

**BGPSEC Protocol Specification**  
**draft-ietf-sidr-bgpsec-protocol-06**

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

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 April 25, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

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

<a href="#">1.</a>	<a href="#">Introduction . . . . .</a>	<a href="#">3</a>
<a href="#">2.</a>	<a href="#">BGPSEC Negotiation . . . . .</a>	<a href="#">3</a>
<a href="#">2.1.</a>	<a href="#">BGPSEC Send Capability . . . . .</a>	<a href="#">3</a>
<a href="#">2.2.</a>	<a href="#">BGPSEC Receive Capability . . . . .</a>	<a href="#">4</a>
<a href="#">2.3.</a>	<a href="#">Negotiating BGPSEC Support . . . . .</a>	<a href="#">5</a>
<a href="#">3.</a>	<a href="#">The BGPSEC_Path Attribute . . . . .</a>	<a href="#">6</a>
<a href="#">3.1.</a>	<a href="#">Secure_Path . . . . .</a>	<a href="#">8</a>
<a href="#">3.2.</a>	<a href="#">Signature_Block . . . . .</a>	<a href="#">9</a>
<a href="#">4.</a>	<a href="#">Generating a BGPSEC Update . . . . .</a>	<a href="#">11</a>
<a href="#">4.1.</a>	<a href="#">Originating a New BGPSEC Update . . . . .</a>	<a href="#">12</a>
<a href="#">4.2.</a>	<a href="#">Propagating a Route Advertisement . . . . .</a>	<a href="#">14</a>
<a href="#">4.3.</a>	<a href="#">Processing Instructions for Confederation Members . . . . .</a>	<a href="#">18</a>
<a href="#">4.4.</a>	<a href="#">Reconstructing the AS_PATH Attribute . . . . .</a>	<a href="#">20</a>
<a href="#">5.</a>	<a href="#">Processing a Received BGPSEC Update . . . . .</a>	<a href="#">21</a>
<a href="#">5.1.</a>	<a href="#">Overview of BGPSEC Validation . . . . .</a>	<a href="#">23</a>
<a href="#">5.2.</a>	<a href="#">Validation Algorithm . . . . .</a>	<a href="#">24</a>
<a href="#">6.</a>	<a href="#">Algorithms and Extensibility . . . . .</a>	<a href="#">28</a>
<a href="#">6.1.</a>	<a href="#">Algorithm Suite Considerations . . . . .</a>	<a href="#">28</a>
<a href="#">6.2.</a>	<a href="#">Extensibility Considerations . . . . .</a>	<a href="#">28</a>
<a href="#">7.</a>	<a href="#">Security Considerations . . . . .</a>	<a href="#">29</a>
<a href="#">8.</a>	<a href="#">IANA Considerations . . . . .</a>	<a href="#">32</a>
<a href="#">9.</a>	<a href="#">Contributors . . . . .</a>	<a href="#">32</a>
<a href="#">9.1.</a>	<a href="#">Authors . . . . .</a>	<a href="#">32</a>
<a href="#">9.2.</a>	<a href="#">Acknowledgements . . . . .</a>	<a href="#">34</a>
<a href="#">10.</a>	<a href="#">Normative References . . . . .</a>	<a href="#">34</a>
<a href="#">11.</a>	<a href="#">Informative References . . . . .</a>	<a href="#">35</a>
	<a href="#">Author's Address . . . . .</a>	<a href="#">35</a>



## **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 that has been 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 through which the update message passes 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 BGP update messages to external peers (eBGP) containing the BGPSEC\_Path needs to have the 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 update messages containing the BGPSEC\_Path attribute.

## **2. BGPSEC Negotiation**

This document defines two new BGP capabilities [6] that allow a BGP speaker to advertise to a neighbor the ability (respectively) to send or to receive BGPSEC update messages (i.e., update messages containing the BGPSEC\_Path attribute).

