Routing Registry Function Automation using RPKI & RPSL

Target Audience

- Knowledge of Internet Routing (particularly BGP)
- Fair idea of Routing Policy
- Familiarity with any IRR Database
- No need to know Cryptography
- Basic knowledge of PKI (Public Key Infrastructure)

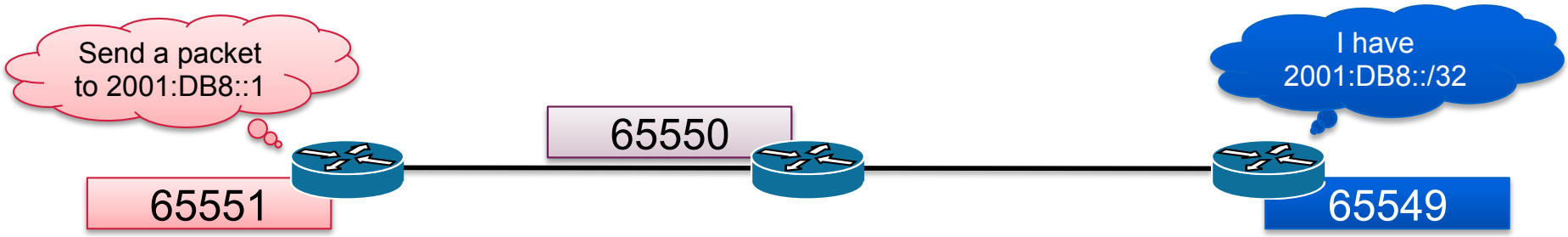
Agenda

- BGP 101
- Routing Policy
- RPSL
  - Internet Routing Registry
  - RPSL Objects and Routing Policy
- RPKI

BGP (AS)

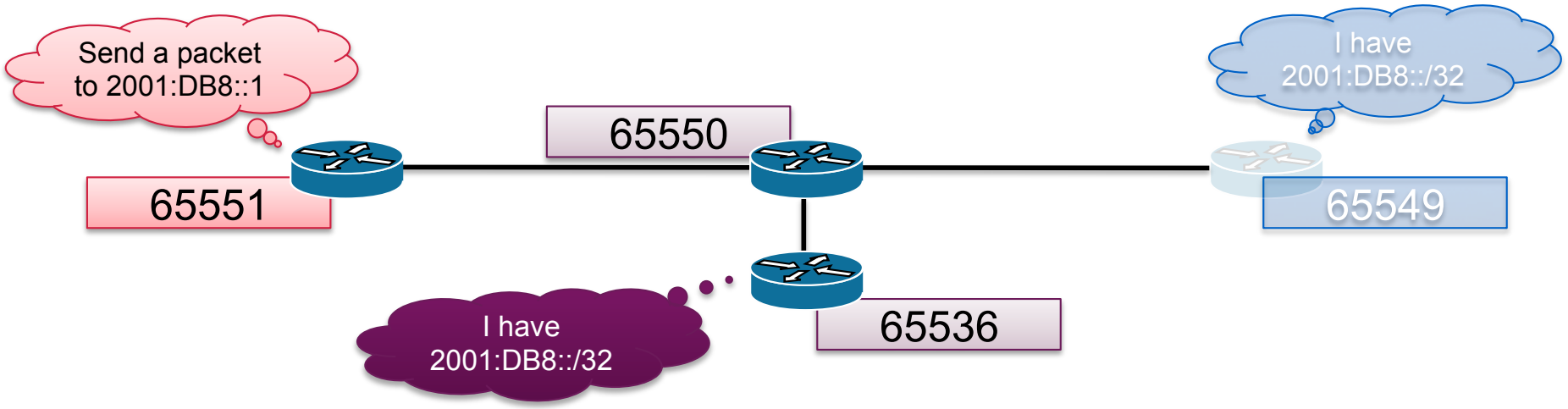
AS Path

2001:DB8::/32 65551 65550 65549 i



AS Path

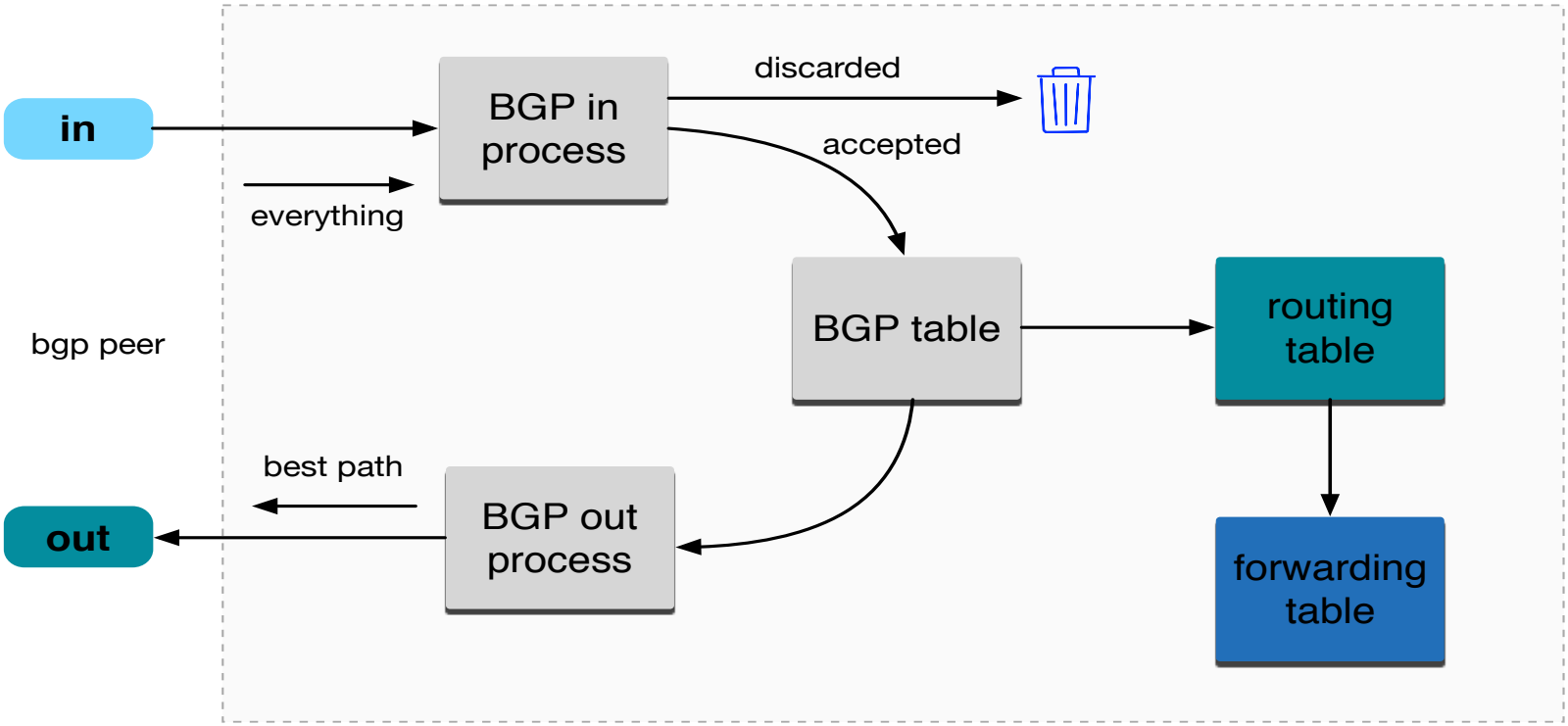
2001:DB8::/32	65551	65550	65549	i
2001:DB8::/48	65551	65550	65536	i



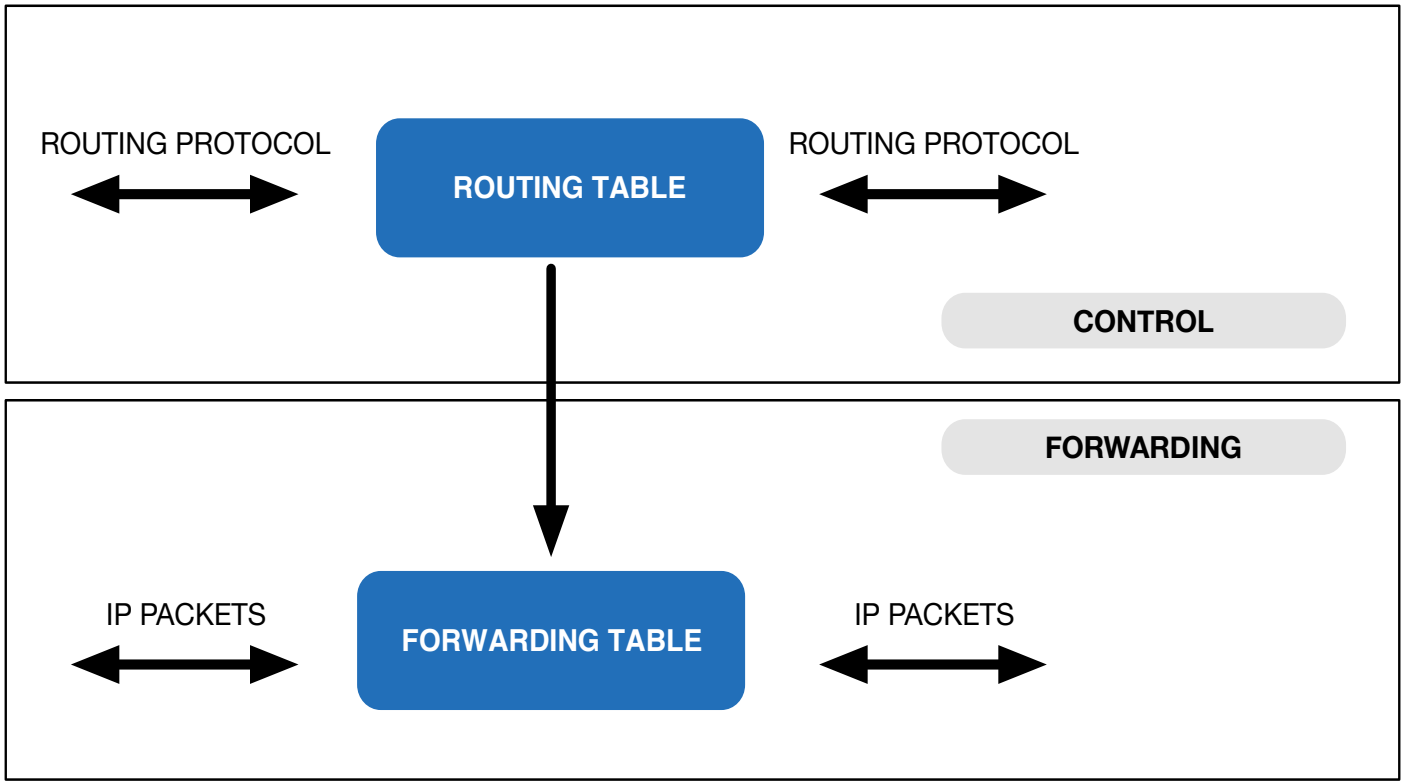
BGP Best Path Calculation

- Drop if own AS in AS-Path
- Prefer path with highest Weight
- Highest Local Preference
- Shortest AS-Path
- Lowest MED
- Path with shortest next hop metric (minimum IGP cost)
- Oldest received path
- Path from lowest neighbour address

Constructing the Forwarding Table



Control Plane and Forwarding Plane



Routing Incident Types

- Incidents
  - Misconfiguration
  - Malicious
  - Targeted Traffic Misdirection
- For theory of positivity, lets call all these as Mis-Origination
- Traffic Hijacking or Prefix Hijacking assumes a negative intent

Historical Incident

April 1997: The "AS 7007 incident" UU/Sprint for 2 days

February 24, 2008: Pakistan's attempt to block YouTube access within their country takes down YouTube entirely.[6]

November 11, 2008: The Brazilian ISP CTBC - Companhia de Telecomunicações do Brasil Central leaked their internal table into the global BGP table.

April 8, 2010: China Telecom originated 37,000 prefixes not belonging to them in 15 minutes, causing massive outage of services globally.

source : http://en.wikipedia.org/wiki/IP_hijacking

Securing Internet Routing

To Secure Internet Routing; we need to check:

A network should only originate his own prefix

1. How do we verify?
2. How do we avoid false advertisement?

A transit network should filter customer prefix

1. Check customer prefix and ASN delegation
2. Transitive trust

Routing Policy

- Public description of the relationship between external BGP peers
- Can also describe internal BGP peer relationship
- Usually registered at an IRR (Internet Routing Registry) such as RADB or APNIC

Routing Policy

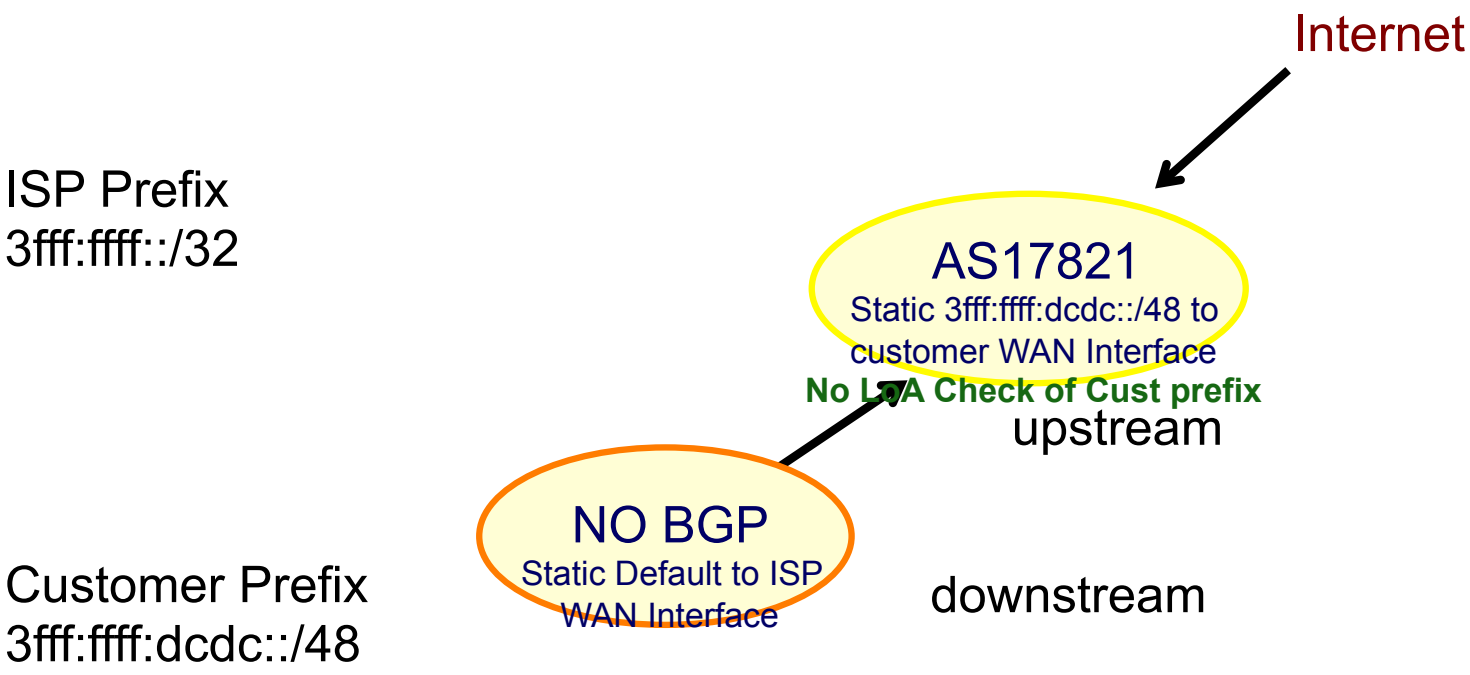
- Who are my BGP peers
- What routes are
  - Originated by a peer
  - Imported from each peer
  - Exported to each peer
  - Preferred when multiple routes exist
- What to do if no route exists

Prefix Advertise to Internet

- Ingress prefix from downstream:
  - Option 1: Customer single home and non portable prefix
    - Customer is not APNIC member prefix received from upstream ISP
  - Option 2: Customer single home and portable prefix
    - Customer is APNIC member receive allocation as service provider but no AS number yet
  - Option 3: Customer multihome and non portable prefix
    - Customer is not APNIC member both prefix and ASN received from upstream ISP
  - Option 4: Customer multihome and portable prefix
    - Customer is APNIC member both prefix and ASN received from APNIC

