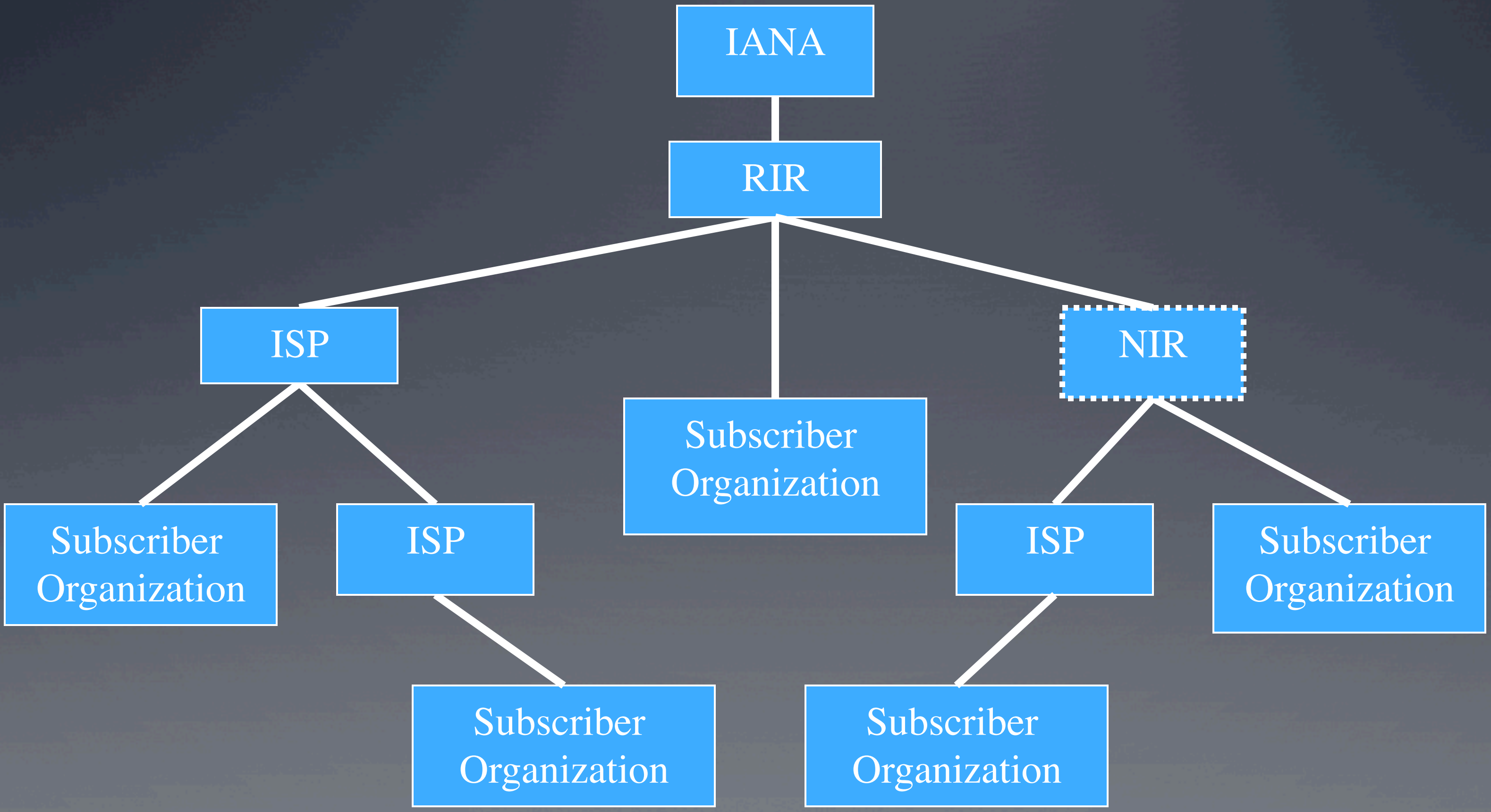
Stephen Kent

BBN Technologies

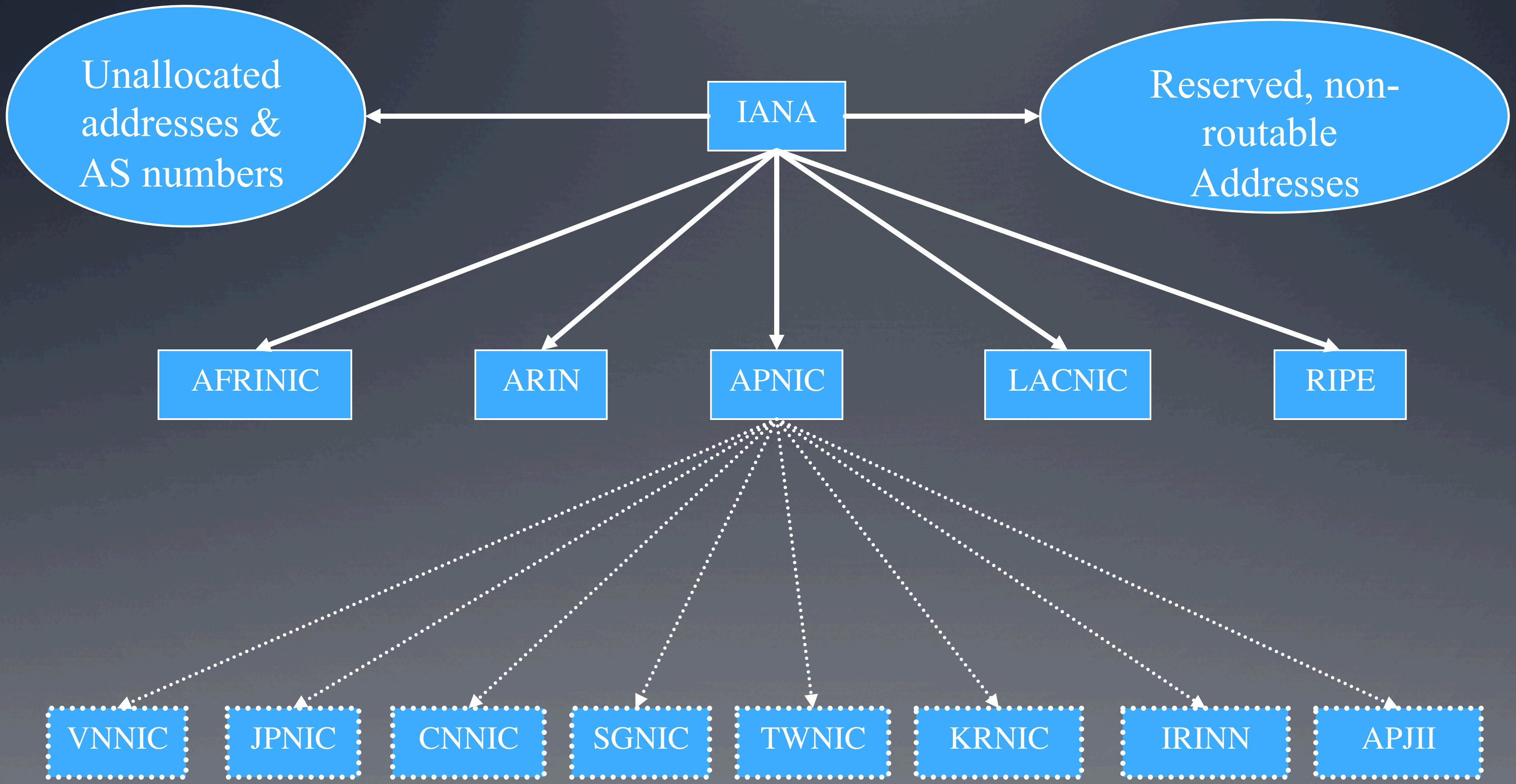
Quick RPKI Review

- The Resource Public Key Infrastructure (RPKI) is designed to enable operators to detect unauthorized (principally accidental) route origination in BGP (e.g., Pakistan Telecom vs. YouTube)
- It is also intended to serve as a basis for route path security enhancement to BGP in the future (BGPSEC)
- The RPKI is aligned with the address and AS # allocation hierarchy. Thus any attempt to assert “holding” of a prefix or an AS # that does not match IANA + RIR records will be rejected by participating ISPs

