BGPsec Protocol Specification
draft-ietf-sidr-bgpsec-protocol-16

Abstract

   This document describes BGPsec, an extension to the Border Gateway
   Protocol (BGP) that provides security for the path of autonomous
   systems through which a BGP update message passes. BGPsec is
   implemented via a new optional non-transitive BGP path attribute that
   carries a digital signature produced by each autonomous system that
   propagates the update message.

Requirements Language

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and
   "OPTIONAL" are to be interpreted as described in RFC 2119 [1] only
   when they appear in all upper case. They may also appear in lower or
   mixed case as English words, without normative meaning.

Table of Contents

[1](#). **Introduction**

This document describes BGPsec, a mechanism for providing path
security for Border Gateway Protocol (BGP) [2] route advertisements.
That is, a BGP speaker who receives a valid BGPsec update has
cryptographic assurance that the advertised route has the following
property: Every 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 a new 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
[19][20] 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 [12] 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 [9] 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).

2. BGPsec Negotiation

This document defines a new BGP capability [6] 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).

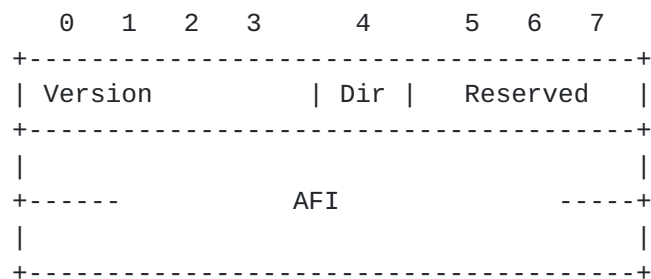## 2.1. The BGPsec Capability

This capability has capability code : TBD

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

The three octets of the capability value are specified as follows.

                    BGPsec Send Capability Value:

```
            0   1   2   3     4     5   6   7
          +---------------------------------------+
          | Version           | Dir |  Reserved   |
          +---------------------------------------+
          |                                       |
          +------          AFI          -----+
          |                                       |
          +---------------------------------------+
```

The first four 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 four bits set to zero). 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 three bits of the first octet are reserved for future
use. These bits are set to zero 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,
AFI values 1 and 2 respectively. 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
support the BGP multiprotocol extension [3]. 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 [3].

In a session where BGP 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 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, we say that the use of (the particular version of)
BGPsec has been negotiated (for a particular address family). BGP
update messages without the BGPsec_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 zero is the only version that has been
successfully negotiated. Any future document which 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 four-byte AS numbers. Therefore, any BGP speaker that announces
the BGPsec capability, MUST also announce the capability for four-
byte AS support [4]. If a BGP speaker sends the BGPsec capability but
not the four-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.

Note that BGPsec update messages can be quite large, therefore any
BGPsec speaker announcing the capability to receive BGPsec messages
SHOULD also announce support for the capability to receive BGP
extended messages [8].

3. The BGPsec_Path Attribute

The BGPsec_Path attribute is a new optional non-transitive BGP path
attribute.