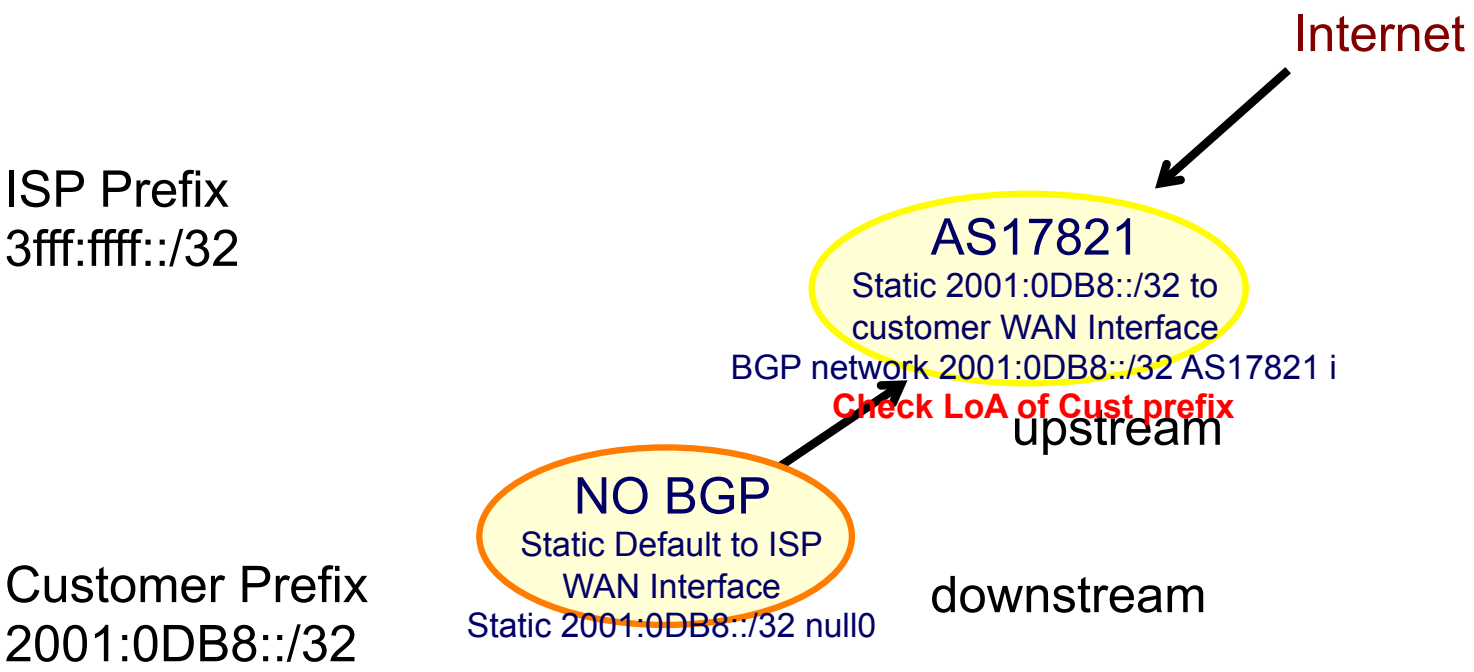
Prefix Filtering BCP [Single home]

- Option 1: Customer single home and non portable prefix



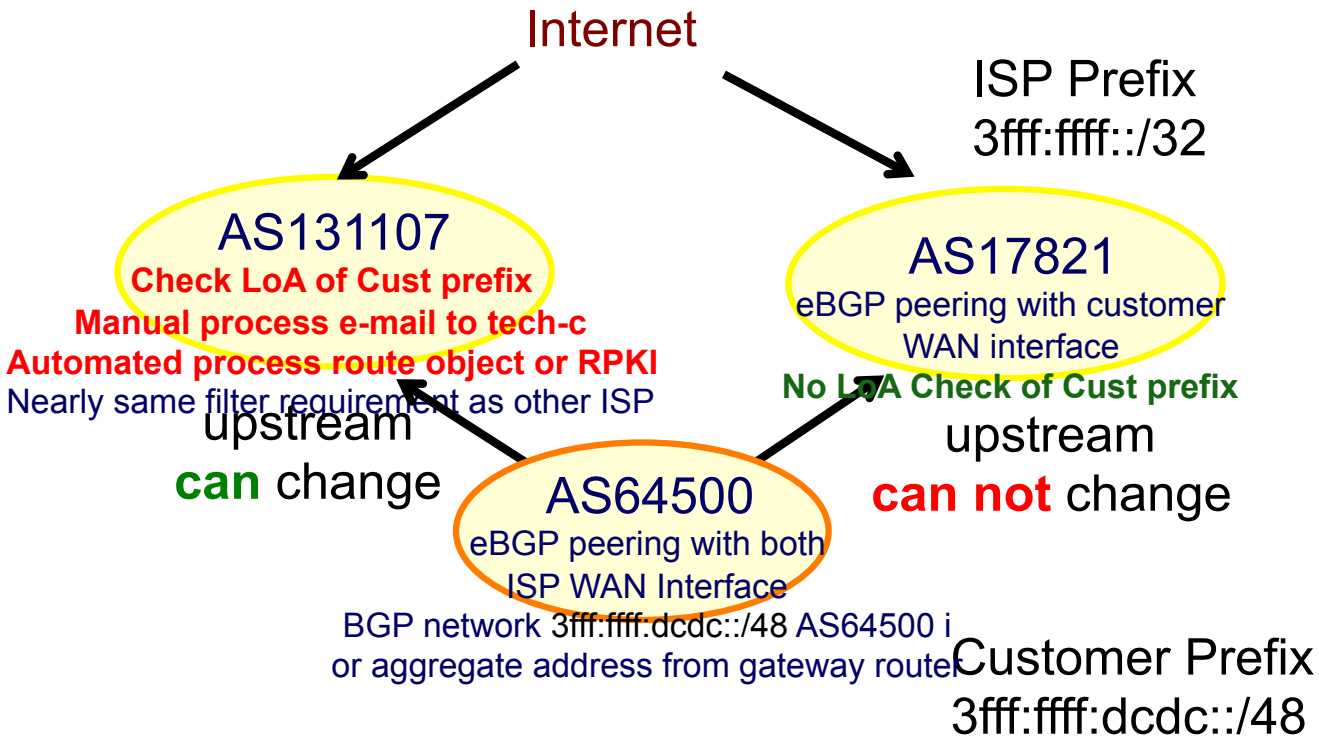
Prefix Filtering BCP [Single home]

- Option 2: : Customer single home and portable prefix



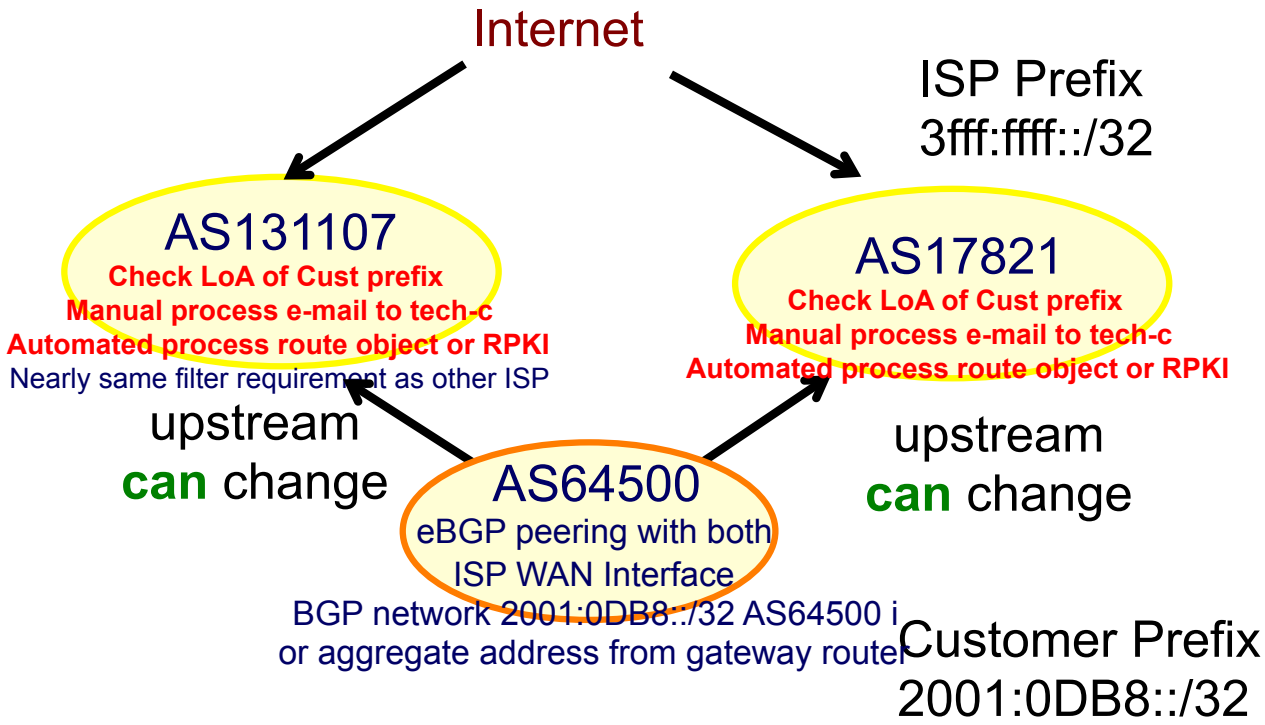
Prefix Filtering [Multihome]

- Option 3: Customer multihome and non portable prefix



Prefix Filtering [Multihome]

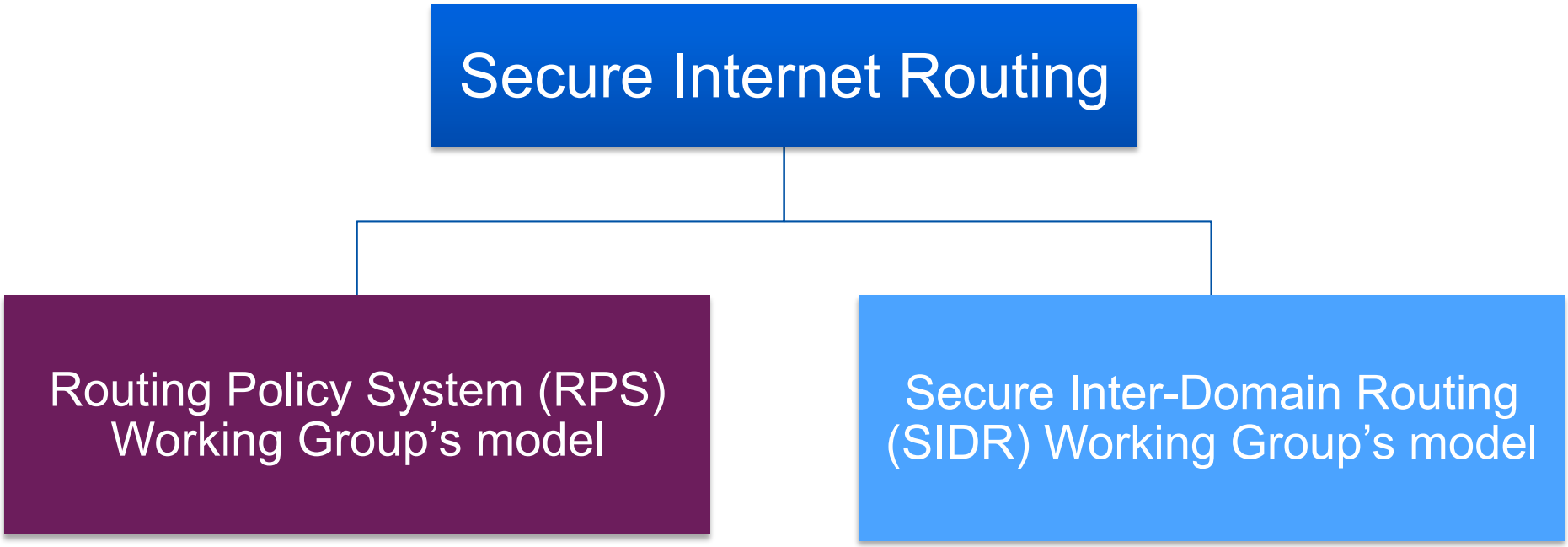
- Option 4: Customer multihome and portable prefix



Why define a Routing Policy

- Documentation
- Provides routing security
  - Can peer originate the route?
  - Can peer act as transit for the route?
- Allows automatic generation of router configurations
- Provides a debugging aid
  - Compare policy versus reality

Secure Internet Routing



RPSL & IRR

What is RPSL

- Routing Policy Specification Language
- RPSL is object oriented
  - These objects are registered in the Internet Routing Registry (IRR)
  - route, autonomous system, router, contact and set objects
- Describes things interesting to routing policy
  - Prefixes
  - AS Numbers
  - Relationships between BGP peers
  - Management responsibility

What is RPSL

- RIPE-81 was the first language deployed in the Internet for specifying routing policies
  - Later replaced by RIPE-181
  - RPSL is a replacement for the RIPE-181 or RFC-1786
  - RPSL addresses RIPE-181's limitations
- For more about RPSL
  - RFC-1786: RIPE-181
  - RFC-2622: Routing Policy Specification Language
  - RFC-2650: Using RPSL in Practice
  - RFC-2726: PGP Authentication for RIPE Database Updates
  - RFC-2725: Routing Policy System Security
  - RFC-2769: Routing Policy System Replication
  - RFC-4012: Routing Policy System Replication next generation

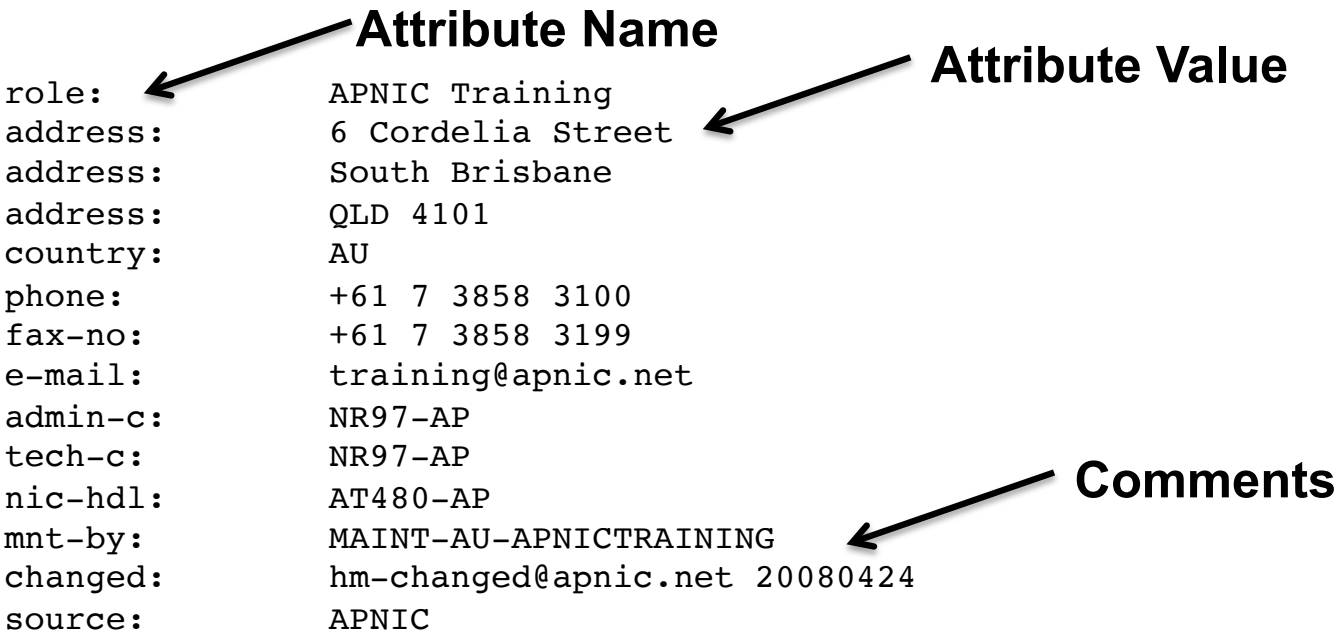
RPSL Objects

- RPSL objects are similar to RIPE-181 objects
- Objects
  - set of attributes
- Attributes
  - mandatory or optional
  - values: single, list, multiple
- Class “key”
  - set of attributes
  - usually one attribute has the same name as the object’s class
  - uniquely identify each object
- Class “key” = primary key
  - must be specified first

RPSL Attributes

- Case insensitive
- Value of an attribute has a type
  - <object-name>
  - <as-number>
  - <ipv4-address>
  - <ipv6-address>
  - <address-prefix>
  - etc
- Complete list of attributes and types in RFC 2622
  - https://www.rfc-editor.org/rfc/rfc2622.txt

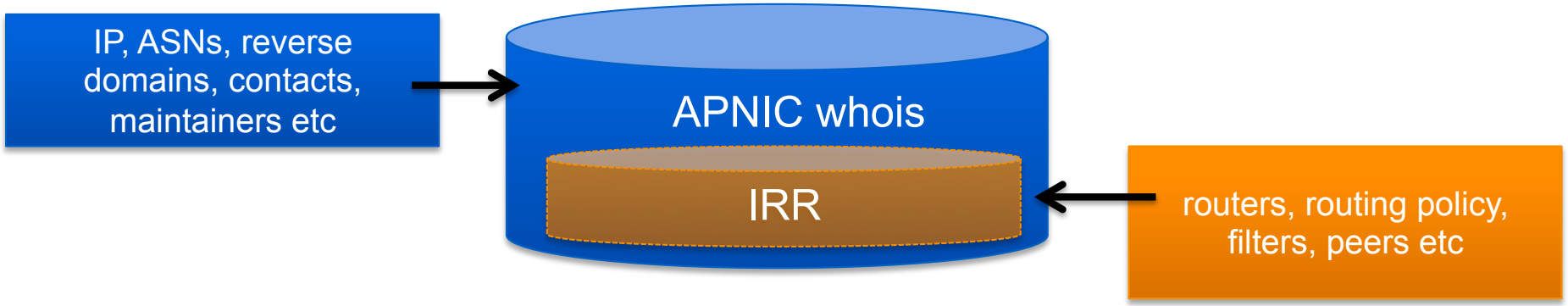
RPSL Objects



Complete list of attributes and types in RFC 2622
https://www.rfc-editor.org/rfc/rfc2622.txt

