Issues in Revising BGP-4 (RFC1771 to RFC4271)
draft-ietf-idr-bgp-issues-05

Abstract

This document records the issues discussed and the consensus reached in the Interdomain Routing (IDR) Working Group during its efforts to revise and bring up to date the base specification for the BGP-4 protocol as documented in RFC1771. The document focuses on the changes tracked from August 2002 when the last major push for revision began. The results of these efforts are encoded in RFC4271, which should be taken as normative for any of the issues that were discussed. The discussion here is intended to record how and why some of the changes to BGP were made.

Table of Contents

1. Introduction

This document records the issues discussed and the consensus reached in the Interdomain Routing (IDR) Working Group during its efforts to revise and bring up to date the base specification for the BGP-4 protocol as documented in RFC1771. The results of these efforts are encoded in RFC4271. The rational for doing this is simple: Experience has demonstrated that the same issues and questions tend to come up again and again. This memo will document not only the decisions on these issues but also how and why the working group reached those conclusions. We hope that this will help make future discussions more fruitful by providing them with a historical context.

This document traces the evolution of the BGP-4 base specification from its incarnation as draft-ietf-idr-bgp4-17.txt through the big revision and update push culminating in draft-ietf-idr-bgp4-19.txt. It is divided into two main sections. The first deals with the issues discussed between -17 and -18, and the second deals with the issues discussed between -18 and -19.

N.B. There is no rhyme or reason to the numbering scheme other than unique tags to address the issues.

2. The Issues from -17 to -18

This section lists the issues discussed on the list from late August to late October 2002.

2.1. IDR WG Charter

Status: Consensus
Change: Yes
Summary: New charter adopted.

Discussion:

A variety of discussions surrounded the new charter. The rough consensus is to accept the new charter that the AD's have proposed, and to push as hard a possible to get the base spec to RFC status so other drafts that are dependent can also move forward.

For our information, Alex has provided these approximate time lines:

```
Stage Anticipated delay Comment
-----------------------------------------------------------------------
AD-review 1-4 weeks The document may go back depending on to the WG
```

for the workload AD-review comments to be addressed; this would introduce additional delay.

IETF LC 2 weeks Same as above

IESG review & 1-2 weeks depending Same as above telechat on when the IETF LC ends
----

Note that if the document is sent back to the WG at some stage, required changes may warrant an additional WG Last Call.

I can personally commit to a 2-week upper bound for the AD-review period. Bill may have a different timer granularity.

The opinions expressed on this were 7 in favor, 4 against.

This thread has messages subjects of "BGP spec and IDR WG charter" and "IDR WG charter".

2.2. TCP Port

Status: Consensus
Change: Yes
Summary:

Change:
"BGP uses TCP port 179 for establishing its connections."

To:
"BGP listens on TCP port 179."

Discussion:

There has been a discussion on clarifying the wording in Section 2, on which port BGP uses. The original text was:

"BGP uses TCP port 179 for establishing its connections."

The proposed new text is:

"BGP listens on TCP port 179."

There seems to be a rough consensus that the new text is better.

This thread has a message subject of "Review: Section 2, TCP Port 179"

2.3. FSM wording for what state BGP accepts connections in

Status: Consensus
Change: No
Summary: No change necessary

Discussion:

An issue was brought up later in the "Review: Section 2, TCP Port 179" thread about the words in the FSM for what state BGP accepts connections in. The consensus is that the existing wording is clear.

2.4. BGP Identifier/Router ID

Status: Consensus
Change: No
Summary: No change necessary to base draft. Perhaps in a BCP.

Discussion:

The "admin dist/gp spec proposal", "Router ID" and "bgp spec proposal" threads discussed the BGP Identifier and how close or not it is to IGP's Router ID. The consensus was that this discussion is better saved for a BCP draft, and that it does not need to be contained in the base spec.

2.5. Direct EBGP Peers

Status: Consensus Change: No Summary: A recollection that ebgp peers must be direct. No text proposed, no discussion.

Discussion:

Jonathan recalled something that stated that ebgp peers must be direct. No specific sections were quoted.

Yakov responded to this with:

Section 5.1.3 talks about both the case where ebgp peers are 1 IP hop away from each other:

2) When sending a message to an external peer X, and the peer is one IP hop away from the speaker:

as well as the case where they are multiple IP hops away from each other:

3) When sending a message to an external peer X, and the peer is

multiple IP hops away from the speaker (aka "multi hop EBGP"):

And emphasized that multi hop EBGP does exist.

This came up in the "bgp draft review" thread.

2.6. Disallow Private Addresses

Status: Consensus
Change: No
Summary: No change necessary

Discussion:

In the thread entitled "bgp draft review":

Mentioned explicitly disallowing private addresses. The consensus was that there is no reason to disallow them. Which IP addresses peers use is an operational issue.

2.7. Renumber Appendix Sections

Status: Consensus
Change: Yes
Summary: Rename/renumber appendix sections so they do not have the same numbers as sections of the main text.

Discussion:

In the tread entitled "bgp draft review":

This thread brought up renaming sections in the appendix to avoid confusion with sections of the same number in the main text.

Yakov responded that he would do so in the next edition.

2.8. Jitter Text

Status: Consensus
Change: Yes
Summary:

Get rid of section 9.2.1.3 ("Jitter"). Move the text to an Appendix: "BGP Timers" Expand text to indicate that jitter applies to all timers, including ConnectRetry.

The text for the appendix is listed at the end of the discussion.

Discussion:

In the tread entitled "bgp draft review": The thread also proposed:

"jitter should be applied to the timers associated with MinASOriginationInterval, Keepalive, and MinRouteAdvertisementInterval"

Be changed to:

"jitter should be applied to the timers associated with ConnectRetry timer"

Yakov agreed with making some changes and suggested that we make sure that jitter is applied to all timers. Specifically, he proposes we get rid of section 9.2.1.3 ("Jitter"), move the text of this section into Appendix "BGP Timers", and expand the text to indicate that jitter applies to ConnectRetry timer as well.

Jonathan, the original commenter, agreed with Yakov's suggestion.

In a follow-up to this issue, there was a question raised about the values we have specified for timers in the document. Specifically:

The ConnectRetry timer is should have a value that is 'sufficiently large to allow TCP initialization. Application of jitter can reduce the this value (by up to 25%). A configuration which the ConnectRetry timer has been pegged at a value close to TCP connection time may cause a connection to be terminated as a result of this jitter. Is this a cause for concern ?

The default value suggested for ConnectRetry (120 seconds) is sufficiently large that event with a jitter of 0.75, it will be greater than TCP's connection establishment timer.

Is adding a jitter to the ConnectRetry timer a standard practice ? What benefit does this provide ?

Curtis responded to this with:

The TCP connection establishment timer is 75 seconds (sysctl yield "net.inet.tcp.keepinit: 75000" in BSD-oids).

The ConnectRetry determines when to make a second attempt after a prior attempt to connect has failed. It is to avoid a rapid succession of retries on immediate failures (for example "Connection refused" if the peer was in the middle of a reboot, Network Unreachable if you can't get there from here, etc) but also covers

the case where the TCP SYN goes off and is never heard from again.

And Jonathan replied with this information about current practice:

It seems to me that if you bring up all bgp peers at once it may lead to load spikes on the network. Cisco seems to wait 27.5 +/- 2.5 seconds for IBGP, and 40 +/- 5 seconds for EBGP--20 sec. from config time to the "open active, delay" jittered delay assignment plus the jittered delay (5 to 10 sec. for IBGP, and 15 to 25 sec. for EBGP). This would also apply for "no neighbor x.x.x.x shutdown". Their value of ConnectRetry is 60sec. though, not sure how this value is used (based on above). Maybe some Cisco folks can chime in on this one???

I did not check Juniper.

Also, interestingly, they do not apply jitter to the other timers (as far as I can tell), but I don't see a problem with this.

Another timer that they use that is not mentioned in the draft/rfc is the next hop resolution timer which is 30 seconds. Although it would be nice to have this in the spec, I will concede that it is out of scope and/or implementation dependent.

So the question that arises from this followup, is how does this question affect the text of the appendix on jitter?

Curtis replied that we need to only state that jitter should be applied to all timers. Whether a vendor does so or not is a minor deficiency and does not bear on interoperability. Therefore, specifying exact details are not necessary.

After Jonathan's response Curtis and Jonathan agreed that jitter should be added to all timers and that we should state so in the text.

Yakov proposed the following text for the appendix to discuss jitter:

I'd like to propose the following text for "BGP Timers" section:

BGP employs five timers: ConnectRetry (see Section 8), Hold Time (see Section 4.2), KeepAlive (see Section 8), MinASOriginationInterval (see Section 9.2.1.2), and MinRouteAdvertisementInterval (see Section 9.2.1.1).

The suggested value for the ConnectRetry timer is 120 seconds.

The suggested value for the Hold Time is 90 seconds.

The suggested value for the KeepAlive timer is 1/3 of the Hold Time.

The suggested value for the MinASOriginationInterval is 15 seconds.

The suggested value for the MinRouteAdvertisementInterval is 30 seconds.

An implementation of BGP MUST allow the Hold Time timer to be configurable, and MAY allow the other timers to be configurable.

To minimize the likelihood that the distribution of BGP messages by a given BGP speaker will contain peaks, jitter should be applied to the timers associated with MinASOriginationInterval, Keepalive, MinRouteAdvertisementInterval, and ConnectRetry. A given BGP speaker shall apply the same jitter to each of these quantities regardless of the destinations to which the updates are being sent; that is, jitter will not be applied on a "per peer" basis.

The amount of jitter to be introduced shall be determined by multiplying the base value of the appropriate timer by a random factor which is uniformly distributed in the range from 0.75 to 1.0.

Jeff & Ben agreed with this.

Justin suggested that we move the range from 0.75 to 1.25 to ensure that the average is around the configured value. Yakov agreed with Justin's changes. Jonathan disagreed, arguing that it was out-of-scope for the task of clarifying the text only. Justin agreed and withdrew his comment.

Curtis liked the general text, but suggested these modifications:

minor improvement (not really an objection) -- s/suggested value/suggested default value/g

Also

s/shall apply the same jitter/may apply the same jitter/ (to each of these quantities regardless of ...).

s/jitter will not be applied/jitter need not be configured/ (on a "per peer" basis).

He stated that in Avici's implementation they allow a lot of granularity in timer settings, so this reflects current practice.

Curtis also suggested changing the last paragraph:

The suggested default amount of jitter shall be determined by multiplying the base value of the appropriate timer by a random factor which is uniformly distributed in the range from 0.75 to 1.0. A new random value should be picked each time the timer is set. The range of the jitter random value MAY be configurable.

This would make it clear that it is possible to have this timer as configurable and still be within spec.

Other comments on Yakov's text pointed out that IOS uses 5 seconds as the default IBGP MinRouteAdvertisementInterval.

Tom pointed out that there seems to be a discrepancy between this text and the FSM: The FSM has an OpenDelay timer. And the FSM suggests a HoldTimer of 4 minutes.

In following up on this issue, Yakov stated:

Here is the final text for the BGP Timers section:

BGP employs five timers: ConnectRetry (see Section 8), Hold Time (see Section 4.2), KeepAlive (see Section 8), MinASOriginationInterval (see Section 9.2.1.2), and MinRouteAdvertisementInterval (see Section 9.2.1.1).

The suggested default value for the ConnectRetry timer is 120 seconds.

The suggested default value for the Hold Time is 90 seconds.

The suggested default value for the KeepAlive timer is 1/3 of the Hold Time.

The suggested default value for the MinASOriginationInterval is 15 seconds.

The suggested default value for the MinRouteAdvertisementInterval is 30 seconds.

An implementation of BGP MUST allow the Hold Time timer to be configurable, and MAY allow the other timers to be configurable.

To minimize the likelihood that the distribution of BGP messages by a given BGP speaker will contain peaks, jitter should be applied to the timers associated with MinASOriginationInterval, Keepalive, MinRouteAdvertisementInterval, and ConnectRetry. A given BGP speaker may apply the same jitter to each of these quantities regardless of the destinations to which the updates are being sent; that is, jitter

need not be configured on a "per peer" basis.

The suggested default amount of jitter shall be determined by multiplying the base value of the appropriate timer by a random factor which is uniformly distributed in the range from 0.75 to 1.0. A new random value should be picked each time the timer is set. The range of the jitter random value MAY be configurable.

With this in mind, I would suggest we mark this issue as closed.

Jonathan suggested adding "per peer" to the text, Yakov responded with this text:

An implementation of BGP MUST allow the Hold Time timer to be configurable on a per peer basis, and MAY allow the other timers to be configurable.

This proposal met with general agreement. This issue is at consensus.

2.9. Reference to RFC904 - EGP Protocol

Status: Consensus
Change: Yes
Summary: Add a reference to RFC904

Discussion:

The "Review Comment: Origin Attribute pg 14" thread suggested adding a reference to RFC904(?), to refer to the EGP protocol. There was no discussion.

Yakov agreed to this, and Jonathan seconded it.

2.10. Extending AS_PATH Attribute

Status: Consensus
Change: Yes
Summary: Add this to 9.2:

If due to the limits on the maximum size of an UPDATE message (see Section 4) a single route doesn't fit into the message, the BGP speaker MUST not advertise the route to its peers and may choose to log an error locally.

Discussion:

The "Extending AS_PATH attribute length en route" thread brought up

the issue of what action should we specify when we receive a route with an AS_PATH that exceeds the defined maximum length. There was some discussion, and it was suggested that, after logging the error, the route not be propagated.

Yakov stated that:

The real issue here is how to handle the case when a route (a single address prefix + path attributes) doesn't fit into 4K bytes (as the max BGP message size is 4 K). To address this issue I would suggest to add the following to 9.2:

After some discussion, Yakov's proposed text's last sentence was dropped and we arrived at:

If due to the limits on the maximum size of an UPDATE message (see Section 4) a single route doesn't fit into the message, the BGP speaker may choose not to advertise the route to its peers.

In response to Andrew's clarification question to the list, Curtis responded:

Wording would be more like:

If the attributes for a specific prefix becomes too large to fit the prefix into the maximum sized BGP UPDATE message, the prefix should not be advertised further. Truncation or omission of attributes should not occur unless policies for such modifications are specifically configured. Such policies may contribute to the formation of route loops and are not within the scope of this protocol specification.

After some additional discussion, it was decided that we add "and may choose to log an error locally." to the end of Yakov's text.

Also, we agreed to change "may choose not to advertise..." to "MUST NOT advertise...".

So the text on the table right now is:

If due to the limits on the maximum size of an UPDATE message (see Section 4) a single route doesn't fit into the message, the BGP speaker MUST not advertise the route to its peers and may choose to log an error locally.

This met with one agreement and no disagreements. We have a consensus.

2.11. Rules for routes from Loc-RIB to Adj-RIB-Out - Section 9.1

Status: Consensus
Change: Yes
Summary: Add this text:

The local speaker SHALL then install that route in the Loc-RIB, replacing any route to the same destination that is currently being held in the Loc-RIB. When the new BGP route is installed in the Rout- ing Table, care must be taken to ensure that existing routes to the same destination that are now considered invalid are removed from the Routing Table. Whether or not the new BGP route replaces an existing non-BGP route in the Routing Table depends on the policy configured on the BGP speaker.

Discussion:

The "Proxy: comments on section 9.1.3" thread brought up some lack of clarity in the section discussing the rules for which routes get propagated from the Loc-RIB into the Adj-RIB-Out. These discussions resulted in a number of suggestions for new text.

The first new text was proposed to clarify the issue that the thread first brought up:

I agree that this could use some clarification. How about adding to b) in section 9.1:

The Loc-RIB must contain only one route per destination; further, it must include all routes that the BGP speaker is using.

changing c) in section 9.1.2 to:

c) is selected as a result of the Phase 2 tie breaking rules specified in 9.1.2.2, or

and adding

d) when routing protocols other than BGP are in use, is determined by some other local means to be the route that will actually be used to reach a particular destination.

This text was never discussed or a consensus formed on putting it in the document.

This modification to 9.1.2 was also proposed to address the same concern:

How about changing the paragraph after c) in 9.1.2 to:

The local speaker SHALL then install that route in the Loc-RIB, replacing any route to the same destination that is currently being held in the Loc-RIB. This route SHALL then also be installed in the BGP speakers forwarding table.

