BGPsec Protocol Specification

Abstract

   This document describes BGPsec, an extension to the Border Gateway
   Protocol (BGP) that provides security for the path of Autonomous
   Systems (ASes) through which a BGP UPDATE message passes. BGPsec is
   implemented via an optional non-transitive BGP path attribute that
   carries digital signatures produced by each AS that propagates the
   UPDATE message. The digital signatures provide confidence that every
   AS on the path of ASes listed in the UPDATE message has explicitly
   authorized the advertisement of the route.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 7841.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   https://www.rfc-editor.org/info/rfc8205.

Table of Contents

1. Introduction

   This document describes BGPsec, a mechanism for providing path
   security for Border Gateway Protocol (BGP) [RFC4271] route
   advertisements. That is, a BGP speaker who receives a valid BGPsec
   UPDATE message has cryptographic assurance that the advertised route
   has the following property: every Autonomous System (AS) on the path
   of ASes listed in the UPDATE message has explicitly authorized the
   advertisement of the route to the subsequent AS in the path.

   This document specifies an optional (non-transitive) BGP path
   attribute, BGPsec_PATH. It also describes how a BGPsec-compliant BGP
   speaker (referred to hereafter as a BGPsec speaker) can generate,
   propagate, and validate BGP UPDATE messages containing this attribute
   to obtain the above assurances.

   BGPsec is intended to be used to supplement BGP origin validation
   [RFC6483] [RFC6811], and when used in conjunction with origin
   validation, it is possible to prevent a wide variety of route
   hijacking attacks against BGP.

   BGPsec relies on the Resource Public Key Infrastructure (RPKI)
   certificates that attest to the allocation of AS number and IP
   address resources. (For more information on the RPKI, see RFC 6480
   [RFC6480] and the documents referenced therein.) Any BGPsec speaker
   who wishes to send, to external (eBGP) peers, BGP UPDATE messages
   containing the BGPsec_PATH needs to possess a private key associated
   with an RPKI router certificate [RFC8209] that corresponds to the
   BGPsec speaker's AS number. Note, however, that a BGPsec speaker
   does not need such a certificate in order to validate received UPDATE
   messages containing the BGPsec_PATH attribute (see Section 5.2).

1.1. 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. BGPsec Negotiation

   This document defines a BGP capability [RFC5492] that allows a BGP
   speaker to advertise to a neighbor the ability to send or to receive
   BGPsec UPDATE messages (i.e., UPDATE messages containing the
   BGPsec_PATH attribute).

2.1. The BGPsec Capability

   This capability has capability code 7.

   The capability length for this capability MUST be set to 3.

   The 3 octets of the capability format are specified in Figure 1.

```
           0   1   2   3       4       5   6   7
         +-------------------------------------+
         | Version           | Dir | Unassigned |
         +-------------------------------------+
         |                                     |
         +------          AFI          -----+
         |                                     |
         +-------------------------------------+
```
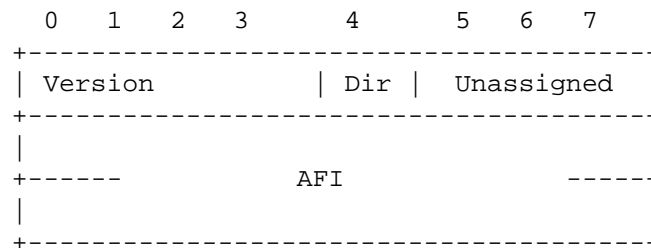
                  Figure 1: BGPsec Capability Format

   The first 4 bits of the first octet indicate the version of BGPsec
   for which the BGP speaker is advertising support. This document
   defines only BGPsec version 0 (all 4 bits set to 0). Other versions
   of BGPsec may be defined in future documents. A BGPsec speaker MAY
   advertise support for multiple versions of BGPsec by including
   multiple versions of the BGPsec capability in its BGP OPEN message.

   The fifth bit of the first octet is a Direction bit, which indicates
   whether the BGP speaker is advertising the capability to send BGPsec
   UPDATE messages or receive BGPsec UPDATE messages. The BGP speaker
   sets this bit to 0 to indicate the capability to receive BGPsec
   UPDATE messages. The BGP speaker sets this bit to 1 to indicate the
   capability to send BGPsec UPDATE messages.

   The remaining 3 bits of the first octet are unassigned and for future
   use. These bits are set to 0 by the sender of the capability and
   ignored by the receiver of the capability.

   The second and third octets contain the 16-bit Address Family
   Identifier (AFI), which indicates the address family for which the
   BGPsec speaker is advertising support for BGPsec. This document only
   specifies BGPsec for use with two address families, IPv4 and IPv6,
   with AFI values 1 and 2, respectively [IANA-AF]. BGPsec for use with
   other address families may be specified in future documents.

2.2. Negotiating BGPsec Support

   In order to indicate that a BGP speaker is willing to send BGPsec
   UPDATE messages (for a particular address family), a BGP speaker
   sends the BGPsec capability (see Section 2.1) with the Direction bit
   (the fifth bit of the first octet) set to 1. In order to indicate
   that the speaker is willing to receive BGP UPDATE messages containing
   the BGPsec_PATH attribute (for a particular address family), a BGP
   speaker sends the BGPsec capability with the Direction bit set to 0.
   In order to advertise the capability to both send and receive BGPsec
   UPDATE messages, the BGP speaker sends two copies of the BGPsec
   capability (one with the Direction bit set to 0 and one with the
   Direction bit set to 1).

   Similarly, if a BGP speaker wishes to use BGPsec with two different
   address families (i.e., IPv4 and IPv6) over the same BGP session,
   then the speaker includes two instances of this capability (one for
   each address family) in the BGP OPEN message. A BGP speaker MUST NOT
   announce BGPsec capability if it does not support the BGP
   multiprotocol extension [RFC4760]. Additionally, a BGP speaker
   MUST NOT advertise the capability of BGPsec support for a particular
   AFI unless it has also advertised the multiprotocol extension
   capability for the same AFI [RFC4760].

   In a BGPsec peering session, a peer is permitted to send UPDATE
   messages containing the BGPsec_PATH attribute if and only if:

   o The given peer sent the BGPsec capability for a particular version
      of BGPsec and a particular address family with the Direction bit
      set to 1, and

   o The other (receiving) peer sent the BGPsec capability for the same
      version of BGPsec and the same address family with the Direction
      bit set to 0.

   In such a session, it can be said that the use of the particular
   version of BGPsec has been negotiated for a particular address
   family. Traditional BGP UPDATE messages (i.e., unsigned, containing
   the AS_PATH attribute) MAY be sent within a session regardless of
   whether or not the use of BGPsec is successfully negotiated.
   However, if BGPsec is not successfully negotiated, then BGP UPDATE
   messages containing the BGPsec_PATH attribute MUST NOT be sent.

   This document defines the behavior of implementations in the case
   where BGPsec version 0 is the only version that has been successfully
   negotiated. Any future document that specifies additional versions
   of BGPsec will need to specify behavior in the case that support for
   multiple versions is negotiated.

BGPsec cannot provide meaningful security guarantees without support
for 4-byte AS numbers. Therefore, any BGP speaker that announces the
BGPsec capability, MUST also announce the capability for 4-byte AS
support [RFC6793]. If a BGP speaker sends the BGPsec capability but
not the 4-byte AS support capability, then BGPsec has not been
successfully negotiated, and UPDATE messages containing the
BGPsec_PATH attribute MUST NOT be sent within such a session.

3. The BGPsec_PATH Attribute

   The BGPsec_PATH attribute is an optional non-transitive BGP path
   attribute.

   This document registers an attribute type code for this attribute:
   BGPsec_PATH (see Section 9).

   The BGPsec_PATH attribute carries the secured information regarding
   the path of ASes through which an UPDATE message passes. This
   includes the digital signatures used to protect the path information.
   The UPDATE messages that contain the BGPsec_PATH attribute are
   referred to as "BGPsec UPDATE messages". The BGPsec_PATH attribute
   replaces the AS_PATH attribute in a BGPsec UPDATE message. That is,
   UPDATE messages that contain the BGPsec_PATH attribute MUST NOT
   contain the AS_PATH attribute, and vice versa.

   The BGPsec_PATH attribute is made up of several parts. The
   high-level diagram in Figure 2 provides an overview of the structure
   of the BGPsec_PATH attribute. ("SKI" as used in Figure 2 means
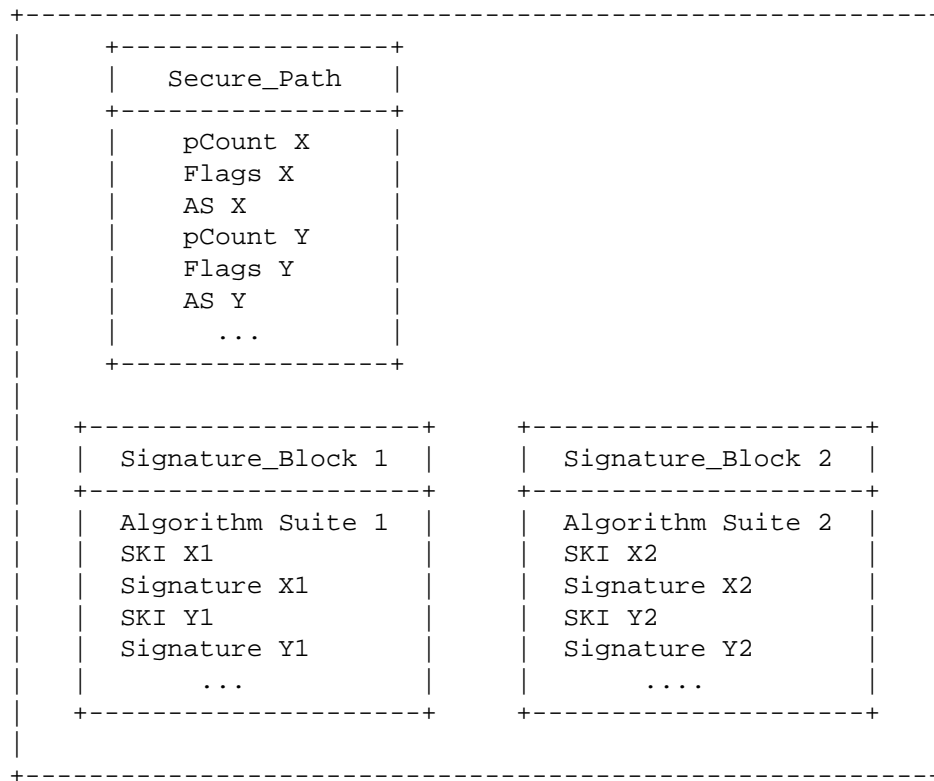   "Subject Key Identifier".)

```
+-------------------------------------------------------------+
|       +-----------------+                                   |
|       |   Secure_Path   |                                   |
|       +-----------------+                                   |
|       |    pCount X     |                                   |
|       |    Flags X      |                                   |
|       |    AS X         |                                   |
|       |    pCount Y     |                                   |
|       |    Flags Y      |                                   |
|       |    AS Y         |                                   |
|       |      ...        |                                   |
|       +-----------------+                                   |
|                                                             |
|    +--------------------+      +--------------------+       |
|    |  Signature_Block 1 |      |  Signature_Block 2 |       |
|    +--------------------+      +--------------------+       |
|    |  Algorithm Suite 1 |      |  Algorithm Suite 2 |       |
|    |  SKI X1            |      |  SKI X2            |       |
|    |  Signature X1      |      |  Signature X2      |       |
|    |  SKI Y1            |      |  SKI Y2            |       |
|    |  Signature Y1      |      |  Signature Y2      |       |
|    |      ...           |      |      ....          |       |
|    +--------------------+      +--------------------+       |
|                                                             |
+-------------------------------------------------------------+
```

Figure 2: High-Level Diagram of the BGPsec_PATH Attribute

Figure 3 provides the specification of the format for the BGPsec_PATH attribute.

```
+-------------------------------------------------------+
| Secure_Path                              (variable)   |
+-------------------------------------------------------+
| Sequence of one or two Signature_Blocks (variable)    |
+-------------------------------------------------------+
```
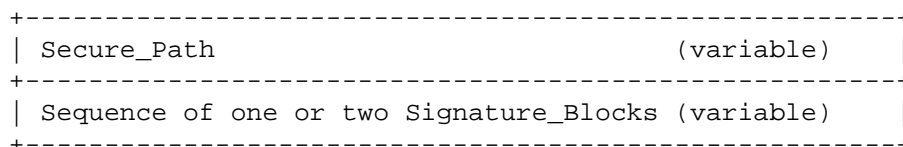
Figure 3: BGPsec_PATH Attribute Format

The Secure_Path contains AS path information for the BGPsec UPDATE message. This is logically equivalent to the information that is contained in a non-BGPsec AS_PATH attribute. The information in the Secure_Path is used by BGPsec speakers in the same way that information from the AS_PATH is used by non-BGPsec speakers. The format of the Secure_Path is described below in Section 3.1.