Integration of whois & IRR

- Integrated APNIC whois database & Internet Routing Registry



APNIC Database Objects and Routing Registry

OBJECT	PURPOSE
person	Technical or administrative contacts responsible for an object
role	Technical or administrative contacts represented by a role, performed by one or more people
inetnum	Allocation or assignment of IPv4 address space
inet6num	Allocation or assignment of IPv6 address space
aut-num	Registered holder of an AS number and corresponding routing policy
domain	in-addr.arpa (IPv4) or ip6.arpa (IPv6) reverse DNS delegations
route / route6	Single IPv4/IPv6 route injected into the Internet routing mesh
mntner	Authorized agent to make changes to an object
irt	Dedicated abuse handling team

person / role Object

- The Person object register contact information

```
person:          [mandatory]  [single]     [lookup key]
address:         [mandatory]  [multiple]   [ ]
country:         [mandatory]  [single]     [ ]
phone:           [mandatory]  [multiple]   [ ]
fax-no:          [optional]   [multiple]   [ ]
e-mail:          [mandatory]  [multiple]   [lookup key]
nic-hdl:         [mandatory]  [single]     [primary/look-up key]
remarks:         [optional]   [multiple]   [ ]
notify:          [optional]   [multiple]   [inverse key]
abuse-mailbox:   [optional]   [multiple]   [inverse key]
mnt-by:          [mandatory]  [multiple]   [inverse key]
changed:         [mandatory]  [multiple]   [ ]
source:          [mandatory]  [single]     [ ]
```

person / role Object

```
person:          Nelly Tan
address:         1000 Jalan Bukit Merah
country:         SG
phone:           +65 6400 7333
fax-no:          +65 6400 7334
e-mail:          nelly@abcinternet.com
nic-hdl:         NT324-AP
mnt-by:          MAINT-SG-ABCINTERNET
changed:         hm-changed@apnic.net 20160503
source:          APNIC
```

intenum / inetnum6 Object

- Contains details of an allocation or assignment of IPv4/IPv6 address space

```
inet6num:       [mandatory]  [single]     [primary/lookup key]
netname:        [mandatory]  [single]     [lookup key]
descr:          [mandatory]  [multiple]   [ ]
country:        [mandatory]  [multiple]   [ ]
geoloc:         [optional]   [single]     [ ]
language:       [optional]   [multiple]   [ ]
admin-c:        [mandatory]  [multiple]   [inverse key]
tech-c:         [mandatory]  [multiple]   [inverse key]
status:         [mandatory]  [single]     [ ]
remarks:        [optional]   [multiple]   [ ]
notify:         [optional]   [multiple]   [inverse key]
mnt-by:         [mandatory]  [multiple]   [inverse key]
mnt-lower:      [optional]   [multiple]   [inverse key]
mnt-routes:     [optional]   [multiple]   [inverse key]
mnt-irt:        [mandatory]  [single]     [inverse key]
changed:        [mandatory]  [multiple]   [ ]
source:         [mandatory]  [single]     [ ]
```

intenum / inetnum6 Object

```
inet6num:       2406:6400::/32
netname:        APNIC-TRAININGIPv6-Lab-AP
descr:          APNIC TRAINING Lab
country:        AU
admin-c:        AT480-AP
tech-c:         AT480-AP
mnt-by:         APNIC-HM
mnt-lower:      MAINT-AU-APNICTRAINING
mnt-routes:     MAINT-AU-APNICTRAINING
status:         ALLOCATED PORTABLE
remarks:        -+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
remarks:        To report network abuse, please contact the IRT
remarks:        For troubleshooting, please contact tech-c and admin-c
remarks:        For assistance, please contact the APNIC Helpdesk
remarks:        -+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
source:         APNIC
mnt-irt:        IRT-APNICTRAINING-AU
changed:        hm-changed@apnic.net 20100216
changed:        hm-changed@apnic.net 20100818
```

mntner Object

- Maintainer objects used for authentication
  - Multiple auth / mnt-by / mntner-s are OR-ed

```
mntner:          [mandatory]  [single]     [primary/lookup key]
descr:           [mandatory]  [multiple]   [ ]
country:         [optional]   [single]     [ ]
admin-c:         [mandatory]  [multiple]   [inverse key]
tech-c:          [optional]   [multiple]   [inverse key]
upd-to:          [mandatory]  [multiple]   [inverse key]
mnt-nfy:         [optional]   [multiple]   [inverse key]
auth:            [mandatory]  [multiple]   [inverse key]
remarks:         [optional]   [multiple]   [ ]
notify:          [optional]   [multiple]   [inverse key]
abuse-mailbox:   [optional]   [multiple]   [inverse key]
mnt-by:          [mandatory]  [multiple]   [inverse key]
referral-by:     [mandatory]  [single]     [inverse key]
changed:         [mandatory]  [multiple]   [ ]
source:          [mandatory]  [single]     [ ]
```

mntner Object Example

```
mntner:         MAINT-AU-APNICTRAINING
descr:          APNIC Training
country:        AU
admin-c:        NR97-AP
tech-c:         NR97-AP
auth:           # Filtered
mnt-by:         MAINT-AU-APNICTRAINING
upd-to:         nurul@apnic.net
referral-by:    APNIC-HM
changed:        hm-changed@apnic.net 20131129
source:         APNIC
```

Hierarchical Authorization

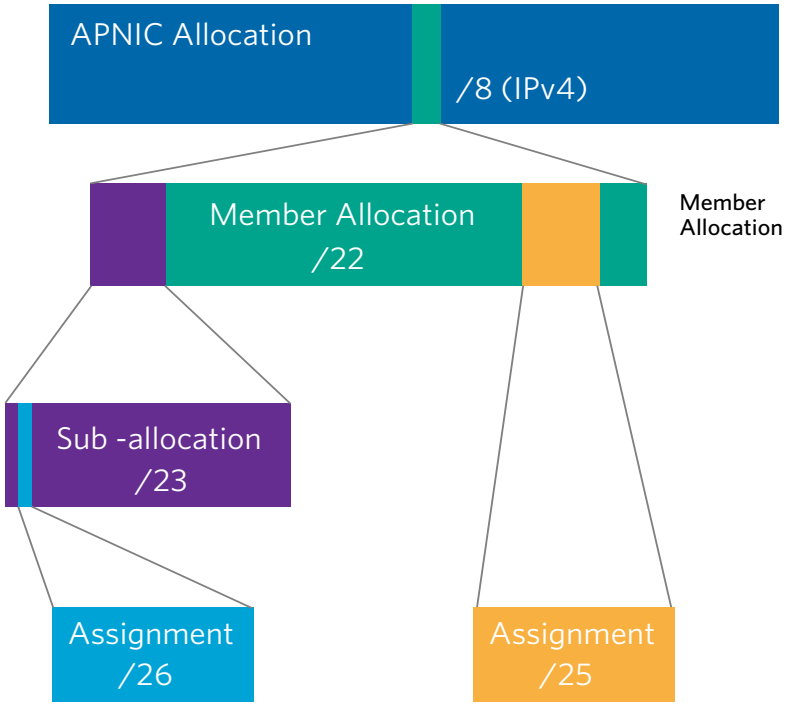
- **mnt-by**
  - Used to protect any object
  - Changes to protected object must satisfy authentication rules of **mntner** object
- **mnt-lower**
  - Used for sub-assignment creation (customer assignment)
- **mnt-routes**
  - Used for the creation of **route** or **route6** objects
  - **inetnum**, **inet6num** and **aut-num** must have the same mnt-route maintainer

Maintainer Hierarchy Diagram

Allocated to APNIC:
mnt-by can only be changed by IANA

Allocated to Member:
mnt-by can only be changed by APNIC

Sub-allocated to Customer:
mnt-by can only be changed by Member



Authorisation Mechanism

```
inet6num:       2406:6400::/32
netname:        APNIC-TRAININGIPv6-Lab-AP
descr:          APNIC TRAINING Lab
descr:          LEVEL 1, 33 PARK RD
country:        AU
admin-c:        AT480-AP
tech-c:         AT480-AP
mnt-by:         APNIC-HM
mnt-lower:      MAINT-AU-APNICTRAINING
mnt-routes:     MAINT-AU-APNICTRAINING
status:         ALLOCATED PORTABLE
```

1. This object can only be modified by **APNIC-HM**
2. Creation of more specific objects within this range has to pass the authentication of **MAINT-AU-APNICTRAINING**
3. Creation of route objects matching/within this range has to pass the authentication of **MAINT-AU-APNICTRAINING**

route/route6 Object

- Use CIDR length format
- Specifies origin AS for a route.
- Use both route and origin fields as the primary key

```
route:          [mandatory]  [single]     [primary/lookup key]
descr:          [mandatory]  [multiple]   [ ]
country:        [optional]   [single]     [ ]
origin:         [mandatory]  [single]     [primary/inverse key]
holes:          [optional]   [multiple]   [ ]
member-of:      [optional]   [multiple]   [inverse key]
inject:         [optional]   [multiple]   [ ]
aggr-mtd:       [optional]   [single]     [ ]
aggr-bndry:     [optional]   [single]     [ ]
export-comps:   [optional]   [single]     [ ]
components:     [optional]   [single]     [ ]
remarks:        [optional]   [multiple]   [ ]
notify:         [optional]   [multiple]   [inverse key]
mnt-lower:      [optional]   [multiple]   [inverse key]
mnt-routes:     [optional]   [multiple]   [inverse key]
mnt-by:         [mandatory]  [multiple]   [inverse key]
changed:        [mandatory]  [multiple]   [ ]
source:         [mandatory]  [single]     [ ]
```

route/route6 Example

```
route6:         2406:6400::/32
descr:          APNIC Training Lab parent block
country:        AU
origin:         AS17821
notify:         training@apnic.net
mnt-by:         MAINT-AU-APNICTRAINING
changed:        hm-changed@apnic.net 20100818
source:         APNIC
```

aut-num Object

- Defines routing policy for an AS
- Uses **import/mp-import:** and **export/mp-export:** attributes to specify policy
- These define the incoming and outgoing routing announcement relationships
- Can reference other registry objects such as *as-sets / route-sets / filter-sets*

aut-num Object

```
aut-num:        [mandatory]  [single]     [primary/lookup key]
as-name:        [mandatory]  [single]     [ ]
descr:          [mandatory]  [multiple]   [ ]
country:        [mandatory]  [single]     [ ]
member-of:      [optional]   [multiple]   [inverse key]
import:         [optional]   [multiple]   [ ]
export:         [optional]   [multiple]   [ ]
default:        [optional]   [multiple]   [ ]
remarks:        [optional]   [multiple]   [ ]
admin-c:        [mandatory]  [multiple]   [inverse key]
tech-c:         [mandatory]  [multiple]   [inverse key]
notify:         [optional]   [multiple]   [inverse key]
mnt-lower:      [optional]   [multiple]   [inverse key]
mnt-routes:     [optional]   [multiple]   [inverse key]
mnt-by:         [mandatory]  [multiple]   [inverse key]
mnt-irt:        [mandatory]  [multiple]   [inverse key]
changed:        [mandatory]  [multiple]   [ ]
source:         [mandatory]  [single]     [ ]
```

aut-num Object Example

```
aut-num:        AS17821
as-name:        APNIC-TRAINING-Lab-AS-AP
descr:          Two-byte AS number for APNIC Training
import:         from as4608 accept ANY
export:         to AS4608 announce AS17821
admin-c:        AT480-AP
tech-c:         AT480-AP
mnt-by:         MAINT-AU-APNICTRAINING
mnt-routes:     MAINT-AU-APNICTRAINING
mnt-irt:        IRT-APNICTRAINING-AU
changed:        hm-changed@apnic.net 20110701
source:         APNIC
```

as-set Object

- Collect together Autonomous Systems with shared properties
- Can be used in policy in place of AS
- RPSL has hierarchical names, can reference other as-set's

Non-Hierarchical : AS-

Hierarchical: <origin-as-number>: AS-CUSTOMERS

<origin-as-number>: AS-PEERS

as-set Object

```
as-set:        [mandatory]  [single]     [primary/lookup key]
descr:         [mandatory]  [multiple]   [ ]
country:       [optional]   [single]     [ ]
members:       [optional]   [multiple]   [ ]
mbrs-by-ref:   [optional]   [multiple]   [inverse key]
remarks:       [optional]   [multiple]   [ ]
tech-c:        [mandatory]  [multiple]   [inverse key]
admin-c:       [mandatory]  [multiple]   [inverse key]
notify:        [optional]   [multiple]   [inverse key]
mnt-by:        [mandatory]  [multiple]   [inverse key]
mnt-lower:     [optional]   [multiple]   [inverse key]
changed:       [mandatory]  [multiple]   [ ]
source:        [mandatory]  [single]     [ ]
```

as-set Object Example

```
as-set:          AS-APNICTRAINING
descr:           AS-SET for APNIC Training
tech-c:          AT480-AP
admin-c:         AT480-AP
mnt-by:          MAINT-AU-APNICTRAINING
changed:         fakrul@apnic.net 20151215
members:         AS17821
source:          APNIC
```

route-set Object

- Defines a set of routes prefixes
- Name must begin with prefix “RS-” or in the format

  ASNUM:RS-<ORGANIZATION>

- Can reference other route-sets, 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

route-set Object

route-set:	[mandatory]	[single]	[primary/lookup key]
descr:	[mandatory]	[multiple]	[ ]
members:	[optional]	[multiple]	[ ]
mp-members:	[optional]	[multiple]	[ ]
mbrs-by-ref:	[optional]	[multiple]	[inverse key]
remarks:	[optional]	[multiple]	[ ]
tech-c:	[mandatory]	[multiple]	[inverse key]
admin-c:	[mandatory]	[multiple]	[inverse key]
notify:	[optional]	[multiple]	[inverse key]
mnt-by:	[mandatory]	[multiple]	[inverse key]
mnt-lower:	[optional]	[multiple]	[inverse key]
changed:	[mandatory]	[multiple]	[ ]
source:	[mandatory]	[single]	[ ]

source : https://www.rfc-editor.org/rfc/rfc2622.txt

route-set Object Example

```
route-set:      RS-APNICTRAINING
descr:          Routes announced by APNIC Training
tech-c:         AT480-AP
admin-c:        AT480-AP
mnt-by:         MAINT-AU-APNICTRAINING
changed:        fakrul@apnic.net 20151215
mp-members:     2406:6400::/32, AS17821
source:         APNIC
```

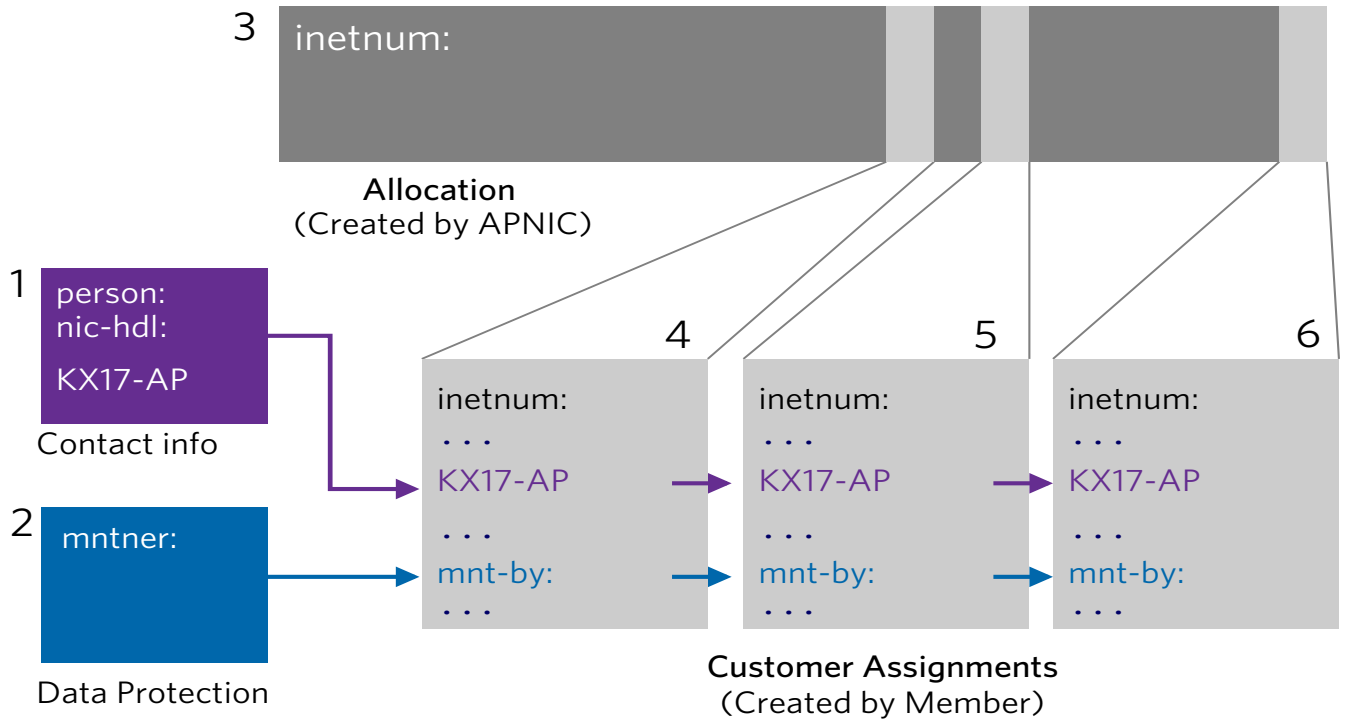
filter-set Object

- 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-”