There was one response in the negative to this change, arguing that is is not necessary.

Yakov replied to this that:

Wrt "adding to b) in section 9.1", the second part (after ";") is redundant as this point is already stated in 3.2. Wrt the first point about Loc-RIB containing just one route per destination, I would suggest to add it to section 3.2, where Loc-RIB is first introduced, rather than adding it to 9.1.

Wrt "changing c)... and adding...", I have no objections to add/ modify the text, as suggested above.

I am not sure though that changing the paragraph after c) in 9.1.2 is really necessary though, so I would prefer to keep it as is.

The "issue 11" thread this was being discussed in then digressed to the topic, now covered in issue 11.3.

Ben re-addressed the original issue with this input:

I have somewhat of an issue with the paragraph after item c section 9.1.2 as discussed.

which is =>

"The local speaker SHALL then install that route in the Loc-RIB, replacing any route to the same destination that is currently being held in the Loc-RIB. If the new BGP route is installed in the Routing Table (as a result of the local policy decision), care must be taken to ensure that invalid BGP routes to the same destination are removed from the Routing Table. Whether or not the new route replaces an already existing non-BGP route in the routing table depends on the policy configured on the BGP speaker."

Can we assume that its OK to have a route present in the Loc-RIB and possibly in the adj-RIB-Out but not in the Routing table due to some policy. Won't we violate rule number 1? Only advertise what you use.

As conversely implied in this sentence =>

"If the new BGP route is installed in the Routing Table (as a result of the local policy decision), care must be taken to ensure that invalid BGP routes to the same destination are removed from the Routing Table"

I would rephrase the paragraph as follows =>

"The local speaker SHALL then install that route in the Loc-RIB, replacing any route to the same destination that is currently being held in the Loc-RIB. When the new BGP route is installed in the Routing Table, care must be taken to ensure that existing routes to the same destination that are now considered invalid are removed from the Routing Table. Whether or not the new BGP route replaces an existing non-BGP route in the routing table depends on the policy configured on the BGP speaker."

Jeff replied:

With the exception that Routing Table should be capitalized throughout, I'd suggest we take this as consensus.

Yakov agreed. We are at consensus.

2.11.1. Rules for routes from Loc-RIB to Adj-RIB-Out - Section 9.1.3

Status: Consensus
Change: Yes
Summary: The text below will be added to the -18 version.

Discussion:

In further discussions around this issue, this text was also proposed:

How about adding to section 9.1.3, at the end:

Any local-policy which results in reachability being added to an Adj-RIB-Out without also being added to the local BGP speaker's forwarding table is beyond the scope of this document.

This suggestion received one response that agreed to this change.

This text will be added to the -18 version, and since there were no objections, this issue has been moved to consensus.

2.11.2. Rules for routes from Loc-RIB to Adj-RIB-Out - Section 2

Status: Consensus
Change: Yes
Summary: Add this text:

In the context of this document we assume that a BGP speaker advertises to its peers only those routes that it itself uses (in this context a BGP speaker is said to "use" a BGP route if it is the most preferred BGP route and is used in forwarding). All other cases are outside the scope of this document.

Discussion:

Additionally this thread produced this section of new text, in section 2:

<OLD>

"one must focus on the rule that a BGP speaker advertises to its peers (other BGP speakers which it communicates with) in neighboring ASs only those routes that it itself uses."

Should be changed to

<NEW #1>

"one must focus on the rule that a BGP speaker advertises to its peers (other BGP speakers which it communicates with) in neighboring ASs only routes whose NLRIs are locally reachable."

<NEW #2>

"one must focus on the rule that a BGP speaker advertises to its peers (other BGP speakers which it communicates with) in neighboring ASs only routes which are locally reachable. Local reachability can be achieved by having any protocol route to the given destination in the routing table."

There were a lot of emails exchanged on this topic with a variety of texts proposed (see early in the "Active Route" thread). This issue reopened with Jonathan, who brought up the issue originally, stating that:

The issue I raised, and would like to be [re-]considered is with:

"one must focus on the rule that a BGP speaker advertises to its peers (other BGP speakers which it communicates with) in neighboring

ASs only those routes that it itself uses."

Curtis replied that:

That is called route origination and it is allowed by:

9.4 Originating BGP routes

A BGP speaker may originate BGP routes by injecting routing information acquired by some other means (e.g. via an IGP) into BGP. [...] The decision whether to distribute non-BGP acquired routes within an AS via BGP or not depends on the environment within the AS (e.g. type of IGP) and should be controlled via configuration.

Advice on what to put in the AS_PATH and NEXT_HOP is in the document.

He continued with:

I don't think there was ever consensus on what to do with the statement "a BGP speaker advertises to its peers (other BGP speakers which it communicates with) in neighboring ASs only those routes that it itself uses". Some reasonable choices are:

1. Omit it (the implied consensus of the rewrite of the paragraph in 32.2).

2. Leave it as is and put it in another paragraph to separate it from the destination based routing statement.

3. Clean up the wording and put it in another paragraph to separate it from the destination based routing statement.

The separate paragraph for 2 would be the exact sentence we now have.

A BGP speaker advertises to its peers (other BGP speakers which it communicates with) in neighboring ASs only those routes that it itself uses.

In possibility 3 we (try to) clear up the ambiguity about the meaning of the word "use" in this sentence.

A BGP speaker advertises to its peers (other BGP speakers which it communicates with) in neighboring ASs only those routes that it itself uses. In this context a BGP speaker is said to "use" a BGP route if it is the most preferred BGP route and is either directly used in forwarding or in a specifically configured case where the BGP route would be forwarded internally but IGP forwarding information is used. The latter case reflects a usage in which the IGP is used for

forwarding but BGP is originated to IBGP to carry attributes that cannot be carried by the IGP (for example, BGP communities [N]). Other special cases such as virtual routers and multiple instances of BGP on a single router are beyond the scope of this document but for each of these the statement "a BGP speaker advertises to its peers (other BGP speakers which it communicates with) in neighboring ASs only those routes that it itself uses" can (and should in the definition of the extension) be made true with an appropriate definition of the word "use".

Unless someone volunteers better wording this may be a good starting point. I thing the last sentence borders on ridiculous in a protocol spec but may be necessary to address specific objections raised on this mailing list. If we want to elaborate on the meaning of the word "use" and address the objections this is what we end up with.

Of course looking at what we ended up with, I'd also go along with the other two options (leave it out or put the one sentence in a separate paragraph as is).

After some additional discussion (in the "issue 11.2" thread), we have come to a consensus on this text:

In the context of this document we assume that a BGP speaker advertises to its peers only those routes that it itself uses (in this context a BGP speaker is said to "use" a BGP route if it is the most preferred BGP route and is used in forwarding). All other cases are outside the scope of this document.

This issue is at consensus.

2.11.3. Documenting IBGP Multipath

Status: Consensus
Change: Yes
Summary: The documenting of IBGP Multipath is left to another Internet Draft. The consensus is that it should not be in the base spec.

Discussion:

This thread began in the "issue 11" discussion. In it it was proposed that:

There is support in some router vendors to allow more than one BGP route to be installed, for the purpose for load balancing. Given that this is a current practice, and seems to be a useful feature as well, should we insist that only one route be installed in the Loc-

RIB ?

I would like to suggest that all sections which use MUST in the context of only one route in Loc-RIB be relaxed a little to a SHOULD, and a section added that states that it is possible for a n implementation to add more than one route to the Loc-RIB for the purposes of load balancing.

While it will be useful to describe how this situation is the handler, it is perhaps sufficient to even state that handling of this situation is outside the scope of this RFC.

I am including some proposed text for this purpose:

For the part:

> The Loc-RIB must contain only one route per destination;

consider instead,

% The Loc-RIB SHOULD contain only one route per destination. % An implementation may choose to install multiple routes to % a destination (for the purposes of load balancing). The % handling of such a configuration, however, is outside the % scope of this RFC.

Perhaps, this can be in section 3.2 instead.

After much discussion back and forth, it was agreed that documenting IBGP Multipath behavior is a good thing. However, it is something that belongs in another draft.

Alex opened this issue up again. There were a flurry of responses, most all of them agreeing with the original consensus that we should document this feature in a different draft, since it doesn't affect the core interoperability requirements, and we want to advance the spec in a timely manner.

Alex persisted in his assertion that this belongs in the base specification. Right now, the issue is still open.

This discussion later expanded in scope to include all BGP multipath.

Curtis laid out a good description of the various flavors of multipath:

In addition to IGP multihop, there are two cases of BGP multipath.

In IGP multihop there is one BGP advertisement but to ways to reach

th BGP NEXT_HOP via the IGP.

In one case of BGP multihop, two (or more) IBGP routers peering with the same external AS have equal routes to a destination and are an equal cost away from a third router. BGP multihop is applicable to that third router. Without BGP multihop, BGP would normally pick the BGP NEXT_HOP of the advertisement from only one of those IBGP peers (using BGP Identifier) and use that. The IGP lookup would yield one next hop. With BGP multihop, BGP uses the BGP NEXT_HOP of both advertisements. Each BGP NEXT_HOP has a different IGP next hop (one or more IGP next hop).

The second case is where all of the candidates routes for BGP multipath are external. Seldom does IGP multipath come into play for EBGP (odd tunneled EBGP multihop cases maybe). Typically the load is split among two (or more) routers in the same AS.

If in EBGP multipath you split among routers in difference AS, an aggregate should be formed. This is still prior to the IGP cost rule in the route selection.

Normally one would not combine IBGP and EBGP in multihop given that the decision point for multihop is after "d" in 9.1.2.2. If the multihop decision was prior to "d", then two routers each with an external peering would forward some of the traffic to each other and for some src/dst pairs, they'd form a loop. [So don't do that!]

This is getting to be a lot to add to the base spec. I hope we've convinced you that we should put it in another document.

Curtis later added specific text, that could serve as a start for the new document (or added to the base spec if the consensus ended up going the other way):

BGP specifies how to select the single best route. OSPF specifically defines procedures for handling equal cost multipath (ECMP) [cite OSPF]. The same technique has been applied to ISIS. A similar technique has been used with BGP. Variations exist but the decision to support BGP multipath, the specific variation of BGP multipath, or not to support it, does not affect interoperability.

A naive implementations of ECMP can cause severe performance degradation for TCP flows. To avoid this, implementations of BGP multipath SHOULD maintain packet ordering within microflows as described in [cite rfc2991, rfc2992].

BGP multipath, if implemented, SHOULD be disabled by default.

In addition to IGP multipath (OSPF ECMP and ISIS equivalent), there are two variations of BGP multipath described here. A BGP implementation may offer both, either one, or neither variation of BGP multipath. Other variations of BGP multipath may exist, but no guarantees can be made in this protocol specification of their properties or impact on interoperability.

Where IGP multipath is used, there is an interaction with BGP learned routes. The lookup of a BGP NEXT_HOP in the IGP can result in the selection of an IGP multipath entry. This is not a variation of BGP multipath. When this occurs, one BGP route is selected as the best but there is more than one way to reach the BGP NEXT_HOP via the IGP.

In one variation of BGP multipath, a set of more than one IBGP routers peering with the same external AS have equal routes to a destination and are an equal IGP cost away from a second set of one or more routers. BGP multipath is applicable to the latter set of routers. Without BGP multipath, BGP would pick the BGP NEXT_HOP of the advertisement from only one of those IBGP peers (using BGP Identifier) and use only that BGP route. With BGP multipath, BGP uses the BGP NEXT_HOP of more than one of these equal cost advertisements, yielding more than one BGP NEXT_HOP. Each BGP NEXT_HOP has a different IGP next hop (one or more IGP next hop if IGP multipath is in use).

The second case is where all of the candidates routes for BGP multipath are external and learned by a single BGP peer. Without BGP multipath this peer would select only one of the BGP routes and obtain only one BGP NEXT_HOP. With BGP multipath, more than one equal cost route is selected yielding more than one BGP NEXT_HOP. Seldom does IGP multipath come into play when looking up an EBGP NEXT_HOP but could in principle be applicable.

If in EBGP multipath traffic is split among routers in difference AS, an aggregate SHOULD be formed so as to propagate a route with an accurate AS_PATH. If the resulting aggregate is not more specific than the components, the AS_SET SHOULD NOT be dropped.

The decision point for multipath is after step "d" in Section 9.1.2.2 (prefer externally learned routes). IBGP learned and EBGP learned routes MUST NOT be combined in multipath. If the multipath decision is prior to "d", then two routers each with an external peering would form a routing loop.

The decision point for multipath is generally after step "e" in Section 9.1.2.2. Some relaxation of the "equal cost" rule (also applicable to IGP multipath) is possible. In addition to the equal cost BGP NEXT_HOPS available at BGP route selection, if the IGP next

hop for other BGP NEXT_HOPs are of lower cost, then those may be used as well. This relaxation of the step "e" is possible but is not widely implemented (and may not be implemented at all).

The consensus of the majority of the IDR WG is to keep this in a separate draft and out of the base spec.

2.12. TCP Behavior Wording

Status: Consensus
Change: No
Summary: In issue 19 we decided to remove this section entirely. As a result the previous consensus on this issue (no change) is needed moot.

Discussion: The subject-less "your mail" thread discussed a wording clarification from:

"An implementation that would "hang" the routing information process while trying to read from a peer could set up a message buffer (4096 bytes) per peer and fill it with data as available until a complete message has been received. "

To something that is more TCP-correct, such as:

"An implementation that would "hang" the routing information process while trying to received from a peer could set up a message buffer (4096 bytes) per peer and fill it with data as available until a complete message has been received. "

(only change: "read" to "received" This was one of a couple of suggested changes.)

This suggestion was quite contentious, and although there were a variety of alternate texts proposed, the only consensus was that this was a very minor issue, and probably not worth changing.

In issue 19 we decided to remove this section entirely.

2.13. Next Hop for Originated Route

Status: Consensus
Change: No
Summary: No responses, assumed consensus to keep things the same.

Discussion:

There was a one-message thread entitled "next hop for originated

route". This message received no response, so the assumption is that there is a consensus to keep things as they are.

For related discussion see issue 61.

2.14. NEXT_HOP to Internal Peer

Status: Consensus
Change: No
Summary: Closed in favor of issue 61.

Discussion:

The thread entitled "NEXT_HOP to internal peer" starts with this question:

When sending a locally originated route to an internal peer, what should NEXT_HOP be set to?

One response suggested that we add a line stating that the NEXT_HOP address originates from the IGP.

Since this issue and issue 61 are basically the same, except 61 proposes text, we'll close this issue in favor of 61.

2.15. Grammar Fix

Status: Consensus
Change: Yes
Summary: Change: "The Prefix field contains IP address prefixes ..."
To: "contains an IP address prefix ..."

Discussion: The thread entitled "Review comment: bottom of page 16" corrects a grammar mistake by suggesting we change:

"The Prefix field contains IP address prefixes ..."

to:

"contains an IP address prefix ..."

Yakov responded that this will be fixed in -18.

The consensus seems to be to correct this, and go with the new text.

2.16. Need ToC, Glossary and Index

Status: Consensus
Change: Yes
Summary: Need to add a Table of Contents (ToC), Glossary and Index to the draft. Will be added in draft -18.

Discussion:

The "Review Comments: draft-ietf-idr-bgp4-17.txt" thread suggests:

1. Document needs, Table of Contents, Glossary, and Index

2. Paths, Routes, and Prefixes need to be defined in the spec early on (like in a glossary), so it is obvious what is implied.

Yakov responded that draft -18 will have a ToC and definition of commonly used terms.

2.17. Add References to other RFC-status BGP docs to base spec

Status: Consensus
Change: Yes
Summary: Add references to other RFC-status BGP docs to the base spec.

Discussion:

The "Review Comments: draft-ietf-idr-bgp4-17.txt" thread then changes titles to: "Review of draft-ietf-idr-bgp4-17.txt" and goes on to suggest:

3. All BGP Extensions described in other documents that made it to RFC status should be at least referenced in the Reference section P.64. This is justifiable since it's the core BGP standard spec.

Yakov responded that this will be added to the -18 review.

Jonathan agreed.