The BGPsec_PATH attribute will contain one or two Signature_Blocks, each of which corresponds to a different algorithm suite. Each of

the Signature_Blocks will contain a Signature Segment for each AS
number (i.e., Secure_Path Segment) in the Secure_Path. In the
most common case, the BGPsec_PATH attribute will contain only a
single Signature_Block. However, in order to enable a transition
from an old algorithm suite to a new algorithm suite (without a
flag day), it will be necessary to include two Signature_Blocks (one
for the old algorithm suite and one for the new algorithm suite)
during the transition period. (See Section 6.1 for more discussion
of algorithm transitions.) The format of the Signature_Blocks is
described below in Section 3.2.

3.1. Secure_Path

   A detailed description of the Secure_Path information in the
   BGPsec_PATH attribute is provided here. The specification for the
   Secure_Path field is provided in Figures 4 and 5.

```
            +-----------------------------------------------+
            | Secure_Path Length              (2 octets) |
            +-----------------------------------------------+
            | One or more Secure_Path Segments   (variable) |
            +-----------------------------------------------+
```
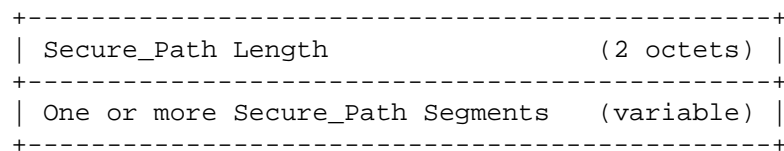
                     Figure 4: Secure_Path Format

   The Secure_Path Length contains the length (in octets) of the entire
   Secure_Path (including the 2 octets used to express this length
   field). As explained below, each Secure_Path Segment is 6 octets
   long. Note that this means the Secure_Path Length is two greater
   than six times the number of Secure_Path Segments (i.e., the number
   of AS numbers in the path).

   The Secure_Path contains one Secure_Path Segment (see Figure 5) for
   each AS in the path to the originating AS of the prefix specified in
   the UPDATE message. (Note: Repeated ASes are "compressed out" using
   the pCount field, as discussed below.)

```
   +-------------------------------------------------------+
   | pCount           (1 octet)                            |
   +-------------------------------------------------------+
   | Confed_Segment flag (1 bit) |  Unassigned (7 bits)    | (Flags)
   +-------------------------------------------------------+
   | AS Number        (4 octets)                           |
   +-------------------------------------------------------+
```
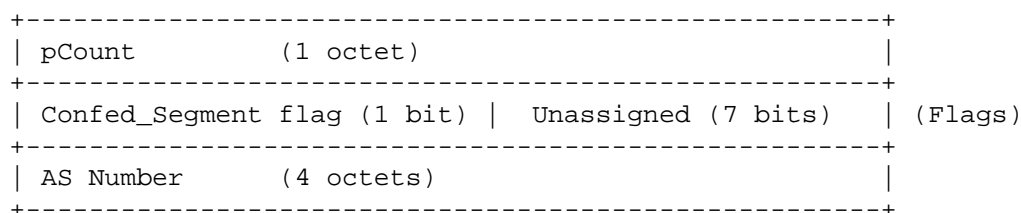
                 Figure 5: Secure_Path Segment Format

The AS Number (in Figure 5) is the AS number of the BGP speaker that
added this Secure_Path Segment to the BGPsec_PATH attribute. (See
Section 4 for more information on populating this field.)

The pCount field contains the number of repetitions of the associated
AS number that the signature covers. This field enables a BGPsec
speaker to mimic the semantics of prepending multiple copies of their
AS to the AS_PATH without requiring the speaker to generate multiple
signatures. Note that Section 9.1.2.2 ("Breaking Ties (Phase 2)") in
[RFC4271] mentions the "number of AS numbers" in the AS_PATH
attribute that is used in the route selection process. This metric
(number of AS numbers) is the same as the AS path length obtained in
BGPsec by summing the pCount values in the BGPsec_PATH attribute.
The pCount field is also useful in managing route servers (see
Section 4.2), AS confederations (see Section 4.3), and AS Number
migrations (see [RFC8206] for details).

The leftmost (i.e., the most significant) bit of the Flags field in
Figure 5 is the Confed_Segment flag. The Confed_Segment flag is set
to 1 to indicate that the BGPsec speaker that constructed this
Secure_Path Segment is sending the UPDATE message to a peer AS within
the same AS confederation [RFC5065]. (That is, a sequence of
consecutive Confed_Segment flags are set in a BGPsec UPDATE message
whenever, in a non-BGPsec UPDATE message, an AS_PATH segment of type
AS_CONFED_SEQUENCE occurs.) In all other cases, the Confed_Segment
flag is set to 0.

The remaining 7 bits of the Flags field are unassigned. They MUST be
set to 0 by the sender and ignored by the receiver. Note, however,
that the signature is computed over all 8 bits of the Flags field.

As stated earlier in Section 2.2, BGPsec peering requires that the
peering ASes MUST each support 4-byte AS numbers. Currently assigned
2-byte AS numbers are converted into 4-byte AS numbers by setting the
two high-order octets of the 4-octet field to 0 [RFC6793].

3.2. Signature_Block

   A detailed description of the Signature_Blocks in the BGPsec_PATH
   attribute is provided here using Figures 6 and 7.

```
                +----------------------------------------+
                | Signature_Block Length       (2 octets) |
                +----------------------------------------+
                | Algorithm Suite Identifier   (1 octet)  |
                +----------------------------------------+
                | Sequence of Signature Segments (variable) |
                +----------------------------------------+
```

                     Figure 6: Signature_Block Format

   The Signature_Block Length in Figure 6 is the total number of octets
   in the Signature_Block (including the 2 octets used to express this
   length field).

   The Algorithm Suite Identifier is a 1-octet identifier specifying the
   digest algorithm and digital signature algorithm used to produce the
   digital signature in each Signature Segment. An IANA registry of
   algorithm suite identifiers for use in BGPsec is specified in the
   BGPsec algorithms document [RFC8208].

   A Signature_Block in Figure 6 has exactly one Signature Segment (see
   Figure 7) for each Secure_Path Segment in the Secure_Path portion of
   the BGPsec_PATH attribute (that is, one Signature Segment for each
   distinct AS on the path for the prefix in the UPDATE message).

```
                +----------------------------------------+
                | Subject Key Identifier (SKI)  (20 octets) |
                +----------------------------------------+
                | Signature Length             (2 octets) |
                +----------------------------------------+
                | Signature                    (variable) |
                +----------------------------------------+
```

                     Figure 7: Signature Segment Format

   The Subject Key Identifier (SKI) field in Figure 7 contains the value
   in the Subject Key Identifier extension of the RPKI router
   certificate [RFC6487] that is used to verify the signature (see
   Section 5 for details on the validity of BGPsec UPDATE messages).
   The SKI field has a fixed size of 20 octets. See Section 6.2 for
   considerations for the SKI size.

   The Signature Length field contains the size (in octets) of the value
   in the Signature field of the Signature Segment.

   The Signature field in Figure 7 contains a digital signature that
   protects the prefix and the BGPsec_PATH attribute (see Sections 4 and
   5 for details on signature generation and validation, respectively).

4. BGPsec UPDATE Messages

   Section 4.1 provides general guidance on the creation of BGPsec
   UPDATE messages -- that is, UPDATE messages containing the
   BGPsec_PATH attribute.

   Section 4.2 specifies how a BGPsec speaker generates the BGPsec_PATH
   attribute to include in a BGPsec UPDATE message.

   Section 4.3 contains special processing instructions for members of
   an AS confederation [RFC5065]. A BGPsec speaker that is not a member
   of such a confederation MUST NOT set the Confed_Segment flag in its
   Secure_Path Segment (i.e., leave the Confed_Segment flag at the
   default value of 0) in all BGPsec UPDATE messages it sends.

   Section 4.4 contains instructions for reconstructing the AS_PATH
   attribute in cases where a BGPsec speaker receives an UPDATE message
   with a BGPsec_PATH attribute and wishes to propagate the UPDATE
   message to a peer who does not support BGPsec.

4.1. General Guidance

   The information protected by the signature on a BGPsec UPDATE message
   includes the AS number of the peer to whom the UPDATE message is
   being sent. Therefore, if a BGPsec speaker wishes to send a BGPsec
   UPDATE message to multiple BGP peers, it MUST generate a separate
   BGPsec UPDATE message for each unique peer AS to whom the UPDATE
   message is sent.

   A BGPsec UPDATE message MUST advertise a route to only a single
   prefix. This is because a BGPsec speaker receiving an UPDATE message
   with multiple prefixes would be unable to construct a valid BGPsec
   UPDATE message (i.e., valid path signatures) containing a subset of
   the prefixes in the received update. If a BGPsec speaker wishes to
   advertise routes to multiple prefixes, then it MUST generate a
   separate BGPsec UPDATE message for each prefix. Additionally, a
   BGPsec UPDATE message MUST use the MP_REACH_NLRI attribute [RFC4760]
   to encode the prefix.

The BGPsec_PATH attribute and the AS_PATH attribute are mutually
exclusive. That is, any UPDATE message containing the BGPsec_PATH
attribute MUST NOT contain the AS_PATH attribute. The information
that would be contained in the AS_PATH attribute is instead conveyed
in the Secure_Path portion of the BGPsec_PATH attribute.

In order to create or add a new signature to a BGPsec UPDATE message
with a given algorithm suite, the BGPsec speaker MUST possess a
private key suitable for generating signatures for this algorithm
suite. Additionally, this private key must correspond to the public
key in a valid RPKI end entity certificate whose AS number resource
extension includes the BGPsec speaker's AS number [RFC8209]. Note
also that new signatures are only added to a BGPsec UPDATE message
when a BGPsec speaker is generating an UPDATE message to send to an
external peer (i.e., when the AS number of the peer is not equal to
the BGPsec speaker's own AS number).

The RPKI enables the legitimate holder of IP address prefix(es) to
issue a signed object, called a Route Origin Authorization (ROA),
that authorizes a given AS to originate routes to a given set of
prefixes (see RFC 6482 [RFC6482]). It is expected that most Relying
Parties (RPs) will utilize BGPsec in tandem with origin validation
(see RFC 6483 [RFC6483] and RFC 6811 [RFC6811]). Therefore, it is
RECOMMENDED that a BGPsec speaker only originate a BGPsec UPDATE
message advertising a route for a given prefix if there exists a
valid ROA authorizing the BGPsec speaker's AS to originate routes to
this prefix.

If a BGPsec router has received only a non-BGPsec UPDATE message
containing the AS_PATH attribute (instead of the BGPsec_PATH
attribute) from a peer for a given prefix, then it MUST NOT attach a
BGPsec_PATH attribute when it propagates the UPDATE message. (Note
that a BGPsec router may also receive a non-BGPsec UPDATE message
from an internal peer without the AS_PATH attribute, i.e., with just
the Network Layer Reachability Information (NLRI) in it. In that
case, the prefix is originating from that AS, and if it is selected
for advertisement, the BGPsec speaker SHOULD attach a BGPsec_PATH
attribute and send a signed route (for that prefix) to its external
BGPsec-speaking peers.)

