

Internet Routing Registry Tutorial

Prerequisites

- You should have some idea of how Internet peering and transit works
- You should have conceptual BGP skills
- You should know how to manipulate objects in a WHOIS database

The IRR

- Concept of “the” Internet Routing Registry system established in 1995
- Web site at <http://www.irr.net>
- Initially RIPE-(1)81 format, shifted to RPSL
- Mirror routing registry data in a common repository for simplified queries - “the union of world-wide routing policy databases”

The IRR

- Today, consists of about 40 registries operated by
 - RIRs (AfrINIC / RIPE)
 - ISPs (NTT / CenturyLink, C&W)
 - Non-affiliated public registries (RADB / ALTDB)

The RADB

- **R**outing **A**rbitrator **D**ata**B**ase (managed by Merit)
- One of the earliest routing registry databases

Why use an IRR?

- Document routing policy
- Register route objects to associate network prefixes with origin AS
- Solves the problem of: What prefixes should my peer / customer be announcing to me?

Why use an IRR?

- A number of transit providers require their customers to register routes and filter customer route announcements based on registry contents.
- Filters prevent unauthorised announcements; protects against route hijacking, denial of service, etc

Querying the IRR

- Historically, IRRs have the “WHOIS” protocol (TCP 43)
- Two primary IRR server implementations
 - RIPE DB from RIPE NCC
 - IRRd server from Merit
- Some IRRs offer Web/REST based queries
- Possible to run your own IRRd.

RPSL specifics

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

Using the IRR

- You need an AS number to use a registry (Ask your RIR)
- You need a `mntner` object (ie. be safe)
- You need an `autnum` object (ie. have an ASN)
- You need `route` object(s)

`mntner` object

- `mntner` is an abbreviation of maintainer
- identifies accounts in the registry
- specifies authentication mechanism in the “auth” attribute. Either:
 - PGP-KEY - PGP/GPG based auth
 - (B)CRYPT-PW / MD5-PW - password auth
 - MAIL-FROM - email based auth

mntner object

- `mntner` is an abbreviation of maintainer
- identifies accounts in the registry
- specifies authentication mechanism in the “auth” attribute. Either:
 - PGP-KEY - PGP/GPG based auth
 - BCRYPT-PW
 - CRYPT-PW / MD5-PW - password auth
 - MAIL-FLOW - email based auth

DEPRECATED

Sample mntner object

mntner:	[mandatory]	[single]	[primary/look-up key]
descr:	[mandatory]	[multiple]	[]
org:	[optional]	[multiple]	[inverse key]
admin-c:	[mandatory]	[multiple]	[inverse key]
tech-c:	[optional]	[multiple]	[inverse key]
upd-to:	[mandatory]	[multiple]	[inverse key]
mnt-notify:	[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]
changed:	[mandatory]	[multiple]	[]
source:	[mandatory]	[single]	[]

aut-num object

- 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:      AS42
as-name:      UNSPECIFIED
descr:        Packet Clearing House - www.pch.net
admin-c:      Bill Woodcock
tech-c:       Bill Woodcock
export:       to AS-ANY    announce AS-PCH
remarks:      peering@pch.net, +1 866 BGP PEER
notify:       radb@pch.net
mnt-by:       MAINT-AS3856
changed:      scg@pch.net 20041121
source:       RADB
```

Alternate aut-num uses

- Often used to register BGP community support offered by service providers

Example: `whois -h whois.radb.net AS1273`

*For a more comprehensive list, see:
<http://www.onesc.net/communities>*

route object

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

Sample route object

```
route:          160.0.0.0/17
descr:         Packet Clearing House
origin:        AS715
notify:        radb@pch.net
mnt-by:        MAINT-AS3856
changed:       kabindra@pch.net 20170705
source:        RADB
```

route object key

- Every RPSL object 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 router and origin fields as the primary key

route object key

- 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

route object key example

- However, many stale objects exists (ISPs are lazy!)

- `whois -h whois.radb.net 158.80.0.0/21`

(look at the dates)

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
G-1050 West Bristol Road
Flint
MI 48507-5508, USA

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
- Functionally equivalent to IPv4

Sample route6 object