Address & ASN Allocation Hierarchy



RPKI Hierarchy (APNIC focus)



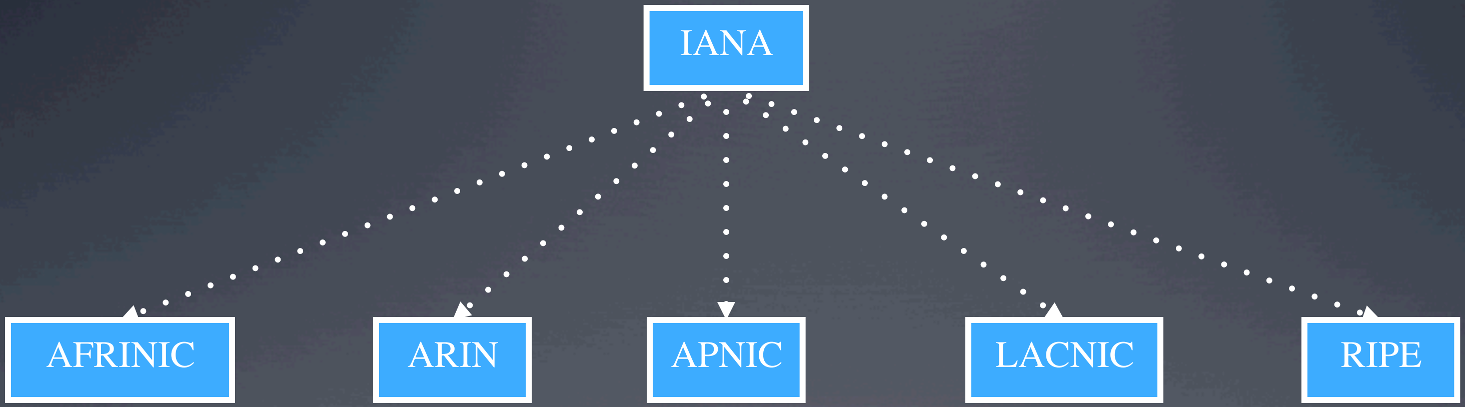
RPKI Principal Features

- The RPKI looks like a typical PKI in most respects
- However, the RPKI makes use of certificates that contain “extensions” defined by RFC 3779
- These extensions represent address space (prefixes) and AS #’s consistent with the allocation hierarchy
- The owner (subject) of an RPKI certificate controls the resources represented in that certificate
  - it can sub-allocate the resources to others
  - or use the certificate to assert the AS # of legitimate originators of routes for a specified prefix

Trust Anchors in the RPKI

- In any PKI there are one or more public keys, and associated data, that are distributed to users (relying parties) in an out-of-band fashion
- Often the public key and associated data are distributed in the for of self-signed certificates
- These keys are referred to formally as trust anchors (TAs), or informally as root certificates
- In the simplest case there would be only one TA for the RPKI, IANA, but for various reasons we currently have at least 5 (the RIRs)

RPKI Trust Anchors



Local TA Management: Why

- There are times when an operator wants to assert ownership of a prefix (or an AS #) in a local context
- In such cases it would be nice to be able to make these assertions, locally, without having RPKI/ BGPSEC software complain (to you, as the operator)
- The obvious case is use of RFC 1918 address space
- If an assertion about an IANA reserved address "escapes" the local context, it will be rejected by operators who make use of the RPKI, so other nets ought not be adversely affected