2.18. IP Layer Fragmentation

Status: Consensus
Change: No
Summary: No need to mention IP Layer Fragmentation in the BGP specification, since this is taken care of at the TCP level.

Discussion:

1. P.6 section 4. Message Formats, its possible for the source BGP peer IP layer to fragment a message such that the receiving BGP peer socket layer would have to reassemble it. Need to mention this, since BGP implementations are required to do this.

The response to this was that, while true, reassembly is something that is inherent in the TCP layer that BGP rides over. Therefore, this is something that is in the TCP spec, and needn't be repeated in the BGP spec. This comment was reaffirmed. There seems to be consensus that this isn't something that needs to be in the BGP spec.

2.19. Appendix Section 6.2: Processing Messages on a Stream Protocol

Status: Consensus
Change: Yes
Summary: Remove the section entirely, as this is something that does not belong in the base spec.

Discussion:

This first came up in response to Issue 17:

There was one comment suggesting that section 6.2 (Processing Messages on a Stream Protocol" mentioned this.

The original reviewer responded that the out-of-scope comment was out-of-place and referred the responder to section 6.2 (appendix 6)

The original reviewer stated that he is happy with just adding a reference to section 6.2 in appendix 6 and leaving it at that.

Curtis suggested we just add a reference to Stevens in appendix 6. 6.2 and be done with it. Specifically:

6.2 Processing Messages on a Stream Protocol

BGP uses TCP as a transport mechanism. If you are unsure as to how to handle asynchronous reads and writes on TCP sockets please refer to Unix Network Programming [RWStevens] or other introductory text for programming techniques for the operating system and TCP implementation that you are using.

There were further suggestions to remove the section entirely as out-of-scope. At least 3 people agreed with this.

Alex responded that he sees no reason to remove it, but wouldn't have a problem if the WG decides to do so.

There seems to be general agreement that this section should be removed.

N.B. This also affects issue 12.

2.20. Wording fix in Section 4.3

Status: Consensus
Change: Yes
Summary: A small change for clarity in section 4.3

Discussion:

This suggestion grew out of the discussion on Issue 18.

The following change was suggested in section 4.3, second line of the first paragraph:

s/UPDATE packet/UPDATE message/

Yakov agreed to this change and updated the draft.

2.21. Authentication Text Update

Status: Consensus
Change: No
Summary: The consensus is that additional references to RFC2385 are not necessary.

Discussion:

P. 10, "Authentication Data:" section you might want to add this, It is also possible to use MD5 (RFC2385) at the transport layer to validate the entire BGP message.

Yakov replied to this:

There is already text that covers this:

"Any authentication scheme used by TCP (e.g., RFC2385 [RFC2385]) may be used in addition to BGP's own authentication mechanisms."

....

"In addition, BGP supports the ability to authenticate its data stream by using [RFC2385]."

So, I see no need to add the text proposed above.

Ishi agreed with Yakov. Jonathan disagreed since he thought no one uses BGP auth. Ishi replied that there are lots of people who do use it. Jonathan replied with a clarification question: "Who uses *BGP's own* authentication mechanisms???" Ron Bonica replied that they use BGP auth. There was some additional discussion over who implements simple password authentication vs. MD5.

After further discussion, the consensus seems to be that we should leave the text as it is for the reasons Yakov pointed out. There was some discussion over opening a new issue to discuss deprecating the BGP auth mechanism discussed in RFC1771 in favor of the mechanism in RFC2385.

The issue of Deprecating BGP AUTH is discussed in issue 62.

2.22. Scope of Path Attribute Field

Status: Consensus
Change: Yes
Summary: This is already being covered by text that has been added to the -18 draft.

Discussion:

P. 12, right after "Path Attributes". The following sentence should be added to this section to clarify the scope of the Path Attribute field. "All attributes in the Path Attribute field represent the characteristics of all the route prefixes defined in the NLRI field of the message".

Yakov replied to this that:

This will be covered by the following text in 3.1 that will be in the -18 version (see also issue 54).

Routes are advertised between BGP speakers in UPDATE messages. Multiple routes that have the same path attributes can be advertised in a single UPDATE message by including multiple prefixes in the NLRI field of the UPDATE message.

Therefore there is no need to add the sentence proposed above.

There were no objections to this statement, so this issue has been moved into consensus.

2.23. Withdrawn and Updated routes in the same UPDATE message

Status: Consensus
Change: No
Summary: For various reasons, not least of which is compatibility with existing implementations, the decision was made to keep thing the way they are.

Discussion:

4. P.16, last paragraph in section 4.3 as stated, "An UPDATE message should not include the same address prefix in the WITHDRAWN ROUTES and Network Layer Reachability Information fields, however a BGP speaker MUST be able to process UPDATE messages in this form. A BGP speaker should treat an UPDATE message of this form as if the WITHDRAWN ROUTES doesn't contain the address prefix."

This complexity could have been avoided if withdrawn routes and NLRI prefixes with their attributes were mutually exclusive of each other and appeared in different update messages. If that was the case, the priority of which field to process first would have been as simple as using "first come, first served" message processing approach.

Yakov commented that this would make the case where they are both in the same message unspecified.

John commented that this is something we don't want to change this late in the game. Although it was acknowledged that this might be a good change if we were working from a clean slate.

Ben acceded that this was somewhat wishful thinking on his part.

Curtis's comment seems to coincide with this message, stating:

The existing rules are very clear.

Summarized:

If an UPDATE contains only a withdraw for a prefix, then withdraw whatever route the peer had previously sent.

If an UPDATE contains the prefix only in the NLRI section, replace whatever route had previously been advertised by the peer or add a route if there was no previous route, in both cases adding a route with the current attributes.

Don't put the same prefix in the same in both the withdraw and NLRI section of the same update.

If you receive an UPDATE with the same prefix in both the withdraw and NLRI, ignore the withdraw. [Some older implementations thought this was a good way to say "delete then add".]

Process UPDATEs from the same peer in the order received.

And goes on to say, that to him, these rules are clear from the existing text.

Consensus is that while this would be nice, we need to stick with what we have, and move on.

2.24. Addition or Deletion of Path Attributes

Status: Consensus
Change: Yes
Summary: Add the following to section 3.1:

Changing the attributes of a route is accomplished by advertising a replacement route. The replacement route carries new (changed) attributes and has the same NLRI as the original route.

Discussion:

5. P. 20 Its not stated how we delete or modify Path Attributes associated with NLRI prefixes.

A response to this comment said that this is implicit in the definition of "route" and the general withdraw/replace behavior and therefore doesn't need to be repeated.

Ben responded saying that, while there was an assumption, there was no well defined mechanism, and this leads to ambiguity.

John responded, no need to define everything explicitly, or we'll be here forever.

Picking this thread up again, Yakov argued:

By *definition* a route is a <path attribute, NLRI> pair. From that definition it follows that changing one or more path attributes of a route means changing a route, which means withdrawing the old route (route with the old attributes) from service and advertising a new route (route with the new attributes). Procedures for doing this are well-defined (see section 3.1), and therefore no new text to cover this is needed.

Jonathan agreed with this statement, but Ben argued that the text in

section 3 is insufficient the way it is currently written. After two iterations, Ben and Yakov agreed on this formulation for an update to section 3.1:

Changing the attributes of a route is accomplished by advertising a replacement route. The replacement route carries new (changed) attributes and has the same NLRI as the original route.

Jeff objected somewhat to the wording, since, because of a bgp route is defined as a <path attribute, NLRI> pair, changing either part of that pair, by definition, changes the route. He acknowledged that this might fall under the category of implementation detail.

Yakov presented the view that he thought we were at consensus with the text he proposed above. Jonathan agreed. There were no objections, so this is moved to Consensus.

2.25. NEXT_HOP Semantics

Status: Consensus
Change: No
Summary: After responders pointed out another sentence, this comment was resolved. Things will stay the way they are.

Discussion:

1. P.28, 2nd to last paragraph. The line that reads, "To be semantically correct, the IP address in the NEXT_HOP must not be the IP address of the receiving speaker, and the NEXT_HOP IP address must either be the sender's IP address (used to establish the BGP session), or the interface associated with the NEXT_HOP IP address must share a common subnet with the receiving BGP speaker..."

This is not always true, what if the current ASBR BGP router is advertising an external AS route (to a IBGP Peer) whose NEXT_HOP IP address is the IP address of the EBGP peer in the other AS?

A response to this pointed out that right before this is a sentence stating that this only applied to eBGP links, and only when the peers are one hop from each other, so a modification is unnecessary. This response was confirmed with another.

The original reviewer acknowledged this and withdrew the comment.

The consensus is to leave things the way they are.

2.26. Attributes with Multiple Prefixes

Status: Consensus
Change: No
Summary: After some discussion, the consensus is to keep things the same since the suggested behavior is defined in the message format.

Discussion:

2. P. 29, Section 6.3. Add this rule near the attribute rules. "Multiple prefixes that require the same attribute type with different values must never appear in the same update message".

A response to this suggested that this text is unnecessary since this behavior is ruled out by the way the message format is defined.

The original commenter agrees with the responder. The consensus is to leave things the way they are.

2.27. Allow All Non-Destructive Messages to Refresh Hold Timer

Status: Consensus
Change: No
Summary: It is agreed that this is a change that exceeds the original goal of this draft revision. This goal is to document existing practice in an interoperable way.

Discussion:

3. P. 29, Section 6.5, Please rewrite this sentence from: "If a system does not receive successive KEEPALIVE and/or UPDATE and/or NOTIFICATION messages within the period specified in the Hold Time field of the OPEN message ..."

To This: "If a system does not receive successive KEEPALIVE and/or UPDATE and/or any other BGP message within the period specified in the Hold Time field of the OPEN message ..."

There is disagreement on this change. It has been discussed in other threads.

The original commenter acknowledged that this is something that would be "adding a new feature" as opposed to the stated goal of "documenting what exists." He suggested that the ADs decide if we should open the door for new features or not.

Yakov replied to this that he would suggest we keep things as is, since the purpose is to document current implementations.

This did not meet with any objections, so this issue has been moved into consensus.

2.28. BGP Identifier as Variable Quantity

Status: Consensus
Change: No
Summary: The consensus is that changing the BGP Identifier in the base draft is out-of-scope at this point in the draft evolution.

Discussion:

4. P. 31, section 6.8, Please rewrite this sentence from: "Comparing BGP Identifiers is done by treating them as (4-octet long) unsigned integers."

To This: "Comparing BGP Identifiers is done by treating them as large numbers based on their IP Address type (e.g. IPv4, IPv6, etc.)."

A response to this was that since BGP Identifier is defined in the base spec as a 4 byte unsigned integer, and not a variable quantity, the sentence as written is acceptable. This was also confirmed by another response.

The original commenter was thinking of IPv6, and providing sufficient space to allow a full v6 address to be used.

Again, responders said that this is out-of-scope for the current draft.

2.29. State Why Unresolveable Routes Should Be Kept in Adj-RIB-In

Status: Consensus
Change: Yes
Summary: Add:

"in case they become resolvable" after the last sentence on p. 46.

Discussion:

5. P.46, last sentence, "However, corresponding unresolvable routes SHOULD be kept in the Adj-RIBs-In." It would helpful if the author states why unresolvable routes should be kept in Adj-RIBs-In?

A response to this stated "In case they become resolvable"

Yakov responded that:

I suggest we add "in case they become resolvable" after the last sentence on p. 46.

The original commenter stated that: Then the point that the peer will not refresh the route if we drop them (unless we use Route Refresh) because they are unreachable should be made.

Yakov also responded that:

This should be clear from the following text in Section 3:

The initial data flow is the portion of the BGP routing table that is allowed by the export policy, called the Adj-Ribs-Out (see 3.2). Incremental updates are sent as the routing tables change. BGP does not require periodic refresh of the routing table.

Jonathan, who was the original commenter, agreed with both the changed text and the clarity of section 3.

2.30. Mention Other Message Types

Status: Consensus
Change: Yes
Summary: Add a reference to RFC2918 at the end of the type code list.

Discussion:

1. P. 7 Type: Need to add the new message types such as, Capability Negotiations (RFC2842), Route Refresh, etc.

One response argued that these are out-of-scope of the base document. One response agreed, but thought that it should be capability and not message type. The original commenter responded about Message type from the capability draft.

Sue mentioned this would be added in the second round.

Yakov replied that:

The only new message type that is covered by an RFC (rather than just an Internet Draft) is the Refresh message. With this in mind how about replacing the following:

The following type codes are defined:

1 - OPEN 2 - UPDATE 3 - NOTIFICATION 4 - KEEPALIVE

with

This document defines the following type codes:

1 - OPEN 2 - UPDATE 3 - NOTIFICATION 4 - KEEPALIVE

[RFC2918] defines one more type code.

Jonathan agreed with this change. This issue has been moved to consensus.

2.31. Add References to Additional Options

Status: Consensus
Change: Yes
Summary: Consensus to add:

[RFC2842] defines another Optional Parameter.

Discussion:

2. P. 9, right after "This document defines the following optional parameters:" Need to mention possible options, such as: Capabilities (RFC2842), Multiprotocol extensions (RFC2858), Route Refresh (RFC2918).

One response agreed that adding references would be fine. A second response agreed.

Yakov replied that:

Please note that only rfc2842 defines an OPEN optional parameter. Neither rfc2858 nor rfc2918 defines an OPEN optional parameter.

With this in mind I would suggest to add the following text:

[RFC2842] defines another Optional Parameter.

The original poster agreed with this modification. This issue is at consensus.

2.32. Clarify EGP Reference

Status: Consensus
Change: No
Summary: The consensus is that this was addressed in 32.1, so we can close this.

Discussion:

3. P. 13, EGP, are there other EGP protocols other than BGP that are in use? If not, change EGP to BGP.

A response to this suggested that we add a reference to [1] (the EGP spec) here.

Another response clarified that this refers to EGP-the-protocol and NOT the class.

Another response disagreed, but suggested that:

IGP = network was explicitly introduced into bgp (network cmd)
INCOMPLETE = network was implicitly introduced into bgp
(redistribute) EGP = other

The original commenter thought that this referred to EGP-the-class of protocols. And why not use BGP therefore, as the only EGP.

There was some discussion over whether or not we should mention something that is historical.

Jeff suggested a footnote in the Origin section about EGP.

Curtis suggested that we state that the EGP in ORIGIN is deprecated, but retain the value to document what it used to mean.

This reviewer thinks a statement about whether this "EGP" origin refers to the protocol or the class or protocols would be useful.

Yakov replied that an EGP reference will be added (see issue 9).

Yakov also stated that he doesn't see what is wrong with the current text, and suggested we keep it. This includes leaving out any reference to the status of the EGP spec. He sees that it is clear from context that we are talking about "the EGP" [RFC904].

Jeff noted that this issue has been sufficiently addressed in the solution to 32.1. This met with agreement. We are at consensus.

2.32.1. EGP ORIGIN Clarification

Status: Consensus
Change: Yes
Summary: Change section 5.1.1 to read:

ORIGIN is a well-known mandatory attribute. The ORIGIN attribute shall be generated by the speaker that originates the associated routing information. Its value SHOULD NOT be changed by any other

speaker."

Consensus to change:

1 EGP - Network Layer Reachability Information learned via the EGP protocol

to:

1 EGP - Network Layer Reachability Information learned via the EGP protocol [RFC904]

Discussion:

This discussion is picked up again in the "Review of draft-ietf-idr-bgp4-17" thread, where specific text is proposed:

Old:

"ORIGIN is a well-known mandatory attribute that defines the origin of the path information. The data octet can assume the following values:

Value Meaning

0 IGP - Network Layer Reachability Information is interior to the originating AS

1 EGP - Network Layer Reachability Information learned via the EGP protocol

2 INCOMPLETE - Network Layer Reachability Information learned by some other means" New:

"ORIGIN is a well-known mandatory attribute that defines the origin of the path information. The data octet can assume the following values:

Value Meaning

0 IGP - NLRI was explicitly introduced into bgp

1 EGP - this value was administratively configured to affect policy decisions or NLRI was learned via the EGP protocol [1]

2 INCOMPLETE - NLRI was implicitly introduced into bgp"

since: 1) The network command sets the origin to IGP and I remember

seeing somewhere that only static routes should be set to IGP. 2) The primary use of EGP value is policy 3) EGP seems to still exist, anyway even if it does not it is not worth re-writing the world.

