                      BGPSEC Protocol Specification
                    draft-ietf-sidr-bgpsec-protocol-09

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

Status of this Memo

Copyright Notice

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 two properties:

1. The route was originated by an AS explicitly authorized by the holder of the IP address prefix to originate route advertisements for that prefix.

2. 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 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 [7] 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 [10] 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.

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).
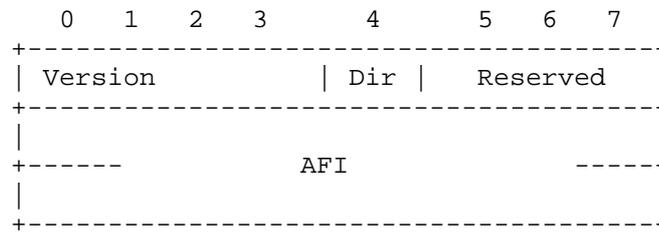
2.1. The BGPSEC Capability

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 SHOULD NOT advertise the capability of BGPSEC support for a particular AFI unless it has also advertised the multiprotocol extension capability for the same AFI combination [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 [9].

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   |           |
    |    |      ...       |        |      ....      |           |
    |    +----------------+        +----------------+           |
    |                                                           |
    +----------------------------------------------------------+
```
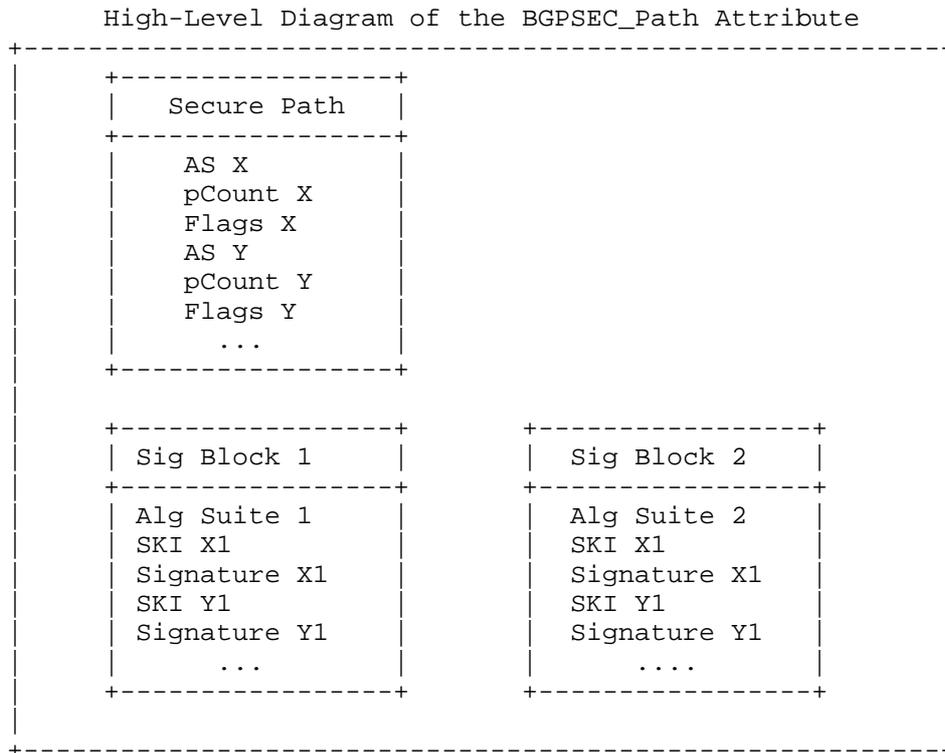
The following is the specification of the format for the BGPSEC_Path
attribute.

                        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. A BGPSEC update message
containing the BGPSEC_Path attribute MUST NOT contain the AS_PATH
attribute. 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 one 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

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

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 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 [11].

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 [10] 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. Generating a BGPSEC Update

   Sections 4.1 and 4.2 cover two cases in which a BGPSEC speaker may
   generate an update message containing the BGPSEC_Path attribute. The
   first case is that in which the BGPSEC speaker originates a new route
   advertisement (Section 4.1). That is, the BGPSEC speaker is
   constructing an update message in which the only AS to appear in the
   BGPSEC_Path is the speaker's own AS. The second case is that in
   which the BGPSEC speaker receives a route advertisement from a peer
   and then decides to propagate the route advertisement to an external
   (eBGP) peer (Section 4.2). That is, the BGPSEC speaker has received
   a BGPSEC update message and is constructing a new update message for
   the same NLRI in which the BGPSEC_Path attribute will contain AS
   number(s) other than the speaker's own AS.

   The remaining case is where the BGPSEC speaker sends the update
   message to an internal (iBGP) peer. When originating a new route
   advertisement and sending it to an internal peer, the BGPSEC speaker
   creates a new BGPSEC_Path attribute with zero Secure_Path segments
   and zero Signature Segments. When propagating a received route
   advertisement to an internal peer, the BGPSEC speaker 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. Note that in the case that a BGPSEC speaker chooses
   to forward to an iBGP peer a BGPSEC update message that has not been
   successfully validated (see Section 5), the BGPSEC_Path attribute
   SHOULD NOT be removed. (See Section 7 for the security ramifications
   of removing BGPSEC signatures.)

   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 which 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.

   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 [10]. 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.

4.1. Originating a New BGPSEC Update

In an update message that originates a new route advertisement (i.e.,
an update whose path will contain only a single AS number), when
sending the route advertisement to an external, BGPSEC-speaking peer,
the BGPSEC speaker creates a new BGPSEC_Path attribute as follows.

First, the BGPSEC speaker constructs the Secure_Path with a single
Secure_Path Segment. The AS in this path is the BGPSEC speaker's own
AS number. In particular, this AS number MUST match an AS number in
the AS number resource extension field of the Resource PKI router
certificate(s) [10] that will be used to verify the digital
signature(s) constructed by this BGPSEC speaker.

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.

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 [8]). Note that validation of a
BGPSEC update message will fail (i.e., the validation algorithm,
specified in Section 5.2, returns 'Not Valid') unless there exists a
valid ROA authorizing the first AS in the Secure_Path portion of the
BGPSEC_Path attribute to originate routes to the prefix being
advertised. Therefore, a BGPSEC speaker SHOULD NOT originate a
BGPSEC update advertising a route for a given prefix unless there
exists a valid ROA authorizing the BGPSEC speaker's AS to originate
routes to this prefix.

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). Setting the pCount field to a value

greater than one permits this repetition without requiring a separate
digital signature for each repetition.

If the BGPSEC speaker is not a member of an autonomous system
confederation [5], then the Flags field of the Secure_Path Segment
MUST be set to zero. (Members of a confederation should follow the
special processing instructions for confederation members in Section
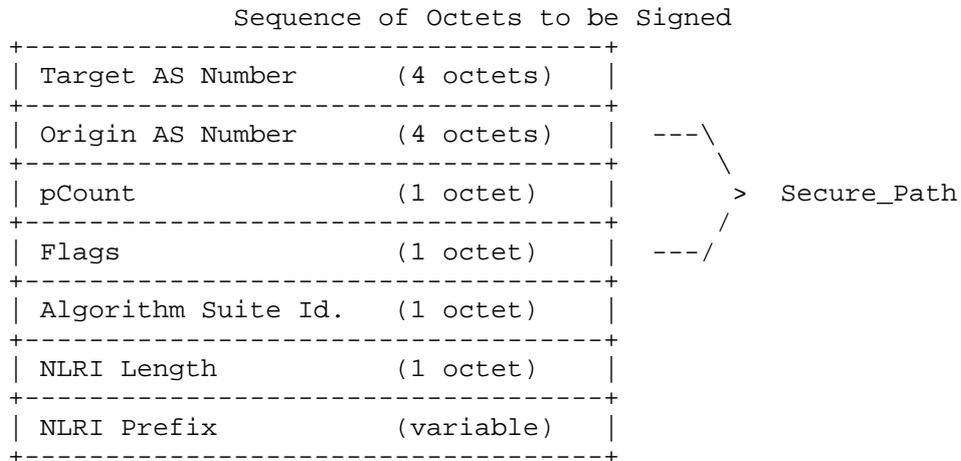4.4.)

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).

When originating a new route advertisement, each Signature_Block MUST
consist of a single Signature Segment. The following describes how
the BGPSEC speaker populates the fields of the Signature_Block.

The Subject Key Identifier field (see Section 3) is populated with
the identifier contained in the Subject Key Identifier extension of
the RPKI router certificate corresponding to the BGPSEC speaker[10].
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 contains a digital signature that binds the NLRI
and BGPSEC_Path attribute to the RPKI router corresponding to the
BGPSEC speaker. The digital signature is computed as follows:

o Construct a sequence of octets by concatenating the Target AS
   Number, the Secure_Path (Origin AS, pCount, and Flags), Algorithm
   Suite Identifier, and NLRI. 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.)

                    Sequence of Octets to be Signed
```
        +----------------------------------+
        | Target AS Number     (4 octets)  |
        +----------------------------------+
        | Origin AS Number     (4 octets)  |  ---\
        +----------------------------------+      \
        | pCount               (1 octet)   |       >  Secure_Path
        +----------------------------------+      /
        | Flags                (1 octet)   |  ---/
        +----------------------------------+
        | Algorithm Suite Id.  (1 octet)   |
        +----------------------------------+
        | NLRI Length          (1 octet)   |
        +----------------------------------+
        | NLRI Prefix          (variable)  |
        +----------------------------------+
```

   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 Signature field.

4.2. Propagating a Route Advertisement

   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.

   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 peers, as specified in Section 4.1.)

   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.

To generate the BGPSEC_Path attribute on the outgoing update message,
the BGPSEC speaker first prepends a new Secure_Path Segment (places
in first position) to the 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[10].

The pCount is typically set to the value 1. A BGPSEC speaker may set
the pCount field to a value greater than 1. (See Section 4.1 for a
discussion of setting pCount to a value greater than 1.) 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. Note that the option of setting
pCount to 0 is intended only for use by route servers that desire not
to increase the effective AS-PATH length of routes they advertise.
The pCount field SHOULD NOT be set to 0 in other circumstances.
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 (i.e., cases where the peer is not acting as a
route server).

If the BGPSEC speaker is not a member of an autonomous system
confederation [5], then the Confed_Segment bit of the Flags field of
the Secure_Path Segment MUST be set to zero. (Members of a
confederation should follow the special processing instructions for
confederation members in Section 4.3.)

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).

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
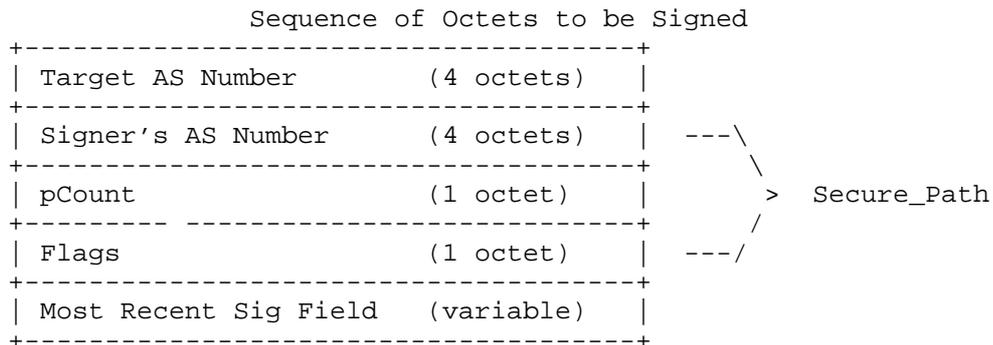discussion of the security ramifications of this design choice.)

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 corresponding to the BGPSEC speaker [10]. 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 Construct a sequence of octets by concatenating the Target AS
   number, the Secure_Path segment that is being added by the BGPSEC
   speaker constructing the signature, and the signature field of the
   most recent Signature Segment (the one corresponding to AS from
   whom the BGPSEC speaker's AS received the announcement). 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 BGPSEC
   update message is sent.