```
route6:      2001:43f8:110::/48
descr:      AFRINIC-RFC5855
origin:     AS37181
mnt-by:     AFRINIC-IT-MNT
source:     AFRINIC # Filtered
```


as-set object

- Provides a way of grouping ASes. Name must begin with the prefix "AS-"
- Frequently used to list downstream/customer AS numbers
- May be referenced in aut-num import/export policy expressions
- Can reference another as-set

Sample as-set object

```
whois -h whois.radb.net AS-PCH
```

as-set: AS-PCH

descr: ASes announced by Packet Clearing House

members: AS3856, AS42, AS715, AS-RS, AS32978, AS32979,
AS35160, AS38052, AS16668, AS44876, AS45170, AS297, AS45494,
AS27678, AS52306, AS52234, AS54145, AS187, AS27, AS54390,
AS11893, AS52304, AS21556, AS19281, AS10886

admin-c: Bill Woodcock

tech-c: Bill Woodcock

notify: radb@pch.net

mnt-by: MAINT-AS3856

changed: kabindra@pch.net 20171013

source: RADB



PeeringDB

Search here for a network, |

[Advanced Search](#)

Packet Clearing House AS42

Organization	Packet Clearing House
Also Known As	Woodynet, PCH
Company Website	http://www.pch.net/
Primary ASN	42
IRR Record	AS-PCH
Route Server URL	
Looking Glass URL	http://lg.pch.net
Network Type	Educational/Research
IPv4 Prefixes	600
IPv6 Prefixes	600

Look familiar?

Pro-tip: Try to make the name something meaningful and easy to guess

More reading

- 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

4byte / 32bit ASNs

- RFC 4893 defines 32bit ASN support
- RFC 5396 standardised representation
 - asplain format uses simple integers (AS327576 vs. AS5.1)
- RPSL implementations and routing registries have 32bit ASN support

<pause>

Sample queries

- IRRs support a number of flag options. eg. “-i” flag performs inverse query
 - “-i mnt-by MAINT-AS3856” returns all routes objects maintained by MAINT-AS3856
 - “-i origin AS42” returns all route objects with an origin of AS42
- -M flag returns more specific router objects for a prefix
 - “-M 70.40.0.0/21” returns more specific objects in the 70.40.0.0/21 prefix

More queries

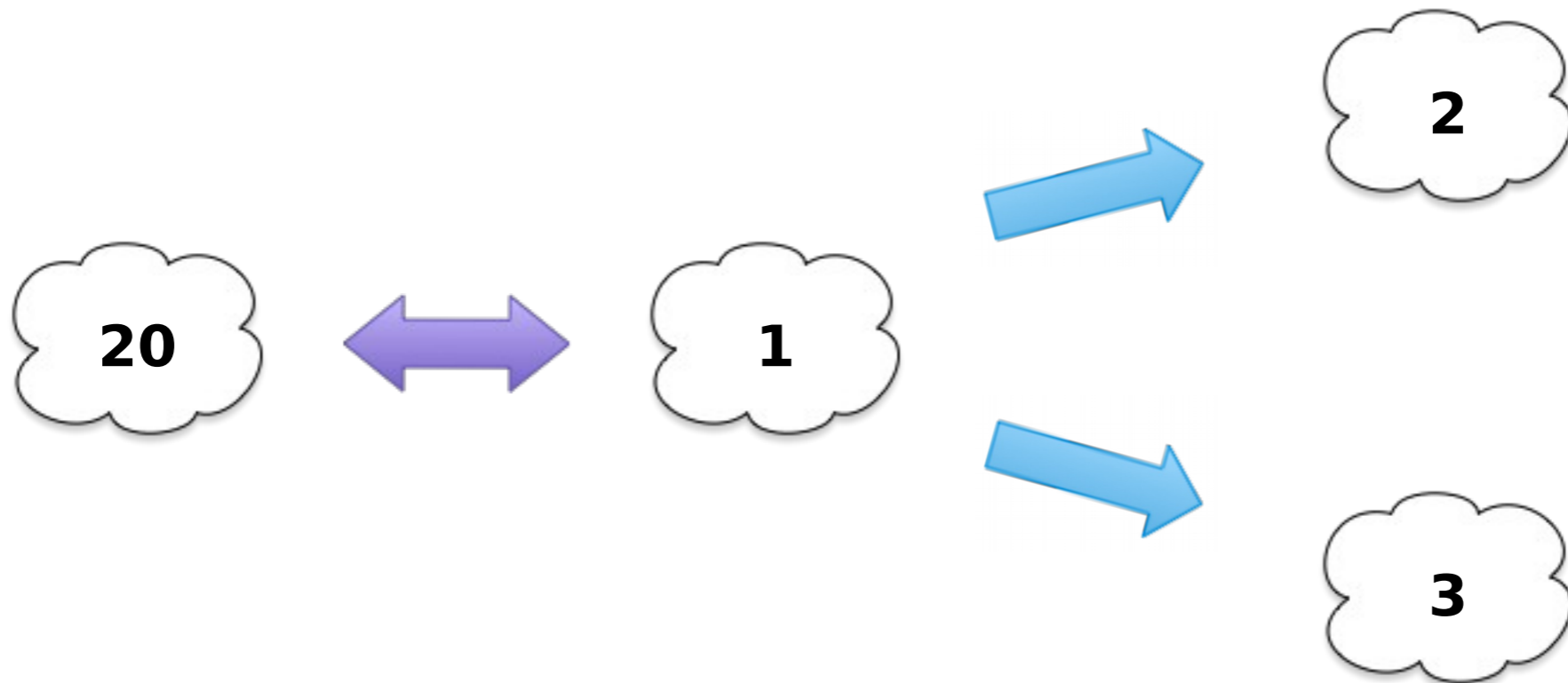
- -s flag limits the sources queried
 - “-s RADB,AFRINIC”
 - -K flag - return primary keys only
 - Useful for router object queries; excludes extraneous fields not usually needed for policy

- “-K 70.40.0.0” returns
route: 70.40.0.0/21
origin: AS42

More on RPSL

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

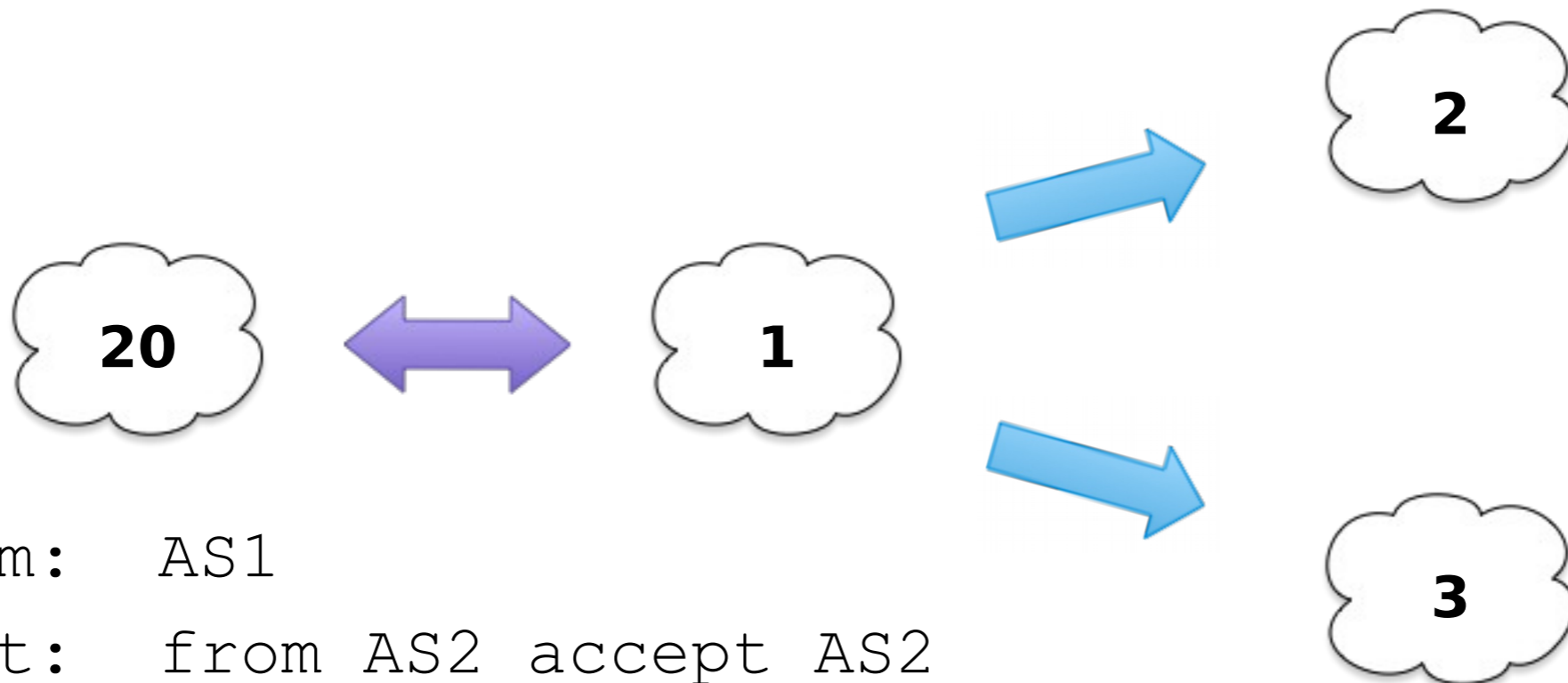
Routing policy



AS1 provides transit to AS2 and AS3

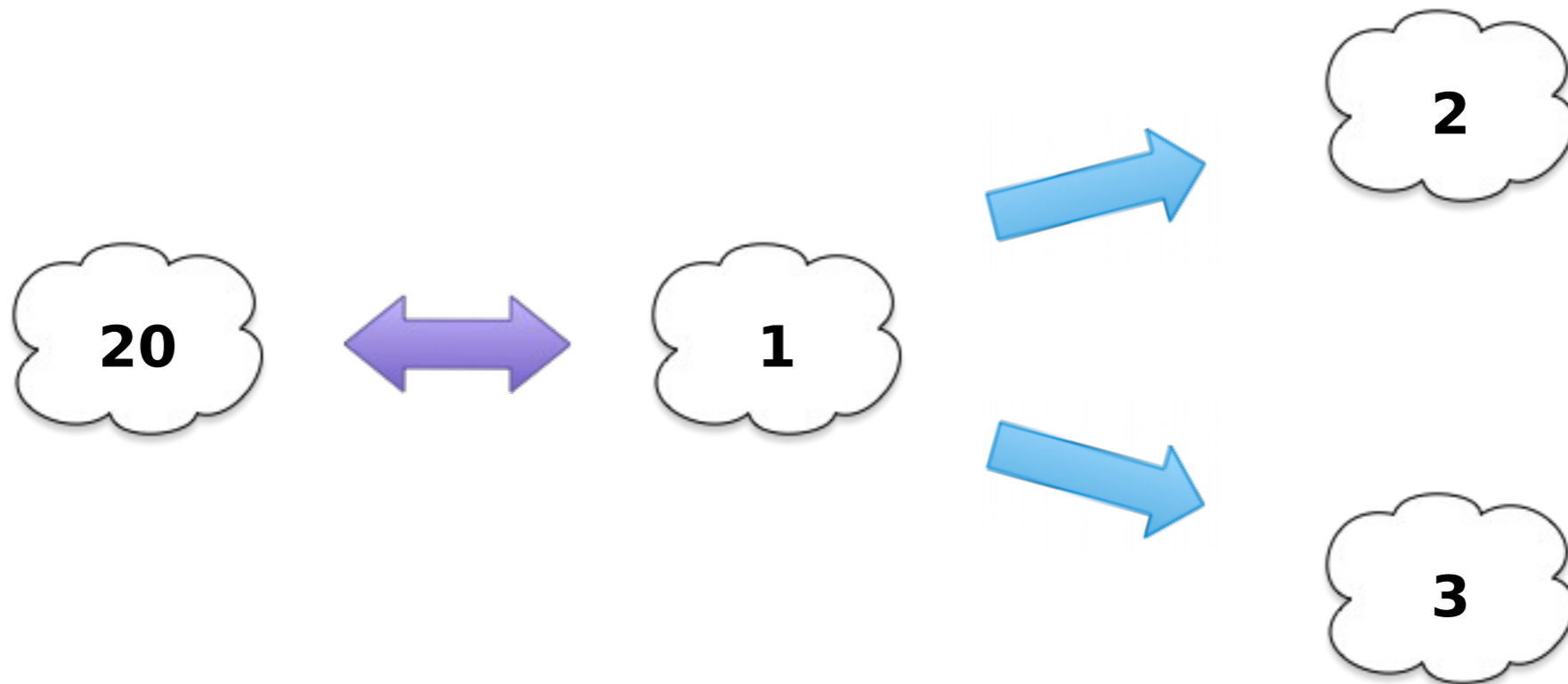
AS1 peers with AS20

in RPSL



```
autnum: AS1
import: from AS2 accept AS2
import: from AS3 accept AS3
import: from AS20 accept AS20
export: to AS2 permit ANY
export: to AS3 permit ANY
export: to AS20 permit AS1 AS2 AS3
```

using as-set



```
autnum: AS-MY-ASONE
```

```
...
```

```
export: to AS20 permit AS-MY-ASONE
```

IRR Tools

- IRRToolSet (<http://irrtolset.isc.org>)
- NET::IRR
- RPSLtool - (<http://www.linux.it/~md/software/>)
- IRRPT (<https://sourceforge.net/projects/irrpt/>)
- bgpq3 (<http://snar.spb.ru/prog/bgpq3/>)
- filtergen (Level 3)
 - `whois -h filtergen.level3.net SOURCE::AS-SET`
 - `whois -h filtergen.level3.net RADB::AS-PCH`

Problems with the IRR

- Accuracy is not maintained
- Verification is not possible
- No consistency in usage

Problems with the IRR

- Accuracy is not maintained
- Verification is not possible
- No consistency in usage

...and we'll cover these
later

Scenario #1:

- You get new IP address space from your RIR. What are your actions?

Scenario #1:

- You get new IP address space from your RIR. What are your actions?

Register new route object.

Origin ASN = your ASN

Scenario #2:

- One of your customers gets new address space from [...]? What are your actions?

Scenario #2:

- One of your non-BGP customers gets new address space from [..]? What are your actions?

Verify the address space using WHOIS

Register a proxy route object using your ASN

Scenario #3:

- You get a new BGP capable customer. What are your actions?

Scenario #3:

- You get a new BGP capable customer. What are your actions?

Get your customer to register their routes (or AS-SET)

Append their AS (or AS-SET) to your AS-SET

IRRPT

Quick intro

Getting it running

- Download it from Github.
- Run `php configure.php`
- Fix issues.
- Profit in time :-)

Generating router configs

Replace Cisco with \$preferred brand

```
root@Graphing:~/irrpt-master# bin/irrpt_pfxgen -f cisco 42
conf t
no ip prefix-list CUSTOMER:42
no ipv6 prefix-list CUSTOMERv6:42
ip prefix-list CUSTOMER:42 permit 4.67.64.0/22 le 24
ip prefix-list CUSTOMER:42 permit 9.9.9.0/24
ip prefix-list CUSTOMER:42 permit 31.135.128.0/19 le 24
ip prefix-list CUSTOMER:42 permit 38.124.249.0/24
<snip>
ipv6 prefix-list CUSTOMERv6:42 permit 2800:110:10::/48
ipv6 prefix-list CUSTOMERv6:42 permit 2801:140:10::/48
end
write mem
```

Generating mikrotik configs

- Mikrotik needs an additional wrapper.
- Download and unzip script into working directory

<https://edd.za.net/download/mkirrpt.zip>

./mk.sh AS42-infilter 42

```
root@Graphing:~/mikrotik# ./mk.sh AS42filters 42
/routing filter set [ find where chain=AS42filters-IPv4 ] comment="deleteme:";
/routing filter set [ find where chain=AS42filters-IPv6 ] comment="deleteme:";
/routing filter add chain=AS42filters-IPv4 prefix=4.67.64.0/22 prefix-length=22-24 action=accept
/routing filter add chain=AS42filters-IPv4 prefix=9.9.9.0/24 action=accept
/routing filter add chain=AS42filters-IPv4 prefix=31.135.128.0/19 prefix-length=19-24 action=accept
/routing filter add chain=AS42filters-IPv4 prefix=38.124.249.0/24 action=accept
/routing filter add chain=AS42filters-IPv4 prefix=45.221.0.0/22 prefix-length=22-24 action=accept
/routing filter add chain=AS42filters-IPv4 prefix=45.221.16.0/22 prefix-length=22-24 action=accept
/routing filter add chain=AS42filters-IPv4 prefix=45.250.60.0/22 prefix-length=22-24 action=accept

<snip>
/routing filter add chain=AS42filters-IPv6 prefix=2801:140:10::/48 action=accept
/routing filter add chain=AS42filters-IPv6 action=reject
/routing filter remove [ find where chain=AS42filters-IPv4 and comment="deleteme:" ]
/routing filter remove [ find where chain=AS42filters-IPv6 and comment="deleteme:" ]
```

Batch filter generation!

- Edit as.txt with asns or route sets
- `./batchmikrotik.sh > rules.txt`
- Copy to mikrotik
- Import \$filename

Want notices when prefixes change?

- Edit `conf/irrdp.conf`
- Cron `bin/irrp_fetch`
- Receive email once it changes.

Other useful things

- Plug it into Rancid,
- Use `Net::Telnet::Cisco` or `JUNOScript` to dump configs to routers

Problems

Suffers with big route sets eg. he.net

bgpq3

Using bgpq3

- We're going to use bgpq3 (because it's fast) to help us create filters for some of our peers.
- Install bgpq3 on a *NIX host (or if you're forced to use Windows ask someone here for a shell)
- Find it in your OS repository, or download from GH:
<https://github.com/snar/bgpq3>

Supplementary tools

- ixgen: <https://github.com/ipcjk/ixgen>
- pinder: <https://github.com/dotwaffle/pinder>

LibreNMS +

Downstream



Local address	Peer address	Type	Family	Remote AS	Peer description	State	Uptime / Updates
br1.bre.inx	» 2001:43f8:1f4::2	iBGP	ipv6.unicast	AS37663 CINX, ZA		start active	Updates ↓ 0 ↑ 0
br1.bre.inx	» 196.223.22.1	eBGP	ipv4.unicast	AS37701 CINX-ROUTESRV, ZA		start idle	Updates ↓ 0 ↑ 0
br1.bre.inx	» 196.223.22.2	eBGP	ipv4.unicast	AS37701 CINX-ROUTESRV, ZA		start idle	Updates ↓ 0 ↑ 0
br1.dpr.inx	» 196.10.54.2	iBGP	ipv4.unicast	AS37663 CINX, ZA		start idle	Updates ↓ 0 ↑ 0
br1.dpr.inx	» 2001:43f8:1f4::2	iBGP	ipv6.unicast	AS37663 CINX, ZA		start idle	Updates ↓ 0 ↑ 0
br1.nls.inx	» 196.10.54.2	iBGP	ipv4.unicast	AS37663 CINX, ZA		start idle	Updates ↓ 0 ↑ 0
br1.nls.inx	» 2001:43f8:1f4::2	iBGP	ipv6.unicast	AS37663 CINX, ZA		start idle	Updates ↓ 0 ↑ 0
br2.pkl.inx	» 196.10.53.84	eBGP	ipv4.unicast	AS42 WOODYNET-1 - WoodyNet, US		start idle	Updates ↓ 0 ↑ 0
br2.pkl.inx	» 2001:43f8:1f3:e00::4	eBGP	ipv6.unicast	AS42 WOODYNET-1 - WoodyNet, US		start idle	Updates ↓ 0 ↑ 0
br2.pkl.inx	» 196.10.53.85	eBGP	ipv4.unicast	AS715 WOODYNET-2 - WoodyNet, US		start idle	Updates ↓ 0 ↑ 0

RPKI

RPKI

- Provides a cryptographically verifiable means to validate information that is in the database.
- Solves the question of: **Is that ASN authorised to originate that prefix**
- Often called: “Origin Validation”

RPKI

- Concept of private and personal keys hasn't changed.
- 2 implementation methods (delegated or hosted)

RPKI Building blocks

- Trust Anchors
- ROAs
- Validators

RPKI

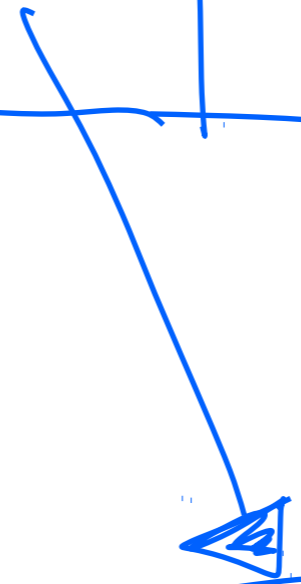
- Builds trust by building a chain of certificates
- TA (Trust Anchor) being the top most CA
- EE certificates at the leaf level (ROA)
- Certificates contain Internet resources
- Validation works by running the chain of trust from root to leaves

TAL

AFRINIC



105/8



105-208-130

ROA

What is a ROA

- A ROA is a **digitally signed object** that provides a means of **verifying** that an **IP Address block holder** has **authorised** an **Autonomous System (AS)** to **originate routes** to one or more prefixes within the address block.

What is a ROA

- A ROA is a **digitally signed object** that provides a means of **verifying** that an **IP Address block holder** has **authorised** an **Autonomous System (AS)** to **originate routes** to one or more prefixes within the address block.

ie. x509 cert ...

ROAs

- Simply construct of:
 - prefix
 - asn
 - min + max prefix_length
 - expiry date

- ROAs can overlap

- Multiple ROAs can exist

Trust anchors

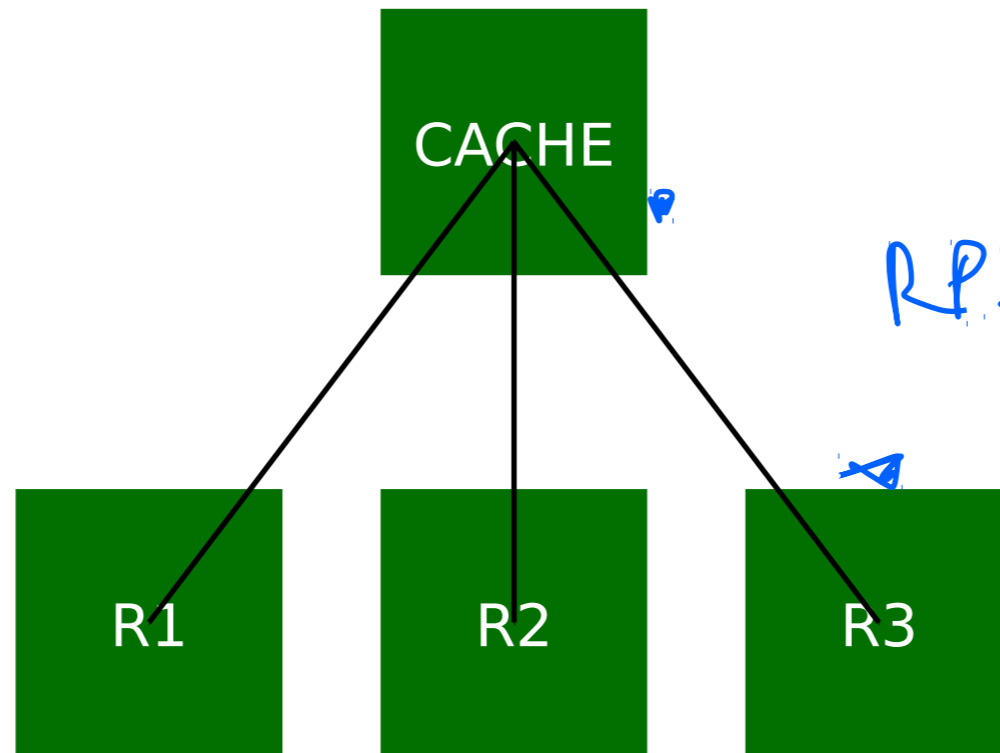
- RIRs have these for the majority blocks
- RIRs have complicated rules for dealing with minority blocks
- 4x RIRs publish these easily; ARIN makes you sign some legal stuff
- A URL and a Public Key that must be able to decrypt the cert found at the URL (so you know you can trust it)

Validators

- Software.
 - Current favorite : Routinator 3000
 - <https://nlnetlabs.nl/projects/rpki/routinator/>
 - RIPE NCC V2 (v3 in dev)
- Speaks rsync to trust anchors to synchronise ROAs
- Performs validation
- Speaks RPKI-RTR protocols to routers


Validators

- Produces a result that is either
 - 0 - NotFound
 - 1 - Valid
 - 2 - Invalid



RPKI - RTR

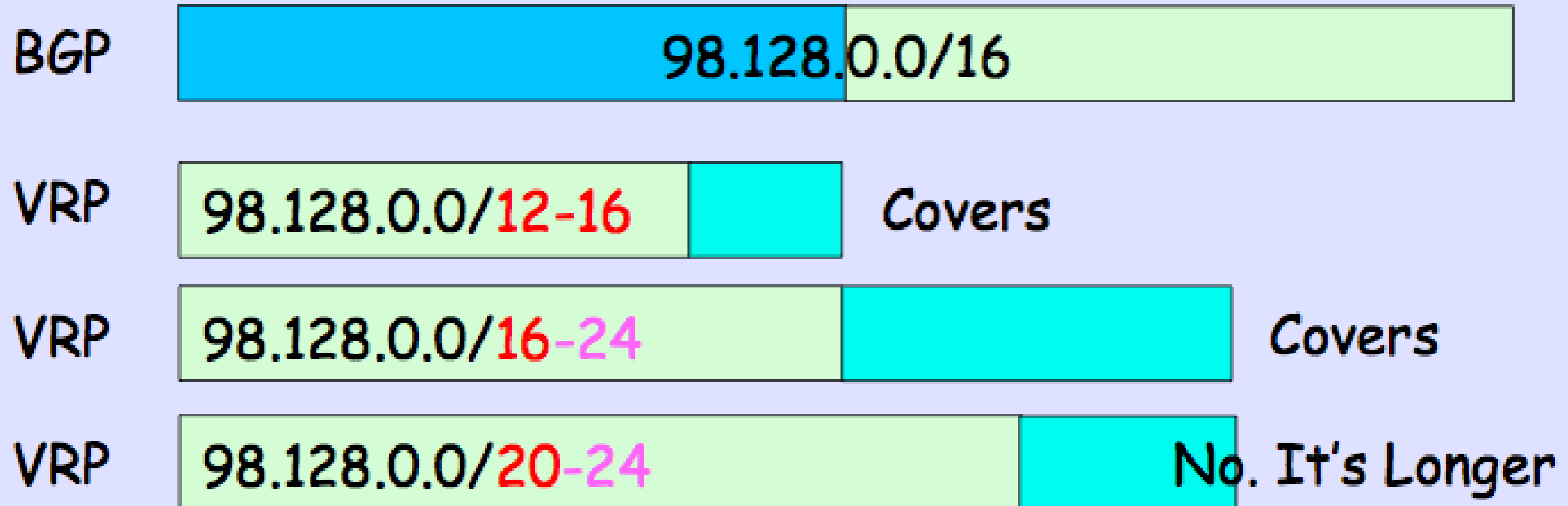
Configuring your device

- <https://www.inx.net.za/display/pub/RPKI+Validation>
- Cisco IOS 15.2+
- Cisco IOS/XR 4.3.2+
- JunOS 12.2+
- Mikrotik v7.x 

What are the BGP / VRRP¹ Matching Rules?

¹ Validated ROA Payload

A Prefix is Covered by a VRP when the VRP prefix length is less than or equal to the Route prefix length



Prefix is Matched by a VRP when the Prefix is Covered by that VRP , prefix length is less than or equal to the VRP max-len, and the Route Origin AS is equal to the VRP's AS

BGP

98.128.0.0/16 AS 42

VRP

98.128.0.0/12-16 AS 42

Matched

VRP

98.128.0.0/16-24 AS 666

No. AS Mismatch

VRP

98.128.0.0/20-24 AS 42

No. VRP Longer

Matching and Validity

VRP₀ 98.128.0.0/16-24 AS 6

VRP₁ 98.128.0.0/16-20 AS 42

BGP 98.128.0.0/12 AS 42 NotFound, shorter than VRPs

BGP 98.128.0.0/16 AS 42 Valid, Matches VRP₁

BGP 98.128.0.0/20 AS 42 Valid, Matches VRP₁

BGP 98.128.0.0/24 AS 42 Invalid, longer than VRP with AS 42

BGP 98.128.0.0/24 AS 6 Valid, Matches VRP₀

In real life

```
conf t
```

```
router bgp 37474
```

```
bgp rpki server tcp 196.10.53.22 port 3323 refresh  
600
```

Practical Use case

```
route-map MatchRPKIState0
```

```
  match rpki valid
```

```
  set local-preference 100
```

```
route-map MatchRPKIState1
```

```
  match rpki not-found
```

```
  set local-preference 50
```

Placing your Caches.