Also, change: "5.1.1 ORIGIN

ORIGIN is a well-known mandatory attribute. The ORIGIN attribute shall be generated by the autonomous system that originates the associated routing information. It shall be included in the UPDATE messages of all BGP speakers that choose to propagate this information to other BGP speakers."

to: "5.1.1 ORIGIN

The value of the ORIGIN attribute shall be set by the speaker that originates the associated NLRI. Its value shall not be changed by any other speaker unless the other speaker is administratively configured to do so to affect policy decisions."

since: 1) It is already defined as well-known mandatory attribute. 2) It may be set differently within the same AS (not saying this is good). 3) It is commonly used for policy, but by default does not get changed. 4) Speakers have no choice, it is mandatory.

After much continued discussion on this in the "issue 32.1" thread, we seem to have come to a consensus that section 5.1.1 should read:

ORIGIN is a well-known mandatory attribute. The ORIGIN attribute shall be generated by the speaker that originates the associated routing information. Its value should not be changed by any other speaker unless the other speaker is administratively configured to do so to affect policy decisions."

This text met with a number of agreements, and one disagreement stating that we shouldn't have the "unless administratively configured" portion.

After some further discussion, we have this text on the table:

ORIGIN is a well-known mandatory attribute. The ORIGIN attribute is generated by the BGP speaker that originates the associated BGP routing information. The attribute shall be included in the UPDATE messages of all BGP speakers that choose to propagate this information to other BGP speakers.

Jonathan suggested that we change "propagate this information" to "forward this route". He also mentioned that he would prefer something more explicit instead of/in addition to "The attribute

shall be included in the UPDATE messages of all BGP speakers that choose to propagate this information to other BGP speakers." such as "other speakers do not change the ORIGIN value."

On the issue of making the EGP ORIGIN type more clear Andrew proposed:

To me, there seems to be sufficient confusion around the "EGP" reference to merit some sort of clarification. The simplest modification would be to change:

1 EGP - Network Layer Reachability Information learned via the EGP protocol

to:

1 EGP - Network Layer Reachability Information learned via the EGP protocol [RFC904]

That would clarify that we're talking about the protocol, and not the class-of-protocols, or EBGP. It would leave unstated that this could theoretically be used to muck with route selection. I think that is ok. If operators want to override ORIGIN to affect some hoho magic, they are welcome to do so, but I don't think it needs to be documented in the base spec.

This met with a number of agreements.

On the second text section we are working on, Jonathan objected to the current working text below and suggested an alternate:

CHANGE:

"ORIGIN is a well-known mandatory attribute. The ORIGIN attribute is generated by the BGP speaker that originates the associated BGP routing information. The attribute shall be included in the UPDATE messages of all BGP speakers that choose to propagate this information to other BGP speakers."

TO:

"ORIGIN is a well-known mandatory attribute. The ORIGIN attribute shall be generated by the speaker that originates the associated routing information. Its value should not be changed by any other speaker unless the other speaker is administratively configured to do so to affect policy decisions."

-or-

"ORIGIN is a well-known mandatory attribute. The ORIGIN attribute shall be generated by the speaker that originates the associated routing information. Its value should not be changed by any other speaker."

Jonathan cited a recent example of someone who was still confused by this section of the text in -17 (not specifically the working text).

Yakov proposed this as final text:

In 4.3:

a) ORIGIN (Type Code 1):

ORIGIN is a well-known mandatory attribute that defines the origin of the path information. The data octet can assume the following values:

Value Meaning

0 IGP - Network Layer Reachability Information is interior to the originating AS

1 EGP - Network Layer Reachability Information learned via the EGP protocol [RFC904]

2 INCOMPLETE - Network Layer Reachability Information learned by some other means

Usage of this attribute is defined in 5.1.1.

In 5.1.1:

ORIGIN is a well-known mandatory attribute. The ORIGIN attribute shall be generated by the speaker that originates the associated routing information. Its value SHOULD NOT be changed by any other speaker."

This met with agreement. This issue is at consensus.

2.32.2. BGP Destination-based Forwarding Paradigm

Status: Consensus
Change: Yes
Summary: After much discussion, this is the consensus: This text in the current draft:

To characterize the set of policy decisions that can be enforced

using BGP, one must focus on the rule that a BGP speaker advertises to its peers (other BGP speakers which it communicates with) in neighboring ASs only those routes that it itself uses. This rule reflects the "hop-by-hop" routing paradigm generally used throughout the current Internet. Note that some policies cannot be supported by the "hop-by-hop" routing paradigm and thus require techniques such as source routing (aka explicit routing) to enforce. For example, BGP does not enable one AS to send traffic to a neighboring AS intending that the traffic take a different route from that taken by traffic originating in the neighboring AS. On the other hand, BGP can support any policy conforming to the "hop-by-hop" routing paradigm. Since the current Internet uses only the "hop-by-hop" inter-AS routing paradigm and since BGP can support any policy that conforms to that paradigm, BGP is highly applicable as an inter-AS routing protocol for the current Internet.

will be replaced in -18 with the following text:

Routing information exchanged via BGP supports only the destination-based forwarding paradigm, which assumes that a router forwards a packet based solely on the destination address carried in the IP header of the packet. This, in turn, reflects the set of policy decisions that can (and can not) be enforced using BGP. Note that some policies cannot be supported by the destination-based forwarding paradigm, and thus require techniques such as source routing (aka explicit routing) to be enforced*. Such policies can not be enforced using BGP either. For example, BGP does not enable one AS to send traffic to a neighboring AS for forwarding to some destination (reachable through but) beyond that neighboring AS intending that the traffic take a different route to that taken by the traffic originating in the neighboring AS (for that same destination). On the other hand, BGP can support any policy conforming to the destination-based forwarding paradigm.

Discussion:

In response to these proposals, Yakov proposed that the real problem is that it is not clear that BGP is build to support the destination-based forwarding paradigm. To fix this, it was proposed that:

<OLD>

To characterize the set of policy decisions that can be enforced using BGP, one must focus on the rule that a BGP speaker advertises to its peers (other BGP speakers which it communicates with) in neighboring ASs only those routes that it itself uses. This rule reflects the "hop-by-hop" routing paradigm generally used throughout the current Internet. Note that some policies cannot be supported by

the "hop-by-hop" routing paradigm and thus require techniques such as source routing (aka explicit routing) to enforce. For example, BGP does not enable one AS to send traffic to a neighboring AS intending that the traffic take a different route from that taken by traffic originating in the neighboring AS. On the other hand, BGP can support any policy conforming to the "hop-by-hop" routing paradigm. Since the current Internet uses only the "hop-by-hop" inter-AS routing paradigm and since BGP can support any policy that conforms to that paradigm, BGP is highly applicable as an inter-AS routing protocol for the current Internet.

<NEW>

Routing information exchanged via BGP supports only the destination-based forwarding paradigm, which assumes that a router forwards a packet based solely on the destination address carried in the IP header of the packet. This, in turn reflects the set of policy decisions that can (and can not) be enforced using BGP. Note that some policies cannot be supported by the destination-based forwarding paradigm and thus require techniques such as source routing (aka explicit routing) to enforce. Such policies can not be enforced using BGP either. For example, BGP does not enable one AS to send traffic to a neighboring AS intending that the traffic take a different route from that taken by traffic originating in the neighboring AS. On the other hand, BGP can support any policy conforming to the destination-based forwarding paradigm.

Curtis thinks the newer text here is more clear.

In response to the new text, Christian Martin proposed a slightly different new text:

Routing information exchanged via BGP supports only the destination-based forwarding paradigm, which assumes that a router forwards a packet based solely on the destination address carried in the IP header of the packet. This, in turn reflects the set of policy decisions that can (and can not) be enforces using BGP. Note that some policies cannot be supported by the destination-based forwarding paradigm and thus require techniques such as source routing (aka explicit routing) to enforce. Such policies can not be enforced using BGP either. For example, BGP does not enable one AS to send traffic to a neighboring AS based on prefixes originating from the local AS. On the other hand, BGP can support any policy conforming to the destination-based forwarding paradigm.

To which Yakov replied:

Routing information exchanged via BGP supports only the destination-

based forwarding paradigm, which assumes that a router forwards a packet based solely on the destination address carried in the IP header of the packet. This, in turn, reflects the set of policy decisions that can (and can not) be enforces using BGP. Note that some policies cannot be supported by the destination-based forwarding paradigm, and thus require techniques such as source routing (aka explicit routing) to enforce. Such policies can not be enforced using BGP either. For example, BGP does not enable one AS to send traffic through a neighboring AS to some destination (which is outside of the neighboring AS, but is reachable through the neighboring AS) intending that the traffic take a different route from that taken by the traffic to the same destination that originating in the neighboring AS. On the other hand, BGP can support any policy conforming to the destination-based forwarding paradigm.

And Chris responded:

Routing information exchanged via BGP supports only the destination-based forwarding paradigm, which assumes that a router forwards a packet based solely on the destination address carried in the IP header of the packet. This, in turn, reflects the set of policy decisions that can (and can not) be enforces using BGP. Note that some policies cannot be supported by the destination-based forwarding paradigm, and thus require techniques such as source routing (aka explicit routing) to enforce. Such policies can not be enforced using BGP either. For example, BGP does not enable one AS to send traffic through a neighboring AS to some destination beyond the neighboring AS intending that the traffic take a different route from that taken by traffic to the same destination which originates in the neighboring AS. In other words, the BGP policy of a local AS cannot affect the downstream (aka, away from the local AS) forwarding policy of a remote AS. On the other hand, BGP can support any policy conforming to the destination-based forwarding paradigm.

Tom Petch preferred Yakov's second formulation, with these changes:

policies can not be enforced using BGP either. For example, BGP does not enable one AS to send traffic ! to a neighboring AS for forwarding to some destination (reachable through but) beyond ! that neighboring AS intending that ! the traffic take a different route to that taken by the traffic ! originating in the neighboring AS (for that same destination).

On the other hand, BGP can support any policy conforming to the destination-based forwarding paradigm.

Yakov agreed to Tom's suggested changes.

2.33. Add "Optional Non-Transitive" to the MED Section

Status: Consensus
Change: Yes
Summary: Add "Optional Non-Transitive" to MED Section Add "well-known mandatory" to the NEXT_HOP Section

Discussion:

4. P.23, change the following:

"The MULTI_EXIT_DISC attribute may be used on external (inter-AS) links to discriminate among multiple exit or entry points to the same neighboring AS ..."

To the following:

"The MULTI_EXIT_DISC is an optional non-transitive attribute which may be used on external (inter-AS) links to discriminate among multiple exit or entry points to the same neighboring AS ..."

A responder disagreed, and stated reasons "covered elsewhere" Original commenter asked for reasons, since the modification seemed obvious to him.

Yakov agreed to make this change in -18.

Jonathan replied that:

5.1.3 NEXT_HOP also, it is missing " well-known mandatory".

Yakov also agreed to make this change.

2.34. Timer & Counter Definition

Status: Consensus
Change: No
Summary: No discussion, no text proposed, defaults to consensus for no change.

Discussion:

5. In section 8, there are a number of Timers, Counters, etc. that need to be explicitly defined before they are used by the FSM. Perhaps these definitions should go in the Glossary section.

There has been no further discussion on this issue. Unless it is brought up again, this issue is in consensus, with no change.

2.35. Fix Typo

Status: Consensus
Change: Yes
Summary: Fix a Typo. No discussion, but this seem clear.

Discussion:

1. P. 41. Typing error, "Each time time the local system...".

2.36. Add Adj-RIB-In, Adj-RIB-Out and Loc-RIB to the Glossary

Status: Consensus
Change: Yes
Summary: This change requires a glossary. Yakov has committed to having a section where commonly used terms are defined in draft 18, so this issue is at consensus.

Discussion:

2. Section 9.1, Need to have Adj-RIB-In, Adj-RIB-Out, and Loc-RIB in the glossary, so when they are used in section 9.1, it is well understood what they are.

Yakov replied:

will be added to the section "Definition of commonly used terms" in -18 version.

2.37. Combine "Unfeasible Routes" and "Withdrawn Routes"

Status: Consensus
Change: Yes
Summary: Add the following terms to the "commonly used terms section":

Feasible route A route that is available for use.

Unfeasible route A previously advertised feasible route that is no longer available for use.

Discussion:

3. P. 45, Phase I, There is no definition of what are unfeasible routes? Are they the same as withdrawn routes? If so, the two should be combined to one name.

Ishi replied to this that he thought that we could combine the two

terms, since there is limited difference from an implementation standpoint.

Yakov replied:

The routes are withdrawn from service because they are unfeasible, not because they are "withdrawn". So, we need to keep the term "unfeasible" to indicate the *reason* why a route could be withdrawn. On the other hand, "withdrawn" is used as a verb, and to the best of my knowledge "unfeasible" can't be used as a verb. With this in mind, I don't think that we can combine the two into a single term.

Ishi replied that he was convinced, and that the terms should stay separate.

Andrew asked the list if we should define these terms in the "commonly used terms" section in draft -18.

Ben replied that if we use them a lot, we should define them, and if not local definitions will suffice.

There was some back and forth about the necessity of defining terms which should be obvious.

mrr actually checked the doc to see if we were consistently using the terms, and found:

It turns out there there is an inconsistency in the usage of the word withdrawn.

Section 3.1:

There are three methods by which a given BGP speaker can indicate that a route has been withdrawn from service:

...

b) a replacement route with the same NLRI can be advertised, or

...

Later, in the definition of Withdrawn Routes Length, we have:

A value of 0 indicates that no routes are being withdrawn from service,

Taken together, this could be construed as meaning that a Withdrawn Routes Length of 0 indicates that all routes included in the UPDATE

represent newly feasible routes... not replacement routes.

Now, it's possible that this problem has been removed by changes to the text that have not yet been incorporated in to a new draft; however, it arose because the text, for the most part, does _not_ use "withdrawn" in the standard way. Instead, it refers to routes included in the WITHDRAWN ROUTES field of an UPDATE message. Consequently, I propose defining a "withdrawn route" as follows:

Withdrawn route: a route included in the WITHDRAW ROUTES field of an UPDATE message.

Regardless of whether or not this definition is included, Section 3.1 should be changed from:

There are three methods by which a given BGP speaker can indicate that a route has been withdrawn from service:

to:

There are three methods by which a given BGP speaker can indicate that a route has been removed from service:

or:

There are three methods by which a given BGP speaker can indicate that a route is now unfeasible:

After some further off-list discussion, mrr agreed that this inconsistency is extremely minor, and withdrew his comment. feasible and unfeasible route will be defined in the "commonly used terms" section to clear up any confusion.

2.38. Clarify Outbound Route Text

Status: Consensus
Change: No
Summary: Consensus that the issue was sufficiently minor to leave things alone.

Discussion:

4. P. 50, line, "If a route in Loc-RIB is excluded from a particular Adj-RIB-Out the previously advertised route in that Adj-RIB-Out must be withdrawn from service by means of an UPDATE message (see 9.2)."

Would like to rephrase the sentence for clarity, "If a route in Loc-RIB is excluded from a particular Adj-RIB-Out and was previously

advertised via Adj-RIB-Out, it must be withdrawn from service by means of an UPDATE message (see 9.2)."

One comment suggested either leave it alone, or remove "via Adj-RIB-Out".

The original commenter withdrew the comment.

2.39. Redundant Sentence Fragments

Status: Consensus
Change: Yes
Summary: Fix typo & parentheses.

Discussion:

5. P. 50, section 9.1.4, The two fragments of this sentence are redundant and don't say anything new or simplify the content. Just keep one fragment.

"A route describing a smaller set of destinations (a longer prefix) is said to be more specific than a route describing a larger set of destinations (a shorted prefix); similarly, a route describing a larger set of destinations (a shorter prefix) is said to be less specific than a route describing a smaller set of destinations (a longer prefix)."

There was a comment that disagreed, thinking that both "more specific" and "less specific" need to be defined. And suggested that only the third and forth parentheses need to be dropped.

The original commenter agreed with the parentheses changes.

Yakov agreed to drop the third and fourth parentheses in the -18 version.

Jonathan replied to this:

Disagree, the text if fine the way it is, except you need to change "shorted" to "shorter".