```
               Sequence of Octets to be Signed
       +------------------------------------+
       | Target AS Number      (4 octets)   |
       +------------------------------------+
       | Signer's AS Number    (4 octets)   |  ---\
       +------------------------------------+      \
       | pCount                (1 octet)    |       >  Secure_Path
       +--------- --------------------------+      /
       | Flags                 (1 octet)    |  ---/
       +------------------------------------+
       | Most Recent Sig Field  (variable)  |
       +------------------------------------+
```

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 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. Furthermore, when
   a confederation member sends a BGPSEC update message to a peer that
   is a member of the same confederation, the BGPSEC speaker 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 in a non-BGPSEC update
   message.

   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 Sections
   4.1 and 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 segments,
      remove all of the consecutive Secure_Path segments that have the
      Confed_Segment flag set to one. Stop this process once a
      Scure_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
   Sections 4.1 and 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 that for a signature
produced by a 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 WXYZ [12].)

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.

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. 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.

   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 RTR protocol), 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 and signed Route Origination Authorizations (ROA). In
   particular, to validate update messages containing the BGPSEC_Path
   attribute, it is necessary that the recipient have access to the
   following data obtained from valid RPKI certificates and ROAs:

   o For each valid RPKI router certificate, the AS Number, Public Key
      and Subject Key Identifier are required,

   o For each valid ROA, the AS Number and the list of IP address
      prefixes.

   Note that the BGPSEC speaker could perform the validation of RPKI
   certificates and ROAs 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. However, BGP route
   selection, and thus the handling of the two validation states is a
   matter of local policy, and is handled using local policy mechanisms.

   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 WXYZ [12]. BGPSEC speakers MUST handle these
   errors using the "treat-as-withdraw" approach as defined in RFC WXYZ
   [12].

   Next, the BGPSEC speaker verifies that the origin AS is authorized to
   advertise the prefix in question. To do this, consult the valid ROA
   data to obtain a list of AS numbers that are associated with the
   given IP address prefix in the update message. Then locate the last
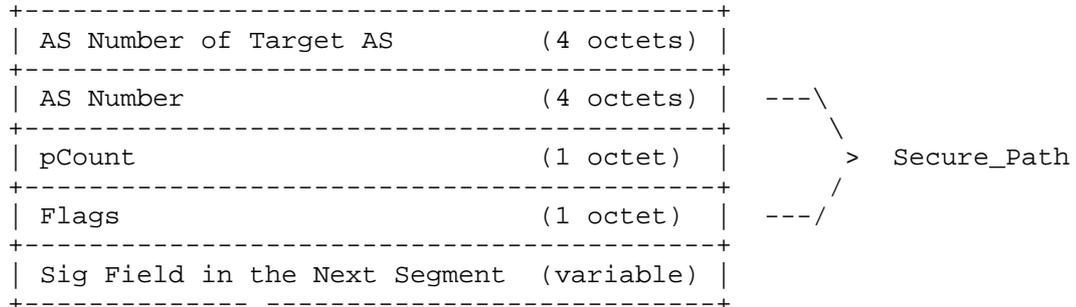   (least recently added) AS number in the Secure_Path portion of the

BGPSEC_Path attribute. If the origin AS in the Secure_Path is not in
the set of AS numbers associated with the given prefix, then the
BGPSEC update message is 'Not Valid' and the validation algorithm
terminates.

Finally, 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 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. If the segment is not the (least
   recently added) segment corresponding to the origin AS, then the
   digest function should be computed on the following sequence of
   octets:

Sequence of Octets to be Hashed

