Routing Registry Tutorial

NANOG 51

Jan 30 – Feb 2, 2011

Miami, Fl

Overview

- Topics to be covered
  - Short historical review
  - Overview of the RPSL and the IRR
  - Why you should use a routing registry
  - Querying IRR's
  - Aut-num policy example
  - Sampling of RPSL tools

Historical Context

- The basic concept of routing registries dates back to the 1980's and NSFNet
- A high-level policy based routing database (PRDB) was used to generate configs
- NSFNet regional networks were required to submit Network Announcement Change Requests (NACR) to update the PRDB
- NACR’s documented connected networks and their Autonomous System numbers

Sample NSFNet NACR

```
netnum:         35.0.0.0
netname:        MERIT-NET
netcc:          US
orgname:        Merit Network Inc.
orgaddr:        1071 Beal Ave.
orgcity:        Ann Arbor
orgstate:       MI
orgzip:         48109
orgcc:          US
orgtype:        N
bbone:          T3
homeas:         177
aslist:         233 237
aup:            N
action:         A
comment:
```

Note classful network

AS used to originate route

Upstream AS numbers

Early European work

- RIPE – Reseaux IP Europeens
- Formed in 1989 to coordinate and promote IP networking in Europe
- Developed a registry for allocation of IP addresses and Autonomous System numbers in Europe (first RIR)
- No routing policy support initially

Initial RIPE routing policy support

- RIPE-81 document was published in Feb., 1993 - extended the RIPE address registry to include basic routing policy information
- Added ability to specify an Autonomous System number for an IP address allocation
- Also allowed the expression of Autonomous System relationships

RIPE-181

- RIPE-181 (RIPE-81++) document was published in Oct, 1994
- Introduced concept of object classes
- Separated routing policy information from IP address allocation information with introduction of the “route” object
- Extended Autonomous System policy expression functionality
- Also adopted a mechanism for grouping Autonomous Systems with the “as-macro”

Sample RIPE-181 route object

```
route:      192.87.45.0/24
descr:      RIPE Network Coordination Centre
origin:     AS3333
comm-list:  SURFNET
changed:    dfk@ripe.net 940427
source:     RIPE
```

Note CIDR format adopted

Sample RIPE-181 aut-num object

```
aut-num: AS1104
descr:   NIKHEF-H Autonomous system
as-in:   from AS1213 100 accept AS1213
as-in:   from AS1913 100 accept AS1913
as-in:   from AS1755 150 accept ANY
as-out:  to AS1213 announce ANY
as-out:  to AS1913 announce ANY
as-out:  to AS1755 announce AS1104 AS1913 AS1213
tech-c:  Rob Blokzijl
admin-c: Eric Wassenaar
guardian: as-guardian@nikhef.nl
changed: ripe-dbm@ripe.net 920910
source:  RIPE
```

Sample RIPE-181 as-macro object

```
as-macro: AS-EBONE
descr:    ASes routed by EBONE
as-list:  AS2121 AS1104 AS2600 AS2122
as-list:  AS1103 AS1755 AS2043
guardian: guardian@ebone.net
   ......
```

RPSL

- In March 1995, the RIPE-181 standard was accepted as an IETF informational document -- RFC 1786
- IETF created the Routing Policy System Working Group to revise and standardize the language under the auspices of the IETF
- Result was known as the Routing Policy Specification Language (RPSL)

RFC 2622

- RFC 2622 was released in June, 1999 and formally defined RPSL standard
- Based on the RIPE-181 standard
  - Significantly extended the functionality of the aut-num object
  - route object also extended
  - as-macro became as-set object
  - Added a number of new object types
  - Included a dictionary based extension mechanism

New object types introduced in RFC 2622

- as-set
- route-set
- filter-set
- rtr-set
- peering-set
- inet-rtr
- mntner, role, and person objects for authentication and contact information

RPSL basics

- Each object type (class) contains mandatory and optional attributes
- All objects must have these attributes
  - mnt-by: identifies mntner object that controls the object
  - changed: lists email and time of change
  - source: identifies the registry name where the object is located

mntner object class

- Mntner is an abbreviation of maintainer
- Identifies accounts in the registry
- Specifies authentication mechanism in the “auth” attribute
  - CRYPT-PW or MD5-PW - password auth
  - PGP-KEY – PGP/GPG based auth
  - MAIL-FROM – email based auth