After minimal further discussion, it was decided we are at a consensus on this issue to fix the typo and drop the third and fourth parentheses.

2.40. Section 9.2.1.1 - Per Peer vs. Per Router MinRouteAdvertisementInterval

Status: Consensus
Change: No
Summary: The consensus is that current practice allows for the MinRouteAdvertisementInterval to be set per peer, so the text should be kept the same.

Discussion:

6. P. 52, section 9.2.1.1 Change this sentence for clarity, "This rate limiting procedure applies on a per-destination basis, although the value of MinRouteAdvertisementInterval is set on a per BGP peer basis."

To This: "This rate limiting procedure applies on a per-destination basis, although the value of MinRouteAdvertisementInterval is set on a BGP router (same value for all peers) basis."

There was a comment disagreeing with this proposal. It was later elaborated on to include that the reason for disagreement was that the proposed changes changed the protocol and not just a practice clarification. Ben responded asking for how this is a protocol change, he saw it as a clarification. Perhaps there is something deeper that needs to be clarified? Again, response to this is that current implementations allow the MinRouteAdvertisementInterval to be set per-peer, not per-router.

Original reviewer conceded the point.

There was some additional discussion on this point. Most of it was along the lines of extracting what was really implemented and supported among various vendors. The conclusion was the same.

2.41. Mention FSM Internal Timers

Status: Consensus
Change: No
Summary: No discussion on this issue. No text proposed. Perhaps this is in the FSM section of the draft? Either way, it defaults to consensus with no change.

Discussion:

7. P. 61, item 6.4. Although all the BGP protocol interfacing timers are mentioned, there are a few FSM internal timers mentioned in the spec that need to be covered here as well.

There has been no discussion on this, it now defaults to consensus with no change.

2.42. Delete the FSM Section

Status: Consensus
Change: No
Summary: There was some confusion on the question: Is the FSM draft going to be a separate document, or incorporated into the base draft. The consensus is that it is going to become part of the base draft, so the FSM section will be kept, and elaborated on.

Discussion:

8. Since there is going to be an FSM spec, do we need to have FSM descriptions in this spec. Maybe the FSM section should be delete.

There was one response agreeing with this. One response asking for clarification: Was this a move to remove section 8. Finite State Machine from the base draft?? The original reviewer said, yes, when Sue's FSM draft becomes a WG document, we should remove section 8 from the base draft. Yakov asked that the AD's provide input on this suggestion.

Alex responded saying that the FSM draft is going to be part of the base spec, and not another document once the FSM words are approved.

2.43. Clarify the NOTIFICATION Section

Status: Consensus
Change: Yes
Summary: Replace:

"If a peer sends a NOTIFICATION message, and there is an error in that message, there is unfortunately no means of reporting this error via a subsequent NOTIFICATION message."

With:

If a peer sends a NOTIFICATION message, and the receiver of the message detects an error in that message, the receiver can not use a NOTIFICATION message to report this error back to the peer.

Discussion:

The "NOTIFICATION message error handling" thread proposed:

Please change" "If a peer sends a NOTIFICATION message, and there is

an error in that message, there is unfortunately no means of reporting this error via a subsequent NOTIFICATION message."

To: "If a peer receives a NOTIFICATION message, and there is an error in that message, there is unfortunately no means of reporting this error via a subsequent NOTIFICATION message."

This reversal of meaning met with disagreement, and this text was proposed instead:

All errors detected while processing the NOTIFICATION message cannot be indicated by sending subsequent NOTIFICATION message back to originating peer, therefore there is no means of reporting NOTIFICATION message processing errors. Any error, such as an unrecognized Error Code or Error Subcode, should be noticed, logged locally, and brought to the attention of the administration of the peer that has sent the message. The means to do this, however, lies outside the scope of this document.

The original posted agreed with the intent of the respondent's text, thought it was too wordy, but did not propose alternate text.

Yakov replied with this proposed text:

If a peer sends a NOTIFICATION message, and the receiver of the message detects an error in that message, the receiver can not use a NOTIFICATION message to report this error back to the peer.

Two responses liked this new text. Unless there are objections, we'll consider that a consensus.

2.44. Section 6.2: OPEN message error handling

Status: Consensus
Change: No
Summary: One commenter observed that the spec seems to specify behavior that doesn't seem to be observed by extant implementations, and suggested modifications to the spec. They were later reminded that the base behavior is acceptable, and agreed.

Discussion:

The "BGP4 draft ; section 6.2" thread began with a discussion of section 6.2: OPEN message error handling. Specifically:

"If one of the optional parameters in the Open message is not recognized, then the error subcode is set to 'unsupported optional parameters"

We have hit on this line when we were testing a BGP connection between a speaker that supported capability negotiation and a speaker that did not.

The speaker that did not support the negotiation closed down the peering session using the error clause mentioned above. Sometimes this lead to the other router to repeat the OPEN message with the Capability optional parameter; a game that went on for minutes.

This router manufacturer stated in a reply to this that : "One should not close down the connection if an optional parameter is unrecognized. That would make this parameter basically mandatory. This is an well known error in the BGP spec. Neither Cisco or Juniper do this"

If this is true it might be good to adapt the text.

The response to this quoted RFC2842, Capabilities Advertisement with BGP-4:

A BGP speaker determines that its peer doesn't support capabilities advertisement, if in response to an OPEN message that carries the Capabilities Optional Parameter, the speaker receives a NOTIFICATION message with the Error Subcode set to Unsupported Optional Parameter. In this case the speaker should attempt to re-establish a BGP connection with the peer without sending to the peer the Capabilities Optional Parameter.

The original poster responded:

This section from the Capabilities Advertisement RFC, is indeed inline with the section 6.2 of the BGP4 specification. For me however the question remains if most implementations do no simply ignore optional parameters that are unknown. And if so, if the text stated above reflects what is implemented by routers that do not have capability advertisement at all.

Yakov replied to this with:

RFC2842 assumes that a router (that doesn't implement RFC2842) would close the BGP session when the router receives an OPEN message with an unrecognized Optional Parameter. Therefore the text in the spec should be left unmodified.

The original poster, Jonathan, agreed with this. This issue moves to consensus.

2.45. Consistent References to BGP Peers/Connections/Sessions

Status: Consensus
Change: Yes
Summary: Stick with "BGP Connection" as the consistent term.

Discussion:

Ben proposed and Yakov responded:

> 1. Throughout the document we have various ways of naming the BGP
> peering communication. 1) BGP Session, 2) BGP Peering Session,

I'll replace "session" with "connection".

> 3) TCP Connection,

The spec doesn't name BGP peering communication as "TCP connection"; TCP connection is used to establish BGP connection. So, TCP connection and BGP connection are two different things.

> 4) BGP Connection,

The spec is going to use this term (see above).

> 5) BGP Peering Connection,

I'll replace "BGP peering connection" with "BGP connection".

> 6) Connection,

The text uses "connection" whenever it is clear from the context that it refers to "BGP connection" (or "TCP connection").

> 7) BGP Speaker Connection.

I'll replace "BGP Speaker Connection" with "BGP connection".

> > BGP router: 1) BGP Speaker, 2) speaker, 3)local speaker

The term "speaker" is used when it is clear from the context that we are talking about "BGP speaker".

> 2. Change Internal peer to IBGP Peer.

IBGP stands for "BGP connection between internal peers". Therefore the term "IBGP Peer" would mean "BGP connection between internal peers peer". That doesn't seem appropriate.

This issue has had some discussion, and section 3 was referenced, specifically:

Refer to Section 3 - Summary of operations which clearly states that " .. a peer in a different AS is referred to as an external peer, while a peer in the same AS may be described as an internal peer. Internal BGP and external BGP are commonly abbreviated IBGP and EBGP"

After more discussion it was decided that we should modify a paragraph on page 4 to read:

If a particular AS has multiple BGP speakers and is providing transit service for other ASs, then care must be taken to ensure a consistent view of routing within the AS. A consistent view of the interior routes of the AS is provided by the IGP used within the AS. For the purpose of this document, it is assumed that a consistent view of the routes exterior to the AS is provided by having all BGP speakers within the AS maintain IBGP with each other. Care must be taken to ensure that the interior routers have all been updated with transit information before the BGP speakers announce to other ASs that transit service is being provided.

This change has consensus.

> 3. Change External peer to EBGP Peer.

Ditto.

Alex responded that having explicit definitions would be nice. This ties into the general glossary suggestion (see issues 16, 34 & 36).

He also suggested that:

"BGP session" which works over a "TCP connection" would be closer to the terminology we're actually using now and would avoid possible confusions when people read terms like "Connection collision")

This was discussed in the "Generial Editorial Comment" thread.

After some further discussion, it was decided that, due to existing implementations, we should go with "BGP connection" as the consistent term. We are at consensus.

2.46. FSM Connection Collision Detection

Status: Consensus
Change: Yes
Summary: Add this to section 8:

There is one FSM per connection. Prior to determining what peer a connection is associated with there may be two connections for a given peer. There should be no more than one connection per peer. The collision detection identifies the case where there is more than one connection per peer and provides guidance for which connection to get rid of. When this occurs, the corresponding FSM for the connection that is closed should be disposed of.

Discussion:

The original reviewer (Tom) commented that the base draft, FSM section, could use some clarification around the area of connection collision detection. Specifically, he argued that it seems like there are actually 2 FSM's depending on which one backs off in the case of a collision. He proposed this text to clear things up:

"8 BGP Finite State Machine

This section specifies BGP operation - between a BGP speaker and its peer over a single TCP connection - in terms of a Finite State Machine (FSM). Following is a brief summary ... "(as before)

Instead of just

"This section specifies BGP operation in terms of a Finite State Machine (FSM). Following is a brief summary ... "(as before).

Curtis responded:

There is one FSM per connection. Prior to determining what peer a connection is associated with there may be two connections for a given peer. There should be no more than one connection per peer. The collision detection identifies the case where there is more than one connection per peer and provides guidance for which connection to get rid of. When this occurs, the corresponding FSM for the connection that is closed should be disposed of.

I'm not sure which document containing an FSM we should be reading at this point, but we could add the above paragraph if we need to explicitly state that the extra connection and its FSM is disposed of when a collision is detected.

When a TCP accept occurs, a connection is created and an FSM is created. Prior to the point where the peer associated with the connection is known the FSM cannot be associated with a peer. The collision is a transient condition in which the rule of "one BGP session per peer" is temporarily violated and then corrected.

This is discussed in the "FSM but FSM of what?" thread.

Sue responded that she would be happy to add Curtis' text to section 8 and solicited any additional comments. There was only one on capitalization, so this issue is at consensus.

2.47. FSM - Add Explicit State Change Wording

Status: Consensus
Change: No
Summary: A desire for explicit state change wording was expressed. No text was proposed. The assumption is that this issue has reached a happy conclusion.

Discussion:

The initial reviewer:

In most places, the actions taken on the receipt of an event include what the new state will be or that it remains unchanged. But there are a significant number of places where this is not done (eg Connect state events 14, 15, 16). I would like to see consistency, always specify the new/unchanged state. Else I may be misreading it.

There was a response asking for specific text, and offering to take the discussion private.

This is discussed in the "FSM words - state changes" thread.

There has been no further discussion on this. The assumption is that is has reached a happy conclusion privately.

2.48. Explicitly Define Processing of Incoming Connections

Status: Consensus
Change: Yes
Summary: Add text that is at the end of the discussion to section 8.

Discussion:

Alex suggested we explicitly define:

- processing of incoming TCP connections (peer lookup, acceptance, FSM creation, collision control,)

Curtis later proposed this text:

BGP must maintain separate FSM for each configured peer. Each BGP

peer paired in a potential connection will attempt to connect to the other. For the purpose of this discussions, the active or connect side of a TCP connection (the side sending the first TCP SYN packet) is called outgoing. The passive or listening side (the sender of the first SYN ACK) is called the an incoming connection.

A BGP implementation must connect to and listen on TCP port 179 for incoming connections in addition to trying to connect to peers. For each incoming connection, a state machine must be instantiated. There exists a period in which the identity of the peer on the other end of an incoming connection is not known with certainty. During this time, both an incoming and outgoing connection for the same peer may exist. This is referred to as a connection collision (see Section x.x, was 6.8).

A BGP implementation will have at most one FSM for each peer plus one FSM for each incoming TCP connection for which the peer has not yet been identified. Each FSM corresponds to exactly one TCP connection.

Jonathan pointed out that there was an inaccuracy in the proposed text. Curtis replied with this:

You're correct in that you must have a collision of IP addresses on the TCP connections and that the BGP Identifier is used only to resolve which gets dropped.

The FSM stays around as long as "BGP Identifier" is not known. Replace "not known with certainty" with "known but the BGP identifier is not known" and replace "for the same peer" with "for the same configured peering".

The first paragraph is unchanged:

BGP must maintain separate FSM for each configured peer. Each BGP peer paired in a potential connection will attempt to connect to the other. For the purpose of this discussions, the active or connect side of a TCP connection (the side sending the first TCP SYN packet) is called outgoing. The passive or listening side (the sender of the first SYN ACK) is called the an incoming connection.

The second paragraph becomes:

A BGP implementation must connect to and listen on TCP port 179 for incoming connections in addition to trying to connect to peers. For each incoming connection, a state machine must be instantiated. There exists a period in which the identity of the peer on the other end of an incoming connection is known but the BGP identifier is not known. During this time, both an incoming and outgoing connection

for the same configured peering may exist. This is referred to as a connection collision (see Section x.x, was 6.8).

The next paragraph then needs to get fixed. Changed "for each peer" to "for each configured peering".

A BGP implementation will have at most one FSM for each configured peering plus one FSM for each incoming TCP connection for which the peer has not yet been identified. Each FSM corresponds to exactly one TCP connection.

Add a paragraph to further clarify the point you made.

There may be more than one connection between a pair of peers if the connections are configured to use a different pair of IP addresses. This is referred to as multiple "configured peerings" to the same peer.

> So multiple simultaneous BGP connection are allowed between the same two > peers, and this behavior is implemented, for example to do load balancing.

Good point.

I hope the corrections above cover your (entirely valid) objections. If you see any more errors please let me know.

Tom replied that:

I take issue with the 'will attempt to connect' which goes too far.

The FSM defines events 4 and 5, 'with passive Transport establishment', so the system may wait and not attempt to connect. The exit from this state is either the receipt of an incoming TCP connection (SYN) or timer expiry.

So we may have a FSM attempting to transport connect for a given source/destination IP pair or we may have an FSM not attempting to connect. (In the latter case, I do not think we can get a collision). In the latter case, an incoming connection should not generate an additional FSM.

I do not believe the concept of active and passive is helpful since a given system can flip from one to the other and it does not help us to clarify the number of FSM involved..

And Curtis suggested that:

Could this be corrected by replacing "will attempt to connect" with "unless configured to remain in the idle state, or configured to remain passive, will attempt to connect". We could also shorten that to "will attempt to connect unless configured otherwise".

Clarification (perhaps an item for a glossary entry): The terms active and passive have been in our vocabulary for almost a decade and have proven useful. The words active and passive have slightly different meanings applied to a TCP connection or applied to a peer. There is only one active side and one passive side to any one TCP connection as per the definition below. When a BGP speaker is configured passive it will never attempt to connect. If a BGP speaker is configured active it may end up on either the active or passive side of the connection that eventually gets established. Once the TCP connection is completed, it doesn't matter which end was active and which end was passive and the only difference is which side of the TCP connection has port number 179.

Tom agreed with Curtis, that he liked the "will attempt to connect unless configured otherwise" verbiage.

This was discussed in the "Generial Editorial Comment" thread.

Sue proposed we add the text above in section 8.2. It is summarized here for clarity:

8.2) Description of FSM

8.2.1) FSM connections

(text below)

8.2.2) FSM Definition

(text now in 8.2)

"BGP must maintain a separate FSM for each configured peer plus Each BGP peer paired in a potential connection unless configured to remain in the idle state, or configured to remain passive, will attempt to to connect to the other. For the purpose of this discussion, the active or connect side of the TCP connection (the side of a TCP connection (the side sending the first TCP SYN packet) is called outgoing. The passive or listening side (the sender of the first SYN ACK) is called an incoming connection. [See section on the terms active and passive below.]