Conversely, if a BGPsec router has received a BGPsec UPDATE message
(with the BGPsec_PATH attribute) from a peer for a given prefix and
it chooses to propagate that peer's route for the prefix, then it
SHOULD propagate the route as a BGPsec UPDATE message containing the
BGPsec_PATH attribute.

Note that removing BGPsec signatures (i.e., propagating a route
advertisement without the BGPsec_PATH attribute) has significant
security ramifications. (See Section 8 for a discussion of the
security ramifications of removing BGPsec signatures.) Therefore,
when a route advertisement is received via a BGPsec UPDATE message,
propagating the route advertisement without the BGPsec_PATH attribute
is NOT RECOMMENDED, unless the message is sent to a peer that did not
advertise the capability to receive BGPsec UPDATE messages (see
Section 4.4).

Furthermore, note that when a BGPsec speaker propagates a route
advertisement with the BGPsec_PATH attribute, it is not attesting to
the validation state of the UPDATE message it received. (See
Section 8 for more discussion of the security semantics of BGPsec
signatures.)

If the BGPsec speaker is producing an UPDATE message that would, in
the absence of BGPsec, contain an AS_SET (e.g., the BGPsec speaker is
performing proxy aggregation), then the BGPsec speaker MUST NOT
include the BGPsec_PATH attribute. In such a case, the BGPsec
speaker MUST remove any existing BGPsec_PATH in the received
advertisement(s) for this prefix and produce a traditional
(non-BGPsec) UPDATE message. It should be noted that BCP 172
[RFC6472] recommends against the use of AS_SET and AS_CONFED_SET in
the AS_PATH of BGP UPDATE messages.

The case where the BGPsec speaker sends a BGPsec UPDATE message to an
iBGP (internal BGP) peer is quite simple. When originating a new
route advertisement and sending it to a BGPsec-capable iBGP peer, the
BGPsec speaker omits the BGPsec_PATH attribute. When originating a
new route advertisement and sending it to a non-BGPsec iBGP peer, the
BGPsec speaker includes an empty AS_PATH attribute in the UPDATE
message. (An empty AS_PATH attribute is one whose length field
contains the value 0 [RFC4271].) When a BGPsec speaker chooses to
forward a BGPsec UPDATE message to an iBGP peer, the BGPsec_PATH
attribute SHOULD NOT be removed, unless the peer doesn't support
BGPsec. In the case when an iBGP peer doesn't support BGPsec, then a
BGP UPDATE message with AS_PATH is reconstructed from the BGPsec
UPDATE message and then forwarded (see Section 4.4). In particular,
when forwarding to a BGPsec-capable iBGP (or eBGP) peer, the
BGPsec_PATH attribute SHOULD NOT be removed even in the case where
the BGPsec UPDATE message has not been successfully validated. (See
Section 5 for more information on validation and Section 8 for the
security ramifications of removing BGPsec signatures.)

All BGPsec UPDATE messages MUST conform to BGP's maximum message
size. If the resulting message exceeds the maximum message size,
then the guidelines in Section 9.2 of RFC 4271 [RFC4271] MUST be
followed.

4.2. Constructing the BGPsec_PATH Attribute

When a BGPsec speaker receives a BGPsec UPDATE message containing a
BGPsec_PATH attribute (with one or more signatures) from an (internal
or external) peer, it may choose to propagate the route advertisement
by sending it to its other (internal or external) peers. When
sending the route advertisement to an internal BGPsec-speaking peer,
the BGPsec_PATH attribute SHALL NOT be modified. When sending the
route advertisement to an external BGPsec-speaking peer, the
following procedures are used to form or update the BGPsec_PATH
attribute.

To generate the BGPsec_PATH attribute on the outgoing UPDATE message,
the BGPsec speaker first generates a new Secure_Path Segment. Note
that if the BGPsec speaker is not the origin AS and there is an
existing BGPsec_PATH attribute, then the BGPsec speaker prepends its
new Secure_Path Segment (places in first position) onto the existing
Secure_Path.

The AS number in this Secure_Path Segment MUST match the AS number in
the Subject field of the RPKI router certificate that will be used to
verify the digital signature constructed by this BGPsec speaker (see
Section 3.1.1 in [RFC8209] and RFC 6487 [RFC6487]).

The pCount field of the Secure_Path Segment is typically set to the
value 1. However, a BGPsec speaker may set the pCount field to a
value greater than 1. Setting the pCount field to a value greater
than 1 has the same semantics as repeating an AS number multiple
times in the AS_PATH of a non-BGPsec UPDATE message (e.g., for
traffic engineering purposes).

To prevent unnecessary processing load in the validation of BGPsec
signatures, a BGPsec speaker SHOULD NOT produce multiple consecutive
Secure_Path Segments with the same AS number. This means that to
achieve the semantics of prepending the same AS number k times, a
BGPsec speaker SHOULD produce a single Secure_Path Segment -- with a
pCount of k -- and a single corresponding Signature Segment.

A route server that participates in the BGP control plane but
does not act as a transit AS in the data plane may choose to set
pCount to 0. This option enables the route server to participate in
BGPsec and obtain the associated security guarantees without
increasing the length of the AS path. (Note that BGPsec speakers

compute the length of the AS path by summing the pCount values in the
BGPsec_PATH attribute; see Section 5.) However, when a route server
sets the pCount value to 0, it still inserts its AS number into the
Secure_Path Segment, as this information is needed to validate the
signature added by the route server. See [RFC8206] for a discussion
of setting pCount to 0 to facilitate AS Number migration. Also, see
Section 4.3 for the use of pCount=0 in the context of an AS
confederation. See Section 7.2 for operational guidance for
configuring a BGPsec router for setting pCount=0 and/or accepting
pCount=0 from a peer.

Next, the BGPsec speaker generates one or two Signature_Blocks.
Typically, a BGPsec speaker will use only a single algorithm suite
and thus create only a single Signature_Block in the BGPsec_PATH
attribute. However, to ensure backwards compatibility during a
period of transition from a 'current' algorithm suite to a 'new'
algorithm suite, it will be necessary to originate UPDATE messages
that contain a Signature_Block for both the 'current' and the 'new'
algorithm suites (see Section 6.1).

If the received BGPsec UPDATE message contains two Signature_Blocks
and the BGPsec speaker supports both of the corresponding algorithm
suites, then the new UPDATE message generated by the BGPsec speaker
MUST include both of the Signature_Blocks. If the received BGPsec
UPDATE message contains two Signature_Blocks and the BGPsec speaker
only supports one of the two corresponding algorithm suites, then the
BGPsec speaker MUST remove the Signature_Block corresponding to the
algorithm suite that it does not understand. If the BGPsec speaker
does not support the algorithm suites in any of the Signature_Blocks
contained in the received UPDATE message, then the BGPsec speaker
MUST NOT propagate the route advertisement with the BGPsec_PATH
attribute. (That is, if it chooses to propagate this route
advertisement at all, it MUST do so as an unsigned BGP UPDATE
message. See Section 4.4 for more information on converting to an
unsigned BGP UPDATE message.)

Note that in the case where the BGPsec_PATH has two Signature_Blocks
(corresponding to different algorithm suites), the validation
algorithm (see Section 5.2) deems a BGPsec UPDATE message to be
'Valid' if there is at least one supported algorithm suite (and
corresponding Signature_Block) that is deemed 'Valid'. This means
that a 'Valid' BGPsec UPDATE message may contain a Signature_Block
that is not deemed 'Valid' (e.g., contains signatures that BGPsec
does not successfully verify). Nonetheless, such Signature_Blocks
MUST NOT be removed. (See Section 8 for a discussion of the security
ramifications of this design choice.)

For each Signature_Block corresponding to an algorithm suite that the BGPsec speaker does support, the BGPsec speaker MUST add a new Signature Segment to the Signature_Block. This Signature Segment is prepended to the list of Signature Segments (placed in the first position) so that the list of Signature Segments appears in the same order as the corresponding Secure_Path Segments. The BGPsec speaker populates the fields of this new Signature Segment as follows.

The Subject Key Identifier field in the new segment is populated with the identifier contained in the Subject Key Identifier extension of the RPKI router certificate corresponding to the BGPsec speaker [RFC8209]. This Subject Key Identifier will be used by recipients of the route advertisement to identify the proper certificate to use in verifying the signature.

The Signature field in the new segment contains a digital signature that binds the prefix and BGPsec_PATH attribute to the RPKI router certificate corresponding to the BGPsec speaker. The digital signature is computed as follows:

o For clarity, let us number the Secure_Path and corresponding Signature Segments from 1 to N, as follows. Let Secure_Path Segment 1 and Signature Segment 1 be the segments produced by the origin AS. Let Secure_Path Segment 2 and Signature Segment 2 be the segments added by the next AS after the origin. Continue this method of numbering, and ultimately let Secure_Path Segment N and Signature Segment N be those that are being added by the current AS. The current AS (Nth AS) is signing and forwarding the UPDATE message to the next AS (i.e., the (N+1)th AS) in the chain of ASes that form the AS path.

o In order to construct the digital signature for Signature Segment N (the Signature Segment being produced by the current AS), first construct the sequence of octets to be hashed as shown in Figure 8. This sequence of octets includes all the data that the Nth AS attests to by adding its digital signature in the UPDATE message that is being forwarded to a BGPsec speaker in the (N+1)th AS. (For the design rationale for choosing the specific structure in Figure 8, please see [Borchert].)

```
                +-----------------------------------+
                | Target AS Number                  |
                +-----------------------------------+----\
                | Signature Segment   : N-1         |     \
                +-----------------------------------+     |
                | Secure_Path Segment : N           |     |
                +-----------------------------------+      \
                       ...                         >  Data from
                +-----------------------------------+      /   N Segments
                | Signature Segment   : 1           |     |
                +-----------------------------------+     |
                | Secure_Path Segment : 2           |     |
                +-----------------------------------+     /
                | Secure_Path Segment : 1           |    /
                +-----------------------------------+---/
                | Algorithm Suite Identifier        |
                +-----------------------------------+
                | AFI                               |
                +-----------------------------------+
                | SAFI                              |
                +-----------------------------------+
                | NLRI                              |
                +-----------------------------------+
```
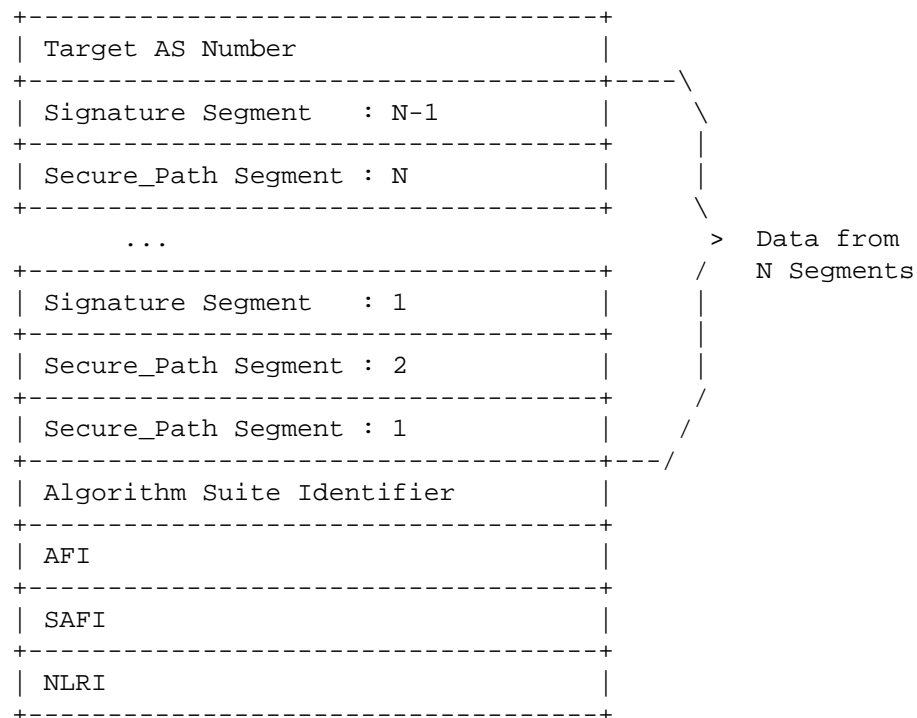
                 Figure 8: Sequence of Octets to Be Hashed

   The elements in this sequence (Figure 8) MUST be ordered exactly
   as shown. The 'Target AS Number' is the AS to whom the BGPsec
   speaker intends to send the UPDATE message. (Note that the
   'Target AS Number' is the AS number announced by the peer in the
   OPEN message of the BGP session within which the UPDATE message is
   sent.) The Secure_Path and Signature Segments (1 through N-1) are
   obtained from the BGPsec_PATH attribute. Finally, the Address
   Family Identifier (AFI), Subsequent Address Family Identifier
   (SAFI), and NLRI fields are obtained from the MP_REACH_NLRI
   attribute [RFC4760]. Additionally, in the Prefix field within the
   NLRI field (see Section 5 in RFC 4760 [RFC4760]), all of the
   trailing bits MUST be set to 0 when constructing this sequence.

   o Apply to this octet sequence (in Figure 8) the digest algorithm
      (for the algorithm suite of this Signature_Block) to obtain a
      digest value.

   o Apply to this digest value the signature algorithm (for the
      algorithm suite of this Signature_Block) to obtain the digital
      signature. Then populate the Signature field (in Figure 7) with
      this digital signature.

