Quagga and BGP Secure Routing Extension

– QuaggaSRx –

Version 0.2

Users Manual

Index

Overview

This document describes the integration of the SRx Server API into Quagga. Quagga uses the SRx server by embedding the SRx server proxy. The SRx proxy manages the necessary TCP connection to the server and provides Quagga with a simple to use API.

Using this approach, the changes within Quagga are kept to a minimum. The majority of changes within the Quagga code are related to configuration settings. Each BGP router instance within Quagga has its own SRx integration.

Installation

The installation of QuaggaSRx requires the BGP-SRx library. The library is part of the BGP-SRx compilation. To compile QuaggaSRx use the provided configure script. The location of the BGP-SRx library must be provided if BGPS-SRx is not installed in the default location /usr/local/{bin|include|lib}. For installation of BGP-SRx refer to the BGP-SRx users guide.

Once BGP-SRx installed the make files have to be created using the ./configure command. An example of the ./configure command can be found in the file INSTALL.SRx_client.txt. Once configured call make followed by make install.

Quagga SRx configuration

All configuration commands added to Quagga are located within the BGP router configuration. All SRx commands start with the key word **srx** followed by the command and its parameters itself. The following command sequence is used to enter the configuration section using the telnet terminal. Using the configuration file, all commands are located within each particular router configuration:

1. enable
2. configure terminal
3. router bgp <asn>

SRx Configuration settings

[no] srx display

Turn on/off additional SRx display information for default show commands.

srx connect <host> <0..65535>

Connect the BGP server instance to the SRx server are the given location.

srx disconnect

Disconnect the BGP server instance from the SRx server. For this command the "keep-window" setting is used.

srx keep-window

Specifies the time in seconds the SRx is requested to hold information after it is deleted! This allows a router reboot without loosing the validation result information within SRx.

srx apply-policy (available in version 0.3.0)

This command is important if either a policy or evaluation method changed while QuaggaSRx is active. Changing policy and/or evaluation mode will have affects on the update/route selection. This command is necessary to allow previous processed updates to be reprocessed.
IMPORTANT: This command is very expensive. Each update stored in the RIB in will be re-evaluated.

srx evaluation (origin_only | bgpsec)

This command enables or disables the policy processing within the decision process as well as activating or deactivating setting of the ignore flag due to an ignore-XXX policy. In addition to enabling the evaluation of validation results this command specifies the mode the evaluation is performed in. Using the *origin_only* mode, only origin validation results are used whereas the *bgpsec* mode activates the evaluation of both, the origin validation result as well as the path validation result.
To disable the evaluation and learn more about the impact, see the next command "no srx evaluation"

origin_only (default)

Using this setting only origin validation is evaluated. Path validation results will still be requested and notifications from SRx will be processed in regards to maintaining the correct data associated with each update but the results of path validation will not be included into the evaluation of validation results.
The following results are possible within origin validation processing:

valid A ROA exists that covers the prefix and origin.
unknown No ROA does exist that covers neither the given prefix nor a less specific prefix.

invalid Only ROA that cover this or a less specific prefix exist, but not the origin.

(*undefined*) Validation not performed yet – Update will be ignored (default policy).

bgpsec

This evaluation mode activates origin validation and path validation. QuaggaSRx uses the validation results of origin validation and path validation to compute the final BGPSEC validation result (valid|invalid|*undefined*). SRx reports prefix-origin validation and path validation independently as soon as they are available. The merging of the result is performed within QuaggaSRx. The following table illustrates BGPSEC validation results:

BGPSEC		Path Validation		
		Valid (V)	**Invalid (I)**	**Undefined (?)**
Prefix – Origin Validation	**Valid**	**V**	**I**	**?**
	Unknown	**I**	**I**	**?**
	Invalid	**I**	**I**	**?**
	Undefined	**?**	**?**	**?**

QuaggaSRx introduces a fourth validation result type, called "*undefined*". This result type allows distinguishing between a validation result and the status if no connection to SRx exists and so far no validation result for the particular update exists. In case of the bgpsec evaluation mode an update validation result is considered undefined as long as one partial validation result (origin or path) is considered undefined.

no srx evaluation

Disable policy processing. In this mode QuaggaSRx performs BGP processing as usual. Regardless if evaluation is disabled, QuaggaSRx will perform validation requests to SRx and react to notifications. To disable SRx communication disconnect from SRx.

SRx Policy Configuration

QuaggaSRx provides three different policy types,

- Ignore updates of selected validation result evaluations
- Modify the local preference of updates depending on the validation result evaluation
- Prefer updates whose validation result is *valid*

By default QuaggaSRx does not enable any policies with the exception of ignore-undefined. Policies do influence the BGP decision process in the following order:

1. Ignore Policies:
   These policies prevent updates entering the decision process. They are stored in the RIB in but will not be considered for route selection
2. Local Preference Modification policies:
   These policies allow a dynamic/fixed modification of the update's local preference values. The dynamic method allows combining other local preference policies with the validation result policies. In case a

dynamic local preference policy reduces the local preference to be less than zero “0” (underflow) the local preference will be adjusted to zero.

3. Prefer Valid:
   This policy prefers updates with the validation state *valid* to updates whose values are different from *valid*. The policy “Prefer Valid” is executed directly after local preference policies.

The QuaggaSRx implementation changes the default decision process the following manner:

1. Weight (kept as is)
2. Local Preference:
   After determining the local pref the srx policy will apply the local pref changes according to the validation result
3. Prefer valid updates to updates with a different validation state.
4. … (kept as is)

It is possible that in certain circumstances updates of different validation states are compared where none of the updates is valid. In this case no ranking is performed because this situation is transient and can occur during the introduction of a new ROA where certain updates are already re-evaluated but others are not yet. In case both updates will eventually be same evaluated, no change in the route selection will be performed later on. Only of the update with the transient state becomes valid.

[no] srx policy (ignore-unknown | ignore-invalid | ignore-undefined)

Activates or deactivates this set of policies that specify which update has to be ignored.

ignore-unknown

Updates with the validation result *unknown* will be flagged as ignored and will not be processed further in the decision process.

ignore-invalid

Updates with the validation result *invalid* will be flagged as ignored and will not be processed further in the decision process

ignore-undefined (default)

Updates that could not be validated by the SRx server are considered undefined. This is an intermediate state and as soon as the SRx server was able to process the validation for the update it will receive one of the final validation results (valid, unknown, invalid).

[no] srx policy local-preference (valid | unknown | invalid) <value> [add | subtract]

Local pref modification policies are only applied to updates with the validation results valid, invalid, or unknown. The value is a positive integer value that is used as a fixed local preference value overwriting a pre-existing value or modifying the pre-existing values. The later one is specified by adding **add** or **subtract** to the policy configuration.

Modifying the local preference by adding of subtracting allows to combine other policies with origin validation / path validation. An example could be a policy where an operator wants to configure the router in such way that all routes of peer A are preferred over routes by peer B except if peer B has a valid route where peer A has only an invalid one. Also routes declared as unknown or undefined of

peer A are still preferred over routes of peer B that are valid. The following configuration would allow such a setup:

Default local preference for peer A: 106
Default local preference for peer B: 100

srx policy local-preference valid 5 add
srx policy local-preference invalid 5 subtract

1: U(A): v | u | ? → LP >= 106, U(B): u | I | ? → LP <= 100, selected update: U(A)
2: U(A): v | u | ? → LP >= 106, U(B): v → LP = 105, selected update: U(A)
3: U(A): i → LP = 101, U(B): I | u | ? → LP <= 100, selected update: U(A)
4: U(A): ii → LP = 101, U(B): v → LP = 105, selected update: **U(B)**

v = valid, u = unknown, i = invalid, ? = undefined

As shown above [4] only "valid" updates of peer B will be chosen over updates of peer A because the updates of peer A are "invalid".

To deactivate a local pref policy only the policy type (local-pref) and the validation type must be specified.

[no] srx policy prefer-valid

This policy indicates that the tiebreaker between two BGP updates is the validation state **valid**. This means updates that are valid are selected over updates whose validation state differ from valid. In case both updates are valid or other than valid, other tiebreakers such as shortest path, MED , router id etc. will be used to determine the route selection.

Display commands

For the display, QuaggaSRx seamless integrated validation information into the standard **show [ip] bgp** commands. The additional information must be enabled or disabled within using the srx display command as described above.

SRx Configuration Display

To display the configuration of the SRx configuration within QuaggaSRx it is necessary to maneuver to the bgp configuration level. This level will be entered by entering the “enabled” mode, “configure terminal” – “router bgp <ASN>”. Once in this level (same as the one used for configuring policies etc.) the console command **show srx-config** displays all configuration settings for the particular SRx connection, configured policies as well as the status of the SRx connection.

```
bgpd(config-router)# show srx-config
SRx configuration settings:
  server.......: localhost
  port.........: 17900
  proxy-id.....: 1
  keep-window..: 900
  evaluation...: roa_only (prefix-origin processing)
  policy.......:
  connected....: true
bgpd(config-router)#
```

SRx configuration display

SRx Related BGP Display

The following described information is only visible is the srx display is configured. The command **[no] srx display** is used to configure the SRx display. By default the command is activated and SRx related information is added to the standards show commands:

Command: **show ip bgp**

```
bgpd> show ip bgp
BGP table version is 0, local router ID is 129.6.140.89
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, R Removed
Validation:    v - valid, u - unknown, i - invalid, ? - undefined
SRx Status:    I - route ignored, D - SRx evaluation deactivated
SRxVal Format: validation result (origin validation, path validation)
Origin codes: i - IGP, e - EGP, ? - incomplete

   Ident    SRxVal SRxLP Status Network          Next Hop            Metric  LocPrf Weight Path
*> DCEBB231 u(u,-)              10.0.0.0         12[redacted]             0              0 44 i
*> EF93DE16 u(u,-)              10.10.0.0/16     12[redacted]             0              0 44 i
*> B27F8F1A v(v,-)              10.10.0.0/20     12[redacted]             0              0 44 i
*> 544B7C0E v(v,-)              10.10.0.0/24     12[redacted]             0              0 44 i
*> 4330684D i(i,-)              10.10.0.0/25     12[redacted]             0              0 44 i
*> EDB0A7E6 v(v,-)              10.10.1.0/24     12[redacted]             0              0 44 i
```

Command: **show ip bgp <network>**

```
bgpd> show ip bgp 10.10.0.0/20
BGP routing table entry for 10.10.0.0/20
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  44
    SRx Information:
      Update ID: 0xB27F8F1A
      Validation:
        prefix-origin: valid
        path processing disabled!
    12[redacted] from 12[redacted] (12[redacted])
      Origin IGP, metric 0, localpref 100, valid, external, best
      Last update: Thu Jan  1 02:57:54 1970
```

Support

Please verify that QuaggaSRx and SRx server are connected and properly communicating. We provide tools such as Wireshark plugins that allow analyzing the traffic in a human readable manner. Also check firewall settings. For questionable results for particular updates look up the update id and query the update information at SRx server. If nothing helps, please contact us and we try to help. In case of crashes, please provide a core dump and a description on how to reproduce the crash.

Email: bgpsrx-dev@nist.gov
Web: www-x.antd.nist.gov/bgpsrx