This document registers a new attribute type code for this attribute
: TBD

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.
 We refer to those update messages that contain the BGPsec_Path
attribute 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 following
high-level diagram provides an overview of the structure of the
BGPsec_Path attribute:

```
        High-Level Diagram of the BGPsec_Path Attribute
   +------------------------------------------------------------+
   |      +-----------------+                                   |
   |      |   Secure Path   |                                   |
   |      +-----------------+                                   |
   |      |     AS X        |                                   |
   |      |     pCount X    |                                   |
   |      |     Flags X     |                                   |
   |      |     AS Y        |                                   |
   |      |     pCount Y    |                                   |
   |      |     Flags Y     |                                   |
   |      |       ...       |                                   |
   |      +-----------------+                                   |
   |                                                            |
   |      +-----------------+        +-----------------+        |
   |      | Sig Block 1     |        |  Sig Block 2    |        |
   |      +-----------------+        +-----------------+        |
   |      | Alg Suite 1     |        |  Alg Suite 2    |        |
   |      | SKI X1          |        |  SKI X1         |        |
   |      | Signature X1    |        |  Signature X1   |        |
   |      | SKI Y1          |        |  SKI Y1         |        |
   |      | Signature Y1    |        |  Signature Y1   |        |
   |      |      ...        |        |     ....        |        |
   |      +-----------------+        +-----------------+        |
   |                                                            |
   +------------------------------------------------------------+
```
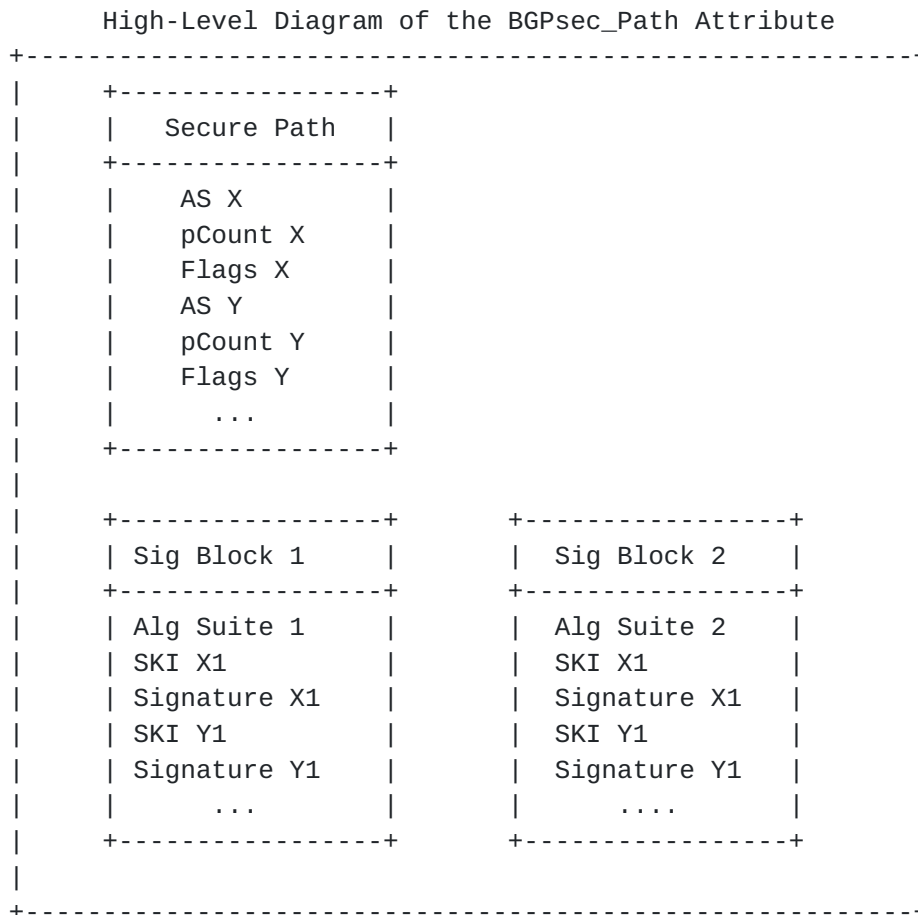
The following is the specification of the format for the BGPsec_Path
attribute.

                       BGPsec_Path Attribute

        +--+
        | Secure_Path   (variable) |
        +--+
        | Sequence of one or two Signature_Blocks (variable) |
        +--+

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
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

Here we provide a detailed description of the Secure_Path information
in the BGPsec_Path attribute.

                             Secure_Path

            +---+
            | Secure_Path Length   (2 octets) |
            +---+
            | One or More Secure_Path Segments (variable) |
            +---+

The Secure_Path Length contains the length (in octets) of the entire
Secure_Path (including the two octets used to express this length

field). As explained below, each Secure_Path segment is six octets
long. Note that this means the Secure_Path Length is two greater
than six times the number Secure_Path Segments (i.e., the number of
AS numbers in the path).

The Secure_Path contains one Secure_Path Segment for each (distinct)
Autonomous System in the path to the originating AS of the NLRI
specified in the update message.

```
               Secure_Path Segment

        +----------------------------+
        | pCount          (1 octet)  |
        +----------------------------+
        | Flags           (1 octet)  |
        +----------------------------+
        | AS Number      (4 octets)  |
        +----------------------------+
```

The AS Number 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
autonomous system 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. The pCount field is also
useful in managing route servers (see Section 4.2) and AS Number
migrations, see [18] for details.