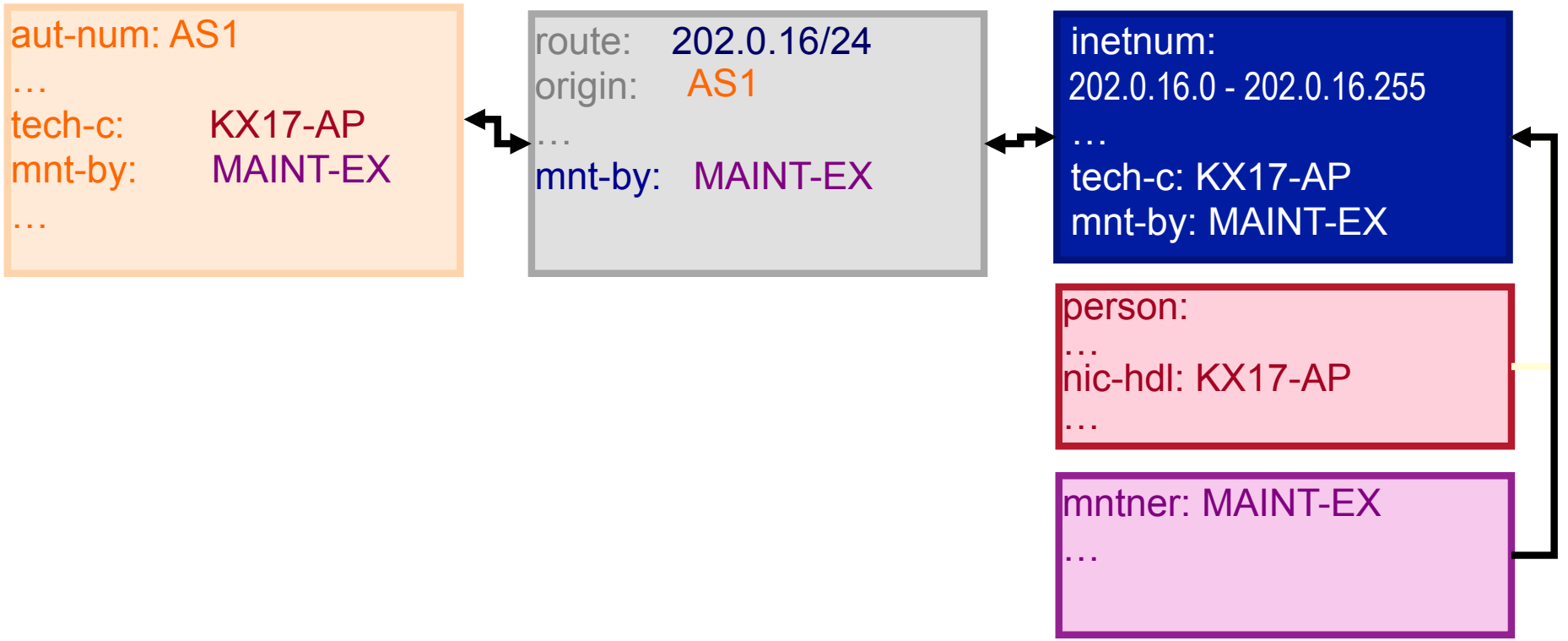
filter-set Object Example

```
filter-set:     fltr-martian-v6
descr:          Current IPv6 MARTIANS
tech-c:         FA129-AP
admin-c:        FA129-AP
mnt-by:         MAINT-AU-APNICTRAINING
changed:        fakrul@apnic.net 20151221
mp-filter:      {
                0000::/8^+,       # loopback, unspecified, v4-mapped
                0064:ff9b::/96^+,  # IPv4-IPv6 Translat. [RFC6052]
                0100::/8^+,        # reserved for Discard-Only Address Block [RFC6666]
                0200::/7^+,        # Reserved by IETF [RFC4048]
                0400::/6^+,        # Reserved by IETF [RFC4291]
                0800::/5^+,        # Reserved by IETF [RFC4291]
                c000::/3^+,        # Reserved by IETF [RFC4291]
                e000::/4^+,        # Reserved by IETF [RFC4291]
                f000::/5^+,        # Reserved by IETF [RFC4291]
                f800::/6^+,        # Reserved by IETF [RFC4291]
                fc00::/7^+,        # Unique Local Unicast [RFC4193]
                fe80::/10^+,       # Link Local Unicast [RFC4291]
                fec0::/10^+,       # Reserved by IETF [RFC3879]
                ff00::/8^+         # Multicast [RFC4291]
                }
remarks:        fltr-martian-v6 from RIPE-NCC
remarks:        this object is manually maintained.
source:         APNIC
```

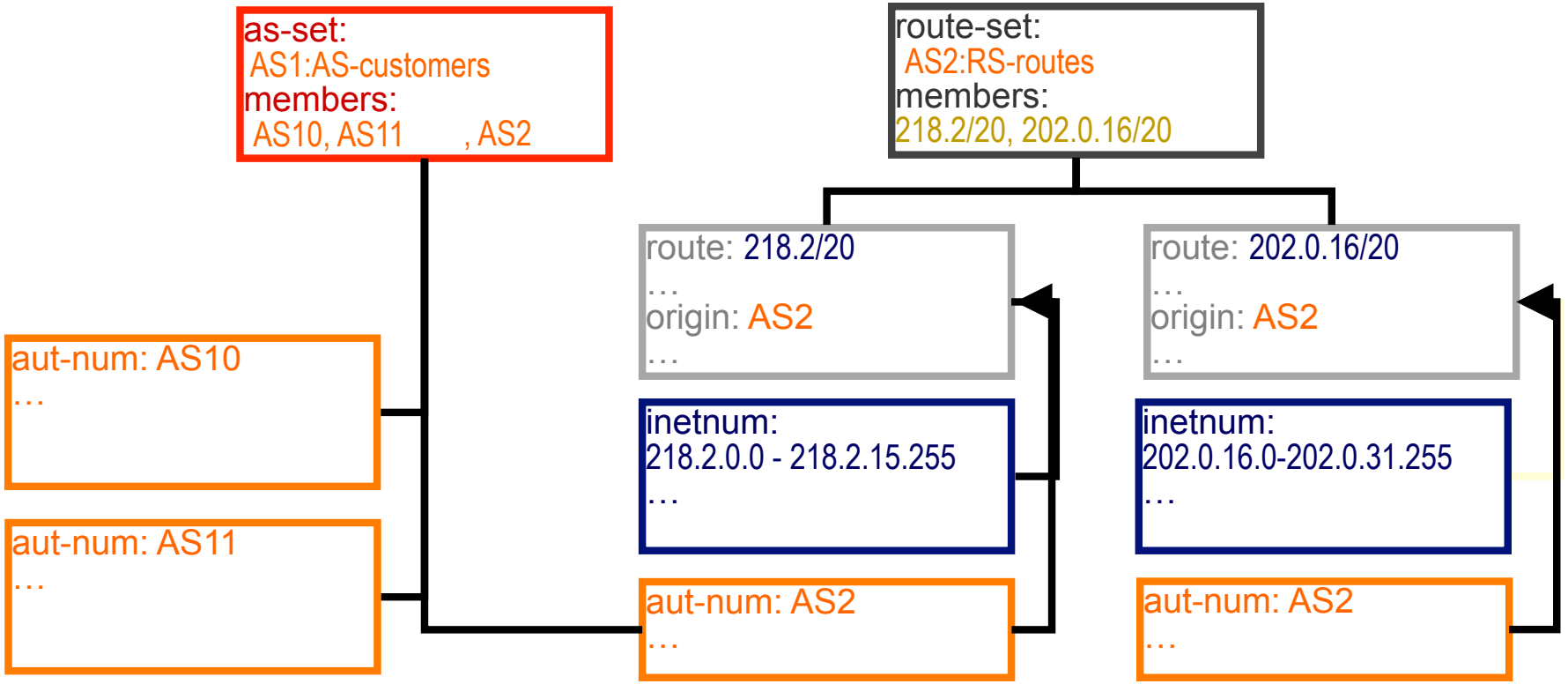
Relation between objects



Inter-related IRR Objects



Inter-related IRR Objects



RPSL Objects
&
Routing Policy

The Internet Routing Registry (IRR)

- Number of public databases that contain routing policy information which mirror each other:
  - APNIC, RIPE, RADB, JPIRR, Level3
  - http://www.irr.net/
- Stability and consistency of routing – network operators share information
- Both public and private databases
- These databases are independent – but some exchange data
  - only register your data in one database
- List of Routing Registry
  - http://www.irr.net/docs/list.html

The Internet Routing Registry (IRR)

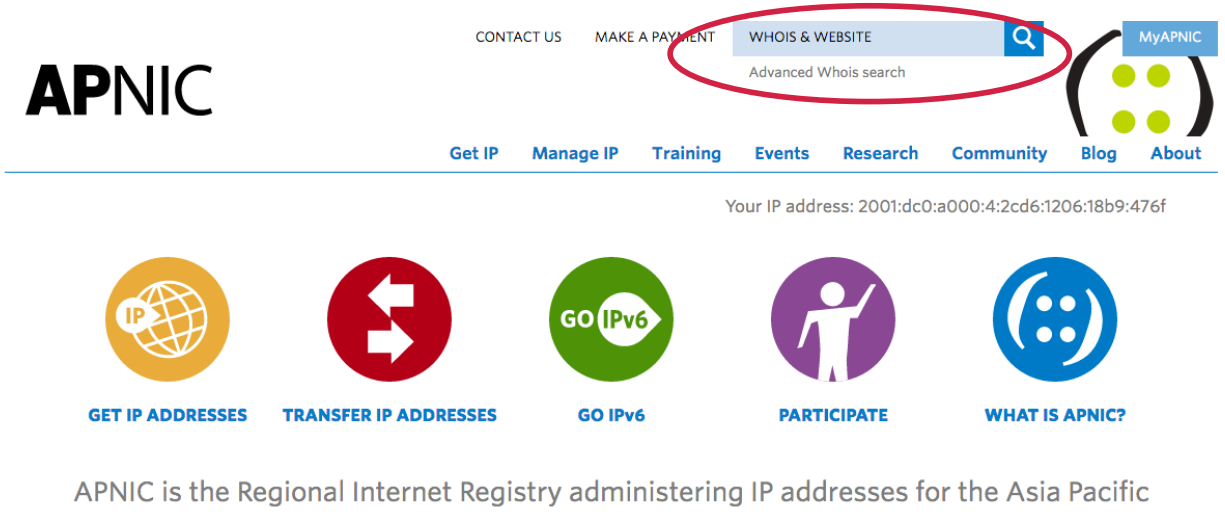
- IRRs are used in at least three distinct ways
  - To publish your own routing intentions
  - To construct and maintain routing filters and router configurations
  - Diagnostic and information service for more general network management

IRR Objects query

- `whois` query from cli

```
whois -h whois.apnic.net 2406:6400::/32
```

- You can also search from APNIC website



IRR objects query flags

- IRR supports a number of flag option
  - **!** RADB Query Flags
  - **-** RIPE/BIRD Query Flags
- `-i` flags for inverse query

```
whois -h whois.apnic.net -i mnt-by MAINT-AU-
APNICTRAINING
```

[All the objects with a matching **mnt-by** attribute]

```
whois -h whois.apnic.net -i origin as17821
```

[**route** and **route6** objects with a matching **origin** attribute]

- -q flag for Informational queries

```
whois -h whois.apnic.net -q sources
```

[list of sources]

IRR objects query flags

- `-K` flags for primary keys of an object are returned

```
whois -h whois.apnic.net -K 2406:6400::/32
```

- IRRd (IRR Daemon) supports service side set expansions (as-set and route-set)

```
whois -h whois.radb.net '!iAS-APNICTRAINING'
```

[returns members of AS-APNICTRAINING as-set object]

- For details please check
  - https://www.apnic.net/apnic-info/whois_search/using-whois/searching/query-options
  - http://www.radb.net/support/query2.php

RPSL Implementation : How to begin

- Need to identify which IRR to use
  - May want to run your own for control
- Need to decide what degree of filtering is desired
  - Prefix filters
  - AS path filters
  - Both
- Register a maintainer object at chosen IRR
  - Usually a “manual” process and could be multi-stage if PGP key authentication required

RPSL Implementation : Checklist

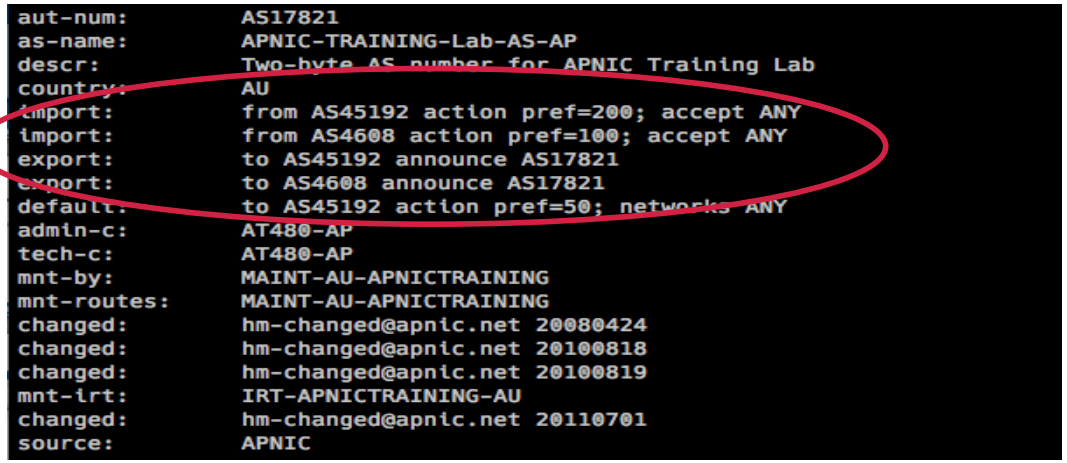
1. Define your routing policy
2. Creating the objects in IRR
3. Use automated tools to generate the configuration

Objects Involved

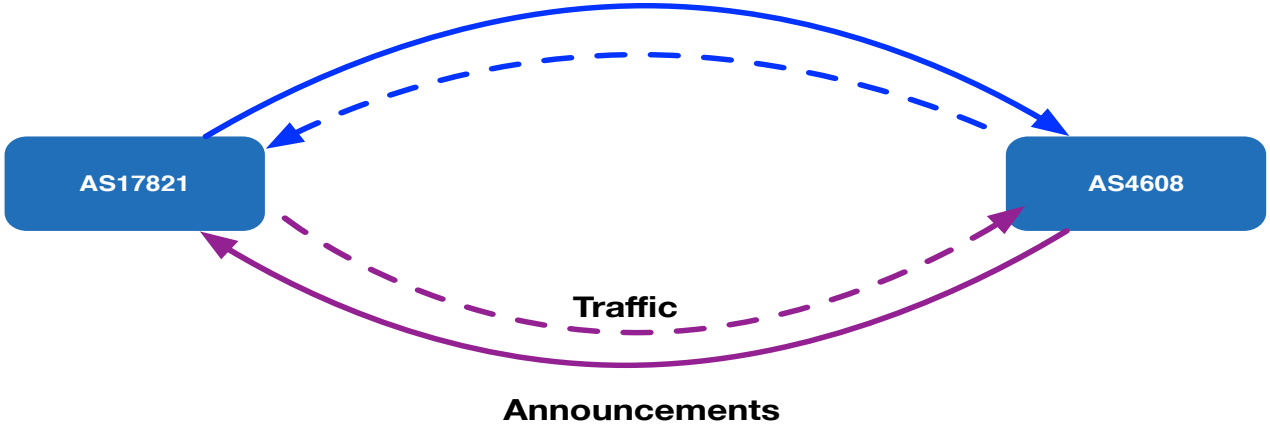
Objects	Functions
route or route6 object	Connects a prefix to an origin AS
aut-num object	Registration record of an AS Number Contains the routing policy
sets	Objects can be grouped in sets, i.e. as-set, route-set
keywords	“ANY” matches every route

Import and Export Attributes

- You can document your routing policy in your aut-num object in the APNIC Database:
  - Import lines describe what routes you accept from a neighbor and what you do with them
  - Export lines describe which routes you announce to your neighbor