A BGP implementation must connect to and listen on TCP port 179 for incoming connections in addition to trying to connect to peers. Fro

each incoming connection, a state machine must be instantiated. There exists a period in which the identity of the peer on the other end of an incoming connection is known but the BGP identifier is not known. During this time, both an incoming and an outgoing connection for the same configured peering may exist. This is referred to as a connection collision (see Section x.x, was 6.8).

A BGP implementation will have at most one FSM for each configured peering plus one FSM for each incoming TCP connection for which the peer has not yet been identified. Each FSM corresponds to exactly one TCP connection.

There may be more than one connections between a pair of peers if the connections are configured to use a different pair of IP addresses. This is referred to as multiple "configured peerings" to the same peer.

8.2.1.1) Terms "active" and "passive"

The terms active and passive have been in our vocabulary for almost a decade and have proven useful. The words active and passive have slightly different meanings applied to a TCP connection or applied to a peer. There is only one active side and one passive side to any one TCP connection per the definition above [and the state machine below.] When a BGP speaker is configured active it may end up on either the active or passive side of the connection that eventually gets established. Once the TCP connection is completed, it doesn't matter which end was active and which end was passive and the only difference is which side of the TCP connection has port number 179.

For additional text, see issue 46.

Sue solicited additional comments, the only one was on capitalization, so it would appear we are at consensus with this issue.

2.49. Explicitly Define Event Generation

Status: Consensus
Change: No
Summary: Suggested that we explicitly define BGP message processing. No text proposed. There has been no further discussion on this issue, it is assumed that the consensus is that things are ok the way they are.

Discussion:

Alex suggested we explicitly define:

- generation of events while processing BGP messages, i.e., the text describing message processing should say where needed that a specific event for the BGP session should be generated.

No text was proposed.

This discussion has received no further comment. Unless someone wants to reopen it, it is assumed it has reached a happy ending.

This was discussed in the "Generial Editorial Comment" thread.

2.50. FSM Timers

Status: Consensus
Change: No
Summary: Discussion tabled, because new document version rendered the discussion moot.

Discussion:

This discussion began with a suggestion that the timers currently in the FSM:

In the 26 Aug text, I find the timer terminology still confusing. Timers can, I find, stop start restart clear set reset expire

Can be cleaned up and simplified to:

start with initial value (spell it out just to be sure) stop expire

A response to this proposal was, that the existing set is clear, and that the proposed set is insufficiently rich to describe a concept like "reset" which encompasses: "Stop the timer, and reset it to its initial value."

This discussion reached an impasse, when Sue pointed out that the text had been updated, and to please review the new text.

This was discussed in the "FSM more words" thread.

2.51. FSM ConnectRetryCnt

Status: Consensus
Change: No
Summary: Discussion tabled, because new document version rendered the discussion moot.

Discussion:

This started with the observation that the ConnectRetryCnt "seems to have lost its purpose." It was suggested that this be made a Session Attribute, along with: OpenDelayTimer and DelayOpenFlag.

Curtis explained that the current purpose of the ConnectRetryCnt is something to be read by the MIB. He also advocated against adding the additional Session Attributes.

2.52. Section 3: Keeping routes in Adj-RIB-In

Status: Consensus
Change: Yes
Summary: Add: To allow local policy changes to have the correct effect without resetting any BGP connections, a BGP speaker SHOULD either (a) retain the current version of the routes advertised to it by all of its peers for the duration of the connection, or (b) make use of the Route Refresh extension [12].

Discussion:

This thread started with a question about why we should retain routes in the Adj-RIB-In, and how it relates to the Route Refresh extension.

mrr proposed this text:

... Therefore, a BGP speaker must either retain the current version of the routes advertised by all of its peers for the duration of the connection, or make use of the Route Refresh extension [12]. This is necessary to allow local policy changes to have the correct effect without requiring the reset of any peering sessions.

If the implementation decides not to retain the current version of the routes that have been received from a peer, then Route Refresh messages should be sent whenever there is a change to local policy.

Yakov later suggested this text, with a slight rewording:

To allow local policy changes to have the correct effect without resetting any BGP connections, a BGP speaker SHOULD either (a) retain the current version of the routes advertised to it by all of its peers for the duration of the connection, or (b) make use of the Route Refresh extension [12].

mrr responded that he was fine with Yakov's suggestions.

This was discussed in the "Proxy: comments on section 3" thread.

2.53. Section 4.3 - Routes v. Destinations - Advertise

Status: Consensus
Change: No
Summary: The text that has reached consensus in issue 54 also addresses this issue.

Discussion:

This issue arose out of this question to the list:

Since:

"For the purpose of this protocol, a route is defined as a unit of information that pairs a set of destinations with the attributes of a path to those destinations. The set of destinations are the systems whose IP addresses are reported in the Network Layer Reachability Information (NLRI) field and the path is the information reported in the path attributes field of the same UPDATE message."

When I read section 4.3:

"An UPDATE message is used to advertise feasible routes sharing common path attribute to a peer, or to withdraw multiple unfeasible routes from service (see 3.1)."

Shouldn't the text read "... advertise feasible [prefixes | destinations] sharing common path attribute(S)to a peer ...", because:

1) A route is defined as quoted above from section 3.1?

or since ...

"An UPDATE message can advertise at most one set of path attributes, but multiple destinations ..."

2) make "routes" in the original singular.

This was discussed in the "Review Comments: Section 4.3: "routes" vs. destinations - advertise" thread.

The text that has reached consensus in issue 54 also addresses this issue.

2.54. Section 4.3 - Routes v. Destinations - Withdraw

Status: Consensus
Change: Yes
Summary: Change the definition of "route" as it currently stands to:

For the purpose of this protocol, a route is defined as a unit of information that pairs a set of destinations with the attributes of a path to those destinations. The set of destinations are systems whose IP addresses are contained in one IP address prefix carried in the Network Layer Reachability Information (NLRI) field of an UPDATE message and the path is the information reported in the path attributes field of the same UPDATE message.

Multiple routes that have the same path attributes can be advertised in a single UPDATE message by including multiple prefixes in the NLRI field of the UPDATE message.

Discussion:

This issue was brought up with this question:

When I read these two paragraphs at the end of section 4.3:

"An UPDATE message can list multiple routes to be withdrawn from service. Each such route is identified by its destination (expressed as an IP prefix), which unambiguously identifies the route in the context of the BGP speaker - BGP speaker connection to which it has been previously advertised.

An UPDATE message might advertise only routes to be withdrawn from service, in which case it will not include path attributes or Network Layer Reachability Information. Conversely, it may advertise only a feasible route, in which case the WITHDRAWN ROUTES field need not be present."

It reads as if one must withdraw the _set of destinations_ advertised with the route instead of just one or more destinations since route is defined in section 3.1 as:

"For the purpose of this protocol, a route is defined as a unit of information that pairs a set of destinations with the attributes of a path to those destinations. The set of destinations are the systems whose IP addresses are reported in the Network Layer Reachability Information (NLRI) field and the path is the information reported in the path attributes field of the same UPDATE message."

Shouldn't the text change "routes" to destinations, or to prefixes?

The original commenter added this clarification later:

I meant to say, the *same* set of destinations as those advertised initially. For example, NLRI with dest-a, dest-b and dest-c with the same attributes is a "route". The withdrawal of the "route" can be read as one must withdraw all destinations originally advertised for that route (dest-a, dest-b, dest-c) as a unit.

A first time reader could be left wondering if the above must be the case, or it is valid for an implementation to withdraw just one of these destinations (e.g. dest-b), leaving the others (dest-a, dest-c) reachable with their attributes intact.

If there is no relationship between destinations when advertised as a set of destinations in a route and those destinations that can be withdrawn should be explicitly stated. Otherwise, the draft should call out that it is not legal to withdraw one prefix only in the set of prefixes advertised as previously as route.

Matt suggested that since the definition seems to cause some confusion, that we update the definition to:

"For the purpose of this protocol, a route is defined as a unit of information that pairs a set of destinations with the attributes of a path to those destinations. The set of destinations are systems whose IP addresses are reported in one prefix present in the Network Layer Reachability Information (NLRI) field of an UPDATE and the path is the information reported in the path attributes field of the same UPDATE message.

This definition allows multiple routes to be advertised in a single UPDATE message by including multiple prefixes in the NLRI field of the UPDATE. All such prefixes must be associated with the same set of path attributes."

Yakov suggested some minor rewording:

For the purpose of this protocol, a route is defined as a unit of information that pairs a set of destinations with the attributes of a path to those destinations. The set of destinations are systems whose IP addresses are contained in one IP address prefix carried in the Network Layer Reachability Information (NLRI) field of an UPDATE message and the path is the information reported in the path attributes field of the same UPDATE message.

Multiple routes that have the same path attributes can be advertised in a single UPDATE message by including multiple prefixes in the NLRI field of the UPDATE message.

Both Jeff and Matt responded that they liked this text.

2.55. Section 4.3 - Description of AS_PATH length

Status: Consensus
Change: Yes
Summary: Replace:

Path segment length is a 1-octet long field containing the number of ASs in the path segment value field.

With:

Path segment length is a 1-octet long field containing the number of ASs (not the number of octets) in the path segment value field.

Discussion:

This question was raised:

Length fields elsewhere in the draft specify the number of bytes that a variable length field uses. For AS_PATH, length is used as the number of 2 byte AS numbers. In the interest of not having to check other sources to be sure, should the description of path segment value:

"The path segment value field contains one or more AS numbers, each encoded as a 2-octets long field."

explicitly mention the number of bytes used by the variable length field?

Or, make the description of length explicitly mention that it is not the length of the variable length field as is the case with other length fields?

One response to this agreed that some more clarification would be good, specifically an ASCII art diagram. No diagram was proposed.

Yakov proposed this change:

How about replacing

Path segment length is a 1-octet long field containing the number of ASs in the path segment value field.

with the following

Path segment length is a 1-octet long field containing the number of ASs (but not the number of octets) in the path segment value field.

Jonathan offered this text:

How about: "Path segment length is a 1-octet long field containing the number of ASs (which is half the number of octets since each AS is 2 octets) in the path segment value field (also note that the path may contain more than 1 segment).

Jeff replied that he preferred Yakov's text, but without the "but". Yakov agreed to that. Andrew also agreed, and asked if there were any objections to moving this issue into consensus. There were no objections.

2.56. Section 6 - BGP Error Handling

Status: Consensus
Change: Yes
Summary: There are a variety of updates to the text that are best described in the discussion section.

Discussion:

This discussion began with some suggestions on ways to clarify the text in section 6 dealing with error handling. The original comments, and Yakov's response are below:

> At the beginning of Section 6. BGP Error Handling:
>
>
> "When any of the conditions described here are detected, a
> NOTIFICATION message with the indicated Error Code, Error Subcode,
> and Data fields is sent, and the BGP connection is closed."
>
> There are several cases where the conditions described in this
section
> do not result in the BGP connection being closed. These conditions
> should either state that the connection stays up. Another
possibility
> is to remove "and the BGP connection is closed." above, and for
each
> listed connection, state what happens to the BGP connection. This
> already takes place for certain conditions, but isn't done
consistently.

How about replacing the above with the following:

When any of the conditions described here are detected, a
NOTIFICATION message with the indicated Error Code, Error Subcode,
and Data fields is sent, and the BGP connection is closed, unless it
is explicitly stated that no NOTIFICATION message is to be sent and
the BGP connection is not to be close.

> I tried to list what I found (which may be wrong or incomplete)
below:
>
>
> "If the NEXT_HOP attribute is semantically incorrect, the error
should
> be logged, and the route should be ignored. In this case, no
> NOTIFICATION message should be sent."
>
> * Append the connection is not closed.

Done.

>
> "(a) discard new address prefixes from the neighbor, or (b)
terminate
> the BGP peering with the neighbor."
>
> * Append "the connection is not closed" to case (a)

added "(while maintaining BGP peering with the neighbor)" to case
(a).

>
> "If the autonomous system number appears in the AS path the
> route may be stored in the Adj-RIB-In,"
>
> * append and the BGP connection stays up.
>
> "but unless the router is configured to accept routes with its
> own autonomous system in the AS path, the route shall not be
> passed to the BGP Decision Process."

I would suggest to move this text to Section 9 (UPDATE message
handling), as receiving a route with your own AS in the AS path isn't
really an error. It is just that usually (but not always) you can't
put this route in your Adj-RIB-In.

> * Q1) does the BGP connection stay up?

yes.

> * Q2) what if the router isn't configured to accept routes with its
> own AS in the AS path? One _may_ store the route in Adj-RIB-In,
but
> if one doesn't want to?

So, don't store them.

> Is the BGP connection closed? If so, what subcode?

The connection is *not* closed.

> "If an optional attribute is recognized, then the value of this
> attribute is checked. If an error is detected, the attribute is
> discarded, and the Error Subcode is set to Optional Attribute
Error.
> The Data field contains the attribute (type, length and value)."
>
> * Append and the BGP connection stays up after "the attribute is
discarded".

Since you have to close the connection, you have to discard all the routes received via this connection, including the route with the incorrect attribute. So, there is no need to say that "the attribute is discarded".

There have been no objections to the updates listed above. This issue seems to have reached a happy ending, so it has been moved into consensus.

This was discussed in the "-17 review Section 6 - BGP Error Handling." thread.

2.57. Section 6.2 - Hold timer as Zero

Status: Consensus
Change: No
Summary: It was suggested that we update the text to say that we MUST reject hold time values of zero. It was pointed out that this is not the case and the text should say the way it is.

Discussion:

In Section 6.2 on OPEN message error handling:

If the Hold Time field of the OPEN message is unacceptable, then the Error Subcode MUST be set to Unacceptable Hold Time. An implementation MUST reject Hold Time values of one or two seconds.

I feel that text similar to:

"An implementation MUST also reject Hold Time values of zero received from a peer in a different AS" should be considered for completeness.

A number of respondents pointed out that zero hold time is legitimate under certain circumstances.

This was discussed in the "-17 review, Section 6.2 - must reject hold time" thread.

2.58. Deprecation of ATOMIC_AGGREGATE

Status: Consensus
Change: Yes
Summary: For new text, please see the end of the discussion.

Discussion:

Jeff opened this discussion with:

Deprecation of ATOMIC_AGGREGATE:

[This is a summary of some discussions from those who have "been there, done that" during the CIDR deployment period and also comments from network operators on the NANOG list.]

When BGP-4 was originally drafted, the topic of aggregation was new enough that people didn't exactly know how it would be used. Additionally, there were some transition issues when aggregated networks would need to be de-aggregated and re-advertised into a classful routing mesh such as BGP-3.

The ATOMIC_AGGREGATE flag was intended to prevent a route from being de-aggregated when de-aggregation would introduce routing loops. Note that de-aggregation in this context specifically means making a less specific route into one or more more-specific routes.

The current BGP draft has two situations where ATOMIC_AGGREGATE should be appended to a route: 1. When a route's AS_PATH is intentionally truncated, such as what happens by default on Cisco's, or using the "brief" option on GateD. Juniper has a similar feature - I'm unsure of the default. 2. When two routes are implicitly aggregated in the LocRib such that a more specific route is not selected when a less specific route is from a given peer.

Note that this particular feature is not implemented anywhere that I am aware of.

3. There is a third case not covered by the specification: Implicit aggregation on export - a more specific route is not exported and only a less specific one is.

When network operators were asked about de-aggregation practices, I received about 40 responses. The majority of these responses confused de-aggregation with leaking existing more-specific routes that are used internally rather than explicitly de-aggregating a less-specific route.

There were a very few cases of explicit de-aggregation. One form was done for purposes of dealing with another ISP creating a routing DoS by advertising IP space that didn't belong to them - leaked more specifics alleviated the problem in many cases. (Note that this is a security issue in the routing system.)

The second case was de-aggregating routes internally (and sending the routes via IBGP marked with NO-ADVERTISE) for purposes of traffic engineering routing internally where a given upstream failed to provide enough routing information to allow traffic flows to be optimized based on supplied prefixes.

My conclusions to this are: 1. De-aggregation is not a common practice. 2. It is no longer a major concern since classful inter-domain routing is pretty much gone. 3. The spec doesn't match reality and should be corrected.