```
    +------------------------------------------+
    | AS Number of Target AS      (4 octets) |
    +------------------------------------------+
    | AS Number                   (4 octets) |   ---\
    +------------------------------------------+       \
    | pCount                      (1 octet)  |         >  Secure_Path
    +------------------------------------------+       /
    | Flags                       (1 octet)  |   ---/
    +------------------------------------------+
    | Sig Field in the Next Segment  (variable) |
    +------------- --------------------------+
```
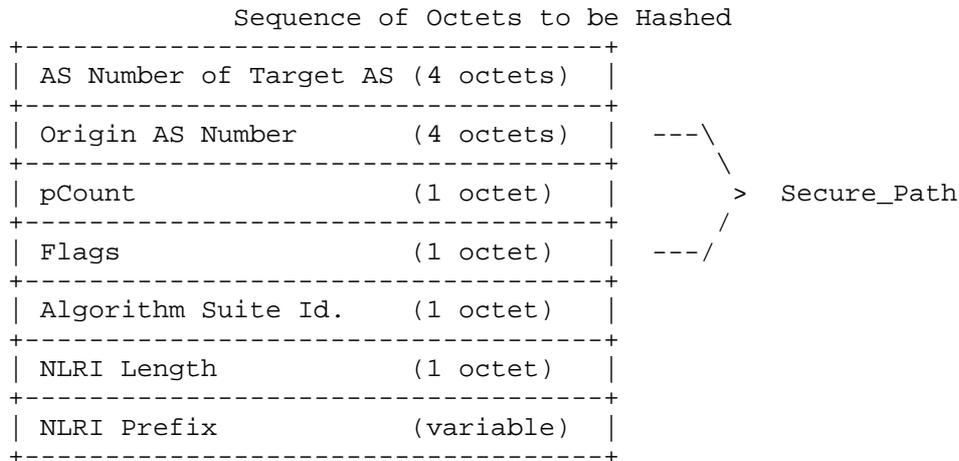
For the first segment to be processed (the most recently added
segment), the 'AS Number of Target AS' is the AS number of the BGPSEC
speaker validating the update message. Note that if a BGPSEC speaker
uses multiple AS Numbers (e.g., the BGPSEC speaker is a member of a
confederation), the AS number used 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 'AS Number of Target AS' 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.)

The AS Number, pCount and Flags fields are taken from the Secure_Path
segment that corresponds to the Signature segment currently being
processed. The 'Signature Field in the Next Segment' is the
Signature field found in the Signature segment that is next to be
processed (that is, the next most recently added Signature Segment).

Alternatively, if the segment being processed corresponds to the
origin AS (i.e., if it is the least recently added segment), then the
digest function should be computed on the following sequence of
octets:

```
                 Sequence of Octets to be Hashed
        +----------------------------------+
        | AS Number of Target AS (4 octets) |
        +----------------------------------+
        | Origin AS Number      (4 octets) |  ---\
        +----------------------------------+      \
        | pCount                (1 octet)  |       > Secure_Path
        +----------------------------------+      /
        | Flags                 (1 octet)  |  ---/
        +----------------------------------+
        | Algorithm Suite Id.   (1 octet)  |
        +----------------------------------+
        | NLRI Length           (1 octet)  |
        +----------------------------------+
        | NLRI Prefix           (variable) |
        +----------------------------------+
```

   The NLRI Length, NLRI Prefix, and Algorithm Suite Identifier are all
   obtained in a straight forward manner from the NLRI of the update
   message or the BGPSEC_Path attribute being validated. The Origin AS
   Number, pCount, and Flags fields are taken from the Secure_Path
   segment corresponding to the Signature Segment currently being
   processed.

   The 'AS Number of Target AS' is the AS Number from the Secure_Path
   segment that was added immediately after the Secure_Path segment
   containing the Origin AS Number. (That is, the Secure_Path segment
   corresponding to the Signature segment that the receiver just
   finished processing prior to the current Signature segment.)

   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 will be created
   which specifies a mandatory-to-use 'current' algorithm suite for use
   by all BGPSEC speakers [11].

   It is anticipated that, in the future mandatory, the 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 discussion of the BGPSEC threat model and related security
   considerations, please see [14].

7.1 Security Guarantees

   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 Secure_Path portion of the BGPSEC_Path attribute corresponds
to a sequence of autonomous systems who have all agreed in principle
to forward packets to the given prefix along the indicated path. (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. 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.

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. 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, or injecting BGPSEC update messages with BGPSEC_Path_Signature attributes that fail validation, or causing the peer to tear-down the BGP session. Therefore, BGPSEC sessions SHOULD be protected by appropriate transport security mechanisms.

8. IANA Considerations

TBD: Need IANA to assign numbers for the two capabilities and the BGPSEC_PATH attribute.

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
   BBN Technologies
   mlepinski.ietf@gmail.com

   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
   Sparta
   weiler+ietf@watson.org

9.2. Acknowledgements

Russ Mundy, Sandy Murphy, Keyur Patel, Mark Reynolds, Heather
Schiller, Jason Schiller, John Scudder, Ruediger Volk and David Ward
for their valuable input and review.

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 4893, May 2007.

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

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

[9]   Patel, K., Ward, D., and R. Bush, "Extended Message support for
       BGP", draft-ietf-idr-bgp-extended-messages (work in progress),
       January 2014.

[10]   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), March 2014.

[11]   Turner, S., "BGP Algorithms, Key Formats, & Signature Formats",
       draft-ietf-sidr-bgpsec-algs (work in progress), July 2014.

[12]   Scudder, J., Chen, E., Mohapatra, P., and K. Patel, "Revised
       Error Handling for BGP UPDATE Messages", draft-ietf-idr-error-
       handling (work in progress), June 2014.

11. Informative References

   [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., "Threat Model for BGP Path Security", draft-ietf-
         sidr-bgpsec-threats (work in progress), December 2013.

   [15] Bush, R. and R. Austein, "The Resource Public Key
         Infrastructure (RPKI) to Router Protocol", RFC 6810, January
         2013.

   [16] Bush, R., Patel, K., and S. Turner, "Router Key PDU for RPKI-
         Router Protocol", draft-ymbk-rpki-rtr-keys (work in progress),
         April 2013.

   [17] Bush, R., "BGPsec Operational Considerations", draft-ietf-sidr-
         bgpsec-ops (work in progress), May 2012.

Author's Address

   Matthew Lepinski (editor)
   BBN Technologies
   10 Moulton St
   Cambridge, MA 55409
   US

   Phone: +1 617 873 5939
   Email: mlepinski.ietf@gmail.com

BGPsec Protocol Specification
draft-ietf-sidr-bgpsec-protocol-23

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 autonomous system 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

Copyright Notice

to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.

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 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 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 [I-D.ietf-sidr-bgpsec-pki-profiles]
   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", "MAY", and "OPTIONAL" in this
   document are to be interpreted as described in RFC 2119 [RFC2119].

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: TBD

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

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

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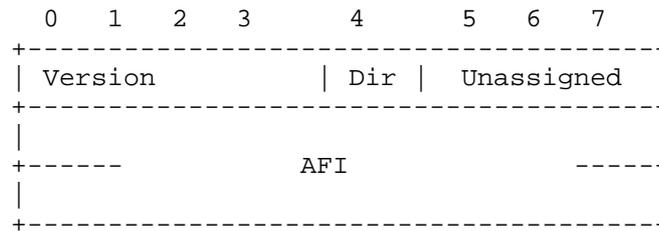
                  Figure 1: BGPsec Capability format.

   The first 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 unassigned and 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 [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
   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 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 [RFC6793]. 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.

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.