Route Announcements vs Traffic Direction

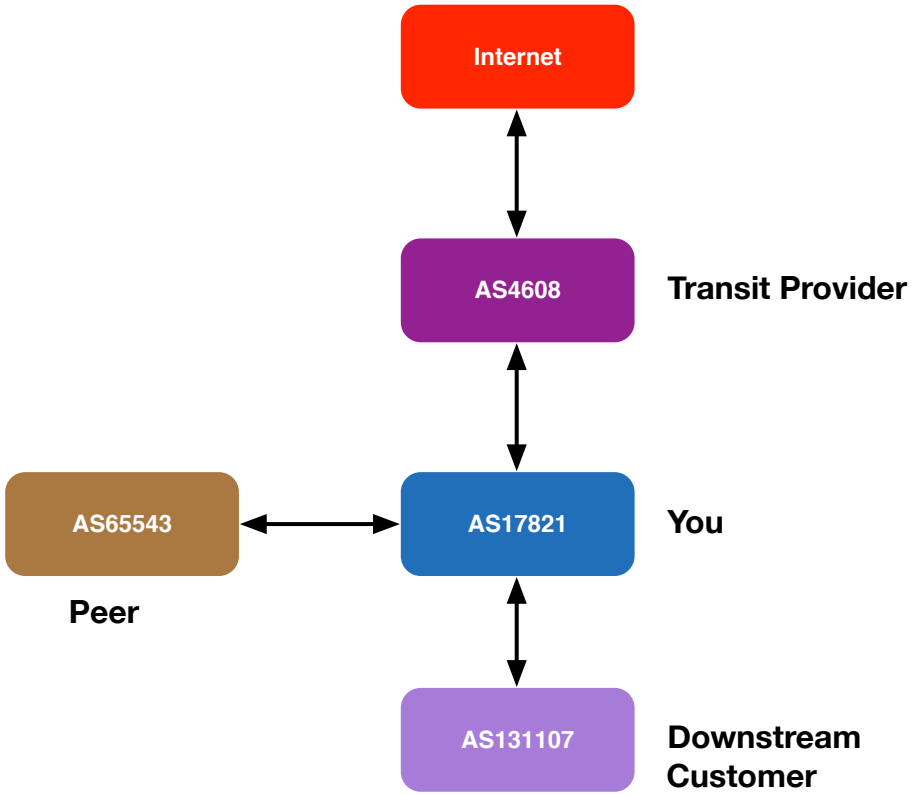


- AS17821 accepting all prefixes from AS4608 so that outbound traffic goes towards AS4608. It also makes localpref to 100
- AS17821 announcing prefixes (originating in AS17821) to AS4608, so that the incoming traffic for AS17821 can flow away from the AS4608

aut-num: AS17821

import: from AS4608 action pref=100; accept ANY
export: to AS4608 announce AS17821

Routing Policy Scenarios



aut-num: AS17821

import: from AS4608 accept ANY
export: to AS4608 announce AS17821 AS131107

import: from AS131107 accept AS131107
export: to AS131107 announce ANY

import: from AS65543 accept AS65543
export: to AS65543 announce AS17821 AS131107

Building an aut-num Object

- RPSL is older than IPv6, the defaults are IPv4
- IPv6 was added later using a different syntax
  - You have to specify that it's IPv6

```
mp-import: afi ipv6.unicast from AS131107 accept AS131107
mp-export: afi ipv6.unicast to AS131107 announce ANY
```

- More information in RFC 4012 RPSLng

Filter List : Regular Expression

AS17821	AS 17821
AS17821*	0 or more occurrences of AS17821
AS17821+	1 or more occurrences of AS17821
AS17821?	0 or 1 occurrence of AS17821
&	Beginning of Path
$	End of Path
\	Escape a regular expression character
_	Beginning, end, white-space, brace
AS17821|AS45192	AS17821 or AS45192
AS17821AS45192	AS17821 followed by AS45192
()	Brackets to contain expression
[]	Brackets to contain numbers

Enclose the expression in “<“ and “>”

RPSL: localpref / prepend

- Controlling the traffic flow:
  - for outbound traffic set the value of local-pref
    - “action pref=NN” in the “import” lines of aut-num object
    - the lower the “pref”, the more preferred the route
  - for inbound traffic, modify as-path length
    - “action aspath.prepend(ASN)” in the “export” lines
    - Longer the as-path, less preferred the route

Note: the direction of traffic is reverse from accepting / announcing routes

RPSL: localpref/prepend Example

Local preference:

```
mp-import:      afi ipv6.unicast from AS65001
2406:6400:10::2 at 2406:6400:10::1 action
community.append(17821:65001); pref=200; accept <^AS65001+
$> AND RS-APNICTRAINING:AS65001
```

Default value is 1000. Setting pref value to 200 mean downgrade the pref value by 200. Local pref will be 800.

Prepend:

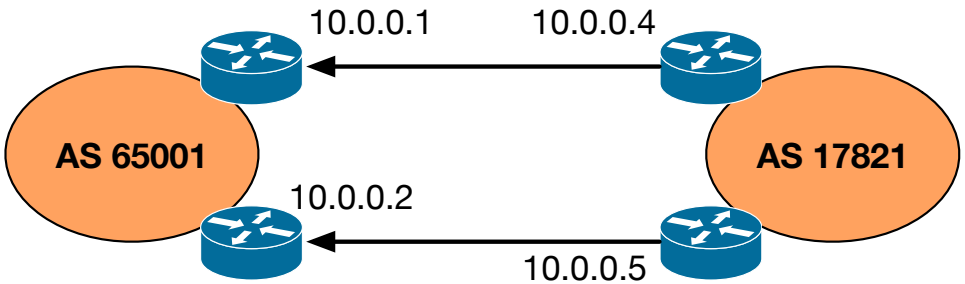
```
mp-export:      afi ipv6.unicast to AS65001 2406:6400:10::2
at 2406:6400:10::1 action aspath.prepend (AS17821,AS17821);
announce ANY AND NOT FLTR-MARTIAN-V6
```

RPSL: Multiple Links / MED

- By setting the value of MED on export lines, the preferred entry point into your AS can be controlled
- The neighbour must agree to honour your MED values
  - Instead of MED, it is possible to use as-path prepend on less preferred link

RPSL: MED Example

```
export: to AS17821 10.0.0.4 at 10.0.0.1 action med=1000;
announce AS65001
export: to AS17821 10.0.0.5 at 10.0.0.2 action med=2000;
announce AS65001
```



RPSL: BGP Communities

- Elegant solution for implementing policies
- Optional tags
  - Can go through many peers
- Can be used for advanced filtering
- Not a routing parameter
- Enables customers to control their own routing policy
  - Publish your communities, and what you do with them
  - Filter incoming announcements accordingly

RPSL: BGP Communities Example

```
mp-import:      afi ipv6.unicast from AS65001
2406:6400:10::2 at 2406:6400:10::1 action
community.append(17821:65001); pref=200; accept <^AS65001+
$> AND RS-APNICTRAINING:AS65001
```

RPSL Tools

- IRRToolkit (written in C++)
  - http://irrtoolset.isc.org/
- Rpsltool (perl, using Template::Toolkit)
  - http://www.linux.it/~md/software
- IRR Power Tools (PHP)
  - http://sourceforge.net/projects/irrpt/
- BGPQ3 (C)
  - http://snar.spb.ru/prog/bgpq3/
- Filtergen (Level 3)
  - Online tool using whois protocol
  - whois -h filtergen.level3.net RIPE::ASxxxx

RPSL Tools

Tool	Advantages	Disadvantages
IRRToolSet	• Full RPSL support • RPSLng support • 32-bit ASN support • Full BGP config generation	• No AS-Set query support • Manual peering configuration on the fly • Difficult to understand
IRR Power Tools	• Route aggregation • AS-SET queries	• No RPSLng support • No 32-bit ASN support
BGPq3	• RPSL support • RPSLng support • 32-bit ASN • AS-SET queries • Easy to use	• Only partial BGP configuration. Can't extract policy from IRR
RPSLtool	• 32-bit ASN • AS-SET queries	• No RPSLng support
Net::IRR	• RPSL and RPSLng support	• Outdated • Doesn't support community attribute from RPSL data • No AS-SET queries
Netconfigs	• Provides peering analysis • Can generate full configuration based on peering relationship	• Doesn't support RPSLng • No command line query • Vendor dependent (CISCO)

Use of RPSL

- Use RtConfig to generate filters based on information stored in our routing registry
  - Avoid filter errors (typos)
  - Filters consistent with documented policy (need to get policy correct though)
  - Engineers don’t need to understand filter rules (it just works :-)
- Some providers have their own tools

Using RPSL to configure routers

- Need to define “policy” for filtering
  - Inbound from customers & peers
  - Outbound to customers & peers
- Need to be aware of shortcomings in router configuration and/or configuration generator
  - Command line length (on cisco this is 512 bytes)
  - Complexity of rules

Filtering philosophy

- Inbound
  - Filter customer by prefix and AS path
  - Filter peer by AS path only but don't accept host routes
  - Filter providers for prefixes longer than a /24
  - Don't accept martians from anyone
- Outbound
  - Filter by BGP community, which indicates the class of the prefix (customer, peer, etc)

Martians

- RtConfig has built in list of martians that can be added automatically to filters by use of command line option
- *-supress_martian* is deprecated
- Properly maintained martian and bogon lists are visible in both the RIPE and Merit whois servers
- You can use following filter-set from APNIC whois
  - `fltr-martian-v4 / fltr-martian-v6`

IRRToolSet : Installation

- Dependency (Debian / Ubuntu)

```
# apt-get install build-essential libtool subversion bison
flex libreadline-dev autoconf automake
```

- Installation

```
# wget
ftp://ftp.isc.org/isc/IRRToolSet/IRRToolSet-5.0.1/
irrtoolset-5.0.1.tar.gz
# tar –zxvf irrtoolset-5.0.1.tar.gz
# cd irrtoolset-5.0.1
# ./configure
# make & make install
```

For details : http://irrtoolset.isc.org/wiki/IRRToolSetInstallation

RtConfig command line options

- Defaults to using RADB
  - -h whois.ra.net / whois.radb.net / whois.apnic.net
  - -p 43
  - Default protocol irrd
- For other RIR use protocol bird
  - -protocol bird/ripe
- Defaults to “cisco” style output
  - -config cisco / -config junos
- -s <list of IRR sources>
  - -s APNIC,RADB,RIPE

RtConfig Syntax

- import / export pair for each link; syntax

```
@RtConfig [import/export] <yourASN> <yourRouterIP>
<neighbourASN> <neighbourRouterIP>
```

- Takes other command also

```
@RtConfig configureRouter <inet-rtr-name>
@RtConfig static2bgp <ASN-1> <rtr-1>
@RtConfg access_list filter <filter>
```

- And many more. But best thing to look `man rtconfig`

IRRToolSet Cisco Example

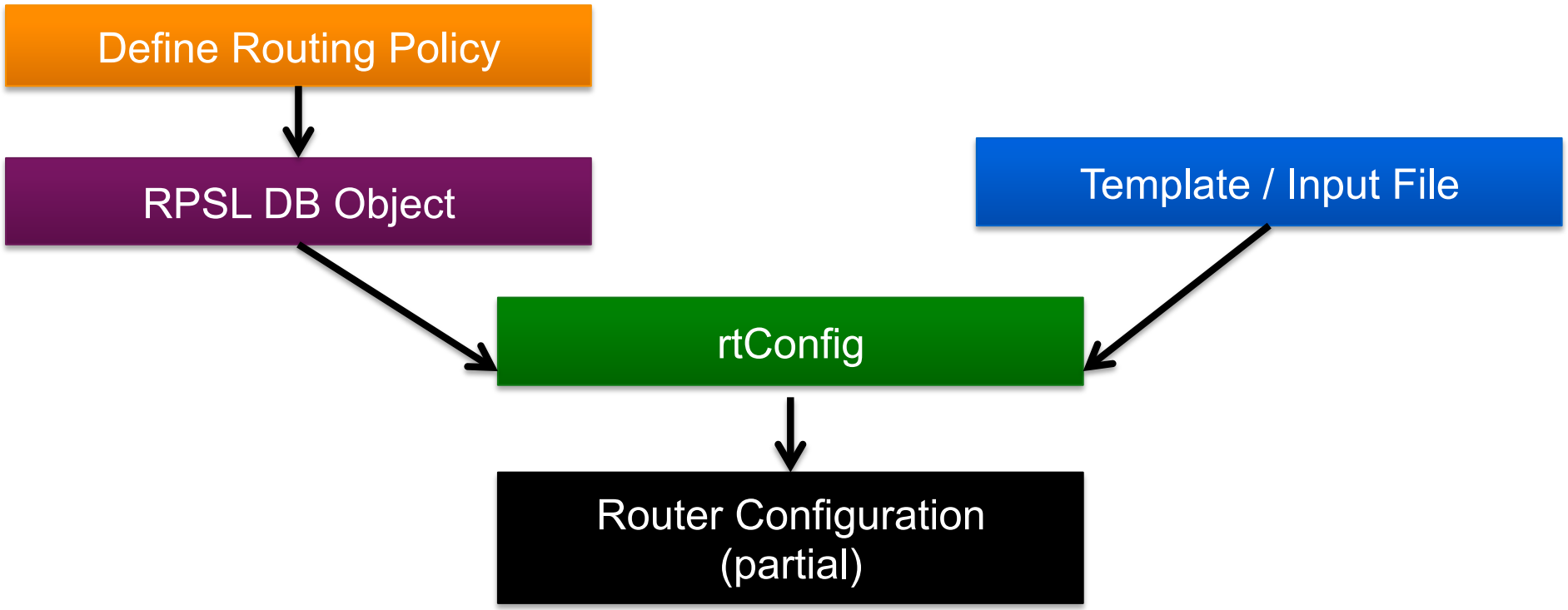
```
bash-3.2$ rtconfig -protocol bird -config cisco -h whois.radb.net

rtconfig> @RtConfig import AS17821 2406:6400:10::1 AS65001 2406:6400:10::2
!
no ipv6 access-list ipv6-500
ipv6 access-list ipv6-500 permit 2406:6400:8000::/48 any
ipv6 access-list ipv6-500 deny any any
!
no ip as-path access-list  500
ip as-path access-list 500 permit ^(_65001)+$

<output truncated>

router bgp 17821
!
 neighbor 2406:6400:10::2 remote-as 65001
 address-family ipv4
  no neighbor 2406:6400:10::2 activate
 address-family ipv6 unicast
  neighbor 2406:6400:10::2 activate
  neighbor 2406:6400:10::2 route-map AS65001-IN in
 exit
```

IRRToolSet JunOS Example

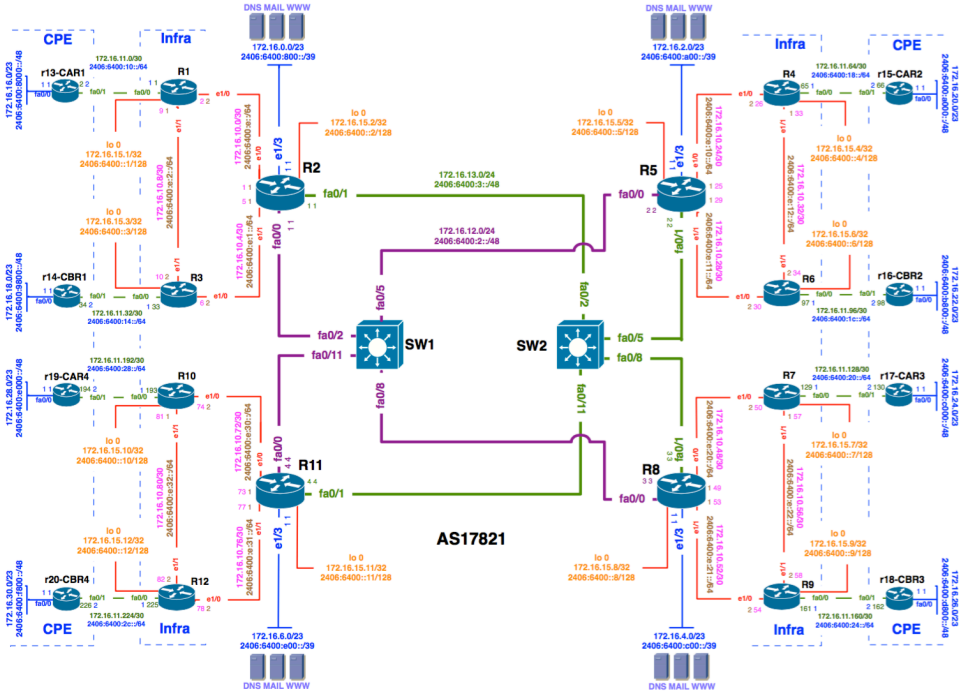
```
bash-3.2$ rtconfig -protocol bird -config junos -h whois.radb.net

rtconfig> @RtConfig import AS17821 2406:6400:10::1 AS65001 2406:6400:10::2
policy-options {
   community community-1 members [17821:65001];
   as-path as-path-1 "( 65001)+";

<output truncated>

protocols {
   bgp {
      group peer-2406:6400:10::2 {
         type external;
         peer-as 65001;
         neighbor 2406:6400:10::2 {
            import policy_65001_1 ;
            family inet6 {
                unicast;
            }
         }
      }
   }
}
```