My suggestions are thus this: Section 5.1.6 should be updated as follows: ATOMIC_AGGREGATE is a well-known discretionary attribute. Its use is deprecated.

When a router explicitly aggregates several routes for the purpose of advertisement to a particular peer, and the AS_PATH of the aggregated route excludes at least some of the AS numbers present in the AS_PATH of the routes that are aggregated (usually due to truncation), the aggregated route, when advertised to the peer, MUST include the ATOMIC_AGGREGATE attribute.

Section 9.1.4 should be updated as follows:
Original text: If a BGP speaker receives overlapping routes, the Decision Process MUST consider both routes based on the configured acceptance policy. If both a less and a more specific route are accepted, then the Decision Process MUST either install both the less and the more specific routes or it MUST aggregate the two routes and install the aggregated route, provided that both routes have the same value of the NEXT_HOP attribute.

If a BGP speaker chooses to aggregate, then it MUST add

ATOMIC_AGGREGATE attribute to the route. A route that carries ATOMIC_AGGREGATE attribute can not be de-aggregated. That is, the NLRI of this route can not be made more specific. Forwarding along such a route does not guarantee that IP packets will actually traverse only ASs listed in the AS_PATH attribute of the route.

Replace with:

It is common practice that more specific routes are often implicitly aggregated by selecting or advertising only a less-specific route when overlapping routes are present. As such, all routes SHOULD be treated as if ATOMIC_AGGREGATE is present and not be made more specific (de-aggregated). De-aggregation may lead to routing loops.

Section 9.2.2 should remain as it is.

Implications of not making the above updates:
ATOMIC_AGGREGATE is not implemented as documented. Current operational practices do not seem to often trigger situations that it was intended to re-mediate. After all, by the way it is currently documented, many of the routes in the Internet would likely have ATOMIC_AGGREGATE.

The original motivation for this investigation (aside from a few years of wondering what this portion of the spec *really* meant) was making sure the MIB is correctly documented. I can do this now, even if the spec is not corrected by simply noting that the values:
lessSpecificRouteNotSelected(1),
lessSpecificRouteSelected(2)

mean:
ATOMIC_AGGREGATE not present
ATOMIC_AGGREGATE present

rather than documenting anything about less and more specific routes.

The v2MIB can be fixed to just call it what it is since it hasn't been RFC'ed yet.

Lastly, the spec would just be incorrect. But all said, nothing bad would really come of this.

Yakov responded to this, saying that he thought these changes were reasonable, and unless there were objections, they would be adopted.

Ishi strongly agreed with the changes.

Curtis stated that:

We used to add ATOMIC_AGGREGATE whenever the AS_PATH was truncated rather than replaced with an AS_SET. It was always purely informational since no one intentionally deaggregated and reannounced.

And suggested that we remove the MUSTs and indicated that this is only informational.

Jeff replied that:

The point is that by definition of the attribute, anywhere that policy is used (and some places where it may not be), ATOMIC_AGGREGATE *should* be there. Since its not, and it would generally be everywhere, you just shouldn't de-aggregate.

At best, leaving it as a method of informationally signalling truncation of a AS_PATH is the best we'll get out of it - and it matches current implementations.

Jonathan agreed with Curtis' idea that we should just move ATOMIC_AGGREGATE to informational.

Curtis proposed this text:

This existing text is fine:

f) ATOMIC_AGGREGATE (Type Code 6)

ATOMIC_AGGREGATE is a well-known discretionary attribute of length 0. Usage of this attribute is described in 5.1.6.

This is the existing text that we are considering changing:

5.1.6 ATOMIC_AGGREGATE

ATOMIC_AGGREGATE is a well-known discretionary attribute.

When a router aggregates several routes for the purpose of advertisement to a particular peer, and the AS_PATH of the aggregated route excludes at least some of the AS numbers present in the AS_PATH of the routes that are aggregated, the aggregated route, when advertised to the peer, MUST include the ATOMIC_AGGREGATE attribute.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute MUST NOT remove the attribute from the route when propagating it to other speakers.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE

attribute MUST NOT make any NLRI of that route more specific (as defined in 9.1.4) when advertising this route to other BGP speakers.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute needs to be cognizant of the fact that the actual path to destinations, as specified in the NLRI of the route, while having the loop-free property, may not be the path specified in the AS_PATH attribute of the route.

Suggested new text:

5.1.6 ATOMIC_AGGREGATE

ATOMIC_AGGREGATE is a well-known discretionary attribute.

When a router aggregates several routes for the purpose of advertisement to a particular peer, the AS_PATH of the aggregated route normally includes an AS_SET formed from the set of AS from which the aggregate was formed. In many cases the network administrator can determine that the aggregate can safely be advertised without the AS_SET and not form route loops.

If an aggregate excludes at least some of the AS numbers present in the AS_PATH of the routes that are aggregated as a result of dropping the AS_SET, the aggregated route, when advertised to the peer, SHOULD include the ATOMIC_AGGREGATE attribute.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute SHOULD NOT remove the attribute from the route when propagating it to other speakers.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute MUST NOT make any NLRI of that route more specific (as defined in 9.1.4) when advertising this route to other BGP speakers.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute needs to be cognizant of the fact that the actual path to destinations, as specified in the NLRI of the route, while having the loop-free property, may not be the path specified in the AS_PATH attribute of the route.

Diffs (for reader convenience):

```
@@ -4,13 +4,19 @@ ATOMIC_AGGREGATE is a well-known discretionary
attribute.

When a router aggregates several routes for the purpose of
- advertisement to a particular peer, and the AS_PATH of the
```

```
- aggregated route excludes at least some of the AS numbers
- present in the AS_PATH of the routes that are aggregated,
- the aggregated route, when advertised to the peer, MUST
- include the ATOMIC_AGGREGATE attribute.
+ advertisement to a particular peer, the AS_PATH of the
+ aggregated route normally includes an AS_SET formed from the
+ set of AS from which the aggregate was formed.  In many cases
+ the network administrator can determine that the aggregate can
+ safely be advertised without the AS_SET and not form route loops.
+
+ If an aggregate excludes at least some of the AS numbers present
+ in the AS_PATH of the routes that are aggregated as a result of
+ dropping the AS_SET, the aggregated route, when advertised to the
+ peer, SHOULD include the ATOMIC_AGGREGATE attribute.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE
- attribute MUST NOT remove the attribute from the route when
+ attribute SHOULD NOT remove the attribute from the route when
+ propagating it to other speakers.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE
```

Current text in 9.1.4:

If a BGP speaker chooses to aggregate, then it MUST add ATOMIC_AGGREGATE attribute to the route. A route that carries ATOMIC_AGGREGATE attribute can not be de-aggregated. That is, the NLRI of this route can not be made more specific. Forwarding along such a route does not guarantee that IP packets will actually traverse only ASs listed in the AS_PATH attribute of the route.

Change to:

If a BGP speaker chooses to aggregate, then it SHOULD either include all AS used to form the aggregate in an AS_SET or add the ATOMIC_AGGREGATE attribute to the route. This attribute is now primarily informational. With the elimination of IP routing protocols that do not support classless routing and the elimination of router and host implementations that do not support classless routing, there is no longer a need to deaggregate. Routes SHOULD NOT be de-aggregated. A route that carries ATOMIC_AGGREGATE attribute in particular MUST NOT be de-aggregated. That is, the NLRI of this route can not be made more specific. Forwarding along such a route does not guarantee that IP packets will actually traverse only ASs listed in the AS_PATH attribute of the route.

This text in 9.2.2.2 need not change.

ATOMIC_AGGREGATE: If at least one of the routes to be aggregated has ATOMIC_AGGREGATE path attribute, then the aggregated route shall have this attribute as well.

The appendix need not change:

Appendix 1. Comparison with RFC1771

[...]

Clarifications to the use of the ATOMIC_AGGREGATE attribute.

This might be a bit more wordy that is necessary. It does address the objections to keeping ATOMIC_AGGREGATE by making the MUST into SHOULD, and explaining that ATOMIC_AGGREGATE is now primarily informational.

Yakov was fine with this text.

Yakov posted the text that represents the WG consensus:

Replace:

5.1.6 ATOMIC_AGGREGATE

ATOMIC_AGGREGATE is a well-known discretionary attribute.

When a router aggregates several routes for the purpose of advertisement to a particular peer, and the AS_PATH of the aggregated route excludes at least some of the AS numbers present in the AS_PATH of the routes that are aggregated, the aggregated route, when advertised to the peer, MUST include the ATOMIC_AGGREGATE attribute.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute MUST NOT remove the attribute from the route when propagating it to other speakers.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute MUST NOT make any NLRI of that route more specific (as defined in 9.1.4) when advertising this route to other BGP speakers.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute needs to be cognizant of the fact that the actual path to destinations, as specified in the NLRI of the route, while having the loop-free property, may not be the path specified in the AS_PATH attribute of the route.

with:

5.1.6 ATOMIC_AGGREGATE

ATOMIC_AGGREGATE is a well-known discretionary attribute.

When a router aggregates several routes for the purpose of advertisement to a particular peer, the AS_PATH of the aggregated route normally includes an AS_SET formed from the set of AS from which the aggregate was formed. In many cases the network administrator can determine that the aggregate can safely be advertised without the AS_SET and not form route loops.

If an aggregate excludes at least some of the AS numbers present in the AS_PATH of the routes that are aggregated as a result of dropping the AS_SET, the aggregated route, when advertised to the peer, SHOULD include the ATOMIC_AGGREGATE attribute.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute SHOULD NOT remove the attribute from the route when propagating it to other speakers.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute MUST NOT make any NLRI of that route more specific (as defined in 9.1.4) when advertising this route to other BGP speakers.

A BGP speaker that receives a route with the ATOMIC_AGGREGATE attribute needs to be cognizant of the fact that the actual path to destinations, as specified in the NLRI of the route, while having the loop-free property, may not be the path specified in the AS_PATH attribute of the route.

In 9.1.4 replace:

If a BGP speaker chooses to aggregate, then it MUST add ATOMIC_AGGREGATE attribute to the route. A route that carries ATOMIC_AGGREGATE attribute can not be de-aggregated. That is, the NLRI of this route can not be made more specific. Forwarding along such a route does not guarantee that IP packets will actually traverse only ASs listed in the AS_PATH attribute of the route.

with:

If a BGP speaker chooses to aggregate, then it SHOULD either include all AS used to form the aggregate in an AS_SET or add the ATOMIC_AGGREGATE attribute to the route. This attribute is now primarily informational. With the elimination of IP routing protocols that do not support classless routing and the elimination of router and host implementations that do not support classless routing, there is no longer a need to deaggregate. Routes SHOULD NOT

be de-aggregated. A route that carries ATOMIC_AGGREGATE attribute in particular MUST NOT be de-aggregated. That is, the NLRI of this route can not be made more specific. Forwarding along such a route does not guarantee that IP packets will actually traverse only ASs listed in the AS_PATH attribute of the route.

This met with agreement. This issue is at consensus.

2.59. Section 4.3 - Move text

Status: Consensus
Change: Yes (minimal)
Summary: Update indentation to allow a new "subsection" heading.
Otherwise no change.

Discussion:

This began with this suggestion:

The text about the minimum length, at first look, gives the impression that it is still part of the NLRI field description and/or the Path Attributes section. A new "subsection" or heading of some sort would be helpful in switching context back to the UPDATE message as a whole and not the path attributes field anymore.

Yakov agreed to update the indentation.

Jonathan agreed, and suggested this text:

" The minimum length of the UPDATE message is 23 octets -- 19 octets for the fixed header + 2 octets for the Withdrawn Routes Length + 2 octets for the Total Path Attribute Length (the value of Withdrawn Routes Length is 0 and the value of Total Path Attribute Length is 0)."

Should be moved up to just after

"... the Total Path Attribute Length field and the Withdrawn Routes Length field."

Yakov responded to this with:

Disagree, as "... the Total Path Attribute Length field and the Withdrawn Routes Length field." explains how to calculate the length of NLRI field (and therefore is part of the NLRI field description), while "The minimum length of the UPDATE message is 23 octets...." has to do with the length of the UPDATE message.

Jonathan also suggested:

" the value of Withdrawn Routes Length is 0 and the value of Total Path Attribute Length is 0)."

Should be changed to

" the min. value of Withdrawn Routes Length is 0 and the min. value of Total Path Attribute Length is 0)."

And Yakov responded with:

Disagree, as the text doesn't talk about what is the minimum value of the Withdrawn Routes Length and Total Path Attribute Length fields, but talks about the value of these fields in the case of a min length UPDATE message.

After Yakov's response and a posting to the list asking that this be moved to consensus, there were no objections, so this is moved to consensus.

This is discussed in the "Review: Comments: Section 4.3: UPDATE min length" thread.

2.60. Section 4.3 - Path Attributes

Status: Consensus
Change: Yes
Summary: Make this change to clarify path attributes in an UPDATE:

To correct the confusion I propose to replace:

A variable length sequence of path attributes is present in every UPDATE.

with:

A variable length sequence of path attributes is present in every UPDATE message, except for an UPDATE message that carries only the withdrawn routes.

Discussion:

This thread began with MikeC pointing out:

The top of page 13 says:

"A variable length sequence of path attributes is present in every

UPDATE."

Is this really true, given that later, in the second to last paragraph of this section (4.3):

"An UPDATE message might advertise only routes to be withdrawn from service, in which case it will not include path attributes or Network Layer Reachability Information."

This could be confusing to a first time reader.

The path attribute length is present in every UPDATE, the path attribute field itself can be left out, both according to the description of the minimum length of the UPDATE message and (implied?) in the picture of the UPDATE message at the beginning of section 4.3.

This met with one agreement.

Yakov then proposed that:

To correct the confusion I propose to replace:

A variable length sequence of path attributes is present in every UPDATE.

with:

A variable length sequence of path attributes is present in every UPDATE message, except for an UPDATE message that carries only the withdrawn routes.

There was one agreement with this proposal.

This is discussed in the thread: "Review: Section 4.3 - Path Attributes"

2.61. Next Hop for Redistributed Routes

Status: Consensus
Change: Yes
Summary: More clearly specify the behavior of NEXT_HOP modification, for the text see the end of the discussion.

Discussion:

Jonathan began this thread with:

I propose adding:

"When announcing a locally originated route to an internal peer, the BGP speaker should use as the NEXT_HOP the interface address of the router through which the announced network is reachable for the speaker; if the route is directly connected to the speaker, or the interface address of the router through which the announced network is reachable for the speaker is the internal peer's address, then the BGP speaker should use for the NEXT_HOP attribute its own IP address (the address of the interface that is used to reach the peer)."

AFTER

"When sending a message to an internal peer, the BGP speaker should not modify the NEXT_HOP attribute, unless it has been explicitly configured to announce its own IP address as the NEXT_HOP."

There has been no discussion on this.

This is discussed in the "Next hop for redistributed routes" thread.

Issue 14 closed in favor of this issue.

In response to Yakov's call for input, Michael responded that:

Section 5.1.3 explicitly states what to do when originating to an external peer. #2 covers one hop away, #3 covers more than on hop away. #1 talks about sending to an iBGP peer, but only when propagating a route received from an eBGP peer.

1) When sending a message to an internal peer, the BGP speaker should not modify the NEXT_HOP attribute, unless it has been explicitly configured to announce its own IP address as the NEXT_HOP.

Text similar to #2 should be added, in their own bullet items to #1 such as what was suggested by Jonathan Natale (text is above.)

Yakov replied with this:

Replace:

1) When sending a message to an internal peer, the BGP speaker should not modify the NEXT_HOP attribute, unless it has been explicitly configured to announce its own IP address as the NEXT_HOP.

with:

1) When sending a message to an internal peer, if the route is not

locally originated the BGP speaker should not modify the NEXT_HOP attribute, unless it has been explicitly configured to announce its own IP address as the NEXT_HOP. When announcing a locally originated route to an internal peer, the BGP speaker should use as the NEXT_HOP the interface address of the router through which the announced network is reachable for the speaker; if the route is directly connected to the speaker, or the interface address of the router through which the announced network is reachable for the speaker is the internal peer's address, then the BGP speaker should use for the NEXT_HOP attribute its own IP address (the address of the interface that is used to reach the peer).