   The Signature Length field (in Figure 7) is populated with the length
   (in octets) of the value in the Signature field.

4.3. Processing Instructions for Confederation Members

   Members of AS confederations [RFC5065] MUST additionally follow the
   instructions in this section for processing BGPsec UPDATE messages.

   When a BGPsec speaker in an AS confederation receives a BGPsec UPDATE
   message from a peer that is external to the confederation and chooses
   to propagate the UPDATE message within the confederation, it first
   adds a signature signed to its own Member-AS (i.e., the 'Target AS
   Number' is the BGPsec speaker's Member-AS Number). In this
   internally modified UPDATE message, the newly added Secure_Path
   Segment contains the public AS number (i.e., Confederation
   Identifier), the segment's pCount value is set to 0, and
   Confed_Segment flag is set to 1. Setting pCount=0 in this case helps
   ensure that the AS path length is not unnecessarily incremented. The
   newly added signature is generated using a private key corresponding
   to the public AS number of the confederation. The BGPsec speaker
   propagates the modified UPDATE message to its peers within the
   confederation.

   Any BGPsec_PATH modifications mentioned below in the context of
   propagation of the UPDATE message within the confederation are in
   addition to the modification described above (i.e., with pCount=0).

   When a BGPsec speaker sends a BGPsec UPDATE message to a peer that
   belongs within its own Member-AS, the confederation member SHALL NOT
   modify the BGPsec_PATH attribute. When a BGPsec speaker sends a
   BGPsec UPDATE message to a peer that is within the same confederation
   but in a different Member-AS, the BGPsec speaker puts its Member-AS
   Number in the AS Number field of the Secure_Path Segment that it adds
   to the BGPsec UPDATE message. Additionally, in this case, the
   Member-AS that generates the Secure_Path Segment sets the
   Confed_Segment flag to 1. Further, the signature is generated with a
   private key corresponding to the BGPsec speaker's Member-AS Number.
   (Note: In this document, intra-Member-AS peering is regarded as iBGP,
   and inter-Member-AS peering is regarded as eBGP. The latter is also
   known as confederation-eBGP.)

   Within a confederation, the verification of BGPsec signatures added
   by other members of the confederation is optional. Note that if a
   confederation chooses not to verify digital signatures within the
   confederation, then BGPsec is not able to provide any assurances
   about the integrity of the Member-AS Numbers placed in Secure_Path
   Segments where the Confed_Segment flag is set to 1.

When a confederation member receives a BGPsec UPDATE message from a
peer within the confederation and propagates it to a peer outside the
confederation, it needs to remove all of the Secure_Path Segments
added by confederation members as well as the corresponding Signature
Segments. To do this, the confederation member propagating the route
outside the confederation does the following:

o First, starting with the most recently added Secure_Path Segment,
   remove all of the consecutive Secure_Path Segments that have the
   Confed_Segment flag set to 1. Stop this process once a
   Secure_Path Segment that has its Confed_Segment flag set to 0 is
   reached. Keep a count of the number of segments removed in this
   fashion.

o Second, starting with the most recently added Signature Segment,
   remove a number of Signature Segments equal to the number of
   Secure_Path Segments removed in the previous step. (That is,
   remove the K most recently added Signature Segments, where K is
   the number of Secure_Path Segments removed in the previous step.)

o Finally, add a Secure_Path Segment containing, in the AS field,
   the AS Confederation Identifier (the public AS number of the
   confederation) as well as a corresponding Signature Segment. Note
   that all fields other than the AS field are populated as per
   Section 4.2.

Finally, as discussed above, an AS confederation MAY optionally
decide that its members will not verify digital signatures added by
members. In such a confederation, when a BGPsec speaker runs the
algorithm in Section 5.2, the BGPsec speaker, during the process of
signature verifications, first checks whether the Confed_Segment flag
in a Secure_Path Segment is set to 1. If the flag is set to 1, the
BGPsec speaker skips the verification for the corresponding signature
and immediately moves on to the next Secure_Path Segment. Note that
as specified in Section 5.2, it is an error when a BGPsec speaker
receives, from a peer who is not in the same AS confederation, a
BGPsec UPDATE message containing a Confed_Segment flag set to 1.

4.4. Reconstructing the AS_PATH Attribute

BGPsec UPDATE messages do not contain the AS_PATH attribute.
However, the AS_PATH attribute can be reconstructed from the
BGPsec_PATH attribute. This is necessary in the case where a route
advertisement is received via a BGPsec UPDATE message and then
propagated to a peer via a non-BGPsec UPDATE message (e.g., because
the latter peer does not support BGPsec). Note that there may be
additional cases where an implementation finds it useful to perform
this reconstruction. Before attempting to reconstruct an AS_PATH for

the purpose of forwarding an unsigned (non-BGPsec) UPDATE message to
a peer, a BGPsec speaker MUST perform the basic integrity checks
listed in Section 5.2 to ensure that the received BGPsec UPDATE
message is properly formed.

The AS_PATH attribute can be constructed from the BGPsec_PATH
attribute as follows. Starting with a blank AS_PATH attribute,
process the Secure_Path Segments in order from least recently added
(corresponding to the origin) to most recently added. For each
Secure_Path Segment, perform the following steps:

1. If the Secure_Path Segment has pCount=0, then do nothing (i.e.,
    move on to process the next Secure_Path Segment).

2. If the Secure_Path Segment has pCount greater than 0 and the
    Confed_Segment flag is set to 1, then look at the most recently
    added segment in the AS_PATH.

    *   In the case where the AS_PATH is blank or in the case where
        the most recently added segment is of type AS_SEQUENCE, add
        (prepend to the AS_PATH) a new AS_PATH segment of type
        AS_CONFED_SEQUENCE. This segment of type AS_CONFED_SEQUENCE
        shall contain a number of elements equal to the pCount field
        in the current Secure_Path Segment. Each of these elements
        shall be the AS number contained in the current Secure_Path
        Segment. (That is, if the pCount field is X, then the segment
        of type AS_CONFED_SEQUENCE contains X copies of the
        Secure_Path Segment's AS Number field.)

    *   In the case where the most recently added segment in the
        AS_PATH is of type AS_CONFED_SEQUENCE, then add (prepend to
        the segment) a number of elements equal to the pCount field in
        the current Secure_Path Segment. The value of each of these
        elements shall be the AS number contained in the current
        Secure_Path Segment. (That is, if the pCount field is X, then
        add X copies of the Secure_Path Segment's AS Number field to
        the existing AS_CONFED_SEQUENCE.)

3. If the Secure_Path Segment has pCount greater than 0 and the
    Confed_Segment flag is set to 0, then look at the most recently
    added segment in the AS_PATH.

    *   In the case where the AS_PATH is blank or in the case where
        the most recently added segment is of type AS_CONFED_SEQUENCE,
        add (prepend to the AS_PATH) a new AS_PATH segment of type
        AS_SEQUENCE. This segment of type AS_SEQUENCE shall contain a
        number of elements equal to the pCount field in the current
        Secure_Path Segment. Each of these elements shall be the AS

number contained in the current Secure_Path Segment. (That
is, if the pCount field is X, then the segment of type
AS_SEQUENCE contains X copies of the Secure_Path Segment's AS
Number field.)

   * In the case where the most recently added segment in the
      AS_PATH is of type AS_SEQUENCE, then add (prepend to the
      segment) a number of elements equal to the pCount field in the
      current Secure_Path Segment. The value of each of these
      elements shall be the AS number contained in the current
      Secure_Path Segment. (That is, if the pCount field is X, then
      add X copies of the Secure_Path Segment's AS Number field to
      the existing AS_SEQUENCE.)

As part of the procedure described above, the following additional
actions are performed in order not to exceed the size limitations of
AS_SEQUENCE and AS_CONFED_SEQUENCE. While adding the next
Secure_Path Segment (with its prepends, if any) to the AS_PATH being
assembled, if it would cause the AS_SEQUENCE (or AS_CONFED_SEQUENCE)
at hand to exceed the limit of 255 AS numbers per segment [RFC4271]
[RFC5065], then the BGPsec speaker would follow the recommendations
in RFC 4271 [RFC4271] and RFC 5065 [RFC5065] of creating another
segment of the same type (AS_SEQUENCE or AS_CONFED_SEQUENCE) and
continue filling that.

Finally, one special case of reconstruction of AS_PATH is when the
BGPsec_PATH attribute is absent. As explained in Section 4.1, when a
BGPsec speaker originates a prefix and sends it to a BGPsec-capable
iBGP peer, the BGPsec_PATH is not attached. So, when received from a
BGPsec-capable iBGP peer, no BGPsec_PATH attribute in a BGPsec UPDATE
message is equivalent to an empty AS_PATH [RFC4271].

5. Processing a Received BGPsec UPDATE Message

Upon receiving a BGPsec UPDATE message from an external (eBGP) peer,
a BGPsec speaker SHOULD validate the message to determine the
authenticity of the path information contained in the BGPsec_PATH
attribute. Typically, a BGPsec speaker will also wish to perform
origin validation (see RFC 6483 [RFC6483] and RFC 6811 [RFC6811]) on
an incoming BGPsec UPDATE message, but such validation is independent
of the validation described in this section.

Section 5.1 provides an overview of BGPsec validation, and
Section 5.2 provides a specific algorithm for performing such
validation. (Note that an implementation need not follow the
specific algorithm in Section 5.2 as long as the input/output
behavior of the validation is identical to that of the algorithm in
Section 5.2.) During exceptional conditions (e.g., the BGPsec

speaker receives an incredibly large number of UPDATE messages at
once), a BGPsec speaker MAY temporarily defer validation of incoming
BGPsec UPDATE messages. The treatment of such BGPsec UPDATE
messages, whose validation has been deferred, is a matter of local
policy. However, an implementation SHOULD ensure that deferment of
validation and status of deferred messages is visible to the
operator.

The validity of BGPsec UPDATE messages is a function of the current
RPKI state. When a BGPsec speaker learns that the RPKI state has
changed (e.g., from an RPKI validating cache via the RPKI-Router
protocol [RFC8210]), the BGPsec speaker MUST rerun validation on all
affected UPDATE messages stored in its Adj-RIB-In [RFC4271]. For
example, when a given RPKI router certificate ceases to be valid
(e.g., it expires or is revoked), all UPDATE messages containing a
signature whose SKI matches the SKI in the given certificate MUST be
reassessed to determine if they are still valid. If this
reassessment determines that the validity state of an UPDATE message
has changed, then, depending on local policy, it may be necessary to
rerun best path selection.