Another Local TA Motivation

- A nation might worry that some entity in the resource allocation hierarchy could (accidentally or maliciously) revoke a certificate for critical infrastructure resources

- A nation could protect nets within its administrative jurisdiction against such mishaps IF it could direct internal nets to rely on a national authority for RPKI data for these critical infrastructure resources

- Note that the protection offered this was has only local impact, so no other nets are affected

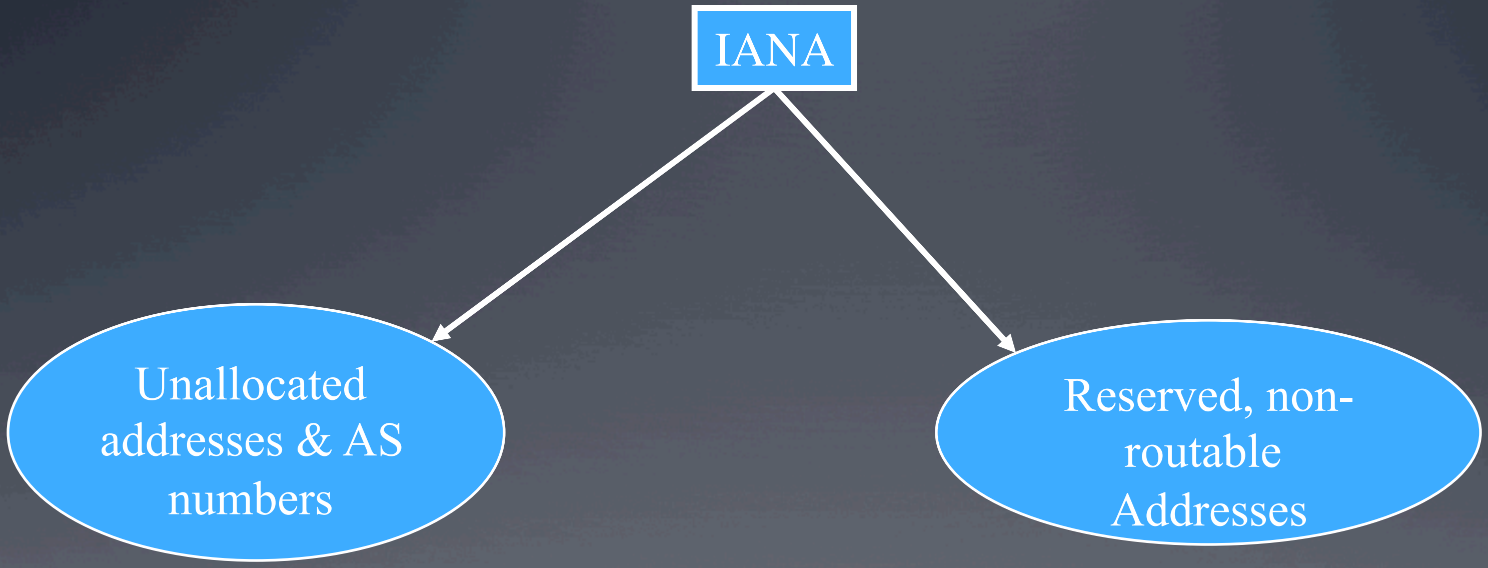
Local TA Management: How

- Local trust anchor management (LTAM) enables operators to make use of reserved address space, and to accommodate national “protection” goals, with minimal impact on RPKI software
- LTAM works by allowing each relying party (operator) to create its own TA, that it controls
- All other would-be TAs are subordinated to the local TA, providing an operator with complete control
- LTAM is a powerful tool and an operator needs to be very careful when using it (don’t shoot yourself in the foot!)

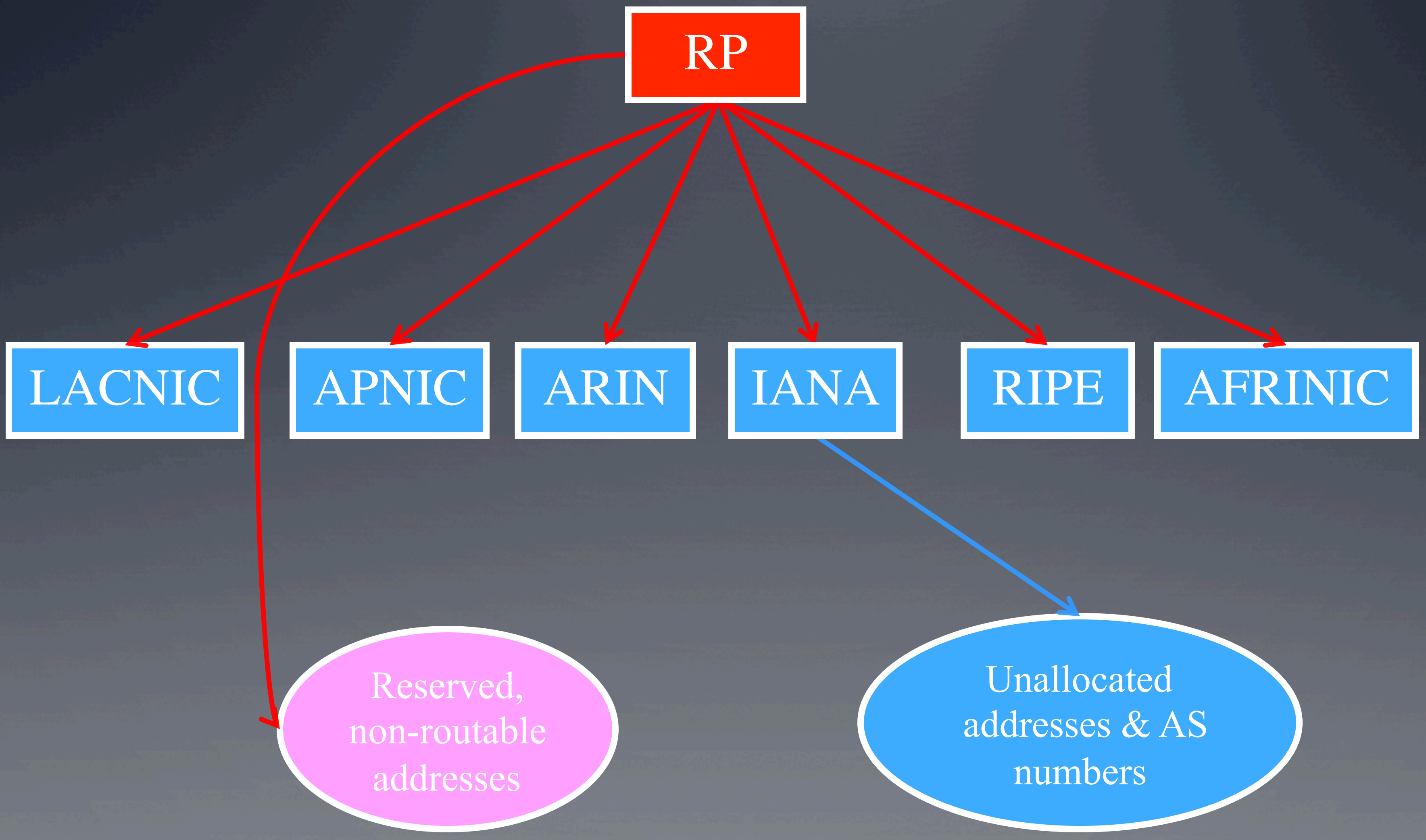
The Idea: The RP is the TA!

- When using LTAM, each RP (operator) recognizes exactly one TA, itself!
- The RP imports putative TAs (typically in the form of self-signed certificates) and re-issues them under itself
- The RP can thus override the RPKI nominal hierarchy, as represented in the repository system (paralleling the allocation hierarchy)
- Because this is a local TA other operators will not see the changes you make, but you can mess up routing in your environment if you make errors!

An RFC 1918 Example


IANA
Unallocated addresses & AS numbers
Reserved, non-routable Addresses

RPKI with LTAM Control


RP
LACNIC
APNIC
ARIN
IANA
RIPE
AFRINIC
Reserved, non-routable addresses
Unallocated addresses & AS numbers

(RP makes use of 10/8 for local routing)

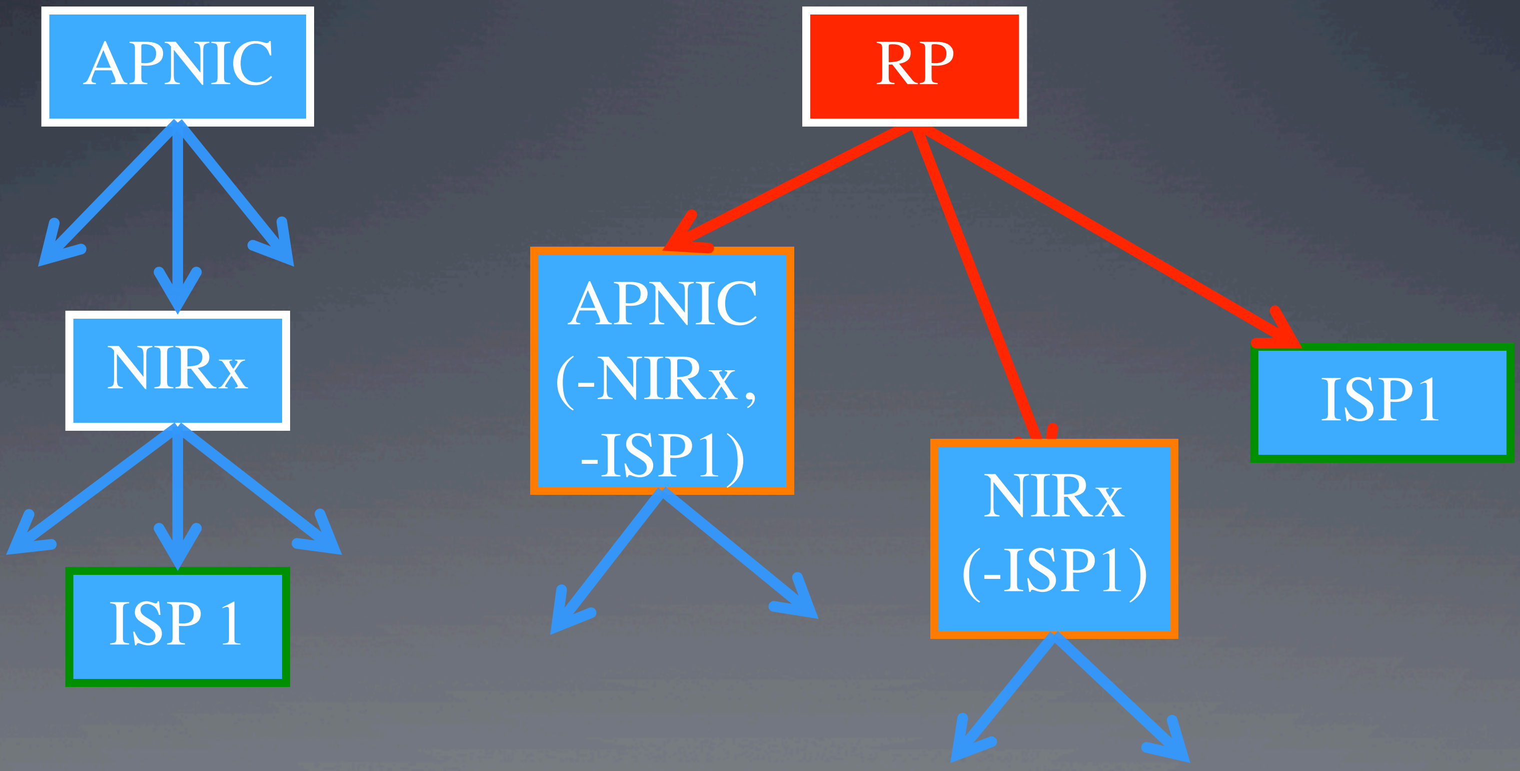
Making this Work in the RPKI

- To implement LTAM, an RP must be able to create new certificates, usually with modified RFC 3779 extensions
- To make this work
  - The self-signed RP TA certificate must contain RFC 3779 extensions encompassing <u>all</u> addresses and <u>all</u> ASNs
  - The RP issues certificates with new 3779 extensions to override the RPKI tree (as needed)
    - Delete overlapping 3779 data (as needed)
    - Re-home targeted certificates under the RP TA
    - Re-home ancestors of re-parented certificates under the RP TA
  - The RP can also override certain fields of re-issued certificates, e.g., expiration dates

A More Detailed Example

As offered by APNIC

As managed by an RP



(RP trusts its own knowledge of ISP1's address allocation and does not want any action by APNIC or NIRx to override that knowledge)

The Constraints File

- The data used by an operator to override the RPKI repository data comes from a constraints file
- This file contains certificates that the RP wants to trust, no matter what the rest of the RPKI says
- It also contains parameters that can be substituted for other fields in a certificate, e.g., a new expiration date
- In the purely local case, an operator manages its own constraints file
- For national protection, a national authority could provide constraints file to operators in its jurisdiction

Constraints File Example

```
PRIVATEKEYMETHOD    <pointer to the RP private key>
TACERTIFICATE       <filename of TA certificate>
CONTROL             <optional flags to control tree processing>
TAG                 <up to 4 optional lines used to change
                     validity dates, CRL distribution points,