```
+-----------------------------------------------------------+
|      +-----------------+                                  |
|      |   Secure Path   |                                  |
|      +-----------------+                                  |
|      |   pCount X      |                                  |
|      |   Flags X       |                                  |
|      |   AS X          |                                  |
|      |   pCount Y      |                                  |
|      |   Flags Y       |                                  |
|      |   AS Y          |                                  |
|      |     ...         |                                  |
|      +-----------------+                                  |
|                                                           |
|      +-----------------+        +-----------------+       |
|      | Sig Block 1     |        | Sig Block 2     |       |
|      +-----------------+        +-----------------+       |
|      | Alg Suite 1     |        | Alg Suite 2     |       |
|      | SKI X1          |        | SKI X2          |       |
|      | Signature X1    |        | Signature X2    |       |
|      | SKI Y1          |        | SKI Y2          |       |
|      | Signature Y1    |        | Signature Y2    |       |
|      |      ...        |        |      ....       |       |
|      +-----------------+        +-----------------+       |
|                                                           |
+-----------------------------------------------------------+
```
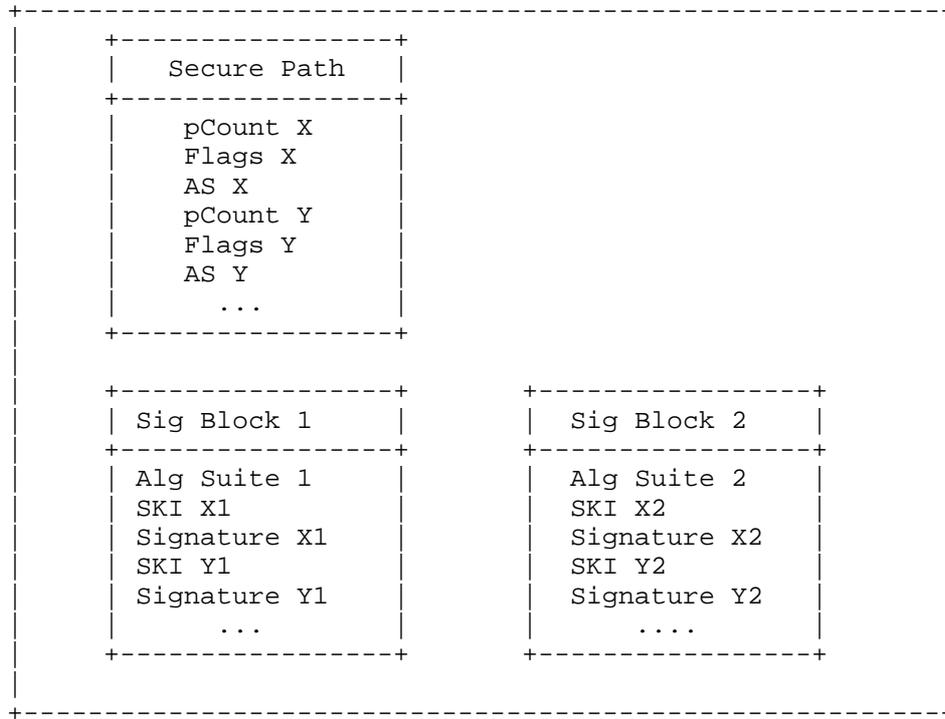
            Figure 2: High-level diagram of the BGPsec_Path attribute.

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

```
+-----------------------------------------------------------+
| Secure_Path                                (variable)     |
+-----------------------------------------------------------+
| Sequence of one or two Signature_Blocks (variable)        |
+-----------------------------------------------------------+
```
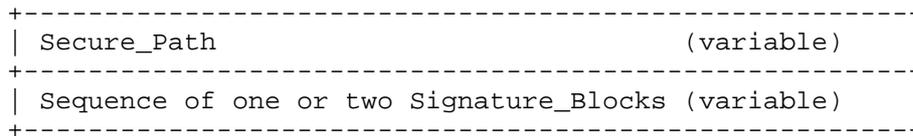
                 Figure 3: BGPsec_Path attribute format.

   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
   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.

```
                +------------------------------------------------+
                | Secure_Path Length                 (2 octets) |
                +------------------------------------------------+
                | One or More Secure_Path Segments   (variable) |
                +------------------------------------------------+
```
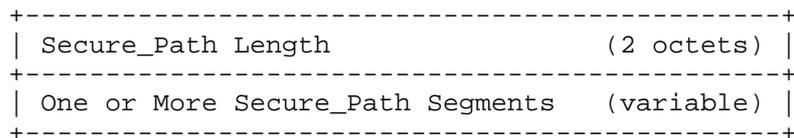
                     Figure 4: Secure_Path format.

   The specification for the Secure_Path field is provided in Figure 4
   and Figure 5. 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 (see Figure 5) for
   each Autonomous System in the path to the originating AS of the
   prefix specified in the update message. (Note: Repeated Autonomous
   Systems 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 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. Note that Section 9.1.2.2 ("Breaking Ties") in [RFC4271] mentions "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 [I-D.ietf-sidr-as-migration] for details).

The left most (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 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 [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 zero.

The remaining seven bits of the Flags are unassigned and 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.

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

3.2. Signature_Block

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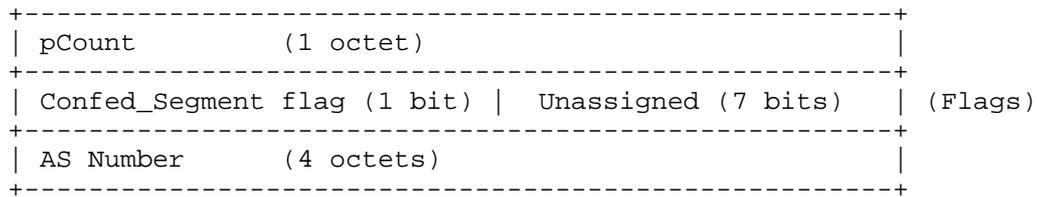

                  +--+
                  | Signature_Block Length   (2 octets) |
                  +--+
                  | Algorithm Suite Identifier   (1 octet) |
                  +--+
                  | Sequence of Signature Segments (variable) |
                  +--+
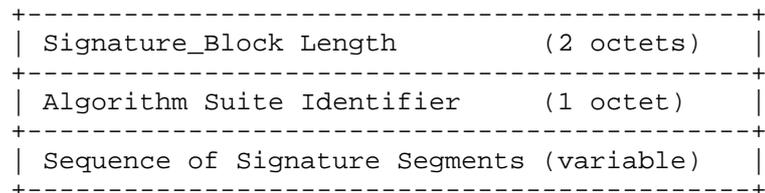
                   Figure 6: Signature_Block format.

   The Signature_Block Length in Figure 6 is the total number of octets
   in the Signature_Block (including the 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 [I-D.ietf-sidr-bgpsec-algs].

   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) |
                  +--+
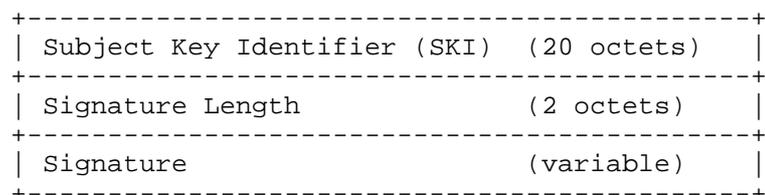
                   Figure 7: Signature Segment format.

   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 validity of BGPsec update messages). The
   SKI field has a fixed 20 octets size. 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 in Figure 7 contains a digital signature that protects
the prefix and the BGPsec_Path attribute (see Section 4 and Section 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 [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 flag
bit at default value 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
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 [RFC4760] attribute
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 Resource PKI end-entity certificate whose AS number
resource extension includes the BGPsec speaker's AS number
[I-D.ietf-sidr-bgpsec-pki-profiles]. 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 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 RFC 6482 [RFC6482]). It is expected
that most relying parties 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 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 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 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 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 [RFC6472]
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
iBGP 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 zero [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 with AS_PATH is reconstructed from the BGPsec update 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 Resource PKI router certificate that will be
   used to verify the digital signature constructed by this BGPsec
   speaker (see Section 3.1.1 in [I-D.ietf-sidr-bgpsec-pki-profiles] 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 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 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
[I-D.ietf-sidr-as-migration] 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 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 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
[I-D.ietf-sidr-bgpsec-pki-profiles]. 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
   to the next AS (i.e. (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
   which 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                              |
      +-----------------------------------+
      | Prefix                            |
      +-----------------------------------+
```
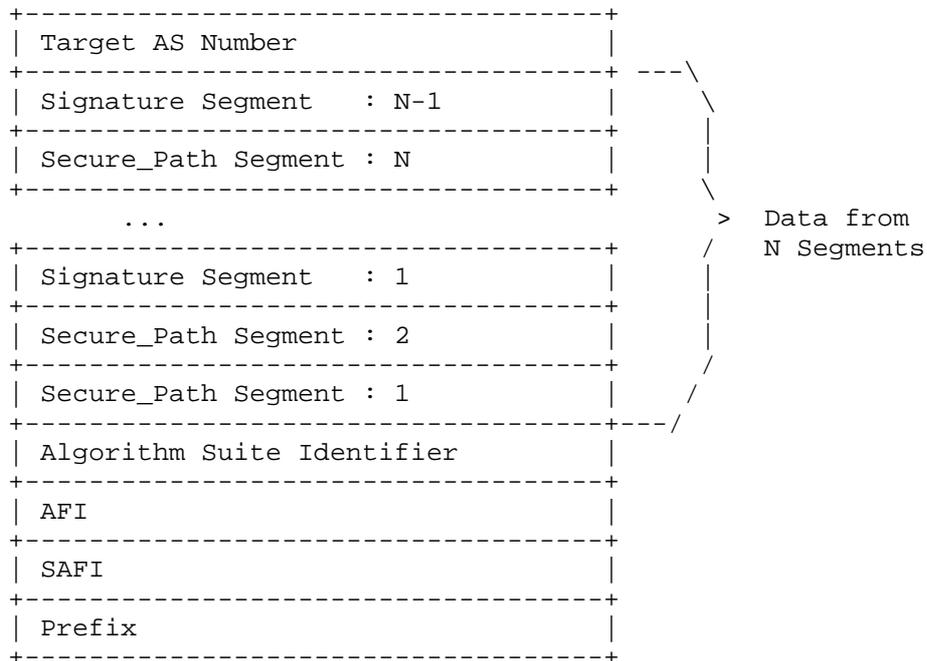
               Figure 8: Sequence of octets to be hashed.