BGPsec UPDATE messages do not contain an AS_PATH attribute. The
Secure_Path contains AS path information for the BGPsec UPDATE
message. Therefore, a BGPsec speaker MUST utilize the AS path
information in the Secure_Path in all cases where it would otherwise
use the AS path information in the AS_PATH attribute. The only
exception to this rule is when AS path information must be updated in
order to propagate a route to a peer (in which case the BGPsec
speaker follows the instructions in Section 4). Section 4.4 provides
an algorithm for constructing an AS_PATH attribute from a BGPsec_PATH
attribute. Whenever the use of AS path information is called for
(e.g., loop detection or the use of the AS path length in best path
selection), the externally visible behavior of the implementation
shall be the same as if the implementation had run the algorithm in
Section 4.4 and used the resulting AS_PATH attribute as it would for
a non-BGPsec UPDATE message.

5.1. Overview of BGPsec Validation

Validation of a BGPsec UPDATE message makes use of data from RPKI
router certificates. In particular, it is necessary that the
recipient have access to the following data obtained from valid RPKI
router certificates: the AS Number, Public Key, and Subject Key
Identifier from each valid RPKI router certificate.

Note that the BGPsec speaker could perform the validation of RPKI
router certificates on its own and extract the required data, or it
could receive the same data from a trusted cache that performs RPKI

validation on behalf of (some set of) BGPsec speakers. (For example,
the trusted cache could deliver the necessary validity information to
the BGPsec speaker by using the Router Key PDU (Protocol Data Unit)
for the RPKI-Router protocol [RFC8210].)

To validate a BGPsec UPDATE message containing the BGPsec_PATH
attribute, the recipient performs the validation steps specified in
Section 5.2. The validation procedure results in one of two states:
'Valid' and 'Not Valid'.

It is expected that the output of the validation procedure will be
used as an input to BGP route selection. That said, BGP route
selection, and thus the handling of the validation states, is a
matter of local policy and is handled using local policy mechanisms.
Implementations SHOULD enable operators to set such local policy on a
per-session basis. (That is, it is expected that some operators will
choose to treat BGPsec validation status differently for UPDATE
messages received over different BGP sessions.)

BGPsec validation need only be performed at the eBGP edge. The
validation status of a BGP signed/unsigned UPDATE message MAY be
conveyed via iBGP from an ingress edge router to an egress edge
router via some mechanism, according to local policy within an AS.
As discussed in Section 4, when a BGPsec speaker chooses to forward a
(syntactically correct) BGPsec UPDATE message, it SHOULD be forwarded
with its BGPsec_PATH attribute intact (regardless of the validation
state of the UPDATE message). Based entirely on local policy, an
egress router receiving a BGPsec UPDATE message from within its own
AS MAY choose to perform its own validation.

5.2. Validation Algorithm

   This section specifies an algorithm for validation of BGPsec UPDATE
   messages. A conformant implementation MUST include a BGPsec update
   validation algorithm that is functionally equivalent to the
   externally visible behavior of this algorithm.

   First, the recipient of a BGPsec UPDATE message performs a check to
   ensure that the message is properly formed. Both syntactical and
   protocol violation errors are checked. The BGPsec_PATH attribute
   MUST be present when a BGPsec UPDATE message is received from an
   external (eBGP) BGPsec peer and also when such an UPDATE message is
   propagated to an internal (iBGP) BGPsec peer (see Section 4.2). The
   error checks specified in Section 6.3 of [RFC4271] are performed,
   except that for BGPsec UPDATE messages the checks on the AS_PATH
   attribute do not apply and instead the following checks on the
   BGPsec_PATH attribute are performed:

   1. Check to ensure that the entire BGPsec_PATH attribute is
       syntactically correct (conforms to the specification in this
       document).

   2. Check that the AS number in the most recently added Secure_Path
       Segment (i.e., the one corresponding to the eBGP peer from which
       the UPDATE message was received) matches the AS number of that
       peer as specified in the BGP OPEN message. (Note: This check is
       performed only at an ingress BGPsec router where the UPDATE
       message is first received from a peer AS.)

   3. Check that each Signature_Block contains one Signature Segment
       for each Secure_Path Segment in the Secure_Path portion of the
       BGPsec_PATH attribute. (Note that the entirety of each
       Signature_Block MUST be checked to ensure that it is well formed,
       even though the validation process may terminate before all
       signatures are cryptographically verified.)

   4. Check that the UPDATE message does not contain an AS_PATH
       attribute.

   5. If the UPDATE message was received from a BGPsec peer that is not
       a member of the BGPsec speaker's AS confederation, check to
       ensure that none of the Secure_Path Segments contain a Flags
       field with the Confed_Segment flag set to 1.

   6. If the UPDATE message was received from a BGPsec peer that is a
       member of the BGPsec speaker's AS confederation, check to ensure
       that the Secure_Path Segment corresponding to that peer contains
       a Flags field with the Confed_Segment flag set to 1.

   7. If the UPDATE message was received from a peer that is not
       expected to set pCount=0 (see Sections 4.2 and 4.3), then check
       to ensure that the pCount field in the most recently added
       Secure_Path Segment is not equal to 0. (Note: See Section 7.2
       for router configuration guidance related to this item.)

   8. Using the equivalent of AS_PATH corresponding to the Secure_Path
       in the UPDATE message (see Section 4.4), check that the local AS
       number is not present in the AS path (i.e., rule out an AS loop).

   If any of these checks fail, it is an error in the BGPsec_PATH
   attribute. BGPsec speakers MUST handle any syntactical or protocol
   errors in the BGPsec_PATH attribute by using the "treat-as-withdraw"
   approach as defined in RFC 7606 [RFC7606]. (Note: Since the AS
   number of a transparent route server does appear in the Secure_Path
   with pCount=0, the route server MAY check to see if its local AS is

listed in the Secure_Path, and this check MAY be included in the
loop-detection check listed above.)

Next, the BGPsec speaker examines the Signature_Blocks in the
BGPsec_PATH attribute. A Signature_Block corresponding to an
algorithm suite that the BGPsec speaker does not support is not
considered in the validation process. If there is no Signature_Block
corresponding to an algorithm suite that the BGPsec speaker supports,
then in order to consider the UPDATE message in the route selection
process, the BGPsec speaker MUST strip the Signature_Block(s),
reconstruct the AS_PATH from the Secure_Path (see Section 4.4), and
treat the UPDATE message as if it were received as an unsigned BGP
UPDATE message.

For each remaining Signature_Block (corresponding to an algorithm
suite supported by the BGPsec speaker), the BGPsec speaker iterates
through the Signature Segments in the Signature_Block, starting with
the most recently added segment (and concluding with the
least recently added segment). Note that there is a one-to-one
correspondence between Signature Segments and Secure_Path Segments
within the BGPsec_PATH attribute. The following steps make use of
this correspondence:

o Step 1: Let there be K AS hops in a received BGPsec_PATH attribute
   that is to be validated. Let AS(1), AS(2), ..., AS(K+1) denote
   the sequence of AS numbers from the origin AS to the validating
   AS. Let Secure_Path Segment N and Signature Segment N in the
   BGPsec_PATH attribute refer to those corresponding to AS(N) (where
   N = 1, 2, ..., K). The BGPsec speaker that is processing and
   validating the BGPsec_PATH attribute resides in AS(K+1). Let
   Signature Segment N be the Signature Segment that is currently
   being verified.

o Step 2: Locate the public key needed to verify the signature (in
   the current Signature Segment). To do this, consult the valid
   RPKI router certificate data and look up all valid (AS Number,
   Public Key, Subject Key Identifier) triples in which the AS
   matches the AS number in the corresponding Secure_Path Segment.
   Of these triples that match the AS number, check whether there is
   an SKI that matches the value in the Subject Key Identifier field
   of the Signature Segment. If this check finds no such matching
   SKI value, then mark the entire Signature_Block as 'Not Valid' and
   proceed to the next Signature_Block.

o Step 3: Compute the digest function (for the given algorithm
   suite) on the appropriate data.

In order to verify the digital signature in Signature Segment N,
construct the sequence of octets to be hashed as shown in Figure 9
(using the notations defined in Step 1). (Note that this sequence
is the same sequence that was used by AS(N) that created the
Signature Segment N (see Section 4.2 and Figure 8).)

```
   +----------------------------------+
   | Target AS Number                 |
   +----------------------------------+----\
   | Signature Segment   : N-1        |     \
   +----------------------------------+     |
   | Secure_Path Segment : N          |     |
   +----------------------------------+      \
          ...                              >  Data from
   +----------------------------------+      /  N Segments
   | Signature Segment   : 1          |     |
   +----------------------------------+     |
   | Secure_Path Segment : 2          |     |
   +----------------------------------+     /
   | Secure_Path Segment : 1          |    /
   +----------------------------------+---/
   | Algorithm Suite Identifier       |
   +----------------------------------+
   | AFI                              |
   +----------------------------------+
   | SAFI                             |
   +----------------------------------+
   | NLRI                             |
   +----------------------------------+
```
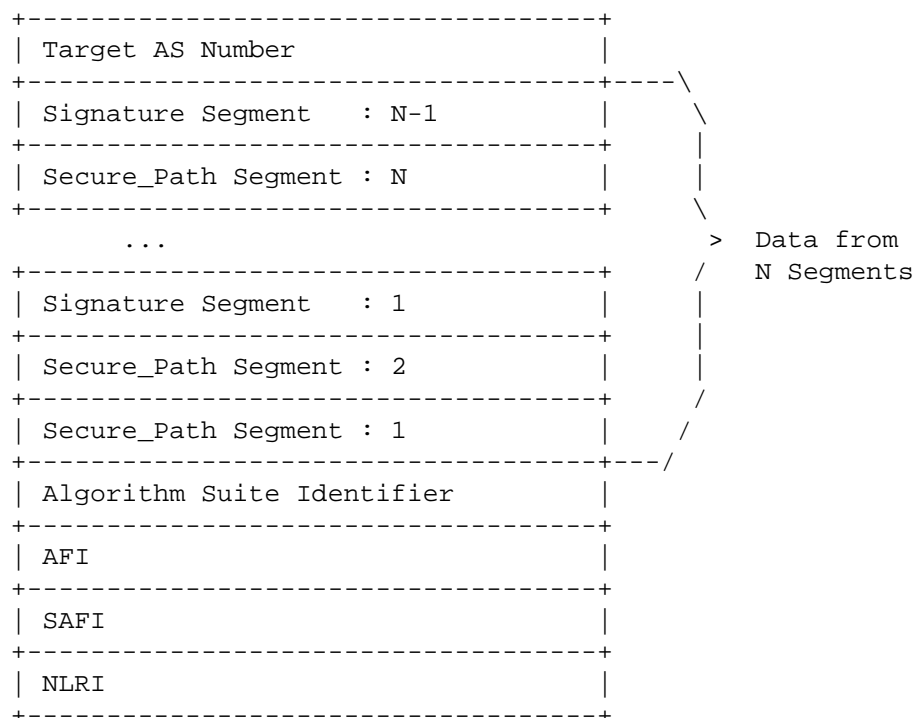
Figure 9: Sequence of Octets to Be Hashed for Signature Verification
  of Signature Segment N; N = 1,2, ..., K, Where K Is the Number of
                AS Hops in the BGPsec_PATH Attribute

The elements in this sequence (Figure 9) MUST be ordered exactly
as shown. For the first segment to be processed (the
most recently added segment (i.e., N = K) given that there are K
hops in the Secure_Path), the 'Target AS Number' is AS(K+1), the
AS number of the BGPsec speaker validating the UPDATE message.
Note that if a BGPsec speaker uses multiple AS Numbers (e.g., the
BGPsec speaker is a member of a confederation), the AS number used
here MUST be the AS number announced in the OPEN message for the
BGP session over which the BGPsec UPDATE message was received.

For each other Signature Segment (N smaller than K), the 'Target
AS Number' is AS(N+1), the AS number in the Secure_Path Segment
that corresponds to the Signature Segment added immediately after
the one being processed (that is, in the Secure_Path Segment that

         corresponds to the Signature Segment that the validator just
         finished processing).

