Securing BGP - RPKI

ThaiNOG2018 - Bangkok
21 May 2018

Tashi Phuntsho (tashi@apnic.net)
Fat-finger/Hijacks/Leaks

• **Amazon (AS16509) Route53 hijack – April 2018**
  
  – AS10279 (eNET) announced/originated more specifics (/24s) of Amazon Route53’s prefix (205.251.192.0/21)
    • 205.251.192.0/24 …… 205.251.199.0/24
  
  – Its peers, like AS6939 (HE), shared these routes with 1000s of their own peers…
  
  – **The motive?**
    • During the period, DNS servers in the hijacked range only responded to queries for [myetherwallet.com](http://myetherwallet.com)
    • Responded with addresses associated with AS41995/AS48693)
Fat-finger/Hijacks/Leaks

• **Route53 hijack (continued....)**
  – Resolvers querying any of the Route53 managed names, would actually be asking the authoritative servers controlled through the BGP hijack

https://blog.cloudflare.com/bgp-leaks-and-crypto-currencies
Fat-finger/Hijacks/Leaks

• Bharti (AS9498) originates 103.0.0.0/10
  – Dec 2017 (~ 2 days)
  – No damage done – more than 8K specific routes!

• Google brings down Internet in Japan
  – Aug 2017 (~ 24 hours)
    – Google (AS15169) leaked >130K prefixes to Verizon (AS701) – in Chicago
      • Normally ~ 50 prefixes
      • ~25K of those were NTT OCN’s (AS4713) more specifics
      • which was leaked onwards to KDDI and IIJ (and accepted)
    – Everyone who received the leaked more specifics, preferred the Verizon-Google path to reach NTT OCN!
Fat-finger/Hijacks/Leaks

- Google leak (contd...)

Fat-finger/Hijacks/Leaks

• Google (AS15169) services downed
  – Nov 2012 (~ 30 minutes)
  – Moratel Id (AS23947) leaked Google prefixes to its upstream (AS3491)
    • AS path: ... 3491 23947 15169

• YouTube (AS36561) Incident
  – Feb 2008 (down for ~ 2 hours)
  – PT (AS17557) announced 208.65.153.0/24 (208.65.152.0/22)
    • Propagated by AS3491 (PCCW)
How do we address these...

• Filters!!!
  – On both ends of a eBGP session
    • AS-PATH, prefix-list, max-prefix limit
  – only announce/originate your own prefix (and your downstream)
  – Only accept your peer’s prefix (and their downstream)
Current practice

Receive request → LOA check → Prefix/AS filter
Tools and techniques

LOA check

Manual

Automated (RPSL)

RPKI
Learning curve... slow (BGP)

RPSL

June 1999

RPKI

January 2013
What is RPKI?
Goals of RPKI

• To authoritatively prove who owns an IP prefix and which AS(es) can originate
  – Attaching digital certificates to network resources (AS Number & IP Address)
Benefits of RPKI

• Prevents **route hijacking**
  – A prefix originated by an AS without authorization
  – *Reason: malicious intent*

• Prevents **mis-origination**
  – A prefix that is mistakenly originated by an AS which does not own it
  – Also route leakage
  – *Reason: configuration mistakes/fat-finger*
RPKI Building Blocks

1. Trust Anchors (RIRs)
2. Route Origination Authorizations (ROA)
3. RPKI Validator
Public Key Encryption Recap

- Public and private key mathematically related to each other
  - Cannot derive one from the other

- Encrypt with one and decrypt with the other
  - Encrypt with private, only public can decrypt
  - Encrypt with public, only private can decrypt
Public-key Infra

- Digital (X.509) certificates
  - associates a public key with an individual or organization
    - public key of the subject!

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Version of X.509</td>
</tr>
<tr>
<td>Serial Number</td>
<td>Uniquely identifies the certificate</td>
</tr>
<tr>
<td>Signature Algorithm</td>
<td>Algorithms used by the CA to sign the cert</td>
</tr>
<tr>
<td>Issuer Name</td>
<td>Id of the CA (that issued the cert)</td>
</tr>
<tr>
<td>Validity Period</td>
<td>Cert validity</td>
</tr>
<tr>
<td>Subject Name</td>
<td>The cert owner</td>
</tr>
<tr>
<td>Subject Public Key</td>
<td>Owner’s public key</td>
</tr>
<tr>
<td>Issuer ID</td>
<td>Extra info (Issuer of the cert)</td>
</tr>
<tr>
<td>Subject ID</td>
<td>Extra info (owner of the cert)</td>
</tr>
<tr>
<td>Extensions (CRL)</td>
<td></td>
</tr>
<tr>
<td>CA Digital Signature</td>
<td>Digital Signature of the CA</td>
</tr>
</tbody>
</table>
RPKI Profile

- Resource certificates are based on the X.509 v3 certificate format (RFC 5280)

- Extended by RFC 3779 – binds a list of resources (IPv4/v6, ASN) to the subject of the certificate

- SIA (subject information access) contains a URI that references the directory where it is published
Trust Anchor (TA)

Allocation Hierarchy

IANA
ARIN
AFRINIC
APNIC
LACNIC
RIPE-NCC

Trust Anchor Certificate

Certificate chain mirrors the allocation hierarchy

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

- Internet Registries (RIR, NIR, Large LIRs)
  - Certificate Authority
    - issues certificates for customers
  - Customers create their ROAs
    - Sign their resources with the certificates

- Publishes the ROA records

![Diagram showing APNIC RPKI Engine publication to rpki.apnic.net]

MyAPNIC GUI
Single Trust anchor

• 27 Feb 2018: a single expanded trust anchor
ROA- Route Origin Authorization

- A digitally signed object that contains a list of address prefixes and the nominated AS number

- It is an authority created by a prefix holder to authorize an ASN to originate one or more prefixes – Which can be verified cryptographically using RPKI

<table>
<thead>
<tr>
<th>Prefix</th>
<th>203.176.189.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max-length</td>
<td>/24</td>
</tr>
<tr>
<td>Origin ASN</td>
<td>AS17821</td>
</tr>
</tbody>
</table>

- Multiple ROAs can exist for the same prefix
Relying Party (RPKI Validator)

• The RPKI Validator
  – Gathers ROA from the distributed RPKI database
  – Validates each entry’s signature
    • Validated cache
Relying Party (RPKI Validator)
Origin Validation

rsync

Global (RPKI) Repository

TA

TA

TA

TA

RPKI Validator/
RPKI Cache server

ROA

2406:6400::/32-48
17821

RPKI-to-Router (RtR)

AS17821

2406:6400::/48

.1/:1

.2/:2

ASXXXX

2406:6400::/32-48
17821
Origin Validation

• Router gets ROA information from the RPKI Cache
  – Crypto is stripped (by the validator)

• The BGP process will check each received BGP update against the ROA information and label
  – Valid
  – Invalid
  – Not Found
Validation States

• **Valid**
  – the prefix and AS pair are found in the database.

• **Invalid**
  – prefix is found, but origin AS is wrong, or
  – the prefix length is longer than the maximum length

• **Not Found / Unknown**
  – No valid ROA found
  – Neither valid nor invalid
    • Perhaps not created!
## RPKI States

<table>
<thead>
<tr>
<th>ROA =&gt;</th>
<th>Origin AS</th>
<th>Prefix</th>
<th>Max Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALID</td>
<td>AS65420</td>
<td>10.0.0.0/16</td>
<td>/18</td>
</tr>
<tr>
<td>VALID</td>
<td>AS65420</td>
<td>10.0.128.0/17</td>
<td></td>
</tr>
<tr>
<td>INVALID</td>
<td>AS65421</td>
<td>10.0.0.0/16</td>
<td>/18</td>
</tr>
<tr>
<td>INVALID</td>
<td>AS65420</td>
<td>10.0.10.0/24</td>
<td></td>
</tr>
<tr>
<td>UNKNOWN</td>
<td>AS65430</td>
<td>10.0.0.0/8</td>
<td>/8</td>
</tr>
</tbody>
</table>
Policies based on validation

• Define your policy based on the validation state
  – Do nothing (observe)
  – Label BGP communities
  – **Modify preference values**
    • RFC7115
  – Drop Invalid announcements (paranoid!)
    • Invalid - but verify against other databases (IRR whois)
RPKI Caveats

• When RTR session goes down, the validation status changes to **NOT FOUND** for all routes after a while
  – *Invalid* => *Not Found*
  – we need several RTR sessions (at least 2) and/or need careful filtering policies

• During router reload, which one is faster: receiving ROAs or receiving BGP updates?
  – If receiving BGP routes is faster than ROA, the router will propagate the invalid routes to its iBGP peers
RPKI Further Reading

• RFC 5280: X.509 PKI Certificates
• RFC 3779: Extensions for IP Addresses and ASNs
• RFC 6481-6493: Resource Public Key Infrastructure
Implementation
Create & publish your ROA

• MyAPNIC portal
  – Resources > RPKI

  – Here is a detailed guide:
Create (publish) your ROA

- Available prefixes for which you can create ROA

### BGP Route Validity

<table>
<thead>
<tr>
<th>Origin AS</th>
<th>Prefix</th>
</tr>
</thead>
<tbody>
<tr>
<td>45192</td>
<td>2001:df2:ee01::/48</td>
</tr>
<tr>
<td>45192</td>
<td>202.125.97.0/24</td>
</tr>
<tr>
<td>131107</td>
<td>2001:df2:ee00::/48</td>
</tr>
<tr>
<td>131107</td>
<td>202.125.96.0/24</td>
</tr>
<tr>
<td>135533</td>
<td>61.45.248.0/24</td>
</tr>
<tr>
<td>135540</td>
<td>61.45.248.0/24</td>
</tr>
</tbody>
</table>

Showing 1 to 6 of 6 entries
Create (publish) your ROA

ROA Configuration

<table>
<thead>
<tr>
<th>Origin ASN</th>
<th>Prefix</th>
<th>Max Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>131107</td>
<td>2001:df2:ee00::/48</td>
<td>48</td>
</tr>
</tbody>
</table>

Add | Add & clone | Clear

Show: 10 entries | Search: 131107

Certified Resources

| 61.45.248.0/21 |
| 202.125.96.0/23 |
| 203.30.127.0/24 |
| 2001:DF0:A::/48 |
| 2001:DF2:EE00::/47 |
| 2406:6400::/32 |
Check your ROA

# whois -h whois.bgpmon.net 2001:df2:ee00::/48

Prefix: 2001:df2:ee00::/48
Prefix description: APNICTRAINING–DC
Country code: AU
Origin AS: 131107
Origin AS Name: APNICTRAINING LAB DC
RPKI status: ROA validation successful
First seen: 2016-06-30
Last seen: 2018-01-21
Seen by #peers: 97

# whois -h whois.bgpmon.net "--roa 131107 2001:df2:ee00::/48"

------------------------
ROA Details
------------------------
Origin ASN: AS131107
Not valid Before: 2016-09-07 02:10:04
Not valid After: 2020-07-30 00:00:00 Expires in 2y190d9h34m23.2000000029802s
Trust Anchor: rpki.apnic.net
Prefixes: 2001:df2:ee00::/48 (max length /48) 202.125.96.0/24 (max length /24)
Check your ROA

https://bgp.he.net/

<table>
<thead>
<tr>
<th>Origin AS</th>
<th>Announcement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS131107</td>
<td>2001:df2:ee00::/48</td>
<td>testing</td>
</tr>
</tbody>
</table>
Deploy RPKI Validator

• Two options:
  
  – RIPE RPKI Validator
  
  – Dragon Research Labs RPKI Toolkit
    [https://github.com/dragonresearch/rpki.net](https://github.com/dragonresearch/rpki.net)
RIPE - Validator

• Download RPKI Validator

```bash
# wget https://lirportal.ripe.net/certification/content/static.validator/rpki-validator-app-2.24-dist.tar.gz
```

• Installation

```bash
tar -zxvf rpki-validator-app-2.23-dist.tar.gz
cd rpki-validator-app-2.23
./rpki-validator.sh start
```

- Need to download ARIN’s TAL separately

```bash
wget https://www.arin.net/resources/rpki/arin-ripevalidator.tal
```

• Move it to “<base-folder>/conf/tal” and restart
Quick Overview of BGP Origin Validation

Trust Anchors are the entry points used for validation in any Public Key Infrastructure (PKI) system. This RPKI Validator is preconfigured with the trust anchors for AFRINIC, APNIC, LACNIC, and RIPE NCC. In order to obtain the trust anchor for the ARIN RPKI repository, you will first have to accept their Relying Party Agreement. Please refer to the README.txt for details on how to add trust anchors to this application.

Configured Trust Anchors

<table>
<thead>
<tr>
<th>Enabled</th>
<th>Trust anchor</th>
<th>Processed Items</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>APNIC from AFRINIC RPKI Root</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>APNIC from ARIN RPKI Root</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>APNIC from IANA RPKI Root</td>
<td>4038</td>
</tr>
<tr>
<td></td>
<td>APNIC from LACNIC RPKI Root</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>APNIC from RIPE RPKI Root</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>ARIN</td>
<td>1549</td>
</tr>
<tr>
<td></td>
<td>AfriNIC RPKI Root</td>
<td>451</td>
</tr>
<tr>
<td></td>
<td>LACNIC RPKI Root</td>
<td>3970</td>
</tr>
<tr>
<td></td>
<td>RIPE NCC RPKI Root</td>
<td>20279</td>
</tr>
</tbody>
</table>

Router Sessions

This table shows all routers connected to this RPKI Validator. Requests and responses are described in RFC 6810. For debugging, please refer to rt.log.

<table>
<thead>
<tr>
<th>Remote Address</th>
<th>Connection Time</th>
<th>Last Request Time</th>
<th>Last Request</th>
<th>Last Reply</th>
</tr>
</thead>
</table>
Dragon Research - Validator

• Installation on Ubuntu 16.04 Xenial

https://github.com/dragonresearch/rpki.net/blob/master/doc/quickstart/xenial-rp.md

• Installation

```bash
# wget -q -O /etc/apt/trusted.gpg.d/rpki.gpg https://download.rpki.net/APTng/apt-gpg-key.gpg

# wget -q -O /etc/apt/sources.list.d/rpki.list https://download.rpki.net/APTng/rpki.xenial.list
```

-q: quite (wget output)
-0: output to <file>

```bash
# apt update
# apt install rpki-rp
```
Dragon Research - Validator

http://rpki-dragonresearch.apnictraining.net/rcyncic/

rcyncic summary 2017-01-03T01:07:37Z

Overview for repository rpki.apnic.net

<table>
<thead>
<tr>
<th>Tainted by stale CRL</th>
<th>Object accepted</th>
<th>Manifest interval overruns</th>
</tr>
</thead>
<tbody>
<tr>
<td>None .cer</td>
<td>28</td>
<td>5981</td>
</tr>
<tr>
<td>None .crl</td>
<td>5948</td>
<td></td>
</tr>
<tr>
<td>None .gbr</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>None .mft</td>
<td>5948</td>
<td>1</td>
</tr>
<tr>
<td>None .roa</td>
<td>5923</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>28</td>
<td>23803</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Repository</th>
<th>.cer</th>
<th>.crl</th>
<th>.gbr</th>
<th>.mft</th>
<th>.roa</th>
</tr>
</thead>
<tbody>
<tr>
<td>ca.rg.net</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ca0.rpki.net</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>localcert.rpki.net</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>repository.lacnic.net</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>rpki-pilot.lab.dtag.de</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>rpki-repository.nic.ad.jp</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>rpki.afnic.net</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>rpki.apnic.net</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>rpki.ripe.net</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tainted by stale CRL</th>
<th>Object accepted</th>
<th>Manifest interval overruns</th>
</tr>
</thead>
<tbody>
<tr>
<td>None .cer</td>
<td>752</td>
<td></td>
</tr>
<tr>
<td>None .crl</td>
<td>748</td>
<td></td>
</tr>
<tr>
<td>None .mft</td>
<td>748</td>
<td></td>
</tr>
<tr>
<td>None .roa</td>
<td>492</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>2740</td>
<td></td>
</tr>
</tbody>
</table>
Configuration - IOS

- Establishing session with the validator

```conf
router bgp 131107
bgp rpki server tcp <validator-IP> port 323 refresh 120
```

- Policies based on validation:

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

- Apply the route-map to inbound updates

```plaintext
router bgp 131107
!---output omitted-----!
address-family ipv4
  neighbor X.X.X.169 activate
  neighbor X.X.X.169 route-map ROUTE-VALIDATION in
  exit-address-family
!
address-family ipv6
  neighbor X6:X6:X6:X6::151 activate
  neighbor X6:X6:X6:X6::151 route-map ROUTE-VALIDATION in
  exit-address-family
!
```
Configuration - JunOS

- Establishing session with the validator

```bash
routing-options {
    autonomous-system 131107;
    validation {
        group rpki-validator {
            session <validator-IP> {
                refresh-time 120;
                port 8282;
                local-address X.X.X.253;
            }
        }
    }
}
```
Router Configuration - JunOS

- Define policies based on the validation states

```plaintext
policy-options {
  policy-statement ROUTE-VALIDATION {
    term valid {
      from {
        protocol bgp;
        validation-database valid;
      }
      then {
        local-preference 110;
        validation-state valid;
        accept;
      }
    }
    term invalid {
      from {
        protocol bgp;
        validation-database invalid;
      }
      then {
        local-preference 90;
        validation-state invalid;
        accept;
      }
    }
    term unknown {
      from {
        protocol bgp;
        validation-database unknown;
      }
      then {
        local-preference 100;
        validation-state unknown;
        accept;
      }
    }
  }
}
```
Router Configuration - JunOS

- Apply the policy to inbound updates

```
protocols {
  bgp {
    group external-peers {
      #output-omitted
      neighbor X.X.X.1 {
        import ROUTE-VALIDATION;
        family inet {
          unicast;
        }
      }
    }
    group external-peers-v6 {
      #output-omitted
      neighbor X6:X6:X6:X6::1 {
        import ROUTE-VALIDATION;
        family inet6 {
          unicast;
        }
      }
    }
  }
}
```
RPKI Verification - IOS

- IOS has only

```
#sh bgp ipv6 unicast rpki ?
    servers Display RPKI cache server information
    table    Display RPKI table entries
```

```
#sh bgp ipv4 unicast rpki ?
    servers Display RPKI cache server information
    table    Display RPKI table entries
```
RPKI Verification - IOS

• Check the RTR session

```bash
#sh bgp ipv4 unicast rpki servers

BGP SOVC neighbor is X.X.X.47/323 connected to port 323
Flags 64, Refresh time is 120, Serial number is 1516477445, Session ID is 8871
InQ has 0 messages, OutQ has 0 messages, formatted msg 7826
Session IO flags 3, Session flags 4008
  Neighbor Statistics:
    Prefixes 45661
    Connection attempts: 1
    Connection failures: 0
    Errors sent: 0
    Errors received: 0

Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled, Minimum incoming TTL 0, Outgoing TTL 255
Local host: X.X.X.225, Local port: 29831
Foreign host: X.X.X.47, Foreign port: 323
```
RPKI Verification - IOS

• Check the RPKI cache

#sh bgp ipv4 unicast rpki table
37868 BGP sovc network entries using 6058880 bytes of memory
39655 BGP sovc record entries using 1268960 bytes of memory

<table>
<thead>
<tr>
<th>Network</th>
<th>Maxlen</th>
<th>Origin-AS</th>
<th>Source</th>
<th>Neighbor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.9.0.0/16</td>
<td>24</td>
<td>4788</td>
<td>0</td>
<td>202.125.96.47/323</td>
</tr>
<tr>
<td>1.9.12.0/24</td>
<td>24</td>
<td>65037</td>
<td>0</td>
<td>202.125.96.47/323</td>
</tr>
<tr>
<td>1.9.21.0/24</td>
<td>24</td>
<td>24514</td>
<td>0</td>
<td>202.125.96.47/323</td>
</tr>
<tr>
<td>1.9.23.0/24</td>
<td>24</td>
<td>65120</td>
<td>0</td>
<td>202.125.96.47/323</td>
</tr>
</tbody>
</table>

#sh bgp ipv6 unicast rpki table
5309 BGP sovc network entries using 976856 bytes of memory
6006 BGP sovc record entries using 192192 bytes of memory

