Routing Policy System Status

Curtis Villamizar (ANS)
Cengiz Alaettinoglu (ISI)

- RPS Documents
- IRR Coding
- IRR New Code Deployment

RPS Document Status

1. **r**ps-auth: This is an Internet-Draft issued as a -00 version and presented to the IETF RPS WG. The -01 version reflects comments since then.

2. **r**ps-dist: The -00-preview version needs some editing to reflect comments from coauthors. The preview is at http://engr.ans.net/rps-auth

3. **r**ps-query: This document has not been written yet. It will define a query interface.

rps-auth: Routing Policy System Security

- The Routing Policy System Security draft was designed to provide an authentication and authorization model.

- Consideration of how the data is entered and used and the business and political relationships are reflected in the authorization model. The model is general though very specific operational conditions and procedures have been considered as a sanity check.

- Deployment, including ongoing support for existing data has been carefully consdidered.

- Database changes are simple. Operational changes are minor. Though simple, these changes provide a sufficient authentication and authorization model to address accidental or malicious router misconfiguration, and eliminate the use of unallocated address space.

rps-dist: Distributed Routing Policy System

- Scaling is not limited by the number of repositories
- Scales according to the size and rate of change of the entire set of routing registries.
- Standardization of the interface between repositories enables and encourages independent implementations.
- Untrustworthy or incompetent repositories are acco-modated in the model by repeating authorization and authentication checks.

rps-query: Routing Policy System Query Interface

This will be a new document. Jake Khuon will extract the current interface definition from the comments and/or code of IRRd and/or rawhoisd. David or Cengiz will describe the thoughts or discussions regarding query interface that have gone on within ISI. Someone will have to propose an initial query syntax with extension mechanism which we can then try to get agreement on. At that point we can issue a draft providing a least common denominator interface that any client can expect to be implemented by the server code from ISI or Merit. We can also discuss what the interface should be if we were to design it from scratch and whether it would be significantly different.

IRR Coding

Coding phases of the transition are taken from notes from a April 2 meeting with Cengiz, David Kessens, Bill Norton, Jake Kahun, and Jaoa (RIPE).

0. Separate the RIPE IP address and AS allocation from route and aut-num policy.
1. RPSL transition based on ISI modified RIPE code base.
2. Implement pgp-key authentication.
3. Implement the rps-auth authorization changes.
4. Real time mirroring modified to use formats in irr-dist.
5. C/C++ implementation of the registry code.
6. No further develpment of the perl code base.

Separate the RIPE IP address and AS allocation

- RIPE does not want disruption of it's most critical role of allocating IP addresses and AS numbers.
- Separation will be accomplished with the existing code used to support a RIPE–181 and RPSL database during transition.
- The existing RIPE code base will run the IP addresses and AS numbers allocation.
- Aut-nums needing RPSL capability will be submitted to the new database (on another port of the same server).
- Route objects will be submitted to the new database.
- This will be done by David Kessens and Jaoa.

RPSL transition based on ISI modified RIPE code base.

- ISI has modified the RIPE code to support RPSL.
- Merit will use the RIPE code and IRRd.
- Merit is being encouraged to use the ISI modified RIPE code if there is any chance that their use of IRRd will slow the RPSL transition.
- ANS to shortly begin regression test of configuration generation using ripe-dbase+rpsl-3.0.0a13.

Implement pgp-key authentication.

The biggest problem here is the US export restrictions. If someone in another country doesn't voluntarily write this, then an application for export of the hooks needed for PGP authorization may be required.

Implement the rps-auth authorization changes.

- New objects:
  1. repository
  2. as-block
- New attributes:
  1. delegated
  2. reclaim and no-reclaim
  3. referral-by and referral-in
  4. mnt-routes
- New semantics:
  1. scoping rules for objects with no hierarchy
  2. traversal rules for objects with hierarchy
  3. specific hierarchy semantics for “inetnum”, “as-block” and ’aut-num”, and “route” objects.

Real time mirroring modified to use formats in irr-dist

1. RIPE near real time mirroring will be modified to use the interchange formats described in irr-dist.

2. This will enable repositories using the perl code base to feed properly authorized data to a set of distributed repositiories.

3. The RIPE perl code will never be able to repeat authorization checks made in other databases due to lack of support to retrieve prior versions of objects.

4. This is a transition phase.

C/C++ implementation of the registry code

1. ISI will provide a C/C++ implementation of the registry code.
2. This will interoperate with perl based repositories through the flooding and exchange format specified in rps-dist.
3. The C/C++ code can repeat authorization and authentication checks made in any remote repository.
4. The perl code cannot repeat authorization checks but can otherwise fully function.
5. Merit may (at their own expense or through funding other than that designated to operate the RADB) provide the same functionality in IRRd.

No further develpment of the perl code base

1. Existing registries will be **s**trongly encouraged to discontinue use of the perl code base.

2. Providers who have unique requirements or wish to support highly customized repository code or their own implementation are free to do so.

3. Any independently implemented repositories are only restricted to provide RPSL as an external format, adhere to the authorization restrictions in rps-auth and use the rps-dist exchange formats.

4. Lightweight mirrors are described in rps-dist as a means to support very efficient local query including the possibility of a repository mirror on a router.

Summary – Documents and Coding

- RPS documents

  1. rps-auth: Routing Policy System Security
  2. rps-dist: Distributed Routing Policy System
  3. rps-query: Routing Policy System Query Interface

- RPS Coding

  0. Separate RIPE IP address and AS from route and policy.
  1. RPSL transition based on ISI modified RIPE code base.
  2. Implement pgp-key authentication.
  3. Implement the rps-auth authorization changes.
  4. Real time mirroring modified to use formats in irr-dist.
  5. C/C++ implementation of the registry code.
  6. No further develpment of the perl code base.

IRR New Code Deployment

- RPSL Phase II
  - RIPE has set up rpslii.ripe.edu to separate RPSL transition from the production number registry.
  - Merit has set up RPSL mirror of RADB and RIPE at rpsl.merit.edu based on ripe-dbase+rpsl and IRRd.
  - MCI is integrating ripe-dbase+rpsl with their internal tools.
  - ANS is integrating ripe-dbase+rpsl and RaToolSet-4 with ANS tools (nearly done).
  - APNIC will be bringing up an RPSL server.
  - Status of CANET (others?) is unknown.
- Latest Versions
  - ripe-dbase+rpsl-3.0a13
  - RaToolSet-3.5.7 (stable)
  - RaToolSet-4.0.2 (limited beta)
  - IRRd 1.4.7 (binary only)