Sample mntner object

```
mntner:     MAINT-AS23323
descr:      Diablo Valley College
admin-c:    Ben Seaberry
tech-c:     Ben Seaberry
upd-to:     bseaberry@DVC.EDU
auth:       CRYPT-PW HIDDENCRYPTPW
notify:     bseaberry@DVC.EDU
mnt-by:     MAINT-AS23323
changed:    rick@extrateam.com 20030311
source:     RADB
```

route object class

- Defines a CIDR prefix and origin AS
- Most common type of object found in routing registries
- Used by a number of ISP's to generate filters on their customer BGP sessions
  - Customers must register all routes in order for their ISP to route them
  - Allows automation of adding new prefixes

Sample route object

```
route:    198.108.0.0/14
descr:    MERIT Network Inc.
          1000 Oakbrook Drive, Suite 200
          Ann Arbor
          MI 48104, US
origin:   AS237
mnt-by:   MAINT-AS237
changed:  ljb@merit.edu 20060919
source:   RADB
```

route object class and keys

- Every RPSL class has a primary “key”
- For most classes, it is simply the main class attribute value
- For example, the mntner class uses the mntner attribute value as the key
- However, route objects use both route and origin fields as the primary key

route object class key (con'd)

- There can be multiple objects for the same prefix with different origins
- This is by design
  - Multi-origin multi-homing
  - When changing to a new origin AS, want routes for both until switched
- However, also many cases of multiples due to stale routes not being cleaned

route object class key example

- The following shows 2 distinct objects
  - Note that routes have different origins

```
route:      158.80.0.0/21
descr:      Baker College
origin:     AS237
mnt-by:     MAINT-AS237
changed:    ljb@merit.edu 20100302  #19:19:56Z
source:     RADB

route:      158.80.0.0/21
descr:      Baker College
origin:     AS20379
mnt-by:     MAINT-AS237
changed:    har@merit.edu 20040916
source:     RADB
```

route6 object class

- Like route object but for IPv6 prefixes
- Defined in RFC4012 RPSLng RPSL extensions for IPv6 and Multicast
- Initial usage was slow
- However, use has been picking up
  - Now over 2000 route6 objects in RIPE registry
  - Over 1000 in the RADB registry

Sample route6 object

```
route6:      2604:AA00::/32
descr:       Lakefield Communications
origin:      AS14159
mnt-by:      MAINT-AS11796
changed:     yach@wins.net 20101208  #16:46:07Z
source:      RADB
```

aut-num object class

- Defines routing policy for an AS
  - Uses import: and export: attributes to specify policy
  - Can be used for highly detailed policy descriptions and automated config generation
  - Can reference other registry objects such as as-sets, route-sets, and filter-sets

Sample aut-num object

```
aut-num: AS52
as-name: UCLA
descr:   University of California, Los Angeles
import:  from AS11422
         accept ANY
import:  from AS2153
         accept ANY
import:  from AS2152
         accept ANY
export:  to AS11422
         announce AS52
export:  to AS2152
          announce AS52
export:  to AS2153
         announce AS52
    .....
```

Some use remarks in aut-num to document communities

```
aut-num:    AS209
descr:      Qwest Communications
   .....
remarks:    ============================================
remarks:    Qwest BGP Local Preference
remarks:    --------------------------------------------
remarks:    Customer default = 100
remarks:    Peer default     = 80
remarks:    ============================================
remarks:    Communities allowed from customers to alter
remarks:    default local preference
remarks:    --------------------------------------------
remarks:    209:90 = Set Local Pref to 90
remarks:    209:80 = Set Local Pref to 80
remarks:    209:70 = Set Local Pref to 70
   .....
```

as-set object class

- Provides a way of grouping AS'es
- Name must begin with prefix “AS-”
- Frequently used to list downstream/customer AS numbers
- Maybe referenced in aut-num import/export policy expressions
- Can reference other as-set's

Sample as-set object

```
as-set: AS-VERIZON
descr:  --------------------------------
        Verizon Internet Services (VIS)
        1880 Campus Commons Drive
        Reston, VA 20191
        --------------------------------
        All AS Announcements from VIS
        --------------------------------
members: AS6350, AS6995, AS7192, AS7021, AS7193, AS8016,
    AS8017, AS8112, AS8113, AS8114, AS8115, AS10719, AS11145,
    AS11146, AS11147, AS4390, AS11279, AS11149, AS20089,
    AS19997, AS268, AS568, AS7925, AS11768, AS11148, AS3783,
    AS13661, AS13387, AS13662, AS295, AS11696, AS11094,
    AS3778, AS2576, AS6485, AS12235, AS8071, AS13673,
    AS14896, AS15308, AS8076, AS2929, AS10448, AS7089,
    AS12065, AS6372, AS13661, AS13662, AS13663, AS13664,
...
```

route-set object class

- Defines a set of routes prefixes
- Name must begin with prefix “RS-”
- Can reference other route-sets
- Can also reference AS's or as-set's
  - In this case, the route-set will include all route object prefixes which have an origin which matches the AS numbers