<table>
<thead>
<tr>
<th>Network</th>
<th>Maxlen</th>
<th>Origin-AS</th>
<th>Source</th>
<th>Neighbor</th>
</tr>
</thead>
<tbody>
<tr>
<td>2001:200::/32</td>
<td>32</td>
<td>2500</td>
<td>0</td>
<td>202.125.96.47/323</td>
</tr>
<tr>
<td>2001:200:900::/40</td>
<td>40</td>
<td>7660</td>
<td>0</td>
<td>202.125.96.47/323</td>
</tr>
<tr>
<td>2001:200:8000::/35</td>
<td>35</td>
<td>4690</td>
<td>0</td>
<td>202.125.96.47/323</td>
</tr>
</tbody>
</table>
Check routes - IOS

#sh bgp ipv4 unicast 202.144.128.0/19
BGP routing table entry for 202.144.128.0/19, version 3814371
Paths: (1 available, best #1, table default)
  Advertised to update-groups:
    2
  Refresh Epoch 15
  4826 17660
    49.255.232.169 from 49.255.232.169 (114.31.194.12)
      Origin IGP, metric 0, localpref 110, valid, external, best
      Community: 4826:5101 4826:6570 4826:51011 24115:17660
      path 7F50C7CD98C8 RPKI State valid
      rx pathid: 0, tx pathid: 0x0

#sh bgp ipv6 unicast 2402:7800::/32
BGP routing table entry for 2402:7800::/32, version 1157916
Paths: (1 available, best #1, table default)
  Advertised to update-groups:
    2
  Refresh Epoch 15
  4826
    2402:7800:10:2::151 from 2402:7800:10:2::151 (114.31.194.12)
      Origin IGP, metric 0, localpref 100, valid, external, best
      path 7F50B266CBD8 RPKI State not found
      rx pathid: 0, tx pathid: 0x0
**RPKI Verification - JunOS**

- Check the RTR session

```
>show validation session
Session  State Flaps   Uptime  #IPv4/IPv6 records
202.125.96.46          Up      75 09:20:59  40894/6747

>show validation session 202.125.96.46
Session  State Flaps   Uptime  #IPv4/IPv6 records
202.125.96.46          Up      75 09:21:18  40894/6747
```
## RPKI Verification - JunOS

- Check the RPKI cache

```plaintext
>show validation database
RV database for instance master

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Origin-AS</th>
<th>Session</th>
<th>State</th>
<th>Mismatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.9.0.0/16–24</td>
<td>4788</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>1.9.12.0/24–24</td>
<td>65037</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>1.9.21.0/24–24</td>
<td>24514</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>1.9.23.0/24–24</td>
<td>65120</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>2001:200::/32–32</td>
<td>2500</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>2001:200:900::/40–40</td>
<td>7660</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>2001:200:c000::/35–35</td>
<td>23634</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>2001:200:e000::/35–35</td>
<td>7660</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
</tbody>
</table>
```

- Would have been nice if they had per AF!
RPKI Verification - JunOS

- Can filter per origin ASN

```bash
>show validation database origin-autonomous-system 45192
RV database for instance master

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Origin-AS</th>
<th>Session</th>
<th>State</th>
<th>Mismatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>202.125.97.0/24-24</td>
<td>45192</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>203.176.189.0/24-24</td>
<td>45192</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>2001:df2:ee01::/48-48</td>
<td>45192</td>
<td>202.125.96.46</td>
<td>valid</td>
<td></td>
</tr>
</tbody>
</table>

IPv4 records: 2
IPv6 records: 1
```

- IOS should have something similar!
Check routes - JunOS

>show route protocol bgp 202.144.128.0

inet.0: 693024 destinations, 693024 routes (693022 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

202.144.128.0/20 *[BGP/170] 1w4d 21:03:04, MED 0, localpref 110, from 202.125.96.254
AS path: 4826 17660 I, validation-state: valid
>to 202.125.96.225 via ge-1/1/0.0

>show route protocol bgp 2001:201::/32

inet6.0: 93909 destinations, 93910 routes (93909 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:201::/32 *[BGP/170] 21:18:14, MED 0, localpref 100, from 2001:df2:ee00::1
AS path: 65332 I, validation-state: unknown
>to fe80::dab1:90ff:fedc:fd07 via ge-1/1/0.0
Configuration - Reference Link

• Cisco

• Juniper

• RIPE:
Questions