O. Borchert
K. Lee
P. Gleichmann
NIST

Experimental
Sept 2020

Secure Router Extension (SRx) Server Protocol
srx-server-protocol

Abstract

This document facilitates the off-loading of RPKI-based security operations such as Route Origin Validation (ROV) and BGPsec path validation (BPV) onto external systems such as the NIST developed Secure Routing Extension (SRx) Server. It describes the communication between the SRx Server and its proxy thin client integrated, within a BGP router or Policy module. The SRx Server provides an interface to the RPKI/ROA Validation Cache using the RPKI to Router protocol [RFC8610] as well as a BGPsec path validation engine.

Status of This Memo

This document specifies a protocol design for the NIST internal reference implementation for ROA processing [RRFC6811] and BGPSEC processing [RFC8205] on the router side. This document describes an experimental protocol which is NOT an Internet standard. Comments and suggestions are welcome and to be send to the author of this document.

Table of Contents

1. Requirements language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Introduction

This document describes the communication between SRx and its proxy. This protocol is of interest to those who decide to implement their own proxy module and therefore choose to communicate directly with the server.

The srx-server-protocol is a TCP based, not encrypted protocol that is intended to be used within a trusted and secure environment, hence it is not needed to add an extra layer of security that would only increase the data volume. If security is desired, it can be tunneled through using ssh.

The SRx server itself does not provide BGPSEC validation in the sense of a combined origin and path validation. SRx provides both validation types independent from one another and therefore the validation requests within this protocol are NOT requests for ROA validation and BGPSEC validation, the requests and result notifications are focused on each component, origin validation and path validation. The consumer can decide on how to interpret / combine the results according to the implementation chosen.

3. Glossary

SRx:
Secure Routing Extension, a framework that allows to out-source the processing of route origin validation and path validation as well as path signing.

Proxy:
The thin client of SRx embedded in the router, policy module or other software that will use the SRx. In case the router chooses to implement the proxy itself, it will take on the role of the proxy.

Router:
A BGP router that uses the SRx to receive validation information about updates.

Policy Module:
A software that generates BGP routing policies that are feed into the BGP router. This policy module might use the SRx to generate/modify policies. In such case the router does not need to use SRx.

Validation Cache:
The validation cache is responsible for performing ROA/RPKI validation. Changes in the validation cache are signaled to the SRx using [RFC8210]

4.   Protocol Data Units (PDU)

4.1. Session Packets

This chapter deals with the session handshake and session tear down.

4.1.1 Hello

The proxy connects to the client and negotiates a session. The negotiation is done by sending a hello packet to the SRx server. The server will answer with a Hello Response packet that contains the connection status and proxy identifier. The packet MUST contain at least one "Peer AS". This information allows the SRx to precalculate signatures while in IDLE mode. The precalculation will only be performed for updates received, not for updates that are originated by the proxy AS. The default prepend count of the own AS number is "1". The "Sign Request" packet allows to specify a different prepend count.

```
0          8           16          24         31
+---------------------------------------------+
|  PDU     |                      |  Proxy    |
|  Type    |      Version         |   ID      |
|   0      |         2            |           |
|---------------------------------------------|
|                                             |
|              Length=Variable                |
|                                             |
|---------------------------------------------|
|                                             |
|        Autonomous System Number (AS)        |
|                                             |
|---------------------------------------------|
|                                             |
|               Number Peers                  |
|                                             |
|---------------------------------------------|
~                                             ~
|---------------------------------------------|
|                                             |
|     Peer AS (Autonomous System Number)      |
|                                             |
+---------------------------------------------+
```

Each connection between a proxy and SRx MUST have a unique identifier. Each proxy can have only one AS number. The router implementation MUST NOT share one proxy instance with multiple internal router instances. Each router instance MUST have its own proxy.

The SRx answers to the hello packet either with a "Hello Response" message or with an error packet. In case the provided proxy identifier has the value "0" zero, SRx will generate one and return it back using the Hello Response message. Otherwise the SRx returns the provided proxy identifier.