The first bit of the Flags field is the Confed_Segment flag. The
Confed_Segment flag is set to one to indicate that the BGPsec speaker
that constructed this Secure_Path segment is sending the update
message to a peer AS within the same Autonomous System confederation
[5]. (That is, the Confed_Segment flag is set in a BGPsec update
message whenever, in a non-BGPsec update message, the BGP speaker's
AS would appear in a AS_PATH segment of type AS_CONFED_SEQUENCE.) In
all other cases the Confed_Segment flag is set to zero.

The remaining seven bits of the Flags MUST be set to zero by the
sender, and ignored by the receiver. Note, however, that the
signature is computed over all eight bits of the flags field.

3.2. Signature_Block

Here we provide a detailed description of the Signature_Blocks in the

BGPsec_Path attribute.

                          Signature_Block

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

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

   The Algorithm Suite Identifier is a one-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 identifiers for use in BGPsec is specified in the BGPsec
   algorithms document [10].

   A Signature_Block has exactly one Signature Segment 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 NLRI in the Update message.)

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

   The Subject Key Identifier contains the value in the Subject Key
   Identifier extension of the RPKI router certificate [9] that is used
   to verify the signature (see Section 5 for details on validity of
   BGPsec update messages).

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

   The Signature contains a digital signature that protects the NLRI 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 autonomous system confederation [5]. A BGPsec speaker that is not
   a member of such a confederation MUST set the Flags field of the
   Secure_Path Segment to zero 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 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 NLRI.
   This is because a BGPsec speaker receiving an update message with
   multiple NLRI would be unable to construct a valid BGPsec update
   message (i.e., valid path signatures) containing a subset of the NLRI
   in the received update. If a BGPsec speaker wishes to advertise
   routes to multiple NLRI, then it MUST generate a separate BGPsec
   update message for each NLRI. Additionally, a BGPsec update message
   MUST use the MP_REACH_NLRI [3] attribute to encode the NLRI.

   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 Resource PKI end-entity certificate whose AS number
resource extension includes the BGPsec speaker's AS number [9]. 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). Therefore, a BGPsec speaker who
only sends BGPsec update messages to peers within its own AS, it does
not need to possess any private signature keys.

The Resource PKI enables the legitimate holder of IP address
prefix(es) to issue a signed object, called a Route Origination
Authorization (ROA), that authorizes a given AS to originate routes
to a given set of prefixes (see [7]). It is expected that most
relying parties will utilize BGPsec in tandem with origin validation
(see [19] and [20]). Therefore, it is RECOMMENDED that a BGPsec
speaker only originate a BGPsec update 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
(without the BGPsec_Path attribute), containing the AS_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 NLRI in it. In that case, the prefix is originating from that AS
and hence the BGPsec speaker SHOULD sign and forward the update 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 7 for 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
7 for more discussion of the security semantics of BGPsec
signatures.)

If the BGPsec speaker is producing an update message which 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 [13]
recommends against the use of AS_SET and AS_CONFED_SET in the AS_PATH
of BGP updates.

The case where the BGPsec speaker sends a BGPsec update message to an
internal (iBGP) peer is quite simple. When originating a new route
advertisement and sending it to an internal peer, the BGPsec speaker
omits the BGPsec_Path attribute. When propagating a received route
advertisement to an internal peer, the BGPsec speaker typically
populates the BGPsec_Path attribute by copying the BGPsec_Path
attribute from the received update message. That is, the BGPsec_Path
attribute is copied verbatim. However, in the case that the BGPsec
speaker is performing an AS Migration, the BGPsec speaker may add an
additional signature on ingress before copying the BGPsec_Path
attribute (see [18] for more details). Note that when a BGPsec
speaker chooses to forward a BGPsec update message to an iBGP peer,
the BGPsec attribute SHOULD NOT be removed, unless the peer doesn't
support BGPsec. In particular, the BGPsec 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 7 for the security ramifications of removing
BGPsec signatures.)

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 to its (internal or external) peers by creating a new
BGPsec advertisement for the same prefix. Similarly, when sending a
new route advertisement to an external, BGPsec-speaking peer, the
BGPsec speaker may send a BGPsec Update message by generating a new
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 AS number resource extension field of the Resource PKI router
certificate(s) that will be used to verify the digital signature(s)
constructed by this BGPsec speaker [9].

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 one 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
pCount of k -- and a single corresponding Signature Segment.