   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 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 Prefix fields are obtained from the MP_REACH_NLRI
   attribute [RFC4760]. Additionally, in the Prefix field all of the
   trailing bits MUST be set to zero 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 autonomous system 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
from a peer that is external to the confederation and chooses to
propagate the update within the confederation, then 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, 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 one. 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 to
its peers within the confederation.

Any BGPsec_Path modifications mentioned below in the context of
propagation of the update 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 one. 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 able to provide no assurances about the
integrity of the 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 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 one. If the flag is set to one,
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 containing a Confed_Segment flag set
to one.

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 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 one, 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 zero, 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 above described procedure, 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
is equivalent to an empty AS_PATH [RFC4271].

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 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 RPKI state has changed
(e.g., from an RPKI validating cache via the RPKI-to-Router protocol
[I-D.ietf-sidr-rpki-rtr-rfc6810-bis]), the BGPsec speaker MUST re-run
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 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. 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 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.

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 using the router key PDU for the RPKI-to-Router protocol [I-D.ietf-sidr-rpki-rtr-rfc6810-bis].)

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 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. Both syntactical and protocol violation errors are checked. BGPsec_Path attribute MUST be present when a BGPsec update is received from an external (eBGP) BGPsec peer and also when such an update 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 updates the checks on the AS_PATH attribute do not apply and instead the following checks on 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 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 routers where the update 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 an 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 one.

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 one.

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

8. Using the equivalent of AS_PATH corresponding to the Secure_Path
    in the update (see Section 4.4), check that the local AS number
    is not present in the AS path (i.e. rule out 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 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 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 validation. If there is no Signature_Block
corresponding to an algorithm suite that the BGPsec speaker supports,
then in order to consider the update 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
as if it was received as an unsigned BGP update.

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, 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 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
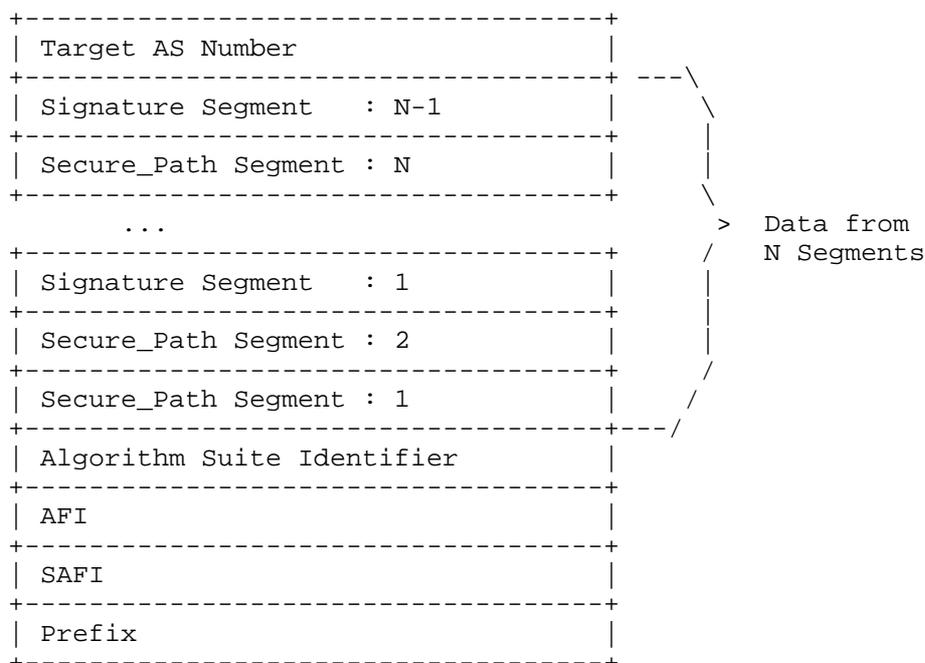Signature Segment N (see Section 4.2 and Figure 8).)

```
    +----------------------------------+
    | 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                             |
    +----------------------------------+
    | Prefix                           |
    +----------------------------------+
```

Figure 9: The 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 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 Address Family Identifier (AFI),
Subsequent Address Family Identifier (SAFI), and Prefix fields are
obtained from the MP_REACH_NLRI attribute [RFC4760].
Additionally, in the Prefix field all of the trailing bits MUST be
set to zero 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 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 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, a mandatory algorithm suites document exists which
specifies a mandatory-to-use 'current' algorithm suite for use by all
BGPsec speakers [I-D.ietf-sidr-bgpsec-algs].

It is anticipated 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, all BGPsec update messages SHOULD simultaneously use both
the 'current' algorithm suite and the 'new' algorithm suite. (Note
that Section 3 and Section 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
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 a RPKI router certificate may vary. The SKI field
in the BGPsec_Path attribute has a fixed 20 octets size (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 zero 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, 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. Operations and Management Considerations

Some operations and management issues that are closely relevant to
BGPsec protocol specification and its deployment are highlighted
here. The Best Current Practices concerning operations and
deployment of BGPsec are provided in [I-D.ietf-sidr-bgpsec-ops].