4.1.2 Hello Response

The Hello Response packet finalizes the handshake. The proxy MUST use the proxy identifier provided within this packet. This value should be the same as the provided one using the Hello packet except the initial value was set to "0" zero. In this case the SRx generated the identifier. In case of a conflict the server assigns a new value.

```
0          8          16         24        31
+---------------------------------------------+
|  PDU     |                     |   Proxy  |
|  Type    |      Version        |    ID    |
|   1      |         2           |          |
|---------------------------------------------|
|                                             |
|                Length=12                    |
|                                             |
+---------------------------------------------+
```

4.1.3 Goodbye PDU

The Goodbye message is sent to orderly disconnect the session between SRx and proxy. This packet is used by both, SRx as well as its proxy.

```
0          8          16         24        31
+---------------------------------------------+
|  PDU     |                     |          |
|  Type    |     Keep Window     |   zero   |
|   2      |                     |          |
|---------------------------------------------|
|                                             |
|                Length=8                     |
|                                             |
+---------------------------------------------+
```

The field Keep Window is a request for both sides to not remove data associated to this session. This value is in seconds and is a request only.

4.2 Origin and Path Validation Request Communication

SRx provides two different services. The first one is the validation of updates received, the second one is the signing of BGP updates selected by the router and send out to its peers.

(1) Origin and Path Validation:

The Verify Request will be performed using two packet types, one for IPv4 prefixes and the other for IPv6 packages. The Verify Request messages are used to request the verification of an update.

Within each request the proxy provides SRx with a predefined validation result (Sections 5.7 & 5.8) to allow SRx to return a preliminary result as soon as SRx generated a unique ID for this update. This ID is the communication interface between SRx, the proxy and the router / Policy module.

The validation packets are used for both request types, Origin validation request as well as Path validation request. Both requests can be either combined into one single packet or send as two separate requests. Eventually all requests for the same update will result in the same update identifier.

Updates that are originated by the router do not need to be verified. Nevertheless, these updates need an update ID to be able to use the SRx signing mechanism. In this case the update MUST be processed using a validation request with the exception that none of the "Validation Type" bits is set. This will prevent the update from being further processed in regard to validation. This is considered the "Safe Only" mode.

(2) Path Signing:

The path signing request starts the signing operation within SRx. SRx is able to precompute signatures for updates that underwent BGPSEC validation. Updates that are originated by the router can not be precomputed due to the fact that one element the signature does cover is the origination time stamp. This will be taken in the moment of sending. All other updates can be precomputed by SRx during idle times.

If the precomputation of update signatures is performed it is up to the SRx implementation to decide if by default the own AS is used once or multiple times. It is recommended though to precompute without any traffic engineering. The signing request can make these requests on an individual base using the

"Prepared Counter" field of the request package.

SRx only monitors verification results only for updates for which it received at least once a validation request. This means in case a validation request was made for origin validation but not path validation, the monitoring is performed for origin validation only. In case an additional request for the same updates is received but this time for path validation, SRx will start adding path validation to the monitor. In this case it will monitor both validations.

This allows to just store updates by submitting a validation request but leaving the field "Flags" empty. This allows the proxy to receive an update id that can be used for later signature requests. (Important for self-originated updates!)

Validation requests normally result in validation receipts. This is necessary to allow the SRx to return the generated update ID. Therefore, the proxy needs to wait for the receipt after each update request. This might create an unnecessary processing delay on the routers side, especially in situations where the update id is already known. In such situations, receipts can be omitted. The SRx then will only send a notification in case a change in validation result occurred. In case the proxy omits all receipts, the proxy has to generate the "update ID" on its own. In this case the proxy MUST assure to use the identical algorithm for generating the ID as the SRx otherwise they will not properly communicate to each other.

In case the validation request uses the receipt flag, the request token allows the proxy to match the request to the notification. Notifications use this flag to assign the receipt to its validation request.

4.2.1 Verify Request IPv4

The following PDU describes the verification request packet for IPv4 Prefix / Update validation.

```
0          8          16         24        31
+-------------------------------------------+
|  PDU     |          |  Origin  |  Path    |
|  Type    |  Flags   |  Result  |  Result  |
|   3      |          |  Source  |  Source  |
|-------------------------------------------|
|              Length=Variable              |
|-------------------------------------------|
|  Origin  |  Path    |          |  Prefix  |
|  Default |  Default |   zero   |  Length  |
|  Result  |  Result  |          |  (0..32) |
|-------------------------------------------|
|               Request Token               |
|-------------------------------------------|
|             IPv4 Prefix Address           |
|-------------------------------------------|
|        Origin AS (Autonomous System)      |
|-------------------------------------------|
|        Length Path Validation Data        |
|-------------------------------------------|
|                     |     BGPsec_PATH     |
|   Number of Hops    |   Attribute Length  |
|                     |                     |
|-------------------------------------------|
|                     |           |         |
|         AFI         |   SAFI    | Length  |
|                     |           |         |
|-------------------------------------------|
|--           IP Prefix Address           --|
|--               16 bytes                --|
|--                                       --|
|-------------------------------------------|
|                  Local AS                 |
|-------------------------------------------|
~                AS Path List               ~
|-------------------------------------------|
~            BGPsec_PATH Attribute          ~
+-------------------------------------------+
```

4.2.2 Verify Request IPv6

The following PDU describes the verification request packet for IPv6 Prefix / Update validation.

```
0          8          16         24        31
+-------------------------------------------+
|  PDU     |          |  Origin  |  Path    |
|  Type    |  Flags   |  Result  |  Result  |
|   4      |          |  Source  |  Source  |
|-------------------------------------------|
|                                           |
|              Length=Variable              |
|                                           |
|-------------------------------------------|
|  Origin  |  Path    |          |  Prefix  |
|  Default |  Default |   zero   |  Length  |
|  Result  |  Result  |          | (0..128) |
|-------------------------------------------|
|               Request Token               |
|-------------------------------------------|
|--         IPv6 Prefix Address          --|
|--              16 bytes                --|
|--                                      --|
|-------------------------------------------|
|        Origin AS (Autonomous System)      |
|-------------------------------------------|
|        Length Path Validation Data        |
|-------------------------------------------|
|                     |    BGPsec_PATH      |
|   Number of Hops    |   Attribute Length  |
|                     |                     |
|-------------------------------------------|
|                     |           |         |
|         AFI         |   SAFI    | Length  |
|                     |           |         |
|-------------------------------------------|
|--           IP Prefix Address          --|
|--              16 bytes                --|
|--                                      --|
|-------------------------------------------|
|                  Local AS                 |
|-------------------------------------------|
~               AS Path List                ~
|-------------------------------------------|
~           BGPsec_PATH Attribute           ~
+-------------------------------------------+
```

4.2.3 Sign Request

This PDU is used to sign the as path and its attributes. This request needs only to reference the peer AS the update will be send to. SRx is capable of precomputing default signatures; signatures with no additional content other than already provided during initial validation request. The field "Prepend Counter" allows traffic engineering in form of path-length within the signature generation.

```
0          8          16         24        31
+-------------------------------------------+
|  PDU     |                     |  Block   |
|  Type    |     ALGORITHM       |  Type    |
|   5      |                     |          |
|-------------------------------------------|
|                                           |
|                Length=20                  |
|                                           |
|-------------------------------------------|
|                                           |
|            Update Identifier              |
|                                           |
|-------------------------------------------|
|                                           |
|             Prepend Counter               |
|                                           |
|-------------------------------------------|
|                                           |
|     Peer AS (Autonomous System Number)    |
|                                           |
+-------------------------------------------+
```

4.3. SRx to Proxy Result Notification"

The PDU's described here are used by SRx to communicate the requested result as well as changes within the results to the proxy and therefore, to the router or policy module. Notifications can occur due to multiple events:

(1) Origin / Path Validation:

Each request for validation will result instantaneously in a result notification. For this kind of notification, the "Receipt" flag is set. This result does not only return a "preliminary/final" result, it also returns the unique update ID that is used for all future communication regarding this particular update.

Repeated validation requests of the exact same update MUST result in the same update ID.

(2) Signature Request:

Different to the validation request, the signature request will result in a signature validation. The router must decide if it only sends updates fully signed or if it allows sending unsigned packages followed by resending the update again once the signature is fully computed.

(3) Validation Changes of ROA's or Signature keys:

SRx monitors the state of updates in such as it received information about ROA expiration, revocations, key expiration etc. These events can change the validation result of a prior processed update. In case a validation state of an update changes, SRx MUST send a notification message to the proxy. This notification messages MUST NOT have the "Receipt" flag set.

4.3.1 Verify Notification

This packet is used by SRx to communicate the validation result to the proxy. The results communicated using this packed must reflect the validation result of SRx as soon as possible. This means that as soon as SRx results are available these results MUST be used, the results provided during the last request are ignored.

The "Receipt" flag specifies if this notification MUST be handles as validation receipt or validation result notification.

```
0          8          16         24        31
+--------------------------------------------+
|  PDU     |  Result  |  Origin  |  Path     |
|  Type    |  Type    |  Result  |  Result   |
|   6      |          |          |           |
|--------------------------------------------|
|                                            |
|                 Length=16                  |
|                                            |
|--------------------------------------------|
|                                            |
|               Request Token                |
|                                            |
|--------------------------------------------|
|                                            |
|              Update Identifier             |
|                                            |
+--------------------------------------------+
```

The "Request Token" field will only be used in case this notification is flagged as "Receipt". This token will help the proxy side to match the receipt to the request.

4.3.2 Signature Notification

This packet is used by SRx to communicate the BGPSEC portion of the update back to the proxy. Depending on the sign request type the data returned contains either only the latest signature block or all signature blocks. This allows the user of the proxy (e.g. router) to either strip all BGPSEC information from the update and keep the memory consumption low or keep the data traffic to a minimum by only transmitting the new signature block. In both cases SRx keeps all BGPSEC related data cached.

```
0          8          16          24         31
+--------------------------------------------+
|  PDU     |                     |           |
|  Type    |       reserved      |   zero    |
|   7      |                     |           |
|--------------------------------------------|
|                                            |
|                Length=Variable             |
|                                            |
|--------------------------------------------|
|                                            |
|               Update Identifier            |
|                                            |
|--------------------------------------------|
|                                            |
|        Length Path Validation Data         |
|                                            |
|--------------------------------------------|
|                                            |
~              PATH SIGNATURE DATA           ~
|                                            |
+--------------------------------------------+
```

4.4 SRx Maintenance and Error Handling

PDU's specified in this section define maintenance packets. They are used to allow synchronization, failure notification, housekeeping, and peer configuration. SRx can implement the functions anticipated behind these messages but does not need to do so. Both, SRx as well as the proxy MUST accept packages of this type according to the communication schematics.

4.4.1 Delete Update

This packet is used to inform the SRx that the specified update is not available anymore in the router. SRx itself does not need to react on this but it will allow SRx to free up resources.

Furthermore, the proxy makes sure it will NOT receive any further notifications related to this update. The "Keep Window" field allows to inform SRx that the update might be requested again within the "Keep Window" time. SRx is not obligated to follow this request. This field is purely a performance setting.

```
  0          8           16          24        31
+--------------------------------------------+
|  PDU     |                     |           |
|  Type    |     Keep Window     |   zero    |
|   8      |                     |           |
|--------------------------------------------|
|                                            |
|                Length=12                   |
|                                            |
|--------------------------------------------|
|                                            |
|             Update Identifier              |
|                                            |
+--------------------------------------------+
```

4.4.2 Peer Change

This packet is used by proxy to indicate a change in the peer configuration.

```
0          8           16         24         31
+--------------------------------------------+
|  PDU     |                      |  Change  |
|  Type    |       reserved       |   type   |
|   9      |                      |          |
|--------------------------------------------|
|                                            |
|                  Length=12                 |
|                                            |
|--------------------------------------------|
|                                            |
|                  Peer AS                   |
|                                            |
+--------------------------------------------+
```

4.4.3 Synchronization Request

This packet is used if SRx assumes that the connection with the proxy is out of sync. This could be due to a session restart or other problems. It is IMPORTANT that the proxy performs a synchronization once the server requested one to assure that SRx has a complete view on the data of proxy. The synchronization is performed by sending a validation request for each update located within the RIB in of the BGP router. Due to the fact that this could be a very expensive operation the SRx implementation should be conservative in the usage of this request.

```
0          8           16         24         31
+--------------------------------------------+
|  PDU     |                      |          |
|  Type    |       reserved       |   zero   |
|   10     |                      |          |
|--------------------------------------------|
|                                            |
|                  Length=8                  |
|                                            |
+--------------------------------------------+
```

4.4.4 Error Packet

This packet is used by SRx in case an error occurred. All errors are considered fatal and are followed by a goodbye if possible.

```
0          8          16         24        31
+--------------------------------------------+
|  PDU     |                     |           |
|  Type    |  Error Code         |   zero    |
|   11     |                     |           |
|--------------------------------------------|
|                                            |
|                 Length=8                   |
|                                            |
+--------------------------------------------+
```

5. PDU Data Fields

This section describes the data fields used within each PDU.

5.1. Proxy Identifier (Proxy ID)

The Proxy Identifier is a unique 4 byte value that allows the SRx to map internal values to the proxy itself. A preferred value is the IPv4 address of the proxy.

During the handshake the identifier is allowed to have the initial value of "0" zero provided by the proxy. In this case the SRx will generate a SRx wide unique identifier for this proxy. Each identifier can only be mapped to one proxy at a time. In case the proxy provides an identifier during handshake and this identifier is currently mapped to an existing session an error will be produced and the handshake fails.

5.2. Number Peer AS

The number of BGP peers the proxies user (most likely the BGP router) has. In case the value of this number is odd, the last two bytes of the packet MUST be filled with "0" zero. The number of peers MUST be greater or equals to "1" one.

5.3. Peer AS (Autonomous System)

Contains the AS number of a peer. This is used to allows SRx to precompute signatures for the moment when the proxy requests a path signature for a particular selected path. This operation can be performed by SRx during idle times.

5.4. Change Type

The change type is used to indicate if the specified peer as is removed or added to router configuration. Adding a peer is done by setting the value to "1" one, removing is done by setting the value to "0" zero.

5.5. Flags

The flags field is bit coded and informs the SRx server what to do.

```
         7 6 5 4 3 2 1 0
        +---------------+
        |x|0|0|0|0|0|x|x|
        +---------------+
         |           | |
         |           | |
         |           | +------VERIFY_PREFIX_ORIGIN =   1
         |           +--------VERIFY_PATH          =   2
         +--------------------REQUEST_RECEIPT      = 128
```

In case none of the bits is set the update will only be stored. This is needed for updates originated by the router that uses the proxy but do not need any validation applied to.

VERIFY_PREFIX_ORIGIN (1)

Request Prefix Origin validation. With this flag set the SRx will perform an origin validation. In case the SRx already performed this validation it is expected to return its result. No new validation needs to be performed. Changes in validation status due to expired or revoked ROA's will be triggered by the validation cache.

VERIFY_PATH (2)

Requests for patch validation. With this flag set the SRx will perform a path validation on the passed update. This setting also indicates that SRx must monitor changes within the key validity. In case a key is revoked or expired its signatures will become invalid and this change will be signaled to the router / proxy via a Validation Notification massage.

REQUEST_RECEIPT (128)

This flag indicates the proxy requests a notification receipt. A receipt is similar to a notification except that the SRx MUST send it regardless if the provided given validation state matches the validation result generated by SRx during a prior validation request. The request receipt is most importantly used to receive the unique update id. In case the proxy generates the update IS it MUST use the same algorithm as SRx to prevent communication problems.

5.6 Origin Result Source / Path Result Source

These two fields specify what to do with the provided default result values. It is possible that the BGP router might call the validation and provides a prior calculated validation result. this can be as a result of a SYNC request from the SRx server itself. It also could be used after a session reboot between SRx and router.

This field can take the following values:

0: SRX
1: ROUTER
2: IGP
3: UNKNOWN

IGNORE: Do not provide default result in case no result is available yet.

All the others: In case the SRx does not already have a validation result it will return the predefined value. In case the validation comes back with a different validation result than the provided one, SRx will notify the router of the change.

5.7. Default Origin Result

A predefined validation result that if used is returned to the proxy in case no current result is available.

The following values can be used:

0: VALID
SRx knows about a ROA that covers this pair of Prefix/Origin.

1: UNKNOWN
The prefix is not covered by any ROA nor is a ROA known that describes a less specific prefix.

2: INVALID
A ROA exists that covers either this prefix or a less specific prefix but none includes the given prefix.

3: UNDEFINED
SRx does not have any result available and no default value was provided.

5.8. Default Path Result

A predefined validation result that if used is returned to the proxy in case no current result is available.

The following values can be used:

0: VALID
SRx could validate the path according to the specifications.

2: INVALID
SRx could not validate the path according to the specifications.

3: UNDEFINED
SRx does not have any result available and no default value was provided.

5.9. Result Type

Identifies which result to use. This field MUST NOT be "0" zero filled.

The result type is bit coded.

```
         7 6 5 4 3 2 1 0
        +---------------+
        |x|0|0|0|0|0|x|x|
        +---------------+
         |           | |
         |           | |
         |           | +------RESULT_TYPE_ORIGIN =   1
         |           +--------RESULT_TYPE_PATH   =   2
         +--------------------RECEIPT_REQUEST    = 128
```

5.10. Update Identifier

Specifies a unique id within the router that is used to identify the update when talking between proxy and SRx. It is expected that the identical update results in the exact same Update Identifier at all times.

5.11. Receipt Token

This field is maintained by the proxy only. It is not used as system wide identifier. the receipt token helps the proxy to assign a receipt notification to the initiating validation request and only if receipts are requested. Otherwise this field will be zero. This token facilitates the timeout management of requests. The client decides on how to tread notifications that do NOT match any requests as well as how to generate this token. SRx simply copies the value from the verification request into the request notification.

5.12. Origin AS

The originator of the update. Will be ignored if origin validation is turned off (VERIFY_PREFIX_ORIGIN not set).

5.13. IPv4 Prefix / IPv6 Prefix

The IPv4 or IPv6 prefix (In Network order).

5.14. Prefix Length

The length of the IP prefix. This value can be 0..32 for IPv4 and 0..128 for IPv4. All other values are resulting in an error response.

5.15. ALGORITHM

BGPsec path validation [RFC8205] only allows two algorithms at the same time for the reason of algorithm change. In case the requested algorithm is not supported, an "Algorithm Not Supported" exception MUST be thrown.

1: ALGORITHM_1:
   The first older algorithm that is either the only currently active algorithm or about to be replaced by the newer algorithm specified as ALGORITHM_2.

2: ALGORITHM_2:
   The newer algorithm that is intended to replace the older algorithm ALGORITHM_1.

0xFFFF:
   This algorithm is just for test purpose until the BGPsec specification is finished. A request for this algorithm MUST not produce an "Algorithm Not Supported" error.

5.16. Block Type

The Block Type specifies if the return value of the signature request MUST result in the complete set of BGPSEC data or only the latest signature. If the bit LATEST_SIGNATURE_ONLY is set to "1" one, only the latest signature is transmitted.

```
           7 6 5 4 3 2 1 0
          +---------------+
          |0|0|0|0|0|0|0|x|
          +---------------+
                         |
                         +------LATEST_SIGNATURE_ONLY = 1
```

5.17. Prepend Counter

By default, SRx uses the current AS only once for (pre)calculating signatures. For traffic engineering reasons a particular update can contain the same prefix multiple times. The "Prepend Counter" allows the calculation of signatures over multiple entities of the own prepended AS.

5.18. Length Path Validation Data

Specifies the length of the data length provided needed for path validation. For ROV validation only, no path data is required. This filed allows to optimize the data volume by setting it to zero were the AS path is not important - ROV only. in case this value is set to zero "0", no further data MUST be read to this package.

5.18.1 Number of Hops

Specifies the number of hops within the path. This umber of hops must include repetitions because it is used to calculate the 4 Byte per entry AS Path List (see 5.18.7)

5.18.2 BGPsec_PATH Attribute Length

This specifies the complete size of the BGPsec_PATH attribute in bytes.

5.18.3 AFI

The Address Family Identifier as specified in RFC 4760

5.18.4 SAFI

The Subsequent Address Family Identifier in RFC 4760

5.18.5 Length

The prefix length in bytes ((prefix_length in bits + 7) div 8). This value is not used to specify the size of the prefix address field size, it is used to identify how many bytes are used within the prefix address filed.

5.18.6 The Prefix Address

The prefix field in network order. The size of this field is 16 bytes regardless of AFI type.

5.18.7 Local AS

The local AS number of the receiver of the UPDATE

5.18.8 AS Path List

An array containing the AS path as consecutive 4 byte ASNs. The right most AS is the originator. AS concatenations must be reflected as concatenations as well.

5.18.9 BGPsec_PATH Attribute

The BGPsec_PATH attribute as specified in RFC 8205. In case the UPDATE was a BGP4 UPDATE, this value will be omitted and the BGPsec_PATH Attribute Length field specified in 5.18.2 must be set to "0" zero.