         The Secure_Path and Signature Segment are obtained from the
         BGPsec_PATH attribute. The AFI, SAFI, and NLRI fields are
         obtained from the MP_REACH_NLRI attribute [RFC4760].
         Additionally, in the Prefix field within the NLRI field (see
         Section 5 in RFC 4760 [RFC4760]), all of the trailing bits MUST be
         set to 0 when constructing this sequence.

   o Step 4: Use the signature validation algorithm (for the given
      algorithm suite) to verify the signature in the current segment.
      That is, invoke the signature validation algorithm on the
      following three inputs: the value of the Signature field in the
      current segment, the digest value computed in Step 3 above, and
      the public key obtained from the valid RPKI data in Step 2 above.
      If the signature validation algorithm determines that the
      signature is invalid, then mark the entire Signature_Block as
      'Not Valid' and proceed to the next Signature_Block. If the
      signature validation algorithm determines that the signature is
      valid, then continue processing Signature Segments (within the
      current Signature_Block).

   If all Signature Segments within a Signature_Block pass validation
   (i.e., all segments are processed and the Signature_Block has not yet
   been marked 'Not Valid'), then the Signature_Block is marked as
   'Valid'.

   If at least one Signature_Block is marked as 'Valid', then the
   validation algorithm terminates and the BGPsec UPDATE message is
   deemed 'Valid'. (That is, if a BGPsec UPDATE message contains two
   Signature_Blocks, then the UPDATE message is deemed 'Valid' if the
   first Signature_Block is marked 'Valid' OR the second Signature_Block
   is marked 'Valid'.)

6. Algorithms and Extensibility

6.1. Algorithm Suite Considerations

   Note that there is currently no support for bilateral negotiation
   (using BGP capabilities) between BGPsec peers to use a particular
   (digest and signature) algorithm suite. This is because the
   algorithm suite used by the sender of a BGPsec UPDATE message MUST be
   understood not only by the peer to whom it is directly sending the
   message but also by all BGPsec speakers to whom the route
   advertisement is eventually propagated. Therefore, selection of an
   algorithm suite cannot be a local matter negotiated by BGP peers but
   instead must be coordinated throughout the Internet.

   To this end, [RFC8208] specifies a mandatory-to-use 'current'
   algorithm suite for use by all BGPsec speakers.

   It is anticipated that, in the future, [RFC8208] or its successor
   will be updated to specify a transition from the 'current' algorithm
   suite to a 'new' algorithm suite. During the period of transition,
   all BGPsec UPDATE messages SHOULD simultaneously use both the
   'current' algorithm suite and the 'new' algorithm suite. (Note that
   Sections 3 and 4 specify how the BGPsec_PATH attribute can contain
   signatures, in parallel, for two algorithm suites.) Once the
   transition is complete, the use of the old 'current' algorithm will
   be deprecated, the use of the 'new' algorithm will be mandatory, and
   a subsequent 'even newer' algorithm suite may be specified as
   "recommended to implement". Once the transition has successfully
   been completed in this manner, BGPsec speakers SHOULD include only a
   single Signature_Block (corresponding to the 'new' algorithm).

6.2. Considerations for the SKI Size

   Depending on the method of generating key identifiers [RFC7093], the
   size of the SKI in an RPKI router certificate may vary. The SKI
   field in the BGPsec_PATH attribute has a fixed size of 20 octets (see
   Figure 7). If the SKI is longer than 20 octets, then use the
   leftmost 20 octets of the SKI (excluding the tag and length)
   [RFC7093]. If the SKI value is shorter than 20 octets, then pad the
   SKI (excluding the tag and length) to the right (least significant
   octets) with octets having "0" values.

6.3. Extensibility Considerations

   This section discusses potential changes to BGPsec that would require
   substantial changes to the processing of the BGPsec_PATH and thus
   necessitate a new version of BGPsec. Examples of such changes
   include:

   o A new type of signature algorithm that produces signatures of
      variable length

   o A new type of signature algorithm for which the number of
      signatures in the Signature_Block is not equal to the number of
      ASes in the Secure_Path (e.g., aggregate signatures)

   o Changes to the data that is protected by the BGPsec signatures
      (e.g., attributes other than the AS path)

   In the case that such a change to BGPsec were deemed desirable, it is
   expected that a subsequent version of BGPsec would be created and
   that this version of BGPsec would specify a new BGP path attribute --

let's call it "BGPsec_PATH_Two" -- that is designed to accommodate
the desired changes to BGPsec. In such a case, [RFC8208] or its
successor would be updated to specify algorithm suites appropriate
for the new version of BGPsec.

At this point, a transition would begin that is analogous to the
algorithm transition discussed in Section 6.1. During the transition
period, all BGPsec speakers SHOULD simultaneously include both the
BGPsec_PATH attribute and the new BGPsec_PATH_Two attribute. Once
the transition is complete, the use of BGPsec_PATH could then be
deprecated, at which point BGPsec speakers should include only the
new BGPsec_PATH_Two attribute. Such a process could facilitate a
transition to a new BGPsec semantics in a backwards-compatible
fashion.

7. Operations and Management Considerations

Some operations and management issues that are closely relevant to
BGPsec protocol specification and deployment are highlighted here.
The best practices concerning operations and deployment of BGPsec are
provided in [RFC8207].

7.1. Capability Negotiation Failure

Section 2.2 describes the negotiation required to establish a
BGPsec-capable peering session. Not only must the BGPsec capability
be exchanged (and agreed on), but the BGP multiprotocol extension
[RFC4760] for the same AFI and the 4-byte AS capability [RFC6793]
MUST also be exchanged. Failure to properly negotiate a BGPsec
session -- due to a missing capability, for example -- may still
result in the exchange of BGP (unsigned) UPDATE messages. It is
RECOMMENDED that an implementation log the failure to properly
negotiate a BGPsec session. Also, an implementation MUST have the
ability to prevent a BGP session from being established if configured
to only use BGPsec.

7.2. Preventing Misuse of pCount=0

A peer that is an Internet Exchange Point (IXP) (i.e., route server)
with a transparent AS is expected to set pCount=0 in its Secure_Path
Segment while forwarding an UPDATE message to a peer (see
Section 4.2). Clearly, such an IXP MUST configure its BGPsec router
to set pCount=0 in its Secure_Path Segment. This also means that a
BGPsec speaker MUST be configured so that it permits pCount=0 from an
IXP peer. Two other cases where pCount is set to 0 are in the
contexts of an AS confederation (see Section 4.3) and of AS migration
[RFC8206]. In these two cases, pCount=0 is set and accepted within
the same AS (albeit the AS has two different identities). Note that

if a BGPsec speaker does not expect a peer AS to set its pCount=0 and
if an UPDATE message received from that peer violates this, then the
UPDATE message MUST be considered to be in error (see the list of
checks in Section 5.2). See Section 8.4 for a discussion of security
considerations concerning pCount=0.

7.3. Early Termination of Signature Verification

During the validation of a BGPsec UPDATE message, route processor
performance speedup can be achieved by incorporating the following
observations. An UPDATE message is deemed 'Valid' if at least one of
the Signature_Blocks is marked as 'Valid' (see Section 5.2).
Therefore, if an UPDATE message contains two Signature_Blocks and the
first one verified is found 'Valid', then the second Signature_Block
does not have to be verified. And if the UPDATE message is chosen
for best path, then the BGPsec speaker adds its signature (generated
with the respective algorithm) to each of the two Signature_Blocks
and forwards the UPDATE message. Also, a BGPsec UPDATE message is
deemed 'Not Valid' if at least one signature in each of the
Signature_Blocks is invalid. This principle can also be used for
route processor workload savings, i.e., the verification for a
Signature_Block terminates early when the first invalid signature is
encountered.

7.4. Non-deterministic Signature Algorithms

Many signature algorithms are non-deterministic. That is, many
signature algorithms will produce different signatures each time they
are run (even when they are signing the same data with the same key).
Therefore, if a BGPsec router receives a BGPsec UPDATE message from a
peer and later receives a second BGPsec UPDATE message from the same
peer for the same prefix with the same Secure_Path and SKIs, the
second UPDATE message MAY differ from the first UPDATE message in the
signature fields (for a non-deterministic signature algorithm).
However, the two sets of signature fields will not differ if the
sender caches and reuses the previous signature. For a deterministic
signature algorithm, the signature fields MUST be identical between
the two UPDATE messages. On the basis of these observations, an
implementation MAY incorporate optimizations in update validation
processing.

7.5. Private AS Numbers

It is possible that a stub customer of an ISP employs a private AS
number. Such a stub customer cannot publish a ROA in the Global RPKI
for the private AS number and the prefixes that they use. Also, the
Global RPKI cannot support private AS numbers (i.e., BGPsec speakers
in private ASes cannot be issued router certificates in the Global

RPKI). For interactions between the stub customer (with the private
AS number) and the ISP, the following two scenarios are possible:

1. The stub customer sends an unsigned BGP UPDATE message for a
    prefix to the ISP's AS. An edge BGPsec speaker in the ISP's AS
    may choose to propagate the prefix to its non-BGPsec and BGPsec
    peers. If so, the ISP's edge BGPsec speaker MUST strip the
    AS_PATH with the private AS number and then (a) re-originate the
    prefix without any signatures towards its non-BGPsec peer and
    (b) re-originate the prefix including its own signature towards
    its BGPsec peer. In both cases (i.e., (a) and (b)), the prefix
    MUST have a ROA in the Global RPKI authorizing the ISP's AS to
    originate it.

2. The ISP and the stub customer may use a local RPKI repository
    (using a mechanism such as one of the mechanisms described in
    [SLURM]). Then, there can be a ROA for the prefix originated by
    the stub AS, and the eBGP speaker in the stub AS can be a BGPsec
    speaker having a router certificate, albeit the ROA and router
    certificate are valid only locally. With this arrangement, the
    stub AS sends a signed UPDATE message for the prefix to the ISP's
    AS. An edge BGPsec speaker in the ISP's AS validates the UPDATE
    message, using RPKI data based on the local RPKI view. Further,
    it may choose to propagate the prefix to its non-BGPsec and
    BGPsec peers. If so, the ISP's edge BGPsec speaker MUST strip
    the Secure_Path and the Signature Segment received from the stub
    AS with the private AS number and then (a) re-originate the
    prefix without any signatures towards its non-BGPsec peer and
    (b) re-originate the prefix including its own signature towards
    its BGPsec peer. In both cases (i.e., (a) and (b)), the prefix
    MUST have a ROA in the Global RPKI authorizing the ISP's AS to
    originate it.

It is possible that private AS numbers are used in an AS
confederation [RFC5065]. The BGPsec protocol requires that when a
BGPsec UPDATE message propagates through a confederation, each
Member-AS that forwards it to a peer Member-AS MUST sign the UPDATE
message (see Section 4.3). However, the Global RPKI cannot support
private AS numbers. In order for the BGPsec speakers in Member-ASes
with private AS numbers to have digital certificates, there MUST be a
mechanism in place in the confederation that allows the establishment
of a local, customized view of the RPKI, augmenting the Global RPKI
repository data as needed. Since this mechanism (for augmenting and
maintaining a local image of RPKI data) operates locally within an AS
or AS confederation, it need not be standard based. However, a
standard-based mechanism can be used (see [SLURM]). Recall that in
order to prevent exposure of the internals of AS confederations, a

BGPsec speaker exporting to a non-member removes all
intra-confederation Secure_Path Segments and Signatures (see
Section 4.3).