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 four-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) updates. 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 for only BGPsec use.

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 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 zero are in the context AS
confederation (see Section 4.3) and AS migration
[I-D.ietf-sidr-as-migration]. 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 received from that peer
violates this, then the update 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, route processor performance
   speedup can be achieved by incorporating the following observations.
   An update is deemed 'Valid' if at least one of the Signature_Blocks
   is marked as 'Valid' (see Section 5.2). Therefore, if an update
   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 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. Also, a
   BGPsec update 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 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 MAY differ from the first update 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 updates. 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 private AS
   number) and the ISP, the following two scenarios are possible:

1. The stub customer sends an unsigned BGP update 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 described in [I-D.ietf-sidr-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 for the prefix to the ISP's AS. An
    edge BGPsec speaker in the ISP's AS validates the update using
    RPKI data based 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]. BGPsec protocol requires that when a BGPsec
update propagates through a confederation, each Member-AS that
forwards it to a peer Member-AS MUST sign the update (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 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 [I-D.ietf-sidr-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] [I-D.ietf-sidr-rpki-rtr-rfc6810-bis]
   [I-D.ietf-sidr-publication] [RFC7115] [I-D.ietf-sidr-bgpsec-ops]
   [I-D.ietf-sidr-delta-protocol]. For example, serial-number based
   incremental update mechanisms are used for efficient transfer of just
   the data records that have changed since last update [RFC6810]
   [I-D.ietf-sidr-rpki-rtr-rfc6810-bis]. 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
   [I-D.ietf-sidr-delta-protocol]. The notification describes the
   location of the files containing the snapshot and 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 and other information during forwarding
   will be the same as 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 updates in BGPsec
   [I-D.ietf-sidr-bgpsec-algs]. 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
   methods described in [FIPS186-4] [SP800-90A]).

7.9. Incremental/Partial Deployment Considerations

   How 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 be possibly 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 contiguous-AS regions spanned
   by their respective groups. Over time they would see those
   contiguous-AS regions 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 updates
   are converted to unsigned updates before forwarding to that AS (see
   Section 4.4). At this point, the assurance that the update
   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
   be still provided, but not beyond that (for the updates 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 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 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 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
(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 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 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 its AS (for example,
its 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 enroute).

A similar argument applies to the case where a BGPsec speaker (for
some reason such as 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 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 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 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.

8.4. 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 length of the AS path.
Two other scenarios where pCount=0 is utilized are in the context AS
confederation (see Section 4.3) and AS migration
[I-D.ietf-sidr-as-migration]. 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 zero) 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 zero is unable to verify for themselves whether
pCount was set to zero legitimately.

There is a possibility of passing a BGPsec update via tunneling
between colluding ASes. For example, say, AS-X does not peer with
AS-Y, but colludes with AS-Y, signs and sends a BGPsec update to AS-Y
by tunneling. AS-Y can then further sign and propagate the BGPsec
update 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 in 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 to a Member-AS in the
confederation, it signs the update 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 to another Member-AS as
received (i.e. without adding a signature). The update 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: (1) The confederation members belong to
one organization and strong internal trust is expected; and (2)
Recall that the signatures that are internal to the confederation
MUST be removed prior to forwarding the update 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 updates 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 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 which are 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
[I-D.ietf-sidrops-bgpsec-rollover] for details.

9. IANA Considerations

   IANA is requested to register a new BGP capability from Section 2.1
   in the BGP Capabilities Code registry's "IETF Review" range. 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).

   IANA is also requested to register a new path attribute from
   Section 3 in the BGP Path Attributes registry. The code for this new
   attribute is "BGPsec_Path". The reference for the new attribute is
   this document (i.e. the RFC that replaces draft-ietf-sidr-bgpsec-
   protocol).

   IANA is requested to define 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                          | [This RFC]|
     |      | Value = 0x0                      |           |
     +------+----------------------------------+-----------+
     | 4    | Direction                        | [This RFC]|
     |      |(Both possible values 0 and 1 are |           |
     |      | fully specified by this RFC)     |           |
     +------+----------------------------------+-----------+
     | 5-7  | Unassigned                       | [This RFC]|
     |      | Value = 000 (in binary)          |           |
     +------+----------------------------------+-----------+
```

              Figure 10: IANA registry for BGPsec Capability.

   The Direction bit (4th bit) has value either 0 or 1, and both values
   are fully specified by this document (i.e. the RFC that replaces
   draft-ietf-sidr-bgpsec-protocol). Future Version values and future
   values of the Unassigned bits are assigned using the "Standards
   Action" registration procedures defined in RFC 5226 [RFC5226].

   IANA is requested to define the "BGPsec_Path Flags" registry in the
   RPKI group. The registry is as shown in Figure 11 with one value
   assigned from Section 3.1:

```
+------+-------------------------------------------+------------+
| Flag | Description                               | Reference  |
+------+-------------------------------------------+------------+
| 0    | Confed_Segment                            | [This RFC] |
|      | Bit value = 1 means Flag set              |            |
|      |               (indicates Confed_Segment)  |            |
|      | Bit value = 0 is default                  |            |
+------+-------------------------------------------+------------+
| 1-7  | Unassigned                                | [This RFC] |
|      | 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 5226
   [RFC5226].

10. Contributors

10.1. 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
      W3C/MIT
      weiler@csail.mit.edu

10.2. Acknowledgements

11. References

11.1. Normative References

   [I-D.ietf-sidr-bgpsec-algs]
              Turner, S. and O. Borchert, "BGPsec Algorithms, Key
              Formats, & Signature Formats", draft-ietf-sidr-bgpsec-
              algs-18 (work in progress), April 2017.

   [I-D.ietf-sidr-bgpsec-pki-profiles]
              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-21 (work in progress), January 2017.

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

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119,
              DOI 10.17487/RFC2119, March 1997,
              <http://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,
              <http://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,
              <http://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,
              <http://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,
              <http://www.rfc-editor.org/info/rfc5065>.

   [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an
              IANA Considerations Section in RFCs", BCP 26, RFC 5226,
              DOI 10.17487/RFC5226, May 2008,
              <http://www.rfc-editor.org/info/rfc5226>.

   [RFC5492] Scudder, J. and R. Chandra, "Capabilities Advertisement
              with BGP-4", RFC 5492, DOI 10.17487/RFC5492, February
              2009, <http://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,
              <http://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,
              <http://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,
              <http://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,
              <http://www.rfc-editor.org/info/rfc7606>.

11.2. Informative References

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

   [FIPS186-4]
              "FIPS Standards Publication 186-4: Digital Signature
              Standard", July 2013,
              <http://nvlpubs.nist.gov/nistpubs/FIPS/
              NIST.FIPS.186-4.pdf>.

   [I-D.ietf-sidr-as-migration]
              George, W. and S. Murphy, "BGPSec Considerations for AS
              Migration", draft-ietf-sidr-as-migration-06 (work in
              progress), December 2016.

   [I-D.ietf-sidr-bgpsec-ops]
              Bush, R., "BGPsec Operational Considerations", draft-ietf-
              sidr-bgpsec-ops-16 (work in progress), January 2017.

   [I-D.ietf-sidr-delta-protocol]
              Bruijnzeels, T., Muravskiy, O., Weber, B., and R. Austein,
              "RPKI Repository Delta Protocol (RRDP)", draft-ietf-sidr-
              delta-protocol-08 (work in progress), March 2017.

   [I-D.ietf-sidr-publication]
              Weiler, S., Sonalker, A., and R. Austein, "A Publication
              Protocol for the Resource Public Key Infrastructure
              (RPKI)", draft-ietf-sidr-publication-12 (work in
              progress), March 2017.

   [I-D.ietf-sidr-rpki-rtr-rfc6810-bis]
              Bush, R. and R. Austein, "The Resource Public Key
              Infrastructure (RPKI) to Router Protocol, Version 1",
              draft-ietf-sidr-rpki-rtr-rfc6810-bis-09 (work in
              progress), February 2017.

   [I-D.ietf-sidr-slurm]
             Mandelberg, D., Ma, D., and T. Bruijnzeels, "Simplified
             Local internet nUmber Resource Management with the RPKI",
             draft-ietf-sidr-slurm-04 (work in progress), March 2017.

   [I-D.ietf-sidrops-bgpsec-rollover]
             Weis, B., Gagliano, R., and K. Patel, "BGPsec Router
             Certificate Rollover", draft-ietf-sidrops-bgpsec-
             rollover-00 (work in progress), March 2017.

   [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,
             <http://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, <http://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,
             <http://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,
             <http://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,
             <http://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,
             <http://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,
             <http://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,
             <http://www.rfc-editor.org/info/rfc7132>.

   [SP800-90A]
              "NIST 800-90A: Deterministic Random Bit Generator
              Validation System", October 2015,
              <http://csrc.nist.gov/groups/STM/cavp/documents/drbg/
              DRBGVS.pdf>.

Authors' Addresses

   Matthew Lepinski (editor)
   NCF
   5800 Bay Shore Road
   Sarasota FL 34243
   USA

   Email: mlepinski@ncf.edu

   Kotikalapudi Sriram (editor)
   NIST
   100 Bureau Drive
   Gaithersburg MD 20899
   USA

   Email: kotikalapudi.sriram@nist.gov

Network Working Group   G. Huston
Internet-Draft   G. Michaelson
Intended status: Informational   APNIC
Expires: January 3, 2015   C. Martinez
                                                               LACNIC
                                                         T. Bruijnzeels
                                                             RIPE NCC
                                                            A. Newton
                                                                 ARIN
                                                             A. Aina
                                                             AFRINIC
                                                         July 2, 2014

                        RPKI Validation Reconsidered
              draft-ietf-sidr-rpki-validation-reconsidered-00.txt

Abstract

   This document reviews the certificate validation procedure specified
   in RFC6487 and highlights aspects of potentially acute operational
   fragility in the management of certificates in the RPKI in response
   to the movement of resources across registries, and the associated
   actions of Certification Authorities to maintain continuity of
   validation of certification of resources during this movement.

Status of this Memo

   This Internet-Draft is submitted in full conformance with the
   provisions of BCP 78 and BCP 79.

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF). Note that other groups may also distribute
   working documents as Internet-Drafts. The list of current Internet-
   Drafts is at http://datatracker.ietf.org/drafts/current/.

   Internet-Drafts are draft documents valid for a maximum of six months
   and may be updated, replaced, or obsoleted by other documents at any
   time. It is inappropriate to use Internet-Drafts as reference
   material or to cite them other than as "work in progress."

   This Internet-Draft will expire on January 3, 2015.

Copyright Notice

This document is subject to BCP 78 and the IETF Trust's Legal
Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info) in effect on the date of
publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with respect
to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.

Table of Contents

1. Introduction

   This document reviews the certificate validation procedure specified
   in RFC6487 and highlights aspects of potentially acute operational
   fragility in the management of certificates in the RPKI in response
   to the movement of resources across registries, and the associated
   actions of Certification Authorities to maintain continuity of
   validation of certification of resources during this movement.

2. Certificate Validation in the RPKI

   As currently defined in section 7.2 of [RFC6487], validation of PKIX
   certificates that conform to the RPKI profile relies on the use of a
   path validation process where each certificate in the validation path
   is required to meet the certificate validation criteria. This can be
   considered to be a recursive validation process where, in the context
   of an ordered sequence of certificates, as defined by each pair of
   certificates in this sequence having a common Issuer and Subject Name
   respectively, a certificate is defined as valid if it satisfies basic
   validation criteria relating to the syntactic correctness, currency
   of validity dates and similar properties of the certificate itself,
   as described in [RFC5280], and also that it satisfies certain
   additional criteria with respect to the previous certificate in the
   sequence (the Issuer part of the pair), and that this previous
   certificate is itself a valid certificate using the same criteria.
   This definition applies recursively to all certificates in the
   sequence apart from the initial sequence element, which is required
   to be a Trust Anchor.

   For RPKI certificates, the additional criteria relating to the
   previous certificate in this sequence is that the certificate's
   number resource set, as defined in [RFC3779], is "encompassed" by the
   number resource set contained in the previous certificate.

   Because [RFC6487] validation demands that all resources in a
   certificate be valid under the parent (and recursively, to the root),
   a digitally signed attestation, such as a Route Origin Authorization
   (ROA) object [RFC6482], which refers only to a subset of RFC3779-
   specified resources from that certificate validation chain can be
   concluded to be invalid, but not by virtue of the relationship
   between the RFC3779 extensions of the certificates on the putative
   certificate validation path and the resources in the ROA, but by
   other resources described in these certificates where the
   "encompassing" relationship of the resources does not hold. Any such
   invalidity along the certificate validation path can cause this
   outcome, not just at the immediate parent of the end entity
   certificate that attests to the key used to sign the ROA.

For example, in the certificate sequence:

    Certificate 1:
      Issuer A, Subject B, Resources 192.0.2.0/24, AS64496-AS64500

    Certificate 2:
      Issuer B, Subject C, Resources 192.0.2.0/24/24, AS64496-AS64511

    Certificate 3:
      Issuer C, Subject D, Resources 192.0.2.0/24

Certificate 3 is considered to be an invalid certificate, because the
resources in Certificate 2 are not encompassed by the resources in
Certificate 1, by virtue of certificate 2 describing the resources of
the range AS64501 - AS64511 in this RFC3779 resource extension.
Obviously, these Autonomous Systems numbers are not related to the
IPv4 resources contained in Certificate 3.

Any non-encompassed resource set can cause invalidation, be it an
ASN, IPv4 or IPv6 resource, if it is not encompassed by the resource
set in the parent (Issuer) certificate.

The underlying observation here is that this definition of
certificate validation treats a collection of resources as
inseparable, so that a single certificate containing a bundle of
number resources is semantically distinct from an equivalent set of
certificates where each certificate contains a single number
resource. This semantic distinction between the whole and the sum of
its parts is an artifice introduced by the particular choice of a
certificate validation procedure, as distinct from meeting any
particular operational requirement, and the result is the
introduction of operational fragility into the handling of RPKI
certificates, particularly in the case where number resources are
moved between the corresponding registries, as described here.

3. Operational Considerations

There are two areas of operational concern with the current RPKI
validation definition.

The first is that of the robustness of the operational management
procedures in the issuance of certificates. If a subordinate
Certification Authority (CA) issues a certificate that contains an
Internet Number Resource (INR) collection that is not either exactly
equal to, or a strict subset of, its parent CA, then this issued
certificate, and all subordinate certificates of this issued
certificate are invalid. These certificates are not only defined as

invalid when being considered to validate an INR that is not in the
parent CA certificate, but are defined as invalid for all INRs in the
certificate.

This constraint creates a degree of operational fragility in the
issuance of certificates, as all CA's are now required to exercise
extreme care in the issuance and reissuance of certificates to ensure
that at no time do they overclaim on the resources described in the
parent CA, as the consequences of an operational lapse or oversight
implies that all the subordinate certificates from the point of INR
mismatch are invalid. It would be preferred if the consequences of
such an operational lapse were limited in scope to the specific INRs
that formed the mismatch, rather than including the entire set of
INRs within the scope of damage from this point of mismatch downward
across the entire sub-tree of descendant certificates in the RPKI
certificate hierarchy.

The second operational consideration described here relates to the
situation where a registry withdraws a resource from the current
holder, and the resource to transferred to another registry, to be
registered to a new holder in that registry. The reason why this is
a consideration in operational deployments of the RPKI lies in the
movement of the "home" registry of number resources during cases of
mergers, acquisitions, business re-alignments, and resource transfers
and the desire to ensure that during this movement all other
resources can continue to be validated.

If the original registry's certification actions are simply to issue
a new certificate for the current holder with a reduced resource set,
and to revoke the original certificate, then there is a distinct
possibility of encountering the situation illustrated by the example
in the previous section. This is a result of an operational process
for certificate issuance by the parent CA being de-coupled from the
certificate operations of child CA.

This de-coupled operation of CAs introduces a risk of unintended
third party damage: since a CA certificate can refer to holdings
which relate to two or more unrelated subordinate certificates, if
this CA certificate becomes invalid due to the reduction in the
resources allocated to this CA relating to one subordinate resource
set, all other subordinate certificates are invalid until the CA
certificate is reissued with a reduced resource set.

In the example provided in the previous section, all subordinate
certificates issued by CA B are invalid, including all certificates
issued by CA C, until CA A issues a new certificate for CA B with a
reduced resource set.

At the lower levels of the RPKI hierarchy the resource sets affected by such movements of resources may not encompass significantly large pools of resources. However, as one ascends through this certification hierarchy towards the apex, the larger the resource set that is going to be affected by a period of invalidity by virtue of such uncoordinated certificate management actions. In the case of a Regional Internet Registry (RIR) or National Internet Registry (NIR), the potential risk arising from uncoordinated certification actions relating to a transfer of resources is that the entire set of subordinate certificates that refer to resources administered by the RIR or the NIR cannot be validated during this period.

Avoiding such situations requires that CA's adhere to a very specific ordering of certificate issuance. In this framework, the common registry CA that describes (directly or indirectly) the resources being shifted from one registry to the other, and also contains in subordinate certificates (direct or indirect) the certificates for both registries who are parties to the resource transfer has to coordinate a specific sequence of actions.

This common registry CA has to first issue a new certificate towards the "receiving" registry that adds to the RFC3779 extension resource set the specific resource being transferred into this receiving registry. The common registry CA then has to wait until all registries in the subordinate certificate chain to the receiving registry have also performed a similar issuance of new certificates, and in each case a registry must await the issuance of the immediate superior certificate with the augmented resource set before it, in turn, can issue its own augmented certificate to its subordinate CA. This is a "top down" issuance sequence."

It is possible for the common registry to issue a certificate to the "sending" registry with the reduced resource set at any time, but it should not revoke the previously issued certificate, nor overwrite this previously issued certificate in its repository publication point without specific coordination. Only when the common registry is assured that the top down certificate issuance process to the receiving registry CA chain has been completed can the common registry commence the revocation of the original certificate for the sending registry, However, it should not so until it is assured that the immediate subordinate registry CA in the path to the sending registry has issued a certificate with a reduced resource set, and so on. This implies that on the sending side the certificate issuance and revocation is a "bottom up" process.

If this process is not carefully followed, then the risk is that some or all or the subordinate certificates of this common registry CA will be unable to be validated until the entire process of

certificate issuance and revocation has been completed. While this
sequenced process is intended to preserve validity of certificates in
the RPKI, it is a complex, fragile and operationally cumbersome
process.

The underlying consideration here is that the operational
coordination of these certificate issuance and revocation actions to
effect a smooth resource transfer across registries is mandated by
the nature of the particular choice of certificate validation process
described in [RFC6487].

4. Alternatives Approaches

If the current definition of the RPKI certificate validation
procedure is considered to introduce unacceptable levels of fragility
and risk into the operational environment, what alternatives exist?

One approach is to remove the semantic requirement to consider the
collection of resources in the extension field of the RPKI
certificate as an indivisible bundle. This would allow for a
certificate to be considered as valid for some subset of the
resources listed in this extension, without necessarily being
considered as valid for all such described resources. The
implications of this approach is that any mismatch between parent and
subordinate over resources where the subordinate certificate lists
resources that are not contained in the parent certificate would
affect validity questions relating to only those particular
resources, rather than invaliding the subordinate certificate for all
resources, and all of its subordinate products. This would appear to
offer a relatively precise match tot he defined problem space, and
limits the scope of consequent third party damage in the event of a
INR mismatch in the RPKI certification hierarchy.

Another approach may involve the alteration of the RPKI provisioning
protocol [RFC6492] to include a specific signal from child to parent
("bottom up") relating to readiness for certificate revocation. At
this stage it is entirely unclear how this signalling mechanism would
operate, nor is it clear that it would alter the elements of
operational fragility nor mitigate to any meaningful extent the risks
of failure to ensure strict INR consistency at all times. This is a
topic for further study.

5. Security Considerations

The Security Considerations of [RFC6487] and [RFC6492] do not address
the topic described here. Obviously, within the current RPKI

validation procedure, any inconsistency in certificates located
towards the apex of the RPKI hierarchy would invalidate the entirety
of the sub-tree located below the point of this inconsistency. If
the RPKI was used to control inter-domain routing in the context of a
secure routing protocol, then the implications of this large scale
invalidation of certificates would have a corresponding massive
impact on the stability of routing. This appears to be a serious
situation.

6. IANA Considerations

No updates to the registries are suggested by this document.

7. Acknowledgements

TBA.

8. References

8.1. Normative References

[RFC3779]   Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP
            Addresses and AS Identifiers", RFC 3779, June 2004.

[RFC5280]   Cooper, D., Santesson, S., Farrell, S., Boeyen, S.,
            Housley, R., and W. Polk, "Internet X.509 Public Key
            Infrastructure Certificate and Certificate Revocation List
            (CRL) Profile", RFC 5280, May 2008.

[RFC6487]   Huston, G., Michaelson, G., and R. Loomans, "A Profile for
            X.509 PKIX Resource Certificates", RFC 6487,
            February 2012.

8.2. Informative References

[RFC6482]   Lepinski, M., Kent, S., and D. Kong, "A Profile for Route
            Origin Authorizations (ROAs)", RFC 6482, February 2012.

[RFC6492]   Huston, G., Loomans, R., Ellacott, B., and R. Austein, "A
            Protocol for Provisioning Resource Certificates",
            RFC 6492, February 2012.

Authors' Addresses

   Geoff Huston
   Asia Pacific Network Information Centre
   6 Cordelia St
   South Brisbane, QLD 4101
   Australia

   Phone: +61 7 3858 3100
   Email: gih@apnic.net

   George Michaelson
   Asia Pacific Network Information Centre
   6 Cordelia St
   South Brisbane, QLD 4101
   Australia

   Phone: +61 7 3858 3100
   Email: ggm@apnic.net

   Carlos M. Martinez
   Latin American and Caribbean IP Address Regional Registry
   Rambla Mexico 6125
   Montevideo 11400
   Uruguay

   Phone: +598 2604 2222
   Email: carlos@lacnic.net

   Tim Bruijnzeels
   RIPE Network Coordination Centre
   Singel 258
   Amsterdam 1016 AB
   The Netherlands

   Email: tim@ripe.net

   Andrew Lee Newton
   American Registry for Internet Numbers
   3635 Concorde Parkway
   Chantilly, VA 20151
   USA

   Email: andy@arin.net

   Alain Aina
   African Network Information Centre (AFRINIC)
   11th Floor, Raffles Tower
   Cybercity, Ebene
   Mauritius

   Phone: +230 403 51 00
   Email: aalain@afrinic.net