RPSL in practice : LAB

RtConfig: The Big Picture

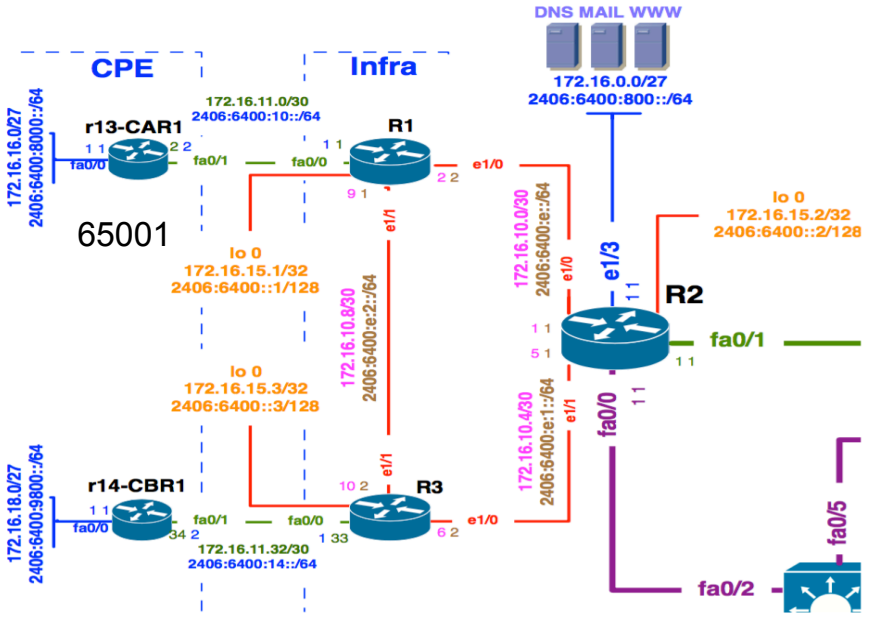


Topology


CPE
Infra
DNS MAIL WWW
172.16.0.0/23
2406:6400:800::/39
172.16.2.0/23
2406:6400:a00::/39
172.16.4.0/23
2406:6400:c00::/39
172.16.6.0/23
2406:6400:e00::/39
r13-CAR1
r14-CBR1
r19-CAR4
r20-CBR4
r15-CAR2
r16-CBR2
r17-CAR3
r18-CBR3
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
SW1
SW2
172.16.13.0/24
2406:6400:3::/48
172.16.12.0/24
2406:6400:2::/48
AS17821

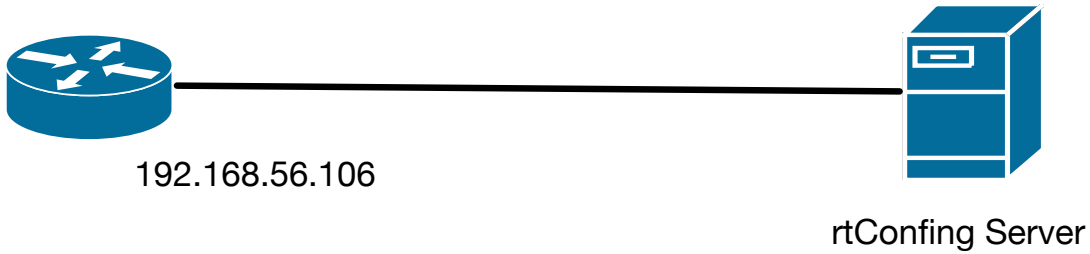
Topology : Region 1

- RPSL Object
  - aut-num : AS17821
  - mnt-by: MAINT-AU-APNICTRAINING
  - route-set: RS-APNICTRAINING
  - fltr-set: FLTR-MARTIAN-V6



Simplified View

- For demonstration we use the following topology



- RPSL Object
  - aut-num : AS17821
  - mnt-by: MAINT-AU-APNICTRAINING
  - route-set: RS-APNICTRAINING
  - fltr-set: FLTR-MARTIAN-V6

IRRToolSet : RPSL Object

```
# whois -h whois.apnic.net as17821


mp-import:      afi ipv6.unicast from AS65001
2406:6400:10::2 at 2406:6400:10::1 action
community.append(17821:65001); pref=200; accept <^AS65001+
$> AND RS-APNICTRAINING:AS65001

mp-export:      afi ipv6.unicast to AS65001 2406:6400:10::2
at 2406:6400:10::1 announce ANY AND NOT FLTR-MARTIAN-V6
```

RtConfig Configuration Template (provision.cfg) – Provision Customer

```
@RtConfig set cisco_map_first_no = 10
@RtConfig set cisco_map_increment_by = 10
@RtConfig set cisco_prefix_acl_no = 100
@RtConfig set cisco_aspath_acl_no = 100
@RtConfig set cisco_pktfilter_acl_no = 100
@RtConfig set cisco_community_acl_no = 10
@RtConfig set cisco_max_preference = 500
!
ip bgp-community new-format
ipv6 unicast-routing
!
! AS65001 CONFIGURATION
@RtConfig set cisco_access_list_no = 500
@RtConfig set cisco_map_name = "AS65001-IMPORT"
@RtConfig import AS17821 2406:6400:10::1 AS65001 2406:6400:10::2
@RtConfig set cisco_access_list_no = 501
@RtConfig set cisco_map_name = "AS65001-EXPORT"
@RtConfig export AS17821 2406:6400:10::1 AS65001 2406:6400:10::2
!
end
```

IRRToolSet : RtConfig Output File

- Now generate the router configuration file

```
rtconfig -protocol bird -cisco_use_prefix_lists -config
cisco -h whois.apnic.net< provision.cfg > /private/
tftpboot/router_config.cfg
```

- You will get output of full configuration
- Configuration will be saved in `/private/tftpboot`

RtConfig Configuration Template (change.cfg) – Update Customer

- Filter customer based on
  - Prefix List
  - AS-PATH access list
- For that we use
  - AS-SET

Simulating Policy Change

- To avoid the impact of the policy change, can do the simulation before publishing your aut-num
  1. Copy the aut-num object into a txt file
  2. Modify the aut-num and save in the new file
  3. Run RtConfig with the flag "**`-f`**"
     - `E.g. "rt –f my_new_asn.txt <rt template> new_router_config"`
     - Other values will be read from the RR (peer aut-nums etc)
  4. Compare new router config output with the old
     - or check if the result describes desired behavior

Upload configuration

- Various ways to upload configuration:
  - SNMP Write
  - NETCONF XML Based
  - Automated Script using expect

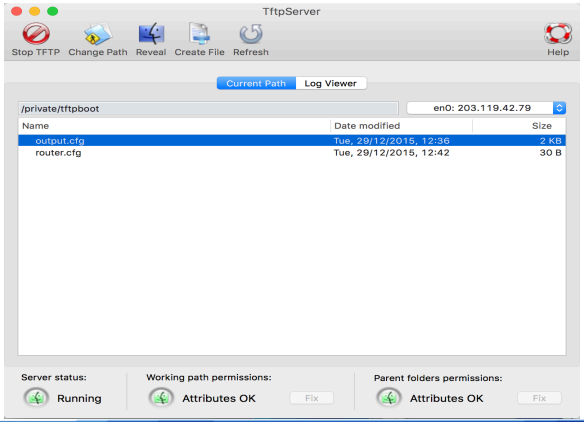
Upload configuration : SNMP

- Enable SNMP:

```
access-list 99 permit 192.168.56.0 0.255.255.255
snmp-server community APNIC rw 99
snmp-server ifindex persist
```

  – Recommended to use SNMPv3.

- Run TFTP server



Upload configuration : SNMP

```
#Set copy method:
snmpset -v 2c -c {community-string} {device-ip-address} 1.3.6.1.4.1.9.9.96.1.1.1.1.2.116
i 1
#Set sourcefile to network file:
snmpset -v 2c -c {community-string} {device-ip-address} 1.3.6.1.4.1.9.9.96.1.1.1.1.3.116
i 1
#Set destination to running-config:
snmpset -v 2c -c {community-string} {device-ip-address} 1.3.6.1.4.1.9.9.96.1.1.1.1.4.116
i 4
#Set TFTP server ip:
snmpset -v 2c -c {community-string} {device-ip-address} 1.3.6.1.4.1.9.9.96.1.1.1.1.5.116
a {ip-address-tftp-server}
#Set desination filename:
snmpset -v 2c -c {community-string} {device-ip-address} 1.3.6.1.4.1.9.9.9a6.1.1.1.1.6.116
s router_config.cfg
#Start tftp upload via via OID ccCopyEntryRowStatus:
snmpset -v 2c -c {community-string} {device-ip-address} 1.3.6.1.4.1.9.9.96.1.1.1.1.14.116
i 1
```

Note: The integer highlighted in **red** is a random integer and you can choose any integer between 1 and 255. Keep in mind to use the same integer for the whole upload procedure! See the integer as a session.

Getting the complete picture

- Automation relies on the IRR being complete
  - Not all resources are registered in an IRR
  - Not all information is correct
- Small mistakes can have a big impact
  - Check your output before using it
- Be prepared to make manual overrides
  - Help others by documenting your policy

RPSL in summary

1. Define Routing Policy

2. Create IRR Object/Objects

3. Run RtConfig to generate config

4. Push config to router/routers

Challenges for the Routing Registries

- Lots of Routing Registries
- Accuracy and completeness
- Not every Routing Registry is linked directly to an Internet Registry
  - Offline verification of the resource holder is needed
- Different authorization methods
- Mirrors are not always up to date

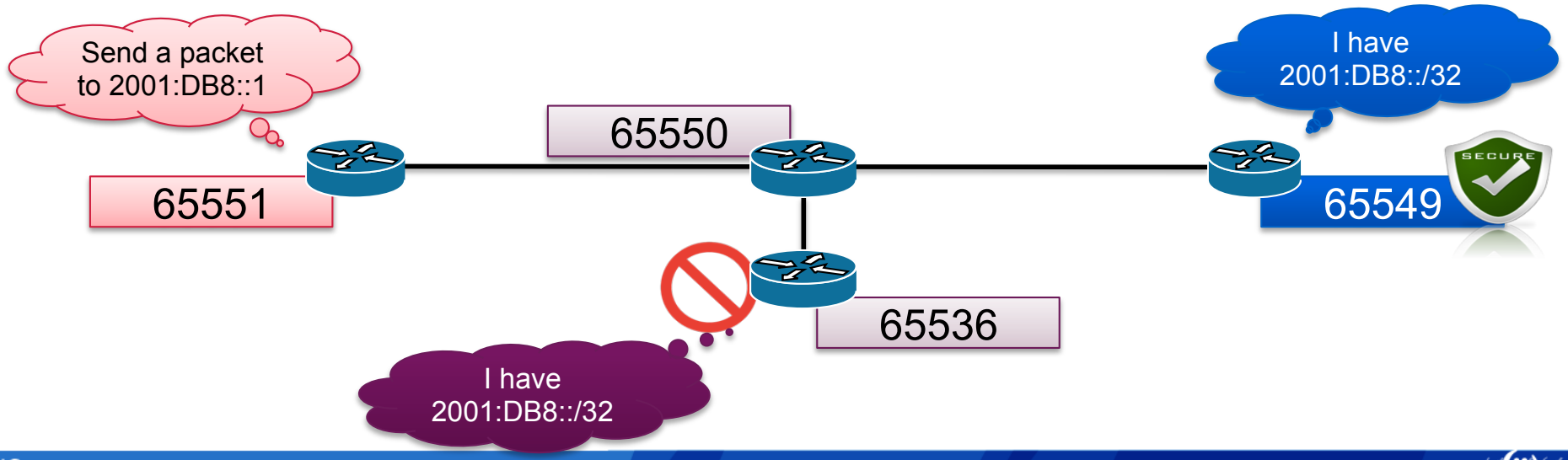
RPKI

Purpose of RPKI

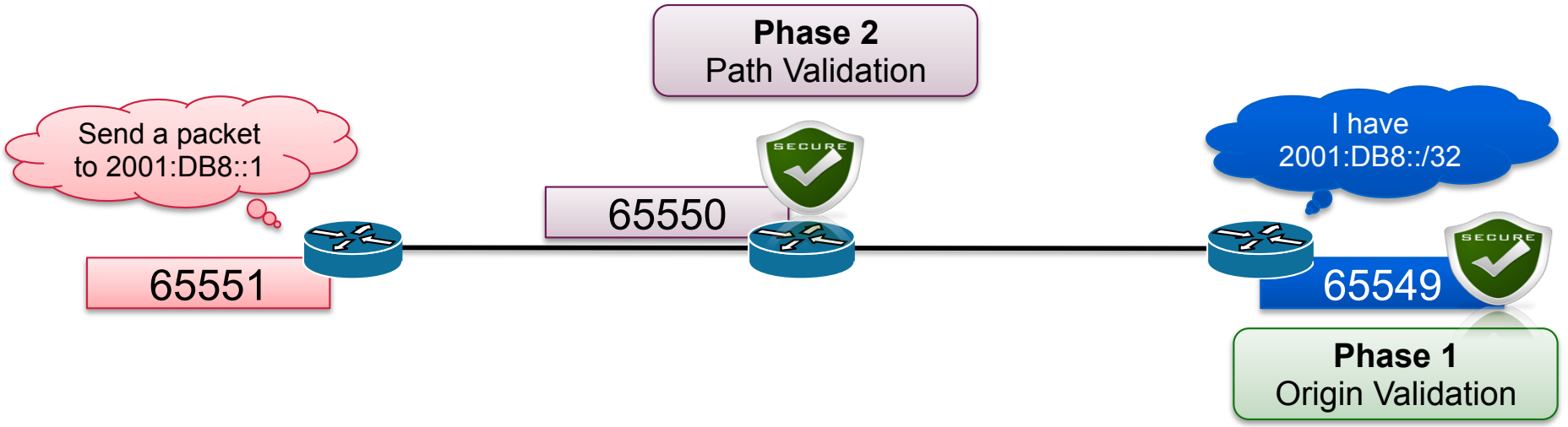
- RPKI replaces IRR or lives side by side?
  - Side by side: different advantages
  - Security, almost real time, simple interface: RPKI
- Purpose of RPKI
  - Is that ASN authorized to originate that address range?

RPKI Origin Validation

2001:DB8::/32	65551	65550	65549	i	VALID
2001:DB8::/48	65551	65550	65536	i	INVALID



RPKI Deployment



Internet Registry (IR) / RIR

- Maintains Internet Resources such as IP addresses and ASNs, and publish the registration information
  - Allocations for Local Internet Registries
  - Assignments for end-users
- APNIC is the Regional Internet Registry(RIR) in the Asia Pacific region
  - National Internet Registry(NIR) exists in several economies

The Eco-System

Goals of RPKI

- Able to authoritatively prove who owns an IP Prefix and what AS(s) may Announce It
  - Reducing routing leaks
  - Attaching digital certificates to network resources (AS Number & IP Address)
- Prefix Ownership Follows the Allocation Hierarchy IANA, RIRs, ISPs, …

Advantage of RPKI

- Useable toolset
  - No installation required
  - Easy to configure manual overrides
- Tight integration with routers
  - Supported routers have awareness of RPKI validity states
- Stepping stone for AS-Path Validation
  - Prevent Attacks on BGP

RPKI Implementation

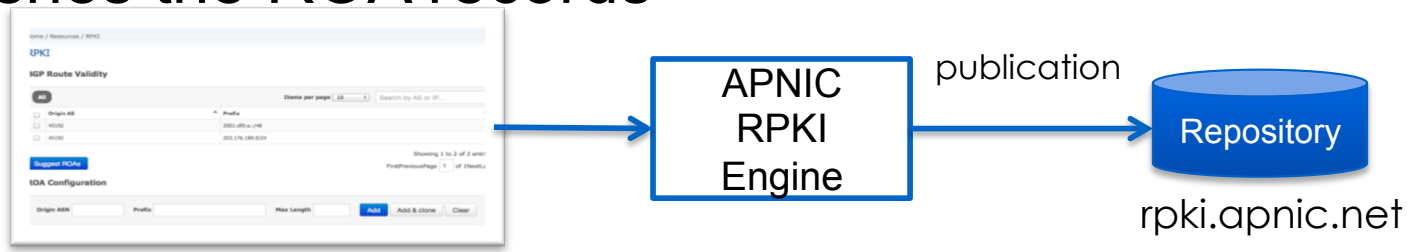
- Two RPKI implementation type
  - **Delegated**: Each participating node becomes a CA and runs their own RPKI repository, delegated by the parent CA.
  - **Hosted**: The RIR runs the CA functionality for interested participants.