                     certificate policy, and the AIA extension>
SKI 00:12:33:44:00:BA:BA:DE:EB:EE:00:99:88:77:66:55:44:33:22:11
    IPv4
        10/8
    IPv6
     2001:DB8::/32
    AS#
     64496
SKI 29:42:83:74:61:EA:CA:1E:E3:CE:01:93:80:78:61:52:45:32:25:16
    IPv4
        172.16/16
    AS#
     65551
```

Using the Constraints File

- The constraints file is used to reissue targeted certificates under the local TA, modifying them as needed
- If any certificate is reissued, its ancestors also have to be reissued, to prevent conflicts in data imported from the RPKI repository system
- Thus, if a targeted certificate is low in the RPKI hierarchy, more parent certificates will have to be modified to accommodate it’s rehoming

Other Processing Details

- It is necessary to ensure that no other certificate anywhere in the RPKI hierarchy interferes with the certificates modified via local processing
- Thus the LTAM algorithm searches the whole RPKI tree looking for certificates that conflict with the targeted certificates
- If it finds any, it “fixes” them!
- In the end, all targeted certificates and their ancestors are re-issued under the local TA
- Certificates that are not targeted, and are not ancestors of targeted certificates are unaffected

Certificate Expiration

- The constraints file allows the RP to specify notBefore and notAfter for all para-certificates
    - This is a global rewrite rule, not a per-certificate rewrite rule
- As a result, expiration of the original certificate need not imply that the reissued certificate expires at the same time

Yes, there is Software!

- BBN's open source (BSD 3-clause license) certificate validation software for RPs (RPSTIR) incorporates a beta version of LTAM
  - RPSTIR is available for 32-bit Linux (Fedora, Ubuntu, CentOS, etc.), FreeBSD, NetBSD, and OpenBSD
  - http://sourceforge.net/projects/rpstir/
- LTAM is not yet a standard, so details may change, and this software will change to match whatever is approved as an RFC
- Feedback on RPSTIR and LTAM is solicited

QUESTIONS ?