### **2.1. BGPSEC Send Capability**

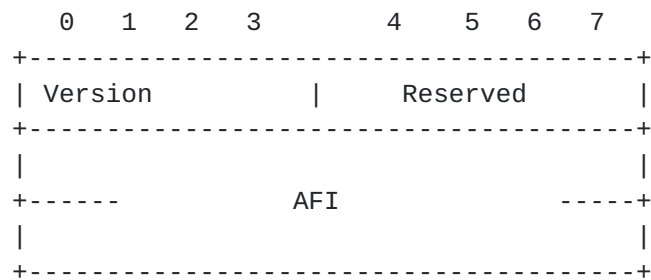
This capability has capability code : TBD

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:



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 remaining four 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. BGPSEC Receive Capability

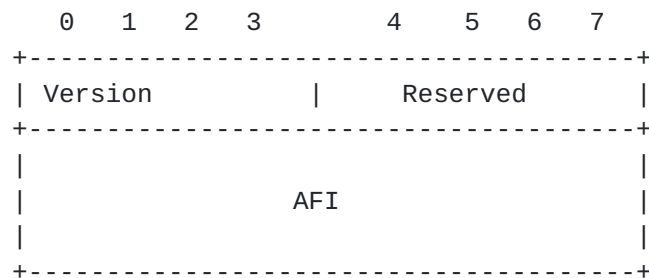
This capability has capability code : TBD

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

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



## BGPSEC Receive Capability Value:



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 remaining four 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 BGPSEC support. 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.

### 2.3. Negotiating BGPSEC Support

A BGP speaker sends the BGPSEC Send Capability (see [Section 2.1](#)) in order to indicate that the speaker is willing to send BGP update messages containing the BGPSEC\_Path attribute (for a particular address family). A BGP speaker sends the BGPSEC Receive Capability (see [Section 2.2](#)) in order to indicate that the speaker is willing to receive messages containing the BPGSEC\_Path attribute.

Note that if the BGPSEC 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 BGPSEC 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 has sent the BGPSEC Send Capability for a particular version of BGPSEC and a particular address family; and
- o The other peer has sent the BGPSEC Receive Capability for the same version of BGPSEC and the same address family.

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. If there exist multiple versions have BGPSEC that are negotiated for a particular session, the behavior of the peers (e.g., which version of BGPSEC shall actually be used) will be specified in a future document.

BGPSEC cannot provide meaningful security guarantees without support for four-byte AS numbers. Therefore, any BGP speaker that announces the capability to send BGPSEC messages, MUST also announce the capability for four-byte AS support [4]. If a BGP speaker sends the BGPSEC send 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.

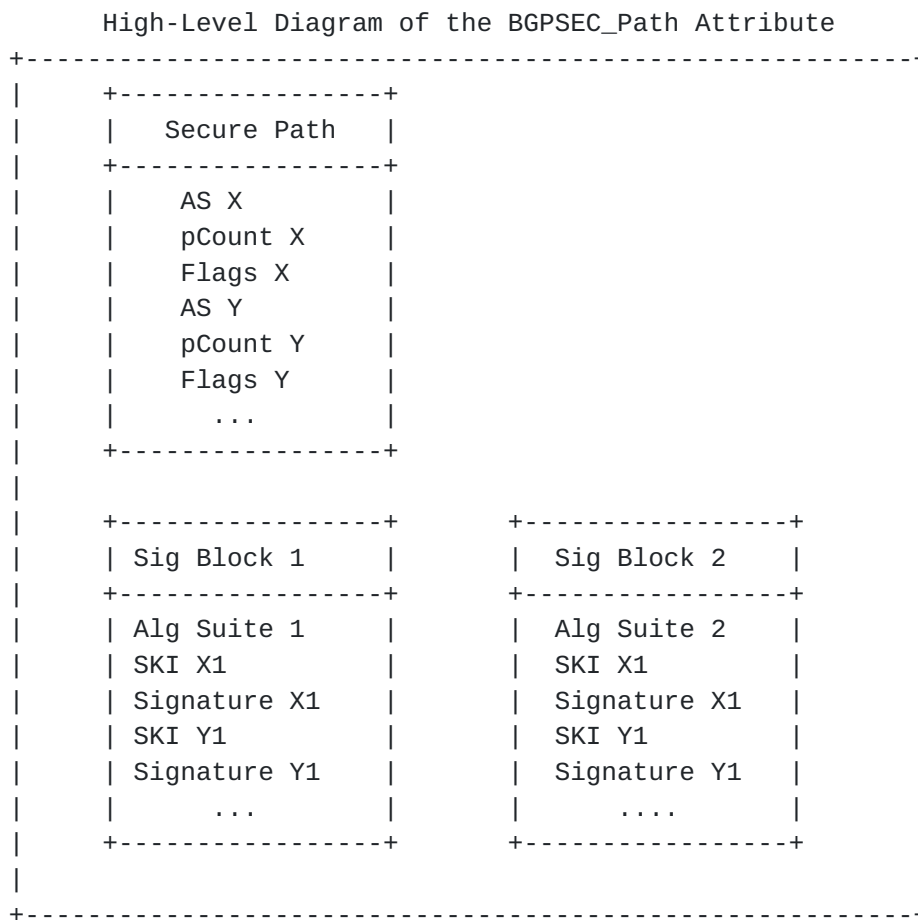
This document registers a new attribute type code for this attribute  
: TBD

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 this 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:



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 created 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.