A route server that participates in the BGP control path, 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
effective length of the AS path. (Note that BGPsec speakers compute
the effective 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 [18] for a
discussion of setting pCount to 0 to facilitate AS Number Migration.)
BGPsec speakers SHOULD drop incoming update messages with pCount set
to zero in cases where the BGPsec speaker does not expect its peer to
set pCount to zero. (That is, pCount is only to be set to zero in
cases such as route servers or AS Number Migration where the BGPsec
speaker's peer expects pCount to be set to zero.)

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 algorithms
suites, then the new update message generated by the BGPsec speaker
SHOULD 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 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
which is not deemed 'Valid' (e.g., contains signatures that the
BGPsec does not successfully verify). Nonetheless, such
Signature_Blocks MUST NOT be removed. (See Section 7 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 adds 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 appear 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 [9].
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 NLRI 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 be
the Secure_Path segment that is being added by the current AS.

o In order to construct the digital signature for Signature Segment
   N (the signature segment being produced by the current AS), first
   construct the following sequence of octets to be hashed.

```
         Sequence of Octets to be Hashed
     +-----------------------------------+
     | Target AS Number                  |
     +-----------------------------------+ -\
     | Signature Segment   : N-1         |   \
     +-----------------------------------+    |
     | Secure_Path Segment : N           |    |
     +-----------------------------------+    \
           ...                                 > For N Hops
     +-----------------------------------+    /
     | Signature Segment   : 1           |    |
     +-----------------------------------+    |
     | Secure_Path Segment : 2           |   /
     +-----------------------------------+ -/
     | Secure_Path Segment : 1           |
     +-----------------------------------+
     | Algorithm Suite Identifier        |
     +-----------------------------------+
     | AFI                               |
     +-----------------------------------+
     | SAFI                              |
     +-----------------------------------+
     | NLRI                              |
     +-----------------------------------+
```

In this sequence, 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 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 Network Layer Reachability Information (NLRI) fields
are obtained from the MP_REACH_NLRI attribute. Additionally, in
the Prefix field of the NLRI (from MP_REACH_NLRI), all of the
trailing bits MUST be set to zero when constructing this
sequence.

o Apply to this octet sequence 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 with this digital
   signature.

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

4.3. Processing Instructions for Confederation Members

Members of autonomous system confederations [5] MUST additionally
follow the instructions in this section for processing BGPsec update
messages.

When a confederation member sends a BGPsec update message to a peer
that is a member of the same confederation, the confederation member
puts its (private) Member-AS Number (as opposed to the public AS
Confederation Identifier) in the AS Number field of the Secure_Path
Segment that it adds to the BGPsec update message. Additionally, in
this case, the confederation member that generates the Secure_Path
Segment sets the Confed_Segment flag to one. This means that in a
BGPsec update message, an AS number appears in a Secure_Path Segment
with the Confed_Segment flag set whenever, in a non-BGPsec update
message, the AS number would appear in a segment of type
AS_CONFED_SEQUENCE.

Within a confederation, the verification of BGPsec signatures added
by other members of the confederation is optional. If a
confederation chooses not to have its members verify signatures added
by other confederation members, then when sending a BGPsec update
message to a peer that is a member of the same confederation, the
confederation members MAY set the Signature field within the
Signature Segment that it generates to be zero (in lieu of
calculating the correct digital signature as described in Section
4.2). Note that if a confederation chooses not to verify digital
signatures within the confederation, then BGPsec is able to provide
no assurances about the integrity of the (private) Member-AS Numbers
placed in Secure_Path segments where the Confed_Segment flag is set
to one.

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 one. Stop this process once a
   Secure_Path segment is reached which has its Confed_Segment flag
   set to zero. 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 that the AS field are populated as per
   Section 4.2.

When validating a received BGPsec update message, confederation
members need to make the following adjustment to the algorithm
presented in Section 5.2. When a confederation member processes
(validates) a Signature Segment and its corresponding Secure_Path
Segment, the confederation member must note the following. For a
signature produced by a peer BGPsec speaker outside of a
confederation, the Target AS will always be the AS Confederation
Identifier (the public AS number of the confederation) as opposed to
the Member-AS Number.

To handle this case, when a BGPsec speaker (that is a confederation
member) processes a current Secure_Path Segment that has the
Confed_Segment flag set to zero, if the next most recently added
Secure_Path segment has the Confed_Segment flag set to one then, when
computing the digest for the current Secure_Path segment, the BGPsec
speaker takes the Target AS Number to be the AS Confederation
Identifier of the validating BGPsec speaker's own confederation.
(Note that the algorithm in Section 5.2 processes Secure_Path
Segments in order from most recently added to least recently added,
therefore this special case will apply to the first Secure_Path
segment that the algorithm encounters that has the Confed_Segment
flag set to zero.)

Finally, as discussed above, an AS confederation may optionally
decide that its members will not verify digital signatures added by
members. In such a federation, when a confederation member runs the

   algorithm in Section 5.2, the confederation member, during processing
   of a Signature Segment, first checks whether the Confed_Sequence flag
   in the corresponding Secure_Path segment is set to one. If the
   Confed_Sequence flag is set to one in the corresponding Secure_Path
   segment, the confederation member does not perform any further checks
   on the Signature Segment and immediately moves on to the next
   Signature Segment (and checks its corresponding 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 containing a Confed_Sequence flag set
   to one. (As discussed in Section 5.2, any error in the BGPsec_Path
   attribute MUST be handled using the "treat-as-withdraw", approach as
   defined in RFC 7606 [11].)

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 to a peer, a BGPsec speaker MUST perform
   the basic integrity checks listed in Section 5.2 to ensure that the
   received BGPsec update is properly formed.

   The AS_PATH attribute can be constructed from the BGPsec_Path
   attribute as follows. Starting with an empty 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 Confed_Segment flag in the Secure_Path segment is set to
       one, then look at the most-recently added segment in the AS_PATH.

       * In the case where the AS_PATH is empty or in the case where
          the most-recently added segment is of type AS_SEQUENCE then
          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.)

2. If the Confed_Segment flag in the Secure_Path segment is set to
    zero, then look at the most-recently added segment in the
    AS_PATH.

     * In the case where the AS_PATH is empty, and the pCount field
       in the Secure_Path segment is greater than zero, 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.)

5. Processing a Received BGPsec Update

   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 [19] and [20]) 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 RPKI state has changed
(e.g., from an RPKI validating cache via the RPKI-to-Router protocol
[15]), the BGPsec speaker MUST re-run validation on all affected
update messages stored in its ADJ-RIB-IN. That is, when a given RPKI
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 re-assessed to determine if they are
still valid. If this reassessment determines that the validity state
of an update has changed then, depending on local policy, it may be
necessary to re-run best path selection.

BGPsec update messages do not contain an AS_PATH attribute.
Therefore, a BGPsec speaker MUST utilize the AS path information in
the BGPsec_Path attribute 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 use of 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.

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 an implementation receives a BGPsec update from a peer
and later receives a second BGPsec update message from the same peer,
the implementation SHOULD treat the second message as a duplicate
update message if it differs from the first update message only in
the Signature fields (within the BGPsec_Path attribute). That is, if
all the fields in the second update are identical to the fields in
the first update message, except for the Signature fields, then the
second update message should be treated as a duplicate of the first
update message. Note that if other fields (e.g., the Subject Key

   Identifier field) within a Signature segment differ between two
   update messages then the two updates are not duplicates.

   With regards to the processing of duplicate update messages, if the
   first update message is valid, then an implementation SHOULD NOT run
   the validation procedure on the second, duplicate update message
   (even if the bits of the signature field are different). If the
   first update message is not valid, then an implementation SHOULD run
   the validation procedure on the second duplicate update message (as
   the signatures in the second update may be valid even though the
   first contained a signature that was invalid).

5.1. Overview of BGPsec Validation

   Validation of a BGPsec update messages makes use of data from RPKI
   certificates. In particular, it is necessary that the recipient have
   access to the following data obtained from valid RPKI 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
   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 using the router key PDU [16] for the RTR protocol
   [15].)

   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, we expect some operators will choose to
   treat BGPsec validation status differently for update messages
   received over different BGP sessions.)

   It is expected that BGP peers will generally prefer routes received
   via 'Valid' BGPsec update messages over both routes received via 'Not
   Valid' BGPsec update messages and routes received via update messages
   that do not contain the BGPsec_Path attribute. However, BGPsec
   specifies no changes to the BGP decision process. (See [17] for
   related operational considerations.)

   BGPsec validation needs only be performed at the eBGP edge. The
   validation status of a BGP signed/unsigned update 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. Specifically, the
   recipient performs the following checks:

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

   2. 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.)

   3. Check that the update message does not contain an AS_PATH
       attribute.

   4. If the update message was received from a 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_Sequence flag set to one.

   5. If the update message was received from a peer that is not
       expected to set pCount equal to zero (see Section 4.2) then check
       to ensure that the pCount field in the most-recently added
       Secure_Path segment is not equal to zero.

   If any of these checks fail, it is an error in the BGPsec_Path
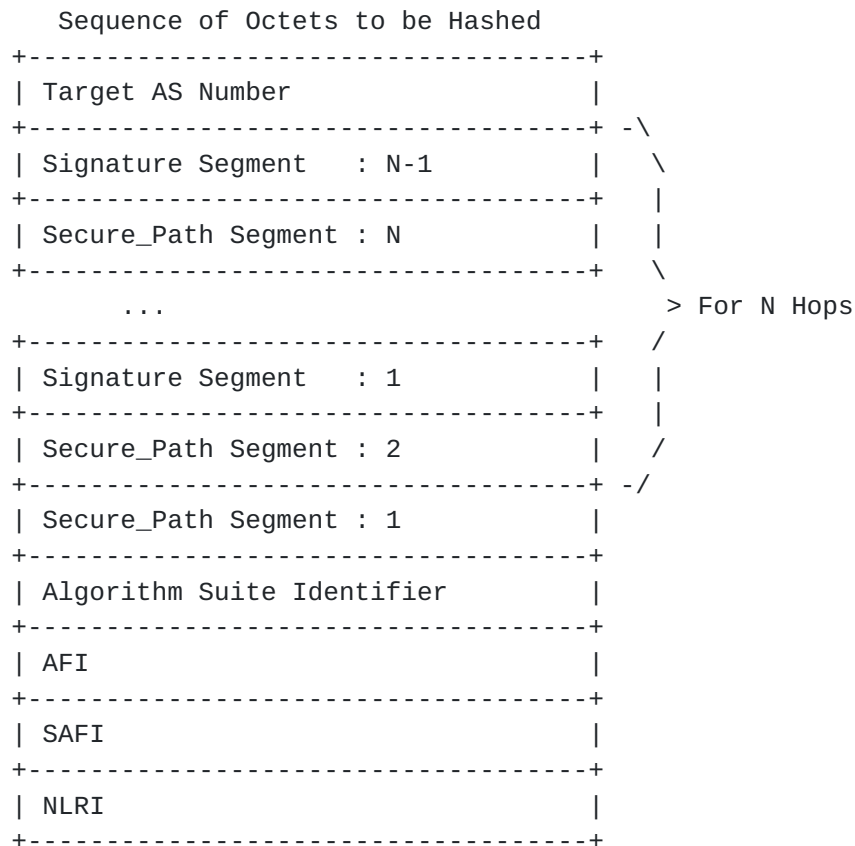   attribute. Any of these errors in the BGPsec_Path attribute are

handled as per RFC 7606 [11]. BGPsec speakers MUST handle these
errors using the "treat-as-withdraw" approach as defined in RFC 7606
[11].

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 validation. If there is no Signature_Block
corresponding to an algorithm suite that the BGPsec speaker supports,
then the BGPsec speaker MUST treat the update message in the same
manner that the BGPsec speaker would treat an (unsigned) update
message that arrived without a BGPsec_Path attribute.

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 0): 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 Signature
   Segment N be the Signature Segment that is currently being
   verified and let Secure_Path Segment N be the corresponding
   Secure_Path Segment.

o (Step I): 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, SKI,
   Public Key) 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 II): 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 following sequence of octets to be hashed.

```
        Sequence of Octets to be Hashed
   +-----------------------------------+
   | Target AS Number                  |
   +-----------------------------------+ -\
   | Signature Segment   : N-1         |   \
   +-----------------------------------+   |
   | Secure_Path Segment : N           |   |
   +-----------------------------------+   \
        ...                                  > For N Hops
   +-----------------------------------+   /
   | Signature Segment   : 1           |   |
   +-----------------------------------+   |
   | Secure_Path Segment : 2           |   /
   +-----------------------------------+ -/
   | Secure_Path Segment : 1           |
   +-----------------------------------+
   | Algorithm Suite Identifier        |
   +-----------------------------------+
   | AFI                               |
   +-----------------------------------+
   | SAFI                              |
   +-----------------------------------+
   | NLRI                              |
   +-----------------------------------+
```

   For the first segment to be processed (the most recently added
   segment), the 'Target AS Number' is 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 was received.

   For each other Signature Segment, the 'Target AS Number' is 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.)

   Additionally, the Secure_Path and Signature Segment are obtained
   from the BGPsec_Path attribute. The Address Family Identifier
   (AFI), Subsequent Address Family Identifier (SAFI), and Network
   Layer Reachability Information (NLRI) fields are obtained from the
   MP_REACH_NLRI attribute. Additionally, in the Prefix field of the
   NLRI (from MP_REACH_NLRI), all of the trailing bits MUST be set to
   zero when constructing this sequence.

   o (Step III): 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 II above; and
the public key obtained from the valid RPKI data in Step I 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 to be '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 of 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 he 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, a mandatory algorithm suites document exists which
specifies a mandatory-to-use 'current' algorithm suite for use by all
BGPsec speakers [10].

We anticipate that, in the future, the mandatory algorithm suites
document will be updated to specify a transition from the 'current'
algorithm suite to a 'new' algorithm suite. During the period of
transition (likely a small number of years), 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, use of
the old 'current' algorithm will be deprecated, use of the 'new'
algorithm will be mandatory, and a subsequent 'even newer' algorithm
suite may be specified as recommend 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. 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, which is designed to accommodate the
desired changes to BGPsec. In such a case, the mandatory algorithm
suites document would be updated to specify algorithm suites
appropriate for the new version of BGPsec.

At this point a transition would begin which 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. Security Considerations

For a discussion of the BGPsec threat model and related security
considerations, please see [14].

7.1 Security Guarantees

When used in conjunction with Origin Validation (see [19] and [20]),
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 autonomous system 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 propagated via the sequences 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
conveying the path indeed propagated along the indicated path.)
Furthermore, the recipient is assured that this path terminates in an
autonomous system 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
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 a matter of local
policy, see Section 5).

7.2 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 SHOULD add its
signature to each of the Signature_Blocks. 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 which are 'Valid' and a set of algorithm B
signatures which 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.

Note also that during a period of partial BGPsec deployment, a
'downstream' entity might reasonably treat unsigned messages
differently from BGPsec updates 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 RPKI state local to his AS (for example,
his AS might not yet have obtained a 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 as 'Not Valid' (due to the
revocation) and not as 'unsigned' (which would happen if the 'Not
Valid' Signature_Blocks were removed).

A similar argument applies to the case where a BGPsec speaker (for
some reason such as lack of viable alternatives) selects as his 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 his 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 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 are valid.
 Instead it is merely asserting that:

o The BGPsec speaker received the given route advertisement with the
   indicated NLRI 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'

7.3 Mitigation of Denial of Service Attacks

The BGPsec update validation procedure is a potential target for
denial of service attacks against a BGPsec speaker. Here we consider
the mitigation only of denial of service attacks that are specific to
BGPsec.

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 less expensive checks (e.g., syntax checks). The
validation algorithm specified in Section 5.2 was chosen so as to
perform checks which 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.

7.4 Additional Security Considerations

The mechanism of setting the pCount field to zero is included in this

specification to enable route servers in the control path to
participate in BGPsec without increasing the effective length of the
AS-PATH. However, entities other than route servers could
conceivably use this mechanism (set the pCount to zero) to attract
traffic (by reducing the effective length of the AS-PATH)
illegitimately. This risk is largely mitigated if every BGPsec
speaker drops incoming update messages that set pCount to zero but
come from a peer that is not a route server. However, note that a
recipient of a BGPsec update message within which an upstream entity
two or more hops away has set pCount to zero is unable to verify for
themselves whether pCount was set to zero legitimately.

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 updates to cause them to fail validation,
injecting (unsigned) BGP update messages without
BGPsec_Path_Signature attributes, injecting BGPsec update messages
with BGPsec_Path_Signature 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 a BGPsec-invalid route is valid. However, as with any BGP
session, BGPsec sessions SHOULD be protected by appropriate transport
security mechanisms.

8. IANA Considerations

This document registers a new capability in the registry of BGP
Capabilities. The description for the new capability is "BGPsec
Capability". The reference for the new capability is this document
(i.e., the RFC that replaces draft-ietf-sidr-bgpsec-protocol), see
Section 2.1.