5.20. Keep Window

Keep Window allows to request the SRx to not delete the data assigned to the current connection after the connection is terminated. If used, the keep widow specifies the time in seconds the proxy user needs to reconnect back to the SRx server. This SHOULD be the reboot time of the BGP router, approx. 15 minutes = 900 seconds. The SRx itself does not need to follow the request but is recommended to reduce the recalculation time.

5.21. Error Code

All fatal errors MUST result in a Goodbye message followed by closing the connection. Errors are only sent out from SRx, SRx itself does not except any errors. In case SRx receives an Error it MUST return an error packet with error code 2 followed by a Goodbye message.

0: Wrong Protocol Version (fatal):
The handshake fails due to a conflict of version number between the speakers.

1: Duplicate Proxy Identifier (fatal):
The handshake fails due to a conflict of the proxy identifier. An other currently active proxy is using the proxy identifier provided.

2: Invalid Packet (fatal):
This error is sent when SRx receives a packet that is either unknown or unexpected. An example could be twice a Hello Packet, a Hello Packet with insufficient number of peers provided, an Error packet send from proxy to SRx, or other.

3: Internal error (fatal):
The SRx has an internal error (memory, etc.) and is forced to abort the communication. This error might be followed by a goodbye message if possible. Otherwise the client can shut down the connection and try to reconnect after some time. For instance, 30 seconds.

4: Algorithm Not Supported:
   This error is thrown when the given algorithm for path signing / validation is not supported. This error is not considered to be fatal. It signals the proxy to resend the signing request using an alternative algorithm.

5: Update Not Found:
   This error is thrown when a signing request for an update cannot be processed because the update can not be found within the database of SRx.

   Even though this error is not considered to be fatal, it should result in a "Synchronization Request" to allow a synchronization between both parties.

6. Communication

6.1. Establish a Connection

```
Proxy                                 SRx
  |                                    |
  | -------------hello----------------> |  P: Send hello
  |                                    |  S: Receive valid hello
  |                                    |     package
  | <---------hello response----------- |  S: response flag: 0
  |                                    |     (connected)

  ~                                    ~
```

The time between sending a "hello" packet and receiving a "hello response" SHOULD not exceed 30 seconds.

In case of an error - for instance wrong version number - SRx will send an error message as response followed by a Goodbye message.

```
Proxy                                 SRx
  |                                    |
  | -------------hello----------------> |  P: Send hello
  |                                    |  S: (ERROR: Wrong Version)
  |                                    |
  | <---------hello response----------- |  S: response flag: 1
  |                                    |     (not connected)
  |                                    |
  | <-----------goodbye---------------- |  S: Send goodbye and close
  |                                    |     TCP connection
```

6.2. SRx server closes the connection

```
   Proxy                                  SRx
     ~                                     ~

     |                                     |
     | <-----------goodbye---------------- |  S: Send goodbye and close
     |                                     |     TCP connection
```

Once the goodbye message is sent out, the server can immediately tear down the connection.

6.3.   Proxy closes the connection

```
   Proxy                                  SRx
     ~                                     ~

     |                                     |
     | ------------goodbye---------------> |  P: Send goodbye and close
     |                                     |     TCP connection
```

Once the goodbye message is sent out, the proxy can immediately tear down the connection. SRx can free up all resources or keep them up for some grace period in case the connection will be reopened. The Keep Window field of the goodbye message specifies a time in seconds the data should be held.

6.4.   Create a validation request

```
   Proxy                                  SRx
     ~                                     ~

     |                                     |
     | -------------validate-------------> |  P: Send validation request
     |                                     |
     | <-------verify notification-------- |  S: Send a notification
     |                                     |     with receipt flag set.

     ~                                     ~
```

The validation request performs two major actions:

A: Initiate a request for Origin / Path validation
B: Generate a unique update id.

With each request the proxy provides validation results for both the requested origin validation result as well as the requested BGPsec path validation result to SRx. SRx uses this results in case no other validation result is available.

Once SRx completed the validation and the result differs from the previous results a result notification message is send to the proxy.