Sample route-set object

```
route-set:   RS-SOFTLAYER-DAL05
members:     50.22.0.0/18,
             173.192.64.0/18,
             67.228.252.0/23,
             67.228.254.0/23
mp-members:  2607:F0D0:1100:0000:0000:0000:0000:0000/40
descr:       SOFTLAYER-DAL05
mnt-by:      MAINT-AS36351
changed:     ipadmin@softlayer.com 20101124  #16:36:36Z
source:      RADB
```

filter-set object class

- Defines a set of routes that are matched by a filter expression
- Similar in concept to route-set's
- Name must begin with prefix “fltr-”

Sample filter-set object

```
filter-set: fltr-unallocated
descr:      Unallocated (by IANA) IPv4 prefixes.
filter:     {39.0.0.0/8^+,
             102.0.0.0/8^+,
             103.0.0.0/8^+,
             104.0.0.0/8^+,
             106.0.0.0/8^+,
             179.0.0.0/8^+,
             185.0.0.0/8^+}
admin-c:    Rob Thomas RT624
tech-c:     Rob Thomas RT624
   .........
```

Additional RPSL IETF documents

- RFC-2650: Using RPSL in Practice
- RFC-2725: Routing Policy System Security
- RFC-2726: PGP Authentication for RIPE Database Updates
- RFC-2769: Routing Policy System Replication
- RFC-4012: RPSLng – RPSL extensions for IPv6 and Multicast

A word about 32-bit ASN's

- RFC4893 defines 32-bit AS number support in BGP
- RPSL specs did not really define ASN size
  - Some implementations enforced 16bit
- RFC5396 standardized representation
  - asplain format uses simple integers
- Most RPSL implementations and routing registries now support 32-bit ASN's

The IRR

- Concept of “the” Internet Routing Registry system established in 1995
- Shares information regarding production Internet Routing Registries
- Web site at http://www.irr.net
- Initially RIPE-181 format, shifted to RPSL
- Mirror Routing Registry data in a common repository for simplified queries

The IRR (con'd)

- The IRR currently consists of roughly 35 operational registries
- Registries operators
  - Regional Internet Registers (RIR's), such as ARIN, RIPE, and APNIC
  - ISP's - SAVVIS, NTT, Level3
  - Non-affiliated public registries – RADB and ALTDB

RADB Routing Registry

- The RADB launched in 1995 as part of NSFNet funded Routing Arbiter project
- The Routing Arbiter project was intended to ease transistion from the NSFNet to the commercial Internet
- Registry was used to configure Route Servers located at designated Network Access Points (NAP's) located in Chicago, Washington, New York, and San Francisco

RADB (con’d)

- RADB transitioned from public NSFNet funding to fee-based model in 1999
- Re-branded Routing Assets Database in 2002 – http://www.radb.net
- The registry can be queried at website and via whois at whois.radb.net
- This server also mirrors the other registries in the IRR as documented at www.irr.net

Why Register?

- Document routing policy
  - In particular, register route objects to associate network prefixes with origin AS
- A number of transit providers require their customers to register routes and filter customer route announcements based on registry contents
- Filters unauthorized announcements to prevent route hijacking, denial of service

Querying the IRR

- Historically, IRR's have used the “whois” protocol
- Basically – a TCP connection on port 43
- Two primary IRR server implementations
  - RIPE DB Server from RIPE NCC
  - IRRd server from Merit
- A number of IRR's offer web based query front-end interfaces and some provide a RESTful interface

Common IRR query flags

- IRR's support a number flag options
- -i flag performs inverse query
  - “-i origin AS237” returns all route objects with an origin of AS237
  - “-i mnt-by MAINT-AS237” returns all routes maintained by MAINT-AS237
- -M flag returns more specific route objects for a prefix
  - “-M 198.108.0.0/14” returns all more specific route objects in the 198.108.0.0/14 prefix

Other IRR query flags

- -s flag limits number of sources queried
  - May not want to query all 30+ IRR db's
  - example, “-s RADB,RIPE”
- -K flag – return primary keys only
  - Useful for route object queries, excludes extraneous fields not needed for policy
  - Often used by tools
- E.g. “whois -h whois.radb.net -- -K 198.108.0.0/14”

route: **198.108.0.0/14**
origin: **AS237**

Advanced IRR queries

- IRRd provides the ability to perform server side set expansions (as-set and route-set)
- This is done with the “!i” query
  - “!iAS-ESNETUS” returns members of AS-ESNETUS as-set object
- Add a “,1” for a recursive expansions
  - “!iAS-ESNETUS,1” will recurse any as-set members and return individual as-members
  - Reduces number of queries to server

More on “!” queries

- Primarily intended for tools use
- Concise output limited to policy related info
- Can also be used from whois command or by telnet – “telnet whois.radb.net 23”
- Remember to escape “!” for Unix
- answer length for tools in initial 'A' line

```
$ whois -h whois.radb.net \!ias-michnet
A171
AS10423 AS11206 AS13325 AS1432 AS19106 AS20379 AS22251 AS236 AS237
AS25609 AS25773 AS27274 AS30154 AS32285 AS33272 AS35874 AS36375 AS39987
AS40044 AS46119 AS46970 AS53263
C
```

Additional “!” queries

- “!!” used to keep connection open for multiple query commands (useful when telnetting to server)
- “!s” used to specify sources (like -s flag)
- “!g” used for inverse query by origin AS (like -i origin flag), returns a list of prefixes only
- “!6” is like !g, but for route6: object prefixes
- More info in IRR User Manual - www.irrd.net

Example dialog with “!” commands

```
$ telnet whois.radb.net 43
Trying 198.108.0.18...
Connected to whois.radb.net.
Escape character is '^]'.
!!
!sRADB
C
!6AS237
A15
2001:48A8::/32
C
quit
```

Advanced query tip

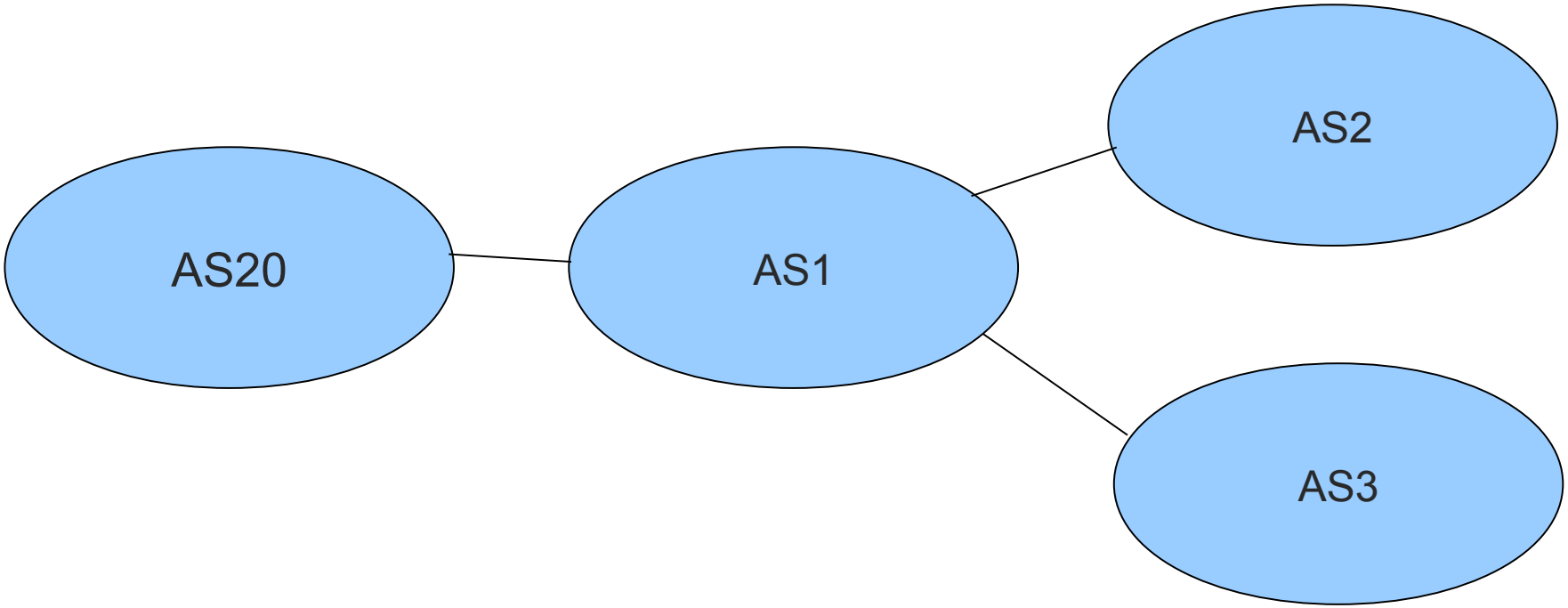
- Route-set's may reference AS numbers or as-set's in addition to prefixes (see below)
- When expanding the route-set, it will include all the prefixes originated by those AS's
- !iRS-MICHNET,1 will generate a list of all prefixes originated by AS numbers in AS-MICHNET in a single query

```
route-set:   RS-MICHNET
members:     AS-MICHNET
```

Advanced RPSL – aut-num

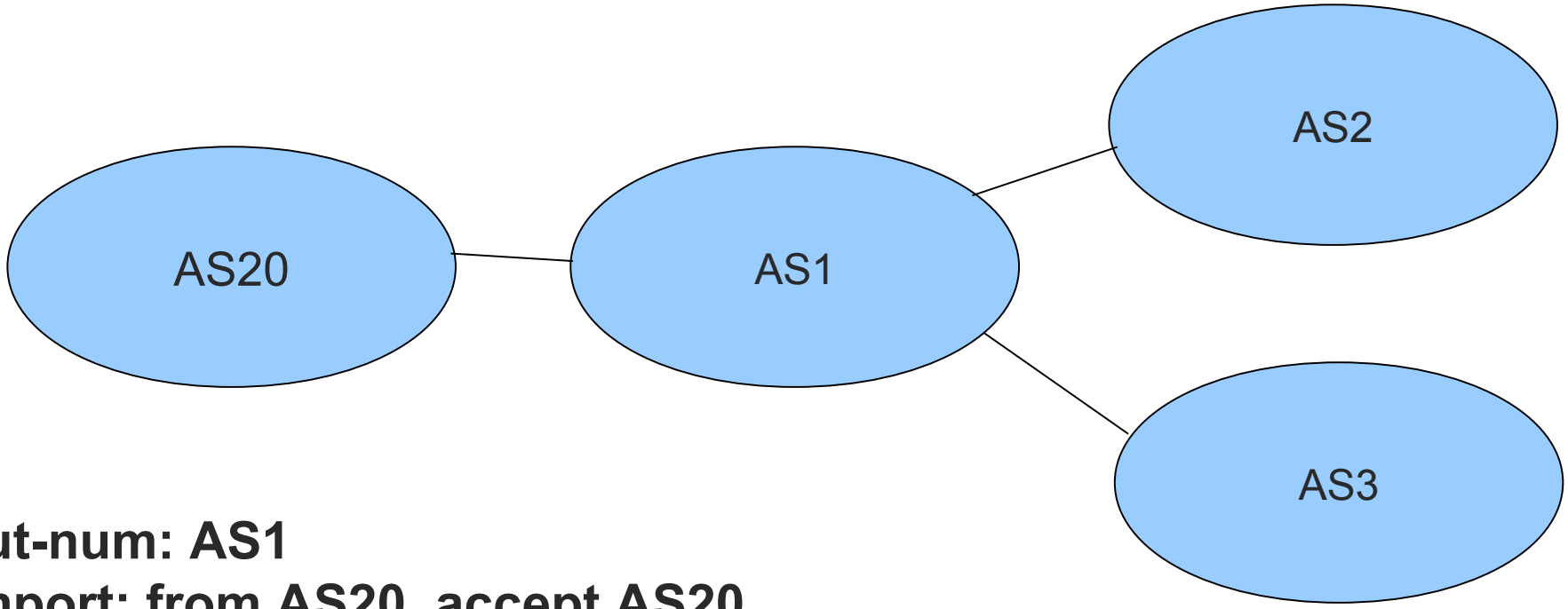
- The aut-num object can be used to express an Autonomous System's routing policy and peering information
- Powerful structured syntax allows for complex policy expressions
- Some operators drive their network configuration off of their RPSL data
- Others simply use it to document AS relationships in a public manner