This document registers a new path attribute in the registry of BGP
Path Attributes. The code for this new attribute is "BGPsec_PATH".
The reference for the new capability is this document (i.e., the RFC
that replaces draft-ietf-sidr-bgpsec-protocol), see Section 3.

This document does not create any new IANA registries.

9. Contributors

9.1. Authors

Rob Austein
Dragon Research Labs
sra@hactrn.net

Steven Bellovin
Columbia University
smb@cs.columbia.edu

Randy Bush
Internet Initiative Japan
randy@psg.com

Russ Housley
Vigil Security
housley@vigilsec.com

Matt Lepinski
New College of Florida
mlepinski@ncf.edu

Stephen Kent
BBN Technologies
kent@bbn.com

Warren Kumari
Google
warren@kumari.net

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

Kotikalapudi Sriram
USA National Institute of Standards and Technology
kotikalapudi.sriram@nist.gov

Samuel Weiler
Parsons
weiler+ietf@watson.org

[9.2](). Acknowledgements

[10](). Normative References

[1]   Bradner, S., "Key words for use in RFCs to Indicate Requirement
       Levels", BCP 14, RFC 2119, March 1997.

[2]   Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border
       Gateway Protocol 4", RFC 4271, January 2006.

[3]   Bates, T., Chandra, R., Katz, D., and Y. Rekhter,
       "Multiprotocol Extensions for BGP-4", RFC 4760, January 2007.

[4]   Vohra, Q. and E. Chen, "BGP Support for Four-Octet AS Number
       Space", RFC 6793, December 2012.

[5]   Traina, P., McPherson, D., and J. Scudder, "Autonomous System
       Confederations for BGP", RFC 5065, August 2007.

[6]   Scudder, J. and R. Chandra, "Capabilities Advertisement with
       BGP-4", RFC 5492, February 2009.

[7]   Lepinski, M., Kent, S., and D. Kong, "A Profile for Route
       Origin Authorizations (ROAs)", RFC 6482, February 2012.

[8]   Patel, K., Ward, D., and R. Bush, "Extended Message support for
       BGP", draft-ietf-idr-bgp-extended-messages (work in progress),
       May 2016.

[9] Reynolds, M., Turner, S., and S. Kent, "A Profile for BGPsec
       Router Certificates, Certificate Revocation Lists, and
       Certification Requests", draft-ietf-sidr-bgpsec-pki-profiles
       (work in progress), June 2016.

[10]   Turner, S., "BGP Algorithms, Key Formats, & Signature Formats",
       draft-ietf-sidr-bgpsec-algs (work in progress), April 2016.

[11]   Chen, E., Scudder, J., Mohapatra, P., and K. Patel, "Revised
       Error Handling for BGP UPDATE Messages", RFC 7606, August 2015.

11. Informative References

[12]   Lepinski, M. and S. Kent, "An Infrastructure to Support Secure
       Internet Routing", RFC 6480, February 2012.

[13]   Kumari, W. and K. Sriram, "Recommendation for Not Using AS_SET
       and AS_CONFED_SET in BGP", RFC 6472, December 2011.

[14]   Kent, S. and A. Chi, "Threat Model for BGP Path Security", RFC
       7132, February 2014.

[15]   Bush, R. and R. Austein, "The Resource Public Key
       Infrastructure (RPKI) to Router Protocol", draft-ietf-sidr-

rpki-rtr-rfc6810-bis (work in progress), March 2016.

[16]   Bush, R., Turner, S., and K. Patel, "Router Keying for BGPsec",
       draft-ietf-sidr-rtr-keying (work in progress), June 2016.

[17]   Bush, R., "BGPsec Operational Considerations", draft-ietf-sidr-
       bgpsec-ops (work in progress), June 2016.

[18]   George, W. and S. Murphy, "BGPsec Considerations for AS
       Migration", draft-ietf-sidr-as-migration (work in progress),
       April 2016.

[19] Huston, G. and G. Michaelson, "Validation of Route Origination
     Using the Resource Certificate Public Key Infrastructure (PKI)
     and Route Origin Authorizations (ROAs)", RFC 6483, February
     2013.

[20] Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R. Austein,
     "BGP Prefix Origin Validation", RFC 6811, January 2013.

Author's Address

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

    Email: mlepinski@ncf.edu

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

    Email: kotikalapudi.sriram@nist.gov