7.6. Robustness Considerations for Accessing RPKI Data

The deployment structure, technologies, and best practices concerning
Global RPKI data to reach routers (via local RPKI caches) are
described in [RFC6810], [RFC8210], [RFC8181], [RFC7115], [RFC8207],
and [RFC8182]. For example, Serial-Number-based incremental update
mechanisms are used for efficient transfer of just the data records
that have changed since the last update [RFC6810] [RFC8210]. The
update notification file is used by Relying Parties (RPs) to discover
whether any changes exist between the state of the Global RPKI
repository and the RP's cache [RFC8182]. The notification describes
the location of (1) the files containing the snapshot and
(2) incremental deltas, which can be used by the RP to synchronize
with the repository. Making use of these technologies and best
practices results in enabling robustness, efficiency, and better
security for the BGPsec routers and RPKI caches in terms of the flow
of RPKI data from repositories to RPKI caches to routers. With these
mechanisms, it is believed that an attacker wouldn't be able to
meaningfully correlate RPKI data flows with BGPsec RP (or router)
actions, thus avoiding attacks that may attempt to determine the set
of ASes interacting with an RP via the interactions between the RP
and RPKI servers.

7.7. Graceful Restart

During Graceful Restart (GR), restarting and receiving BGPsec
speakers MUST follow the procedures specified in [RFC4724] for
restarting and receiving BGP speakers, respectively. In particular,
the behavior of retaining the forwarding state for the routes in the
Loc-RIB [RFC4271] and marking them as stale, as well as not
differentiating between stale routing information and other
information during forwarding, will be the same as the behavior
specified in [RFC4724].

7.8. Robustness of Secret Random Number in ECDSA

The Elliptic Curve Digital Signature Algorithm (ECDSA) with curve
P-256 is used for signing UPDATE messages in BGPsec [RFC8208]. For
ECDSA, it is stated in Section 6.3 of [FIPS186-4] that a new secret
random number "k" shall be generated prior to the generation of each
digital signature. A high-entropy random bit generator (RBG) must be
used for generating "k", and any potential bias in the "k" generation
algorithm must be mitigated (see the methods described in [FIPS186-4]
and [SP800-90A]).

7.9. Incremental/Partial Deployment Considerations

   What will migration from BGP to BGPsec look like? What are the
   benefits for the first adopters? Initially, small groups of
   contiguous ASes would be doing BGPsec. There would possibly be one
   or more such groups in different geographic regions of the global
   Internet. Only the routes originated within each group and
   propagated within its borders would get the benefits of cryptographic
   AS path protection. As BGPsec adoption grows, each group grows in
   size, and eventually they join together to form even larger
   BGPsec-capable groups of contiguous ASes. The benefit for early
   adopters starts with AS path security within the regions of
   contiguous ASes spanned by their respective groups. Over time, they
   would see those regions of contiguous ASes grow much larger.

   During partial deployment, if an AS in the path doesn't support
   BGPsec, then BGP goes back to traditional mode, i.e., BGPsec UPDATE
   messages are converted to unsigned UPDATE messages before forwarding
   to that AS (see Section 4.4). At this point, the assurance that the
   UPDATE message propagated via the sequence of ASes listed is lost.
   In other words, for the BGPsec routers residing in the ASes starting
   from the origin AS to the AS before the one not supporting BGPsec,
   the assurance can still be provided, but not beyond that (for the
   UPDATE messages in consideration).

8. Security Considerations

   For a discussion of the BGPsec threat model and related security
   considerations, please see RFC 7132 [RFC7132].

8.1. Security Guarantees

   When used in conjunction with origin validation (see RFC 6483
   [RFC6483] and RFC 6811 [RFC6811]), a BGPsec speaker who receives a
   valid BGPsec UPDATE message containing a route advertisement for a
   given prefix is provided with the following security guarantees:

   o The origin AS number corresponds to an AS that has been
      authorized, in the RPKI, by the IP address space holder to
      originate route advertisements for the given prefix.

   o For each AS in the path, a BGPsec speaker authorized by the holder
      of the AS number intentionally chose (in accordance with local
      policy) to propagate the route advertisement to the subsequent AS
      in the path.

That is, the recipient of a valid BGPsec UPDATE message is assured
that the UPDATE message propagated via the sequence of ASes listed in
the Secure_Path portion of the BGPsec_PATH attribute. (It should be
noted that BGPsec does not offer any guarantee that the data packets
would flow along the indicated path; it only guarantees that the BGP
UPDATE message conveying the path indeed propagated along the
indicated path.) Furthermore, the recipient is assured that this
path terminates in an AS that has been authorized by the IP address
space holder as a legitimate destination for traffic to the given
prefix.

Note that although BGPsec provides a mechanism for an AS to validate
that a received UPDATE message has certain security properties, the
use of such a mechanism to influence route selection is completely a
matter of local policy. Therefore, a BGPsec speaker can make no
assumptions about the validity of a route received from an external
(eBGP) BGPsec peer. That is, a compliant BGPsec peer may (depending
on the local policy of the peer) send UPDATE messages that fail the
validity test in Section 5. Thus, a BGPsec speaker MUST completely
validate all BGPsec UPDATE messages received from external peers.
(Validation of UPDATE messages received from internal peers is also a
matter of local policy; see Section 5.)

8.2. On the Removal of BGPsec Signatures

There may be cases where a BGPsec speaker deems 'Valid' (as per the
validation algorithm in Section 5.2) a BGPsec UPDATE message that
contains both a 'Valid' and a 'Not Valid' Signature_Block. That is,
the UPDATE message contains two sets of signatures corresponding to
two algorithm suites, and one set of signatures verifies correctly
and the other set of signatures fails to verify. In this case, the
protocol specifies that a BGPsec speaker choosing to propagate the
route advertisement in such an UPDATE message MUST add its signature
to each of the Signature_Blocks (see Section 4.2). Thus, the BGPsec
speaker creates a signature using both algorithm suites and creates a
new UPDATE message that contains both the 'Valid' and the 'Not Valid'
set of signatures (from its own vantage point).

To understand the reason for such a design decision, consider the
case where the BGPsec speaker receives an UPDATE message with both a
set of algorithm A signatures that are 'Valid' and a set of algorithm
B signatures that are 'Not Valid'. In such a case, it is possible
(perhaps even likely, depending on the state of the algorithm
transition) that some of the BGPsec speaker's peers (or other
entities further downstream in the BGP topology) do not support
algorithm A. Therefore, if the BGPsec speaker were to remove the
'Not Valid' set of signatures corresponding to algorithm B, such
entities would treat the message as though it were unsigned. By

including the 'Not Valid' set of signatures when propagating a route
advertisement, the BGPsec speaker ensures that downstream entities
have as much information as possible to make an informed opinion
about the validation status of a BGPsec UPDATE message.

Note also that during a period of partial BGPsec deployment, a
downstream entity might reasonably treat unsigned messages
differently from BGPsec UPDATE messages that contain a single set of
'Not Valid' signatures. That is, by removing the set of 'Not Valid'
signatures, the BGPsec speaker might actually cause a downstream
entity to 'upgrade' the status of a route advertisement from
'Not Valid' to unsigned. Finally, note that in the above scenario,
the BGPsec speaker might have deemed algorithm A signatures 'Valid'
only because of some issue with the RPKI state local to its AS (for
example, its AS might not yet have obtained a Certificate Revocation
List (CRL) indicating that a key used to verify an algorithm A
signature belongs to a newly revoked certificate). In such a case,
it is highly desirable for a downstream entity to treat the UPDATE
message as 'Not Valid' (due to the revocation) and not as 'unsigned'
(which would happen if the 'Not Valid' Signature_Blocks were removed
en route).

A similar argument applies to the case where a BGPsec speaker (for
some reason, such as a lack of viable alternatives) selects as its
best path (to a given prefix) a route obtained via a 'Not Valid'
BGPsec UPDATE message. In such a case, the BGPsec speaker should
propagate a signed BGPsec UPDATE message, adding its signature to the
'Not Valid' signatures that already exist. Again, this is to ensure
that downstream entities are able to make an informed decision and
not erroneously treat the route as unsigned. It should also be noted
that due to possible differences in RPKI data observed at different
vantage points in the network, a BGPsec UPDATE message deemed 'Not
Valid' at an upstream BGPsec speaker may be deemed 'Valid' by another
BGP speaker downstream.

Indeed, when a BGPsec speaker signs an outgoing UPDATE message, it is
not attesting to a belief that all signatures prior to its own
signature are valid. Instead, it is merely asserting that:

o The BGPsec speaker received the given route advertisement with the
   indicated prefix, AFI, SAFI, and Secure_Path, and

o The BGPsec speaker chose to propagate an advertisement for this
   route to the peer (implicitly) indicated by the 'Target AS
   Number'.

8.3. Mitigation of Denial-of-Service Attacks

   The BGPsec update validation procedure is a potential target for
   denial-of-service attacks against a BGPsec speaker. The mitigation
   of denial-of-service attacks that are specific to the BGPsec protocol
   is considered here.

   To mitigate the effectiveness of such denial-of-service attacks,
   BGPsec speakers should implement an update validation algorithm that
   performs expensive checks (e.g., signature verification) after
   performing checks that are less expensive (e.g., syntax checks). The
   validation algorithm specified in Section 5.2 was chosen so as to
   perform checks that are likely to be expensive after checks that are
   likely to be inexpensive. However, the relative cost of performing
   required validation steps may vary between implementations, and thus
   the algorithm specified in Section 5.2 may not provide the best
   denial-of-service protection for all implementations.

   Additionally, sending UPDATE messages with very long AS paths (and
   hence a large number of signatures) is a potential mechanism to
   conduct denial-of-service attacks. For this reason, it is important
   that an implementation of the validation algorithm stops attempting
   to verify signatures as soon as an invalid signature is found. (This
   ensures that long sequences of invalid signatures cannot be used for
   denial-of-service attacks.) Furthermore, implementations can
   mitigate such attacks by only performing validation on UPDATE
   messages that, if valid, would be selected as the best path. That
   is, if an UPDATE message contains a route that would lose out in best
   path selection for other reasons (e.g., a very long AS path), then it
   is not necessary to determine the BGPsec-validity status of the
   route.

8.4. Additional Security Considerations

   The mechanism of setting the pCount field to 0 is included in this
   specification to enable route servers in the control path to
   participate in BGPsec without increasing the length of the AS path.
   Two other scenarios where pCount=0 is utilized are in the contexts of
   an AS confederation (see Section 4.3) and of AS migration [RFC8206].
   In these two scenarios, pCount=0 is set and also accepted within the
   same AS (albeit the AS has two different identities). However,
   entities other than route servers, confederation ASes, or migrating
   ASes could conceivably use this mechanism (set the pCount to 0) to
   attract traffic (by reducing the length of the AS path)
   illegitimately. This risk is largely mitigated if every BGPsec
   speaker follows the operational guidance in Section 7.2 for
   configuration for setting pCount=0 and/or accepting pCount=0 from a
   peer. However, note that a recipient of a BGPsec UPDATE message

within which an upstream entity two or more hops away has set pCount
to 0 is unable to verify for themselves whether pCount was set to 0
legitimately.

There is a possibility of passing a BGPsec UPDATE message via
tunneling between colluding ASes. For example, let's say that AS-X
does not peer with AS-Y but colludes with AS-Y, and it signs and
sends a BGPsec UPDATE message to AS-Y by tunneling. AS-Y can then
further sign and propagate the BGPsec UPDATE message to its peers.
It is beyond the scope of the BGPsec protocol to detect this form of
malicious behavior. BGPsec is designed to protect messages sent
within BGP (i.e., within the control plane) -- not when the control
plane is bypassed.

A variant of the collusion by tunneling mentioned above can happen in
the context of AS confederations. When a BGPsec router (outside of a
confederation) is forwarding an UPDATE message to a Member-AS in the
confederation, it signs the UPDATE message to the public AS number of
the confederation and not to the member's AS number (see
Section 4.3). The Member-AS can tunnel the signed UPDATE message to
another Member-AS as received (i.e., without adding a signature).
The UPDATE message can then be propagated using BGPsec to other
confederation members or to BGPsec neighbors outside of the
confederation. This kind of operation is possible, but no grave
security or reachability compromise is feared for the following
reasons:

o The confederation members belong to one organization, and strong
   internal trust is expected.

o Recall that the signatures that are internal to the confederation
   MUST be removed prior to forwarding the UPDATE message to an
   outside BGPsec router (see Section 4.3).

