RPKI on Juniper Routers

John Scudder

NANOG 67, June 13, 2016
What, When, Where

• Support for Origin Validation and RPKI-RTR (draft form) first added in Junos 12.2R1 (September 2012).
  • And all subsequent releases.

• Currently:
  • RFC 6810 (“The Resource Public Key Infrastructure (RPKI) to Router Protocol”),
  • RFC 6811 (“Prefix Origin Validation”),
  • draft-ietf-sidr-origin-validation-signaling-08.

• Supported on all products (physical and virtual) running Junos.
Talking to the Local RPKI Cache

```
user@R0# show routing-options
  autonomous-system 64496;
  validation {
    group test {
      session 192.0.2.1;
    }
  }
```

That’s the minimum configuration. Various options exist for tuning session parameters, configuring redundant servers, etc. There’s also the possibly-interesting `static` option, for configuring static, local RPKI objects.
Validating Routes

• Origin Validation is invoked using normal Junos policy, with the `validation-database` match condition.
• Policy operates as normal, to do the usual things
  • Set internal state (e.g., the `validation-state` variable)
  • Set other state (e.g., `origin-validation-signaling` community)
  • Accept, reject, adjust `LocalPref`, etc.
Policy Example

- Invokes the Origin Validation machinery (*validation-database*).
- Based on what the OV check returns,
  - Sets the internal *validation-state* variable (to one of *valid*, *invalid*, *unknown*).
  - Adds the community for draft-ietf-sidr-origin-validation-signaling.
  - Sets a LocalPref (110 for valid, leaves default of 100 for unknown).
  - Rejects invalid. (Could have applied a different LocalPref and accepted, if that’s how you prefer to do it.)
- Note definition of OV communities at the end.
Mark valid routes

policy-statement validation {
  term valid {
    from {
      protocol bgp;
      validation-database valid;
    }
    then {
      local-preference 110;
      validation-state valid;
      community add origin-validation-state-valid;
      accept;
    }
  }
}
Mark invalid routes

term invalid {
  from {
    protocol bgp;
    validation-database invalid;
  }
  then {
    validation-state invalid;
    community add origin-validation-state-invalid;
    reject;
  }
}
Anything else is unknown, plus define some community names

term unknown {
    from protocol bgp;
    then {
        validation-state unknown;
        community add origin-validation-state-unknown;
        accept;
    }
}

community origin-validation-state-invalid members 0x4300:2;
community origin-validation-state-unknown members 0x4300:1;
community origin-validation-state-valid members 0x4300:0;
Management and troubleshooting

- Tracing (within “validation” stanza, for RPKI-RTR operation)
- Show commands
  - `show route`
  - `show validation statistics`
  - `show validation database`
  - `show validation replication`
  - `show validation group`
  - `show validation session`
- `request validation policy`
  - Re-run validation, optionally against only specified routes
show route

user@R1> show route
inet.0: 3 destinations, 3 routes (2 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both
2.2.0.2/32    *[BGP/170] 01:06:58, localpref 110, from 1.0.1.1
               AS path: 200 I, validation-state: valid
               > to 10.0.0.2 via lt-1/2/0.1
172.16.1.1/32 *[BGP/170] 00:40:52, localpref 90, from 1.0.1.1
               AS path: 200 I, validation-state: invalid
               Unusable
192.168.2.3/32 *[BGP/170] 01:06:58, localpref 100, from 1.0.1.1
               AS path: 200 I, validation-state: unknown
               > to 10.0.0.2 via lt-1/2/0.1 224.0.0.5/32
Validation states

• Internal validation states can be any of the usual RFC 6811 states,
  • Valid
  • Invalid
  • Unknown
• But also another state that means “validation was not run against this at all”,
  • Unverified
• Unverified is different from unknown
  • A route that is unverified might be any of valid, invalid, or unknown, if validation were attempted
show validation statistics

user@R0> show validation statistics
Total RV records: 3
Total Replication RV records: 3
  Prefix entries: 3
  Origin-AS entries: 3
Memory utilization: 9789 bytes
Policy origin-validation requests: 114
  Valid: 32
  Invalid: 54
  Unknown: 28
BGP import policy reevaluation notifications: 156
  inet.0, 156
  inet6.0, 0
show validation [replication] database

user@R0> 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>2.0.0.0/8-32</td>
<td>200</td>
<td>10.0.0.10</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>10.0.0.0/8-32</td>
<td>200</td>
<td>10.0.0.10</td>
<td>valid</td>
<td></td>
</tr>
<tr>
<td>172.0.0.0/8-12</td>
<td>200</td>
<td>10.0.0.10</td>
<td>invalid</td>
<td></td>
</tr>
</tbody>
</table>

IPv4 records: 3
IPv6 records: 0
show validation group, session

user@R0> show validation group
Master
    Group: test, Maximum sessions: 2
    Session 10.0.0.10, State: Up, Preference: 100

user@R0> show validation session
Session                     State  Flaps  Uptime  #IPv4/IPv6 records
10.0.0.10                    Up      0  00:02:28  1/0
More Info

• Much of this presentation was gleefully cribbed from the Junos documentation.
• The documentation has much more detail, of course.
Thank you