aut-num RPSL example



AS1 provides transit to AS2 and AS3
AS1 provides local routes to AS20

aut-num RPSL example



aut-num: AS1
import: from AS20 accept AS20
import: from AS2 accept AS2
import: from AS3 accept AS3
export: to AS20 announce AS1
export: to AS2 announce ANY
export: to AS3 announce ANY

aut-num expression tip

The keyword PeerAS can be used instead of the AS number of the peer AS. PeerAS is particularly useful when the peering is specified using an AS expression. For example:

```
as-set: AS-MY-CUSTOMERS
members: AS2, AS3, AS4

aut-num: AS1
import: from AS-MY-CUSTOMERS accept PeerAS
```

is same as:

```
aut-num: AS1
import: from AS2 accept AS2
import: from AS3 accept AS3
import: from AS4 accept AS4
```

RPSL Tools

- Several tools have been developed to facilitate the use of RPSL registry data in the configuration of networks
- Tools range from sophisticated and powerful to simple and limited
- Use the IRR by querying over the whois protocol
- Some ISP's use in-house developed tools which process RPSL database files directly

Sample of RPSL Tools

- IRRToolSet
- NET::IRR
  - Perl module supporting basic IRR queries
- IRR Power Tools
  - IRR based router configuration – PHP + CVS
- Rpsltool – generates cisco configs - Perl

IRRToolSet

- Based on original RAToolSet used in NSF Routing Arbiter project
- Written in C++ and now maintainer by ISC
- rtconfig tool uses templates to generate router configs from IRR data
- Other provided tools include
  - peval – low level policy evaluation tools
  - rpslcheck – verfies RPSL syntax of objects

IRRToolSet Cisco Example

- The following generates a Cisco prefix list

```
$ rtconfig -cisco_use_prefix_lists
rtconfig> @RtConfig access_list filter AS38
!
no ip prefix-list pl100
ip prefix-list pl100 permit 72.36.64.0/18
ip prefix-list pl100 permit 128.174.0.0/16
ip prefix-list pl100 permit 130.126.0.0/16
ip prefix-list pl100 permit 192.17.0.0/16
ip prefix-list pl100 permit 192.17.8.0/24
ip prefix-list pl100 deny 0.0.0.0/0 le 32
```

IRRToolSet Junos Example

- The following generates a Junos prefix list

```
$ rtconfig -config junos
rtconfig> @RtConfig access_list filter AS38
   policy-statement prefix-list-100 {
      term prefixes {
         from {
            route-filter 72.36.64.0/18 exact accept;
            route-filter 128.174.0.0/16 exact accept;
            route-filter 130.126.0.0/16 exact accept;
            route-filter 192.17.0.0/16 exact accept;
            route-filter 192.17.8.0/24 exact accept;
         }
      }
      term catch-rest {
          then reject;
      }
   }
```

NET::IRR

- Perl CPAN module
- Provide several useful Perl functions
  - get_routes_by_origin
  - get_ipv6_routes_by origin
  - get_as_set
  - get_route_set
  - route_search

IRR Power Tools

- PHP based toolset
  - sourceforge.net/projects/irrpt
- Allows ISP to easily track, manage and utilize IRR data
- Performs tracking with CVS
- Can email notifications of updates
- irrpt_pfxgen script can generate router configs in Cisco/Foundry, Juniper, Extreme, and Force10 formats

IRR Power Tools Example

```
$ ./irrpt_pfxgen 8001
conf t
no ip prefix-list CUSTOMER:8001
ip prefix-list CUSTOMER:8001 permit 4.17.225.0/24
ip prefix-list CUSTOMER:8001 permit 4.17.226.0/23 le 24
ip prefix-list CUSTOMER:8001 permit 4.17.251.0/24
ip prefix-list CUSTOMER:8001 permit 4.17.252.0/23 le 24
$ ./irrpt_pfxgen -f juniper 8001
policy-options {
replace: policy-statement CUSTOMER:8001 {
term prefixes {
from {
route-filter 4.17.225.0/24 upto /24;
route-filter 4.17.226.0/23 upto /24;
```

Routing Registry Futures

- RPKI work will likely have impact on routing registry usage
- While RPKI ROA's largely obviate need for route objects, RPSL is still need to express other policy

Further Resources

- RFC2650 – Using RPSL in practice
- RIPE NCC Routing Registry Training Course Material

  http://www.ripe.net/training/material.html

Questions?

- Contact Info
  - mkarir@merit.edu, ljb@merit.edu