BGPsec does not provide protection against attacks at the transport
layer. As with any BGP session, an adversary on the path between a
BGPsec speaker and its peer is able to perform attacks such as
modifying valid BGPsec UPDATE messages to cause them to fail
validation, injecting (unsigned) BGP UPDATE messages without
BGPsec_PATH attributes, injecting BGPsec UPDATE messages with
BGPsec_PATH attributes that fail validation, or causing the peer to
tear down the BGP session. The use of BGPsec does nothing to
increase the power of an on-path adversary -- in particular, even an
on-path adversary cannot cause a BGPsec speaker to believe that a
BGPsec-invalid route is valid. However, as with any BGP session,
BGPsec sessions SHOULD be protected by appropriate transport security
mechanisms (see the Security Considerations section in [RFC4271]).

There is a possibility of replay attacks, defined as follows. In the
context of BGPsec, a replay attack occurs when a malicious BGPsec
speaker in the AS path suppresses a prefix withdrawal (implicit or
explicit). Further, a replay attack is said to occur also when a
malicious BGPsec speaker replays a previously received BGPsec
announcement for a prefix that has since been withdrawn. The
mitigation strategy for replay attacks involves router certificate
rollover; please see [ROLLOVER] for details.

9. IANA Considerations

   IANA has registered a new BGP capability described in Section 2.1 in
   the "Capability Codes" registry's "IETF Review" range [RFC8126]. The
   description for the new capability is "BGPsec Capability". This
   document is the reference for the new capability.

   IANA has also registered a new path attribute described in Section 3
   in the "BGP Path Attributes" registry. The code for this new
   attribute is "BGPsec_PATH". This document is the reference for the
   new attribute.

   IANA has defined the "BGPsec Capability" registry in the "Resource
   Public Key Infrastructure (RPKI)" group. The registry is as shown in
   Figure 10, with values assigned from Section 2.1:

```
      +------+------------------------------------+------------+
      | Bits | Field                              | Reference  |
      +------+------------------------------------+------------+
      | 0-3  | Version                            | [RFC8205]  |
      |      | Value = 0x0                        |            |
      +------+------------------------------------+------------+
      | 4    | Direction                          | [RFC8205]  |
      |      |(Both possible values 0 and 1 are   |            |
      |      | fully specified by this RFC)       |            |
      +------+------------------------------------+------------+
      | 5-7  | Unassigned                         | [RFC8205]  |
      |      | Value = 000 (in binary)            |            |
      +------+------------------------------------+------------+
```

              Figure 10: IANA Registry for BGPsec Capability

   The Direction bit (fourth bit) has a value of either 0 or 1, and both
   values are fully specified by this document. Future Version values
   and future values of the Unassigned bits are assigned using the
   "Standards Action" registration procedures defined in RFC 8126
   [RFC8126].

IANA has defined the "BGPsec_PATH Flags" registry in the "Resource
Public Key Infrastructure (RPKI)" group. The registry is as shown in
Figure 11, with one value assigned from Section 3.1:

```
+------+------------------------------------------+-----------+
| Flag | Description                              | Reference |
+------+------------------------------------------+-----------+
| 0    | Confed_Segment                           | [RFC8205] |
|      | Bit value = 1 means Flag set             |           |
|      |             (indicates Confed_Segment)   |           |
|      | Bit value = 0 is default                 |           |
+------+------------------------------------------+-----------+
| 1-7  | Unassigned                               | [RFC8205] |
|      | Value: All 7 bits set to zero            |           |
+------+------------------------------------------+-----------+
```

Figure 11: IANA Registry for BGPsec_PATH Flags Field

Future values of the Unassigned bits are assigned using the
"Standards Action" registration procedures defined in RFC 8126
[RFC8126].

10. References

10.1. Normative References

[IANA-AF] IANA, "Address Family Numbers",
           <https://www.iana.org/assignments/address-family-numbers>.

[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>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A
           Border Gateway Protocol 4 (BGP-4)", RFC 4271,
           DOI 10.17487/RFC4271, January 2006,
           <https://www.rfc-editor.org/info/rfc4271>.

[RFC4724] Sangli, S., Chen, E., Fernando, R., Scudder, J., and Y.
           Rekhter, "Graceful Restart Mechanism for BGP", RFC 4724,
           DOI 10.17487/RFC4724, January 2007,
           <https://www.rfc-editor.org/info/rfc4724>.

[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>.

   [RFC5065] Traina, P., McPherson, D., and J. Scudder, "Autonomous
              System Confederations for BGP", RFC 5065,
              DOI 10.17487/RFC5065, August 2007,
              <https://www.rfc-editor.org/info/rfc5065>.

   [RFC5492] Scudder, J. and R. Chandra, "Capabilities Advertisement
              with BGP-4", RFC 5492, DOI 10.17487/RFC5492, February
              2009, <https://www.rfc-editor.org/info/rfc5492>.

   [RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route
              Origin Authorizations (ROAs)", RFC 6482,
              DOI 10.17487/RFC6482, February 2012,
              <https://www.rfc-editor.org/info/rfc6482>.

   [RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for
              X.509 PKIX Resource Certificates", RFC 6487,
              DOI 10.17487/RFC6487, February 2012,
              <https://www.rfc-editor.org/info/rfc6487>.

   [RFC6793] Vohra, Q. and E. Chen, "BGP Support for Four-Octet
              Autonomous System (AS) Number Space", RFC 6793,
              DOI 10.17487/RFC6793, December 2012,
              <https://www.rfc-editor.org/info/rfc6793>.

   [RFC7606] Chen, E., Ed., Scudder, J., Ed., Mohapatra, P., and K.
              Patel, "Revised Error Handling for BGP UPDATE Messages",
              RFC 7606, DOI 10.17487/RFC7606, August 2015,
              <https://www.rfc-editor.org/info/rfc7606>.

   [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for
              Writing an IANA Considerations Section in RFCs", BCP 26,
              RFC 8126, DOI 10.17487/RFC8126, June 2017,
              <https://www.rfc-editor.org/info/rfc8126>.

   [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
              2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
              May 2017, <https://www.rfc-editor.org/info/rfc8174>.

   [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>.

   [RFC8209] Reynolds, M., Turner, S., and S. Kent, "A Profile for
              BGPsec Router Certificates, Certificate Revocation Lists,
              and Certification Requests", RFC 8209,
              DOI 10.17487/RFC8209, September 2017,
              <https://www.rfc-editor.org/info/rfc8209>.

10.2. Informative References

   [Borchert] Borchert, O. and M. Baer, "Subject: Modification request:
              draft-ietf-sidr-bgpsec-protocol-14", message to the IETF
              SIDR WG Mailing List, 10 February 2016,
              <https://mailarchive.ietf.org/arch/msg/
              sidr/8B_e4CNxQCUKeZ_AUzsdnn2f5Mu>.

   [FIPS186-4]
              National Institute of Standards and Technology, "Digital
              Signature Standard (DSS)", NIST FIPS Publication
              186-4, DOI 10.6028/NIST.FIPS.186-4, July 2013,
              <http://nvlpubs.nist.gov/nistpubs/FIPS/
              NIST.FIPS.186-4.pdf>.

   [RFC6472] Kumari, W. and K. Sriram, "Recommendation for Not Using
              AS_SET and AS_CONFED_SET in BGP", BCP 172, RFC 6472,
              DOI 10.17487/RFC6472, December 2011,
              <https://www.rfc-editor.org/info/rfc6472>.

   [RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support
              Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480,
              February 2012, <https://www.rfc-editor.org/info/rfc6480>.

   [RFC6483] Huston, G. and G. Michaelson, "Validation of Route
              Origination Using the Resource Certificate Public Key
              Infrastructure (PKI) and Route Origin Authorizations
              (ROAs)", RFC 6483, DOI 10.17487/RFC6483, February 2012,
              <https://www.rfc-editor.org/info/rfc6483>.

   [RFC6810] Bush, R. and R. Austein, "The Resource Public Key
              Infrastructure (RPKI) to Router Protocol", RFC 6810,
              DOI 10.17487/RFC6810, January 2013,
              <https://www.rfc-editor.org/info/rfc6810>.

   [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>.

   [RFC7093] Turner, S., Kent, S., and J. Manger, "Additional Methods
              for Generating Key Identifiers Values", RFC 7093,
              DOI 10.17487/RFC7093, December 2013,
              <https://www.rfc-editor.org/info/rfc7093>.

   [RFC7115] Bush, R., "Origin Validation Operation Based on the
              Resource Public Key Infrastructure (RPKI)", BCP 185,
              RFC 7115, DOI 10.17487/RFC7115, January 2014,
              <https://www.rfc-editor.org/info/rfc7115>.

   [RFC7132] Kent, S. and A. Chi, "Threat Model for BGP Path Security",
              RFC 7132, DOI 10.17487/RFC7132, February 2014,
              <https://www.rfc-editor.org/info/rfc7132>.

   [RFC8181] Weiler, S., Sonalker, A., and R. Austein, "A Publication
              Protocol for the Resource Public Key Infrastructure
              (RPKI)", July 2017,
              <https://www.rfc-editor.org/info/rfc8181>.

   [RFC8182] Bruijnzeels, T., Muravskiy, O., Weber, B., and R. Austein,
              "The RPKI Repository Delta Protocol (RRDP)", RFC 8182,
              DOI 10.17487/RFC8182, July 2017,
              <https://www.rfc-editor.org/info/rfc8182>.

   [RFC8206] George, W. and S. Murphy, "BGPsec Considerations for
              Autonomous System (AS) Migration", RFC 8206,
              DOI 10.17487/RFC8206, September 2017,
              <https://www.rfc-editor.org/info/rfc8206>.

   [RFC8207] Bush, R., "BGPsec Operational Considerations", BCP 211,
              RFC 8207, DOI 10.17487/RFC8207, September 2017,
              <https://www.rfc-editor.org/info/rfc8207>.

   [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>.

   [ROLLOVER] Weis, B., Gagliano, R., and K. Patel, "BGPsec Router
              Certificate Rollover", Work in Progress,
              draft-ietf-sidrops-bgpsec-rollover-01, August 2017.

   [SLURM]   Mandelberg, D., Ma, D., and T. Bruijnzeels, "Simplified
              Local internet nUmber Resource Management with the RPKI",
              Work in Progress, draft-ietf-sidr-slurm-04, March 2017.

   [SP800-90A]
              National Institute of Standards and Technology,
              "Recommendation for Random Number Generation Using
              Deterministic Random Bit Generators", NIST SP 800-90A
              Rev 1, DOI 10.6028/NIST.SP.800-90Ar1, June 2015,
              <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/
              NIST.SP.800-90Ar1.pdf>.

Acknowledgements

Contributors

   The following people have made significant contributions to this
   document and should be considered co-authors:

   Rob Austein
   Dragon Research Labs
   Email: sra@hactrn.net

   Steven Bellovin
   Columbia University
   Email: smb@cs.columbia.edu

   Russ Housley
   Vigil Security
   Email: housley@vigilsec.com

   Stephen Kent
   BBN Technologies
   Email: kent@alum.mit.edu

   Warren Kumari
   Google
   Email: warren@kumari.net

   Doug Montgomery
   USA National Institute of Standards and Technology
   Email: dougm@nist.gov

   Chris Morrow
   Google, Inc.
   Email: morrowc@google.com

   Sandy Murphy
   SPARTA, Inc., a Parsons Company
   Email: sandy@tislabs.com

   Keyur Patel
   Arrcus
   Email: keyur@arrcus.com

   John Scudder
   Juniper Networks
   Email: jgs@juniper.net

   Samuel Weiler
   W3C/MIT
   Email: weiler@csail.mit.edu

Authors' Addresses

   Matthew Lepinski (editor)
   New College of Florida
   5800 Bay Shore Road
   Sarasota, FL 34243
   United States of America

   Email: mlepinski@ncf.edu

   Kotikalapudi Sriram (editor)
   USA National Institute of Standards and Technology
   100 Bureau Drive
   Gaithersburg, MD 20899
   United States of America

   Email: kotikalapudi.sriram@nist.gov