In case the validation request did not specify a validation method, no further validation is started and in such case no validation notification will be sent to the proxy. This is useful in regard to updates that originate from the router that implements/uses the proxy instance. In case an update is originated the update Id is also necessary for path signing.

6.5. Create a validation result notification

```
   Proxy                                       SRx
     ~                                          ~

     |                                          |
     | <-------verify notification-------- |  S: Send a notification
     |                                          |     that results are
     |                                          |     available.
     |                                          |     Receipt flag is NOT
     |                                          |     set.

     ~                                          ~
```

A verify notification is send for validation prior requests only. In case a verification was requested for origin validation only, only changes within origin validation are performed. Same with path validation.

6.6. Synchronization Request

The synchronization request is initiated by the SRx due to the believe that SRx and proxy might be out of sync. The proxy need not to answer this request but it is strongly recommended. Furthermore it is recommended that the proxy provides the results once received from SRx to reduce the back traffic due to new notifications. In addition it is recommended that the proxy SHOULD NOT set the "receipt flag" to prevent receiving receipts for already known update results. In case a result differs from the provided default result, SRx will send notifications. In case nothing changed in the validation no further traffic has to be processed and therefore no change should be triggered within the decision process of the router.

```
   Proxy                                  SRx
     ~                                     ~

     |                                     |
     | <----------sync request------------ |  S: Send a sync request.
     |                                     |
     | -------------validate-------------> |  P: Send validation request
     |                                     |     (SHOULD NOT SET FLAG
     |                                     |      REQUEST_RECEIPT)
     ~                                     ~
     | -------------validate-------------> |  P: Send validation request
     |                                     |     (NO REQUEST_RECEIPT)

     ~                                     ~
```

6.7.   Error Communication

At any point in time if an error occurs, SRx initiates an error message indicating the error. In case the error is considered fatal the connection MUST be closed.

```
   Proxy                                  SRx
     ~                                     ~

     |                                     |
     | <-------------error---------------- |  S: An error occurred
     |                                     |

     In case the error is fatal followed by:

     |                                     |
     | <------------goodbye--------------- |  S: Send goodbye and close
     |                                     |     TCP connection.
```

Errors that are considered fatal require a reboot of the session.

7. Implementation Suggestions

This section deals with implementation modes this protocol tries to address. In general, it is thought that a router might access the SRx through a proxy API. The proxy API then needs to implement this protocol to talk to SRx and communicate validation requests and request results between the router and SRx. This communication can be performed in two modes, synchronous and asynchronous.

Regardless of the mode the proxy is operated in some parts are and will always be asynchronous. The synchronization defined here is in regards to validation requests and their initial result value. The router itself receives four values from SRx that need to be added to each update entry within the router.

Update Identifier
   The Update Identifier (Section 5.10)

Validation Result
   Origin (Section 5.7) Path (Section 5.9)

8. Acknowledgements

The Author wants to thank Doug Montgomery and Sebastian Spiess for their input.

9. References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <https://www.rfc-editor.org/info/rfc4760>.

[RFC6811] Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R. Austein, "BGP Prefix Origin Validation", RFC 6811, DOI 10.17487/RFC6811, January 2013, <https://www.rfc-editor.org/info/rfc6811>.

[RFC8205] Lepinski, M., Ed., and K. Sriram, Ed., "BGPsec Protocol Specification", RFC 8205, DOI 10.17487/RFC8205, September 2017, <https://www.rfc-editor.org/info/rfc8205>.

[RFC8210] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol, Version 1", RFC 8210, DOI 10.17487/RFC8210, September 2017, <https://www.rfc-editor.org/info/rfc8210>.

[RFC8208] Turner, S. and O. Borchert, "BGPsec Algorithms, Key Formats, and Signature Formats", RFC 8208, DOI 10.17487/RFC8208, September 2017, <https://www.rfc-editor.org/info/rfc8208>.

Authors' Addresses

Oliver Borchert
NIST
100 Bureau Drive
Gaithersburg, MD 20899
United States of America

Email: oliver.borchert@nist.gov

Kyehwan Lee
NIST
100 Bureau Drive
Gaithersburg, MD 20899
United States of America

Email: kyehwanl@nist.gov

Patrick Gleichmann
NIST
100 Bureau Drive
Gaithersburg, MD 20899
United States of America

Email: patrick.gleichmann@nist.gov