Two Components

- Certificate Authority (CA)
  - Internet Registries (RIR, NIR, Large LIR)
  - Issue certificates for customers
  - Allow customers to use the CA's GUI to issue ROAs for their prefixes
- Relying Party (RP)
  - Software which gathers data from CAs

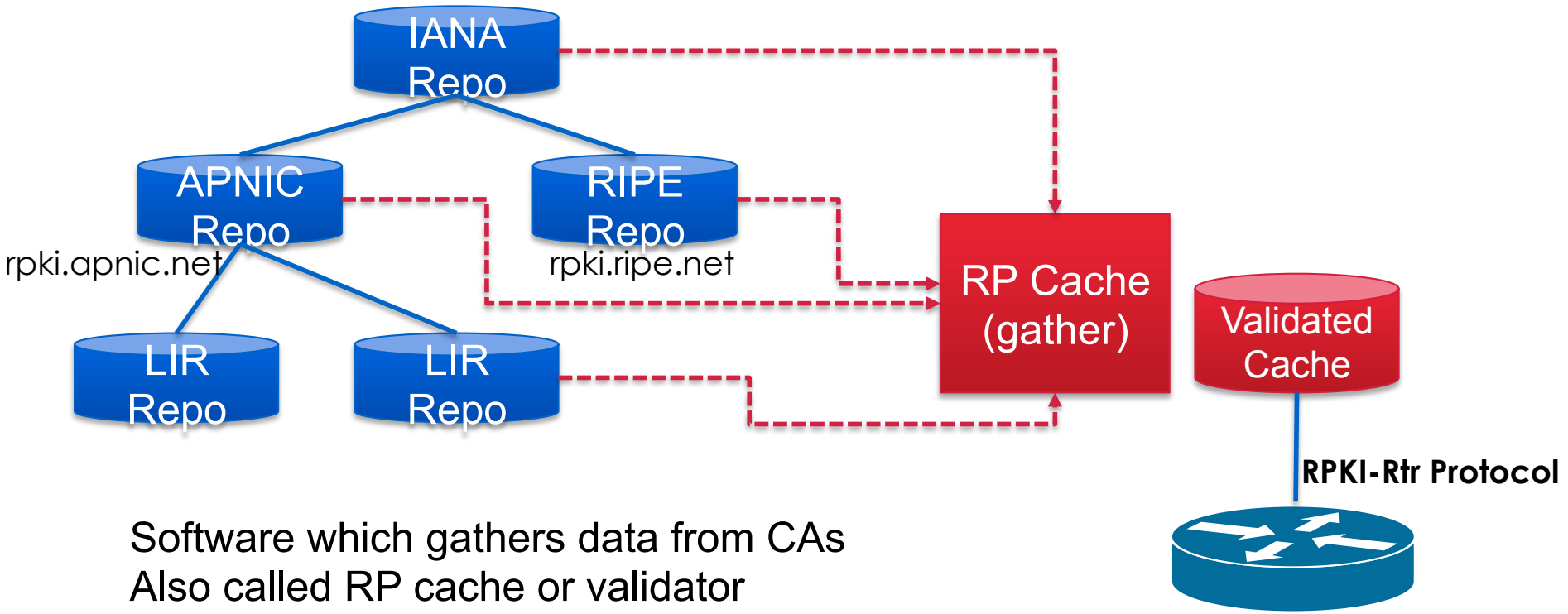
Issuing Party

- Internet Registries (RIR, NIR, Large LIRs)
- Acts as a Certificate Authority and issues certificates for customers
- Provides a web interface to issue ROAs for customer prefixes
- Publishes the ROA records



MyAPNIC GUI

Relying Party (RP)



Software which gathers data from CAs
Also called RP cache or validator

RPKI Building Blocks

1. Trust Anchors (RIR's)
2. Route Origination Authorizations (ROA)
3. Validators

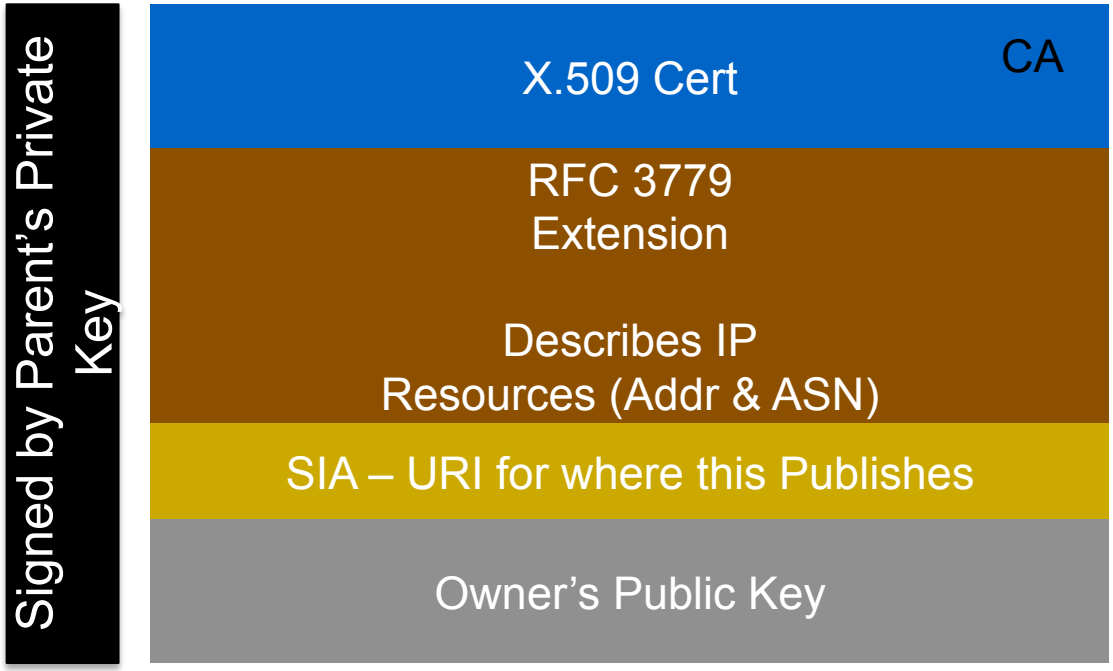
1. PKI & Trust Anchors

Public Key Concept

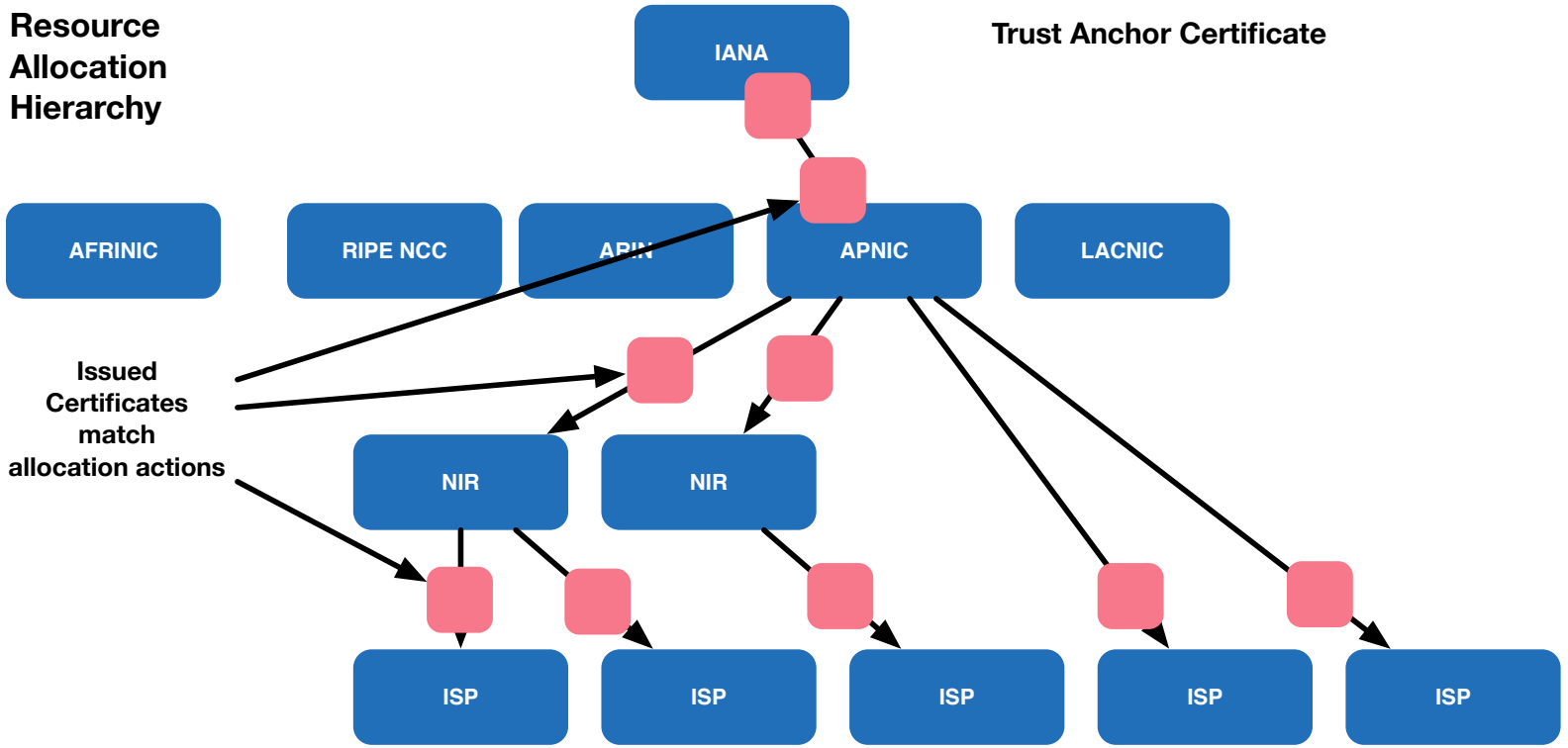
- **Private key**: This key must be known only by its owner.
- **Public key**: This key is known to everyone (it is public)
- **Relation between both keys**: What one key encrypts, the other one decrypts, and vice versa. That means that if you encrypt something with my public key (which you would know, because it's public :-), I would need my private key to decrypt the message.
- Same alike http with SSL aka https

X.509 Certificates 3779 EXT

Certificates are X.509 certificates that conform to the PKIX profile [PKIX]. They also contain an extension field that lists a collection of IP resources (IPv4 addresses, IPv6 addresses and AS Numbers) [RFC3779]



Trust Anchor



Source : http://isoc.org/wp/ietfjournal/?p=2438

RPKI Chain of Trust

- The RIRs hold a self-signed root certificate for all the resources that they have in the registry
  - They are the trust anchor for the system
- That root certificate is used to sign a certificate that lists your resources
- You can issue child certificates for those resources to your customers
  - When making assignments or sub allocations

2. ROA
Route Origin Authorizations

Route Origination Authorizations (ROA)

- A ROA is a **digitally signed object** that provides a means of **verifying** that an **IP address block holder** has **authorized** an **Autonomous System (AS)** to originate routes to one or more **prefixes** within the address block.
- With a **ROA**, the **resource holder is attesting** that the **origin AS** number is **authorized** to **announce** the **prefix(es)**. The attestation can be verified cryptographically using RPKI.

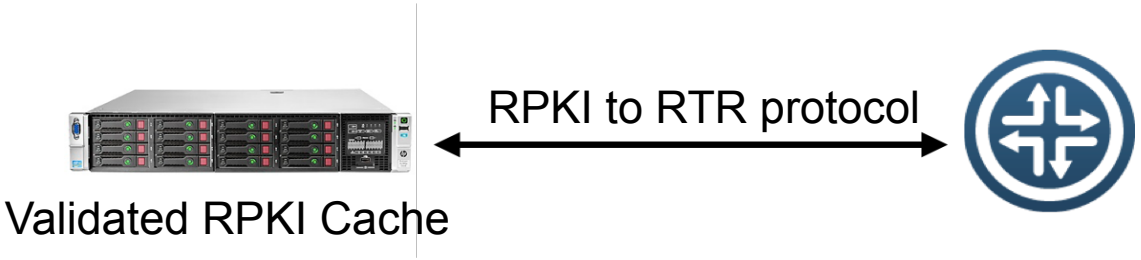
Route Origination Authorizations (ROA)

- Next to the prefix and the ASN which is allowed to announce it, the ROA contains:
  - A minimum prefix length
  - A maximum prefix length
  - An expiry date
  - Origin ASN
- Multiple ROAs can exist for the same prefix
- ROAs can overlap

3. Validators

Origin Validation

- Router gets ROA information from the RPKI Cache
  - RPKI verification is done by the RPKI Cache
- The BGP process will check each announcement with the ROA information and label the prefix



Result of Check

- **Valid** – Indicates that the prefix and AS pair are found in the database.
- **Invalid** – Indicates that the prefix is found, but either the corresponding AS received from the EBGP peer is not the AS that appears in the database, or the prefix length in the BGP update message is longer than the maximum length permitted in the database.
- **Not Found / Unknown**– Indicates that the prefix is not among the prefixes or prefix ranges in the database.

Valid > Unknown > Invalid

ROA Example

Prefix: 10.0.0.0/16
ASN: 65420

ROA	65420	10.0.0.0/16	/18
	Origin AS	**Prefix**	**Max Length**
VALID	AS65420	10.0.0.0/16	
VALID	AS65420	10.0.128.0/17	
INVALID	AS65421	10.0.0.0/16	
INVALID	AS65420	10.0.10.0/24	
UNKNOWN	AS65430	10.0.0.0/8	

Local Policy

- You can define your policy based on the outcomes
  - Do nothing
  - Just logging
  - Label BGP communities
  - Modify preference values
  - Rejecting the announcement

In summary

- As an announcer/LIR
  - You choose if you want certification
  - You choose if you want to create ROAs
  - You choose AS, max length
- As a Relying Party
  - You can choose if you use the validator
  - You can override the lists of valid ROAs in the cache, adding or removing valid ROAs locally
  - You can choose to make any routing decisions based on the results of the BGP Verification (valid/invalid/unknown)

RPKI Caveats

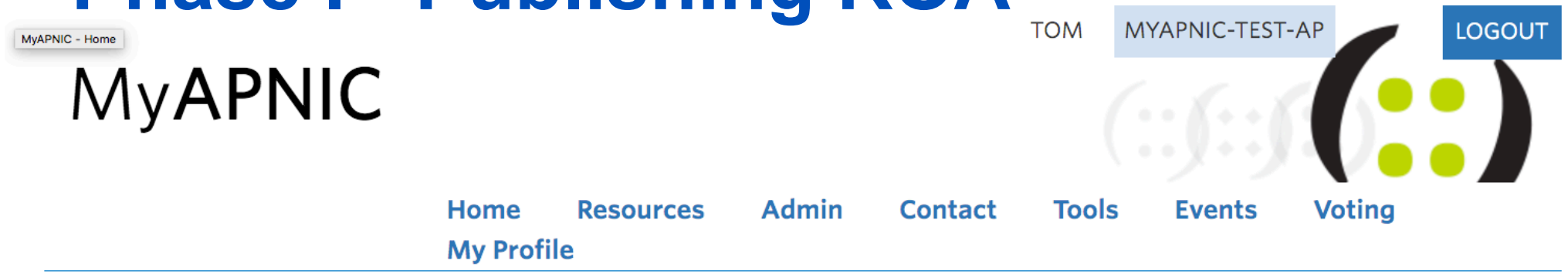
- When RTR session goes down, the RPKI status will be “not found” for all the bgp routes after a while
  - Invalid => not found
  - we need several RTR sessions or care your filtering policy
- In case of the router reload, which one is faster, receiving ROAs or receiving BGP routes?
  - If receiving BGP is match faster than ROA, the router propagate the invalid route to others
  - We need to put our Cache validator within our IGP scope

RPKI Configuration

RPKI Configuration

- Resources:
  - AS : 45192 [APNIC-TRAINING-DC-AS-AP]
  - IPv4 : 203.176.189.0/24
  - IPv6: 2001:DF0:A::/48
- Process
  - Create ROA
  - Setup cache validation server
  - Validate the ROA

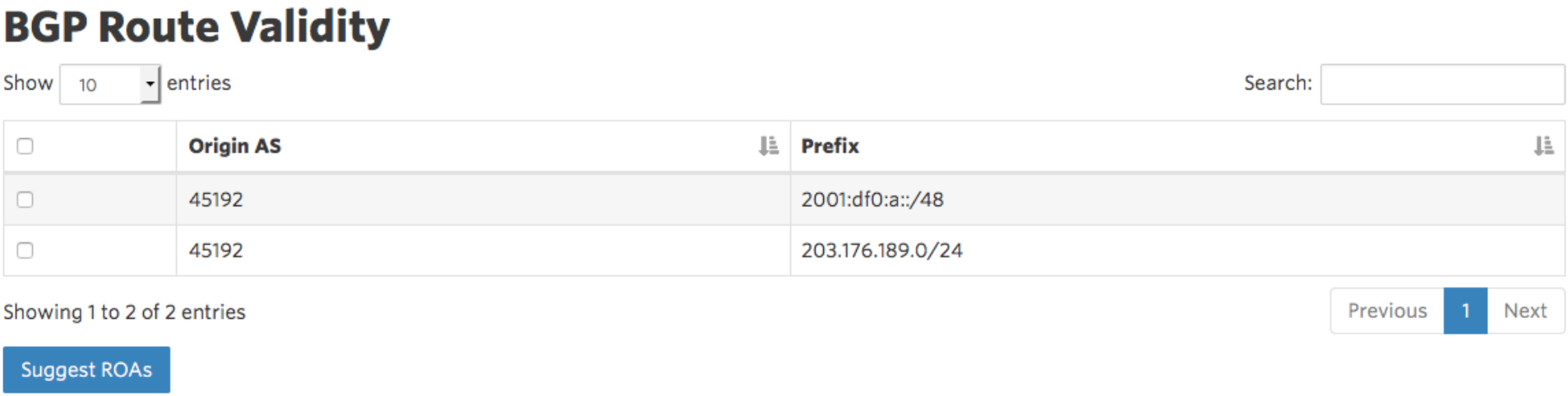
Phase I - Publishing ROA



Login to your MyAPNIC portal
Requires a valid certificate
Go to Resources > Certification Tab

Phase I - Publishing ROA

- Show available prefix for which you can create ROA



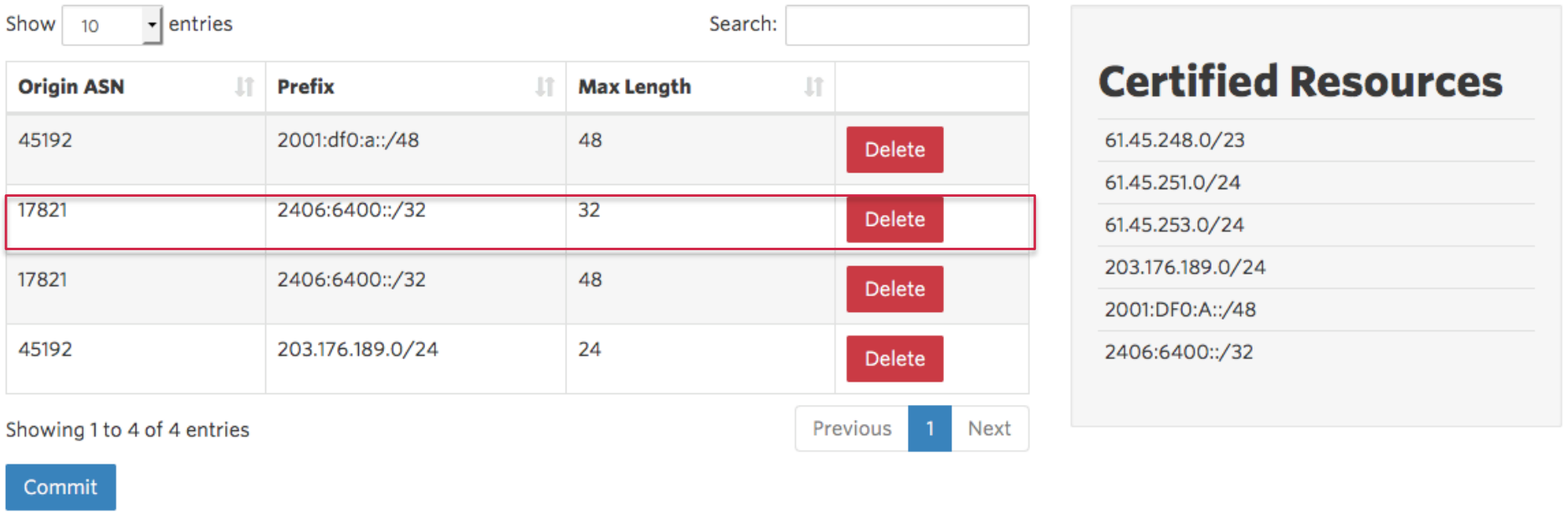
Phase I - Publishing ROA

ROA Configuration

1. Write your ASN
2. Your IP Block
3. Subnet
4. Click Add

Phase I - Publishing ROA

Create ROA for smaller block.



Phase I - Check your ROA

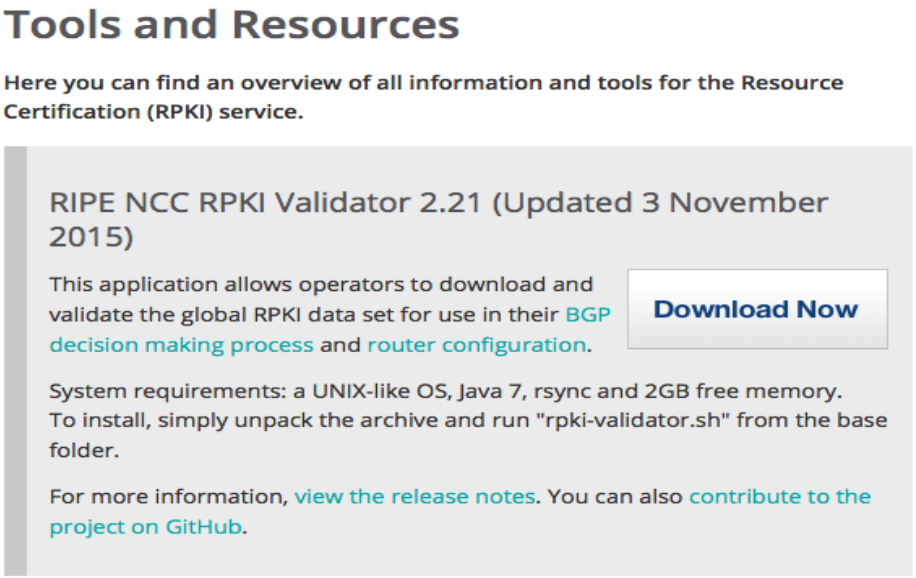
```
whois -h whois.bgpmon.net 2001:df0:a::/48

Prefix:                 2001:df0:a::/48
Prefix description:     APNIC Training data centre
Country code:           AU
Origin AS:              45192
Origin AS Name:         Two-byte AS number for APNIC Training
Data Centre
RPKI status:            ROA validation successful
First seen:             2013-12-11
Last seen:              2016-01-03
Seen by #peers:         170
```

Phase I - Check your ROA

```
whois -h whois.bgpmon.net " --roa 45192 2001:df0:a::/48"

0 - Valid
------------------------
ROA Details
------------------------
Origin ASN:       AS45192
Not valid Before: 2016-01-02 02:30:14
Not valid After:  2020-07-27 00:00:00  Expires in
4y204d23h46m30.4000000059605s
Trust Anchor:     rpki.apnic.net
Prefixes:         2001:df0:a::/48 (max length /48)
                  203.176.189.0/24 (max length /24)
```

Phase II - RPKI Validator

- Download RPKI Validator
  - http://www.ripe.net/lir-services/resource-management/certification/tools-and-resources

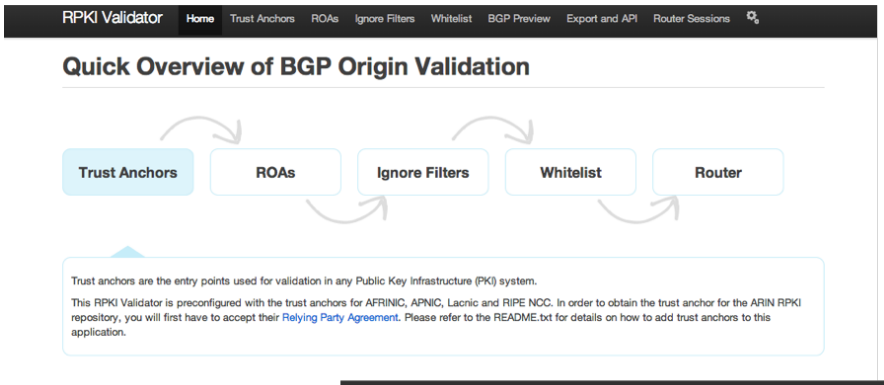


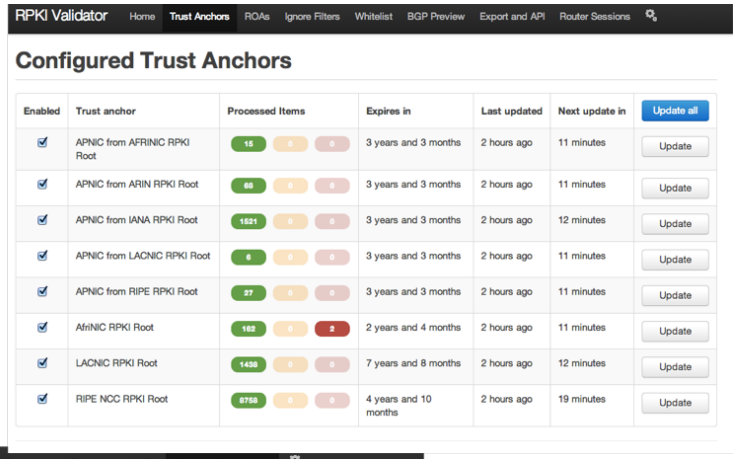
Phase II - RPKI Validator

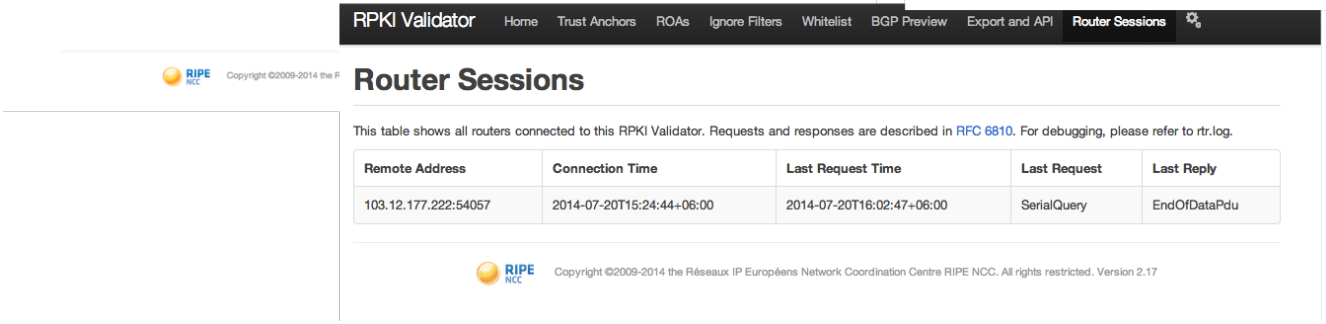
```
# tar -zxvf rpki-validator-app-2.21-dist.tar.gz
# cd rpki-validator-app-2.21
# ./rpki-validator.sh start
```

Phase II - RPKI Validator

http://ip_address:8080







Phase III - Router Configuration

Junos

```
set routing-options validation group RPKI session 202.4.96.221 refresh-time
120
set routing-options validation group RPKI session 202.4.96.221 hold-time 180
set routing-options validation group RPKI session 202.4.96.221 port 8282
set routing-options validation group RPKI session 202.4.96.221 local-address
103.21.75.1
```

IOS

```
router bgp 64500
  bgp log-neighbor-changes
  bgp rpki server tcp 202.4.96.221 port 8282 refresh 120
```

Phase III - Router Configuration

Junos

```
set policy-options policy-statement ROUTE-VALIDATION term valid from protocol bgp
set policy-options policy-statement ROUTE-VALIDATION term valid from validation-database valid
set policy-options policy-statement ROUTE-VALIDATION term valid then local-preference 110
set policy-options policy-statement ROUTE-VALIDATION term valid then validation-state valid
set policy-options policy-statement ROUTE-VALIDATION term valid then accept
```

```
set policy-options policy-statement ROUTE-VALIDATION term invalid from protocol bgp
set policy-options policy-statement ROUTE-VALIDATION term invalid from validation-database
invalid
set policy-options policy-statement ROUTE-VALIDATION term invalid then local-preference 90
set policy-options policy-statement ROUTE-VALIDATION term invalid then validation-state invalid
set policy-options policy-statement ROUTE-VALIDATION term invalid then accept
```

```
set policy-options policy-statement ROUTE-VALIDATION term unknown from protocol bgp
set policy-options policy-statement ROUTE-VALIDATION term unknown from validation-database
unknown
set policy-options policy-statement ROUTE-VALIDATION term unknown then local-preference 100
set policy-options policy-statement ROUTE-VALIDATION term unknown then validation-state unknown
set policy-options policy-statement ROUTE-VALIDATION term unknown then accept
```

Phase III - Router Configuration

2. Configure policy to tag ROA

IOS

```
!
route-map ROUTE-VALIDATION permit 10
 match rpki invalid
 set local-preference 90
!
route-map ROUTE-VALIDATION permit 20
 match rpki not-found
 set local-preference 100
!
route-map ROUTE-VALIDATION permit 30
 match rpki valid
 set local-preference 110
```

Phase III - Router Configuration

3. Push policy to the BGP neighbour

Junos

```
set protocols bgp import ROUTE-VALIDATION
```

IOS

```
router bgp 64500
  bgp log-neighbor-changes
  !other neighbour related configuration
  neighbor 10.1.1.2 route-map ROUTE-VALIDATION in
```

Check your prefix

Junos

```
show route protocol bgp 203.176.189.0

inet.0: 575802 destinations, 575803 routes (575802 active, 0
holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

203.176.189.0/24   *[BGP/170] 1w3d 23:50:49, localpref 100, from
79.141.168.1
                      AS path: 33926 2828 7545 24130 4608 45192
17821 I, validation-state: unknown
                    > to 193.34.50.1 via em0.0
```

Check your prefix

IOS

```
rpki-rtr>show ip bgp 203.176.189.0/24
BGP routing table entry for 203.176.189.0/24, version 70470025
Paths: (2 available, best #2, table default)
  Not advertised to any peer
  Refresh Epoch 1
  3333 1273 4637 1221 4608 45192
    193.0.19.254 from 193.0.3.5 (193.0.0.56)
      Origin IGP, localpref 110, valid, external
      Community: 83449328 83450313
      path 287058B8 RPKI State valid
```

Commands

Command (Junos)	Description
`show validation session detail`	Check session status of cache validator server
`show validation statistics`	Statistics on valid/invalid prefixes
`show validation database`	Full validation database
`show route protocol bgp validation-state valid/invalid/ unknown`	Find valid/invalid/unknown routes

!Caution!

```
CMD: 'show ip bgp ' 18:26:21 BDT Mon Mar 17 2014

CMD: 'show ip bgp ' 18:26:34 BDT Mon Mar 17 2014

CMD: 'show ip bgp ' 18:27:55 BDT Mon Mar 17 2014

CMD: 'show ip bgp ' 18:29:20 BDT Mon Mar 17 2014

CMD: 'show ip bgp rpki table ' 18:29:31 BDT Mon Mar 17 2014

CMD: 'show ip bgp rpki servers ' 18:29:34 BDT Mon Mar 17 2014

CMD: 'show ip bgp rpki table ' 18:29:49 BDT Mon Mar 17 2014

Exception to IOS Thread:
Frame pointer 0x7F3A8AA51EE0, PC = 0x8DA4DA

UNIX-EXT-SIGNAL: Segmentation fault(11), Process = BGP Router
-Traceback= 1#270a78af3c82800fb448b5d32a66d575  :400000+4DA4DA :400000+73AB56B :4000
400000+4980EA :400000+4A64DD :400000+496ED5

Fastpath Thread backtrace:
-Traceback= 1#270a78af3c82800fb448b5d32a66d575  c:7F3B7C28C000+BDDD2

Auxiliary Thread backtrace:
-Traceback= 1#270a78af3c82800fb448b5d32a66d575  pthread:7F3B774EB000+A7C9

RAX = 0000000000000008  RBX = 00007F3A8AA520A0
RCX = 8039F30F00000000  RDX = 0000000000000000
RSP = 00007F3A8AA51EE0  RBP = 00007F3A8AA51FE0
RSI = A020A58A3A7F0000  RDI = D8803CB53A7F0000
R8  = A020A58A3A7F0000  R9  = 00007F3AB53C80D8
R10 = 00007F3A83A6B221  R11 = 0000000000000001
R12 = 00007F3AB53C80D8  R13 = 00007F3A8AA52110
R14 = FFF7000600000000  R15 = 00007F3A8AA52094
RFL = 0000000000010293  RIP = 00000000008DA4DA
CS = 0033  FS = 0000  GS = 0000
ST0 = 0000 0000000000000000  ST1 = 0000 0000000000000000
ST2 = 0000 0000000000000000  ST3 = 0000 0000000000000000
ST4 = 0000 0000000000000000  ST5 = 0000 0000000000000000
ST6 = 0000 0000000000000000  ST7 = 0000 0000000000000000
X87CW = 037F  X87SW = 0000  X87TG = 0000  X87OP = 0000
X87IP = 0000000000000000  X87DP = 0000000000000000
XMM0  = A81F718A3A7F00009802598A3A7F0000
```

Testbed

- **Cisco (hosted by the RIPE NCC)**
  - Public Cisco router: rpki-rtr.ripe.net
  - Telnet username: ripe / No password
- **Juniper (hosted by Kaia Global Networks)**
  - Public Juniper routers: 193.34.50.25, 193.34.50.26
  - Telnet username: rpki / Password: testbed

Configuration - Reference Link

- **Cisco**
  - http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_bgp/command/irg-cr-book/bgp-m1.html#wp3677719851
- **Juniper**
  - http://www.juniper.net/techpubs/en_US/junos12.2/topics/topic-map/bgp-origin-as-validation.html

www.apnic.net/roa

QUESTIONS?

www.facebook.com/APNIC

www.twitter.com/apnic

www.youtube.com/apnicmultimedia

www.flickr.com/apnic

www.weibo.com/APNICrir