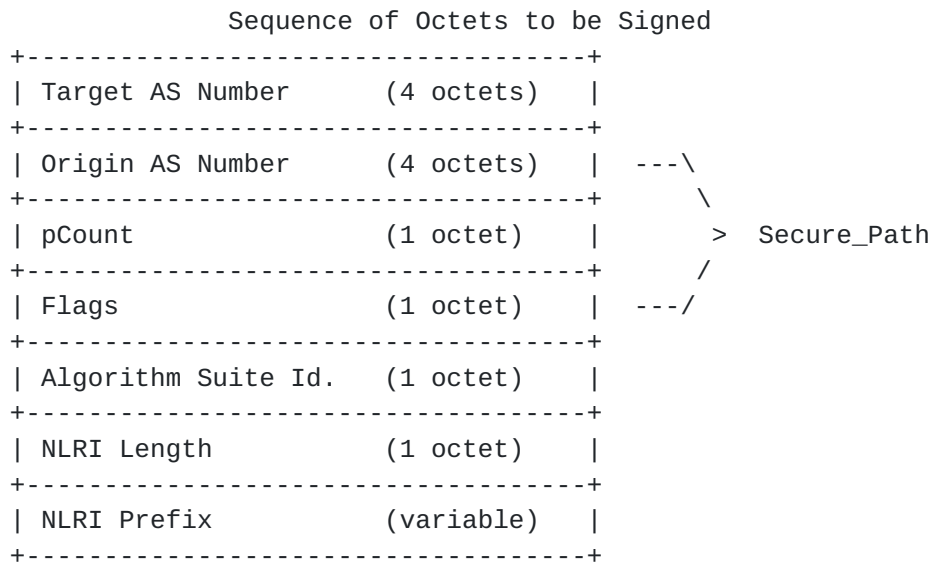
The Subject Key Identifier field (see [Section 3](#)) is populated with the identifier contained in the Subject Key Identifier extension of the RPKI router certificate corresponding to the BGPSEC speaker[10]. This Subject Key Identifier will be used by recipients of the route advertisement to identify 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.)







- 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 and if it chooses to propagate that peer's route for the prefix, then it MUST NOT attach any BGPSEC\_Path attribute to the corresponding update being propagated. (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. However, the BGPSEC speaker MAY propagate the route as a (unsigned) BGP update message without 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 standard (non-BGPSEC) update message. It should be noted that [BCP 172](#) [[12](#)] recommends against the use of AS\_SET and AS\_CONFED\_SET in AS\_PATH in 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 algorithm 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 there are two Signature\_Blocks (corresponding to different algorithm suites) that 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 then 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 appears in the same order as the corresponding Secure\_Path segments in the Secure\_Path portion of the BGPSEC\_Path attribute. 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. Note that this means that in a BGPSEC update message, an AS number appears in a Secure\_Path Segment with the Confed\_Segment flag set to one, in precisely those circumstances where 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 to have its members not 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 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 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 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](#), when processing a Signature\_Segment, the confederation member 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 for a BGPSEC speaker to receive a BGPSEC update messages containing a Secure\_Path segment with the Confed\_Sequence flag set to one from a peer who is not a member of the same AS confederation. (Such an error is treated in exactly the same way as receipt of a non-BGPSEC update message containing an AS\_CONFED\_SEQUENCE from a peer that is not a member of the same AS confederation.)

#### **4.4. Reconstructing the AS\_PATH Attribute**

BGPSEC update messages do not contain the AS\_PATH attribute. Note, however, that 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. 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 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. Note that 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 containing an AS Number extension, 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 [\[15\]](#) for the RTR protocol [\[14\]](#).)

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 shall be handled using local policy mechanisms. It is expected that BGP peers will generally prefer routes received via 'Valid' BGPSEC update messages over routes received via 'Not Valid' BGPSEC update messages as well as routes received via update messages that do not contain the BGPSEC\_Path attribute. However, BGPSEC specifies no changes to the BGP decision process. (See [\[16\]](#) for related operational considerations.)

BGPSEC validation needs only be performed at 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 identify an error in the BGPSEC\_Path attribute, then the implementation should notify the operator that an error has occurred and treat the update in a manner consistent with other BGP errors (i.e., following [RFC 4271\[2\]](#) or any future updates to that document).

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 does not exist a 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) |
+-----+

```

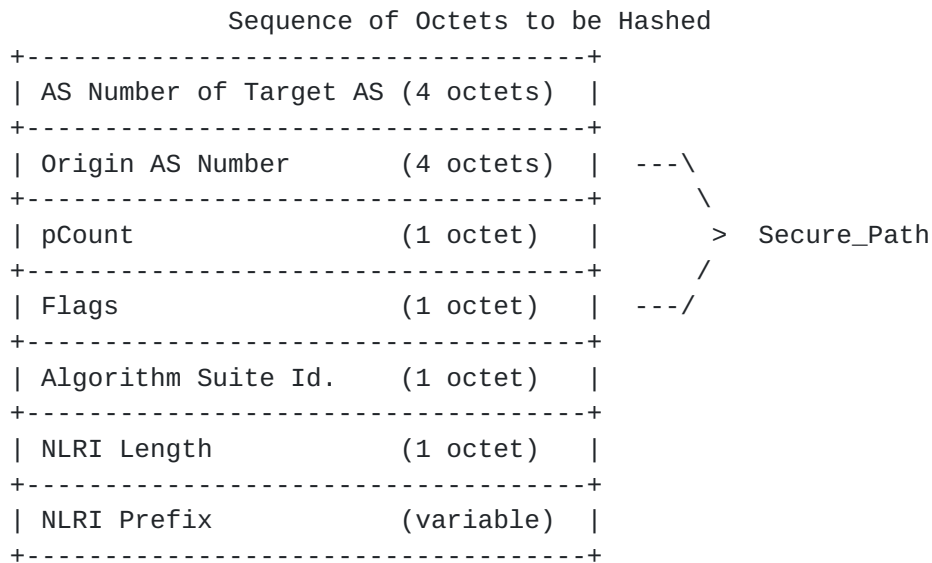
For the first segment to be processed (the most recently added segment), the 'AS Number of Target AS' is the AS number of the BGPSEC speaker validating the update message. Note that if a BGPSEC speaker uses multiple AS Numbers (e.g., the BGPSEC speaker is a member of a confederation), the AS number used 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:





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 between BGPSEC peers to use of a particular (digest and signature) algorithm suite using BGP capabilities. 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 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 [\[13\]](#).

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 propagate 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](#)).

Note that 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 if the BGPSEC speaker propagates the route advertisement received in such an update message then the BGPSEC speaker **SHOULD** add its signature to each of the `Signature_Blocks` using both the corresponding algorithm suite. 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 quite likely) 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 different 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 route to a given prefix a route obtained via a 'Not Valid' BGPSEC update message. (That is, a BGPSEC update containing only 'Not Valid' Signature\_Blocks.) 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 may also be noted here that due to possible differences in RPKI data at different vantage points in the network, a BGPSEC update that was deemed 'Not Valid' at an upstream BGPSEC speaker may indeed be deemed 'Valid' at another BGP speaker downstream.

Therefore, it is important to note that 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'

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.

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 in which an upstream entity that is two or more hops away set pCount to zero is unable to verify for themselves whether pCount was set to zero legitimately.

Finally, 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@hacitrn.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  
lepinski@bbn.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**

The authors would like to thank Luke Berndt, Sharon Goldberg, Ed Kern, Chris Morrow, Doug Maughan, Pradosh Mohapatra, 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", July 2012.
- [10] Reynolds, M., Turner, S., and S. Kent, "A Profile for BGPSEC Router Certificates, Certificate Revocation Lists, and Certification Requests", April 2012.
- [11] Turner, S., "BGP Algorithms, Key Formats, & Signature Formats", March 2012.

## **11. Informative References**

- [12] Kumari, W. and K. Sriram, "Recommendation for Not Using AS\_SET



and AS\_CONFED\_SET in BGP", [RFC 6472](#), December 2011.

- [13] Kent, S., "Threat Model for BGP Path Security", February 2012.
- [14] Bush, R. and R. Austein, "The RPKI/Router Protocol", February 2012.
- [15] Bush, R., Patel, K., and S. Turner, "Router Key PDU for RPKI-Router Protocol", October 2012.
- [16] Bush, R., "BGPsec Operational Considerations", May 2012.

#### Author's Address

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

Phone: +1 617 873 5939  
Email: [mlepinski@bbn.com](mailto:mlepinski@bbn.com)