And stated the change would be made if there were no objections. There have been no objections, so this is at consensus.

2.62. Deprecate BGP Authentication Optional Parameter from RFC1771

Status: Consensus
Change: Yes
Summary: We are at consensus, in that we agree that we should deprecate the BGP Authentication Optional Parameter as described in RFC1771 in favor of the mechanism described in RFC2385. The textual changes are listed at the end of the discussion section of this issue.

Discussion:

This discussion started in issue 21: Authentication Text Update.

This topic has come up before (July time frame), but was recently refreshed in the context of issue 21. It began with some questions to the list as to who used what sort of authentication mechanisms. From the responses it was clear that MD5 (RFC2385) was the preferred method.

Eric Gray's message helps to flesh this out:

The question is not whether MD5 authentication is used, it is how is it implemented in real BGP implementations or - more importantly - where is the authentication information located in the packets sent between two BGP peers? This is not strictly an implementation issue because authentication information is located in different places depending on which version of MD5 authentication is in use.

As is generally known, options are not necessarily good. Currently, between RFC 1771 and RFC 2385, there are two very distinct ways to accomplish a semantically identical function. If the mechanism defined in RFC 1771 is not supported by a number of interoperable

implementations, it must be deprecated per RFC 2026 prior to advancing the specification to Draft Standard. If it is not implemented and actually in use, it should be deprecated if for no other reason than to make the

To this Yakov responded:

To be more precise,

In cases in which one or more options or features have not been demonstrated in at least two interoperable implementations, the specification may advance to the Draft Standard level only if those options or features are removed.

So, the relevant question is whether we have at least two implementations that support authentication as described in rfc1771.

Folks who implemented authentication, as described in rfc1771, please speak up.

There have been no responses to Yakov's question.

There seems to be a consensus that, since it is little used, and since there are better mechanisms, namely MD5 authentication, we should deprecate the BGP Authentication Optional Parameter from RFC1771.

Ok, after some discussion, this is a list of the text that we are adding, changing or removing:

1) Remove the reference to the authentication optional parameter:

I would suggest to remove the following text from the draft:

This document defines the following Optional Parameters:

a) Authentication Information (Parameter Type 1):

This optional parameter may be used to authenticate a BGP peer. The Parameter Value field contains a 1-octet Authentication Code followed by a variable length Authentication Data.

```
0 1 2 3 4 5 6 7 8 +-+-+-+-+-+-+-+-+ | Auth.  Code |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | | |
Authentication Data | | |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Authentication Code:

This 1-octet unsigned integer indicates the authentication mechanism being used. Whenever an authentication mechanism is specified for use within BGP, three things must be included in the specification:

- the value of the Authentication Code which indicates use of the mechanism, - the form and meaning of the Authentication Data, and - the algorithm for computing values of Marker fields.

Note that a separate authentication mechanism may be used in establishing the transport level connection.

Authentication Data:

Authentication Data is a variable length field that is interpreted according to the value of the Authentication Code field.

2) Update the introduction:

In section 2 (Introduction), sixth paragraph

From:

BGP runs over a reliable transport protocol. This eliminates the need to implement explicit update fragmentation, retransmission, acknowledgment, and sequencing. Any authentication scheme used by the transport protocol (e.g., RFC2385 [10]) may be used in addition to BGP's own authentication mechanisms. The error notification mechanism used in BGP assumes that the transport protocol supports a "graceful" close, i.e., that all outstanding data will be delivered before the connection is closed.

To:

BGP uses TCP [RFC793] as its transport protocol. This eliminates the need to implement explicit update fragmentation, retransmission, acknowledgment, and sequencing. BGP listens on TCP port 179. Any authentication scheme used by TCP (e.g., RFC2385 [RFC2385]) may be used. The error notification mechanism used in BGP assumes that TCP supports a "graceful" close, i.e., that all outstanding data will be delivered before the connection is closed.

3) Update the message header format section:

From:

Marker:

This 16-octet field contains a value that the receiver of the message

can predict. If the Type of the message is OPEN, or if the OPEN message carries no Authentication Information (as an Optional Parameter), then the Marker must be all ones. Otherwise, the value of the marker can be predicted by some a computation specified as part of the authentication mechanism (which is specified as part of the Authentication Information) used. The Marker can be used to detect loss of synchronization between a pair of BGP peers, and to authenticate incoming BGP messages.

To:

Marker:

This 16-octet field is included for compatibility; it must be set to all ones.

4) Update the Message Header error handling section:

In section 6.1 (Message Header error handling), second paragraph

From:

The expected value of the Marker field of the message header is all ones if the message type is OPEN. The expected value of the Marker field for all other types of BGP messages determined based on the presence of the Authentication Information Optional Parameter in the BGP OPEN message and the actual authentication mechanism (if the Authentication Information in the BGP OPEN message is present). If the Marker field of the message header is not the expected one, then a synchronization error has occurred and the Error Subcode is set to Connection Not Synchronized.

To:

The expected value of the Marker field of the message header is all ones. If the Marker field of the message header is not as expected, then a synchronization error has occurred and the Error Subcode is set to Connection Not Synchronized.

5) Remove a paragraph from the OPEN message error handling section (section 6.2):

If the OPEN message carries Authentication Information (as an Optional Parameter), then the corresponding authentication procedure is invoked. If the authentication procedure (based on Authentication Code and Authentication Data) fails, then the Error Subcode is set to Authentication Failure.

6) Update the "Differences from RFC1771 Appendix"

Text not listed here

7) Fix the hole in the numbering by updating:

From:

This document defines the following Optional Parameters:

a) Authentication Information (Parameter Type 1):

To:

This document defines the following Optional Parameters:

a) Optional parameter type 1, Authentication Information, has been deprecated.

We are at consensus with these changes.

2.63. Clarify MED Removal Text

Status: Consensus
Change: Yes
Summary: Modify text to clear up MED removal behavior. Text is at the end of the discussion.

Discussion:

This discussion began when Jonathan posted a question to the list:

In reference to:

"A BGP speaker MUST IMPLEMENT a mechanism based on local configuration which allows the MULTI_EXIT_DISC attribute to be removed from a route"

Does anybody know how this can be done in IOS??? Looks like it cannot.

Juniper???

Other code???

Change to "SHOULD"???

Enke responded that:

As the MED value is treated as zero when the MED attribute is missing, removing the MED attribute is really equivalent to setting the value to zero. Based on this logic, one can argue that "MED removal" is supported by multiple vendors.

However, I do see that the current text can be consolidated and cleaned up:

5.1.4 MULTI_EXIT_DISC

...

A BGP speaker MUST IMPLEMENT a mechanism based on local configuration which allows the MULTI_EXIT_DISC attribute to be removed from a route. This MAY be done prior to determining the degree of preference of the route and performing route selection (decision process phases 1 and 2).

An implementation MAY also (based on local configuration) alter the value of the MULTI_EXIT_DISC attribute received over an external link. If it does so, it shall do so prior to determining the degree of preference of the route and performing route selection (decision process phases 1 and 2).

How about this:

A BGP speaker MUST implement a mechanism based on local configuration which allows the value of MULTI_EXIT_DISC attribute of a received route to be altered. This shall be done prior to determining the degree of preference of the route and performing route selection (decision process phases 1 and 2).

In responding to a question, Enke also added:

> Humm. I thought with a missing MED it was preferable to be treated
> as worst not as 0.

It was changed a long time ago. Please see the following text in Sect. 9.1.2.2 of the draft:

In the pseudo-code above, MED(n) is a function which returns the value of route n's MULTI_EXIT_DISC attribute. If route n has no MULTI_EXIT_DISC attribute, the function returns the lowest possible

MULTI_EXIT_DISC value, i.e. 0.

Curtis replied to Enke:

If Juniper treats missing MULTI_EXIT_DISC as worst and Cisco has a knob to treat missing MULTI_EXIT_DISC as worst, then IMHO we should change the spec to say that MED(n) returns the largest value possible is MULTI_EXIT_DISC is missing since this has better loop suppression behavior if the placement of MULTI_EXIT_DISC removal is moved in its position in the flow from Adj-In-RIB to Loc-RIB to Adj-Rib-Out. In other words, no matter where the removal takes place, we are loop free without special rules about comparing EBGP before MED removal and then IBGP after MED removal. The only argument for MED(n) going to zero for missing MULTI_EXIT_DISC was that Cisco routers did that (and change would pose an operational issue if there wasn't a knob to facilitate the change).

Note that when explicitly jamming a MULTI_EXIT_DISC value, such as zero, the issue of where in the decision process the MULTI_EXIT_DISC learned from the EBGP peers could still be used becomes a concern again. Unfortunately these implementation hints are necessary to remain loop free and so its hard to declare them out of scope, unless we forbid changing MULTI_EXIT_DISC and just allow it to be removed (which does not reflect current practice).

Curtis also added:

The other issue was MED handling. In "5.1.4 MULTI_EXIT_DISC":

A BGP speaker MUST IMPLEMENT a mechanism based on local configuration which allows the MULTI_EXIT_DISC attribute to be removed from a route. This MAY be done prior to determining the degree of preference of the route and performing route selection (decision process phases 1 and 2).

An implementation MAY also (based on local configuration) alter the value of the MULTI_EXIT_DISC attribute received over an external link. If it does so, it shall do so prior to determining the degree of preference of the route and performing route selection (decision process phases 1 and 2).

This doesn't sufficiently address the issue.

The MED step in the decision process is (in 9.1.2.2):

c) Remove from consideration routes with less-preferred MULTI_EXIT_DISC attributes. MULTI_EXIT_DISC is only comparable between routes learned from the same neighboring AS. Routes which do

not have the MULTI_EXIT_DISC attribute are considered to have the lowest possible MULTI_EXIT_DISC value.

This is also described in the following procedure:

```
for m = all routes still under consideration
for n = all routes still under consideration
if (neighborAS(m) == neighborAS(n)) and (MED(n) < MED(m))
remove route m from consideration
```

In the pseudo-code above, MED(n) is a function which returns the value of route n's MULTI_EXIT_DISC attribute. If route n has no MULTI_EXIT_DISC attribute, the function returns the lowest possible MULTI_EXIT_DISC value, i.e. 0.

Similarly, neighborAS(n) is a function which returns the neighbor AS from which the route was received.

The problem is that a route loop can be formed.

To avoid the route loop, two suggestions were made (2-3 years ago and nothing was done). One was to make MED(n) return infinity if there was no MULTI_EXIT_DISC. The other was to consider MULTI_EXIT_DISC in the decision process only for the purpose of selecting among the EBGP peers, then remove MULTI_EXIT_DISC, then use that best route in comparisons to IBGP learned routes.

The statement in 5.1.4 "This MAY be done prior to determining the degree of preference of the route and performing route selection (decision process phases 1 and 2)" does not sufficiently address this. This implies that you MAY also remove after route selection, in which case field experience indicates you WILL get burned (unless you know about the caveat above). Initially this came up as an interoperability issue but later it was proven (in the field) that a Cisco and another Cisco could form a route loop until Cisco made this change.

Additional wording is needed either in 5.1.4 or in 9.1.2.2. I suggest we put a forward reference in 5.1.4:

[...]. This MAY be done prior to determining the degree of preference of the route and performing route selection (decision process phases 1 and 2). See section 9.1.2.2 for necessary restricts on this.

Then in 9.1.2.2 add a clarification to the neighborAS(n) function and add to the existing text.

Similarly, neighborAS(n) is a function which returns the neighbor AS from which the route was received. If the route is learned via IBGP, it is the neighbor AS from which the other IBGP speaker learned the route, not the internal AS.

If a MULTI_EXIT_DISC attribute is removed before redistributing a route into IBGP, the MULTI_EXIT_DISC attribute may only be considered in the comparison of EBGP learned routes, them removed, then the remaining EBGP learned route may be compared to the remaining IBGP learned routes, without considering the MULTI_EXIT_DISC attribute for those EBGP learned routes whose MULTI_EXIT_DISC will be dropped before advertising to IBGP. Including the MULTI_EXIT_DISC of an EBGP learned route in the comparison with an IBGP learned route, then dropping the MULTI_EXIT_DISC and advertising the route has been proven to cause route loops.

The loop is the classic I prefer your and you prefer mine problem. It occurs when the router is configured to remove MULTI_EXIT_DISC and advertise out a route so other routers can use IGP cost to select the best route. If two routers do this, as soon as they hear the route with no MULTI_EXIT_DISC from the other peer they start forwarding toward that peer but they continue to advertise to it since others IBGP peers are expected to select among the MULTI_EXIT_DISC free IBGP learned routes using the next step in the decision process, IGP cost.

In this case, what you want is each router to prefer its own EBGP route, even though it has a MULTI_EXIT_DISC and the IBGP learned route from the same neighbor AS has had its MULTI_EXIT_DISC stripped off (or didn't have one, you can't tell which it is). You then want all of the IBGP peers with a MULTI_EXIT_DISC free route from that AS, or that have stripped the MULTI_EXIT_DISC to form one, to advertise them. Others in the AS will then use IGP cost to further resolve which exit point to use. It make a difference when the route is the aggregate route of another major provider. IGP cost yields what ISPs call "hot potatoe routing" and MED selects among multiple heavily loaded provider interconnects.

[Aside: Having a missing MULTI_EXIT_DISC default to infinity would do exactly what the ISPs want it to do in the first place and be a lot easier to explain but we didn't fix that 2-3 years ago when the issue came up even though we had WG consensus that it was the right thing to do. The authors didn't act on it at the time (because Cisco didn't do it that way and the authors preferred to sit on the draft). At this point we might as well adequately document what we ended up with.... End of soap box. I don't take myself all that seriously so others shouldn't either. :-)]

After some more discussion on this, we have this text:

In 5.1.4 replace:

An implementation MAY also (based on local configuration) alter the value of the MULTI_EXIT_DISC attribute received over EBGP. If it does so, it shall do so prior to determining the degree of preference of the route and performing route selection (decision process phases 1 and 2).

with:

An implementation MAY also (based on local configuration) alter the value of the MULTI_EXIT_DISC attribute received over EBGP. This MAY be done prior to determining the degree of preference of the route and performing route selection (decision process phases 1 and 2). See section 9.1.2.2 for necessary restricts on this.

In 9.1.2.2 replace:

Similarly, neighborAS(n) is a function which returns the neighbor AS from which the route was received.

with:

Similarly, neighborAS(n) is a function which returns the neighbor AS from which the route was received. If the route is learned via IBGP, and the other IBGP speaker didn't originate the route, it is the neighbor AS from which the other IBGP speaker learned the route. If the route is learned via IBGP, and the other IBGP speaker originated the route, it is the local AS.

If a MULTI_EXIT_DISC attribute is removed before re-advertising a route into IBGP, the MULTI_EXIT_DISC attribute may only be considered in the comparison of EBGP learned routes, then removed, then the remaining EBGP learned route may be compared to the remaining IBGP learned routes, without considering the MULTI_EXIT_DISC attribute for those EBGP learned routes whose MULTI_EXIT_DISC will be dropped before advertising to IBGP. Including the MULTI_EXIT_DISC of an EBGP learned route in the comparison with an IBGP learned route, then dropping the MULTI_EXIT_DISC and advertising the route has been proven to cause route loops.

There have been no objections to this, so we are at consensus.

2.64. MED for Originated Routes

Status: Consensus
Change: No
Summary: The consensus is that there is not need to specify default

values for MED in the base draft.

Discussion:

This issue began when it was pointed out that we do not specify the default values for MED in the base draft.

There were a variety of responses, but the consensus is that since this is not relevant for interoperability, this does not need to be in the base spec.

2.65. Rules for Aggregating with MED and NEXT_HOP

Status: Consensus
Change: Yes
Summary: Clear up the text on aggregating with a MED. See the discussion for the text.

Discussion:

There is a proposal to relax this statement:

"Routes that have the following attributes shall not be aggregated unless the corresponding attributes of each route are identical: MULTI_EXIT_DISC, NEXT_HOP."

In his reply to the original mail, Curtis asserted that we should leave the MED rules alone, but perhaps we could relax the NEXT_HOP statement.

This was revisited in the "aggregating with MED and NEXT_HOP" thread.

Yakov suggested we replace:

Routes that have the following attributes shall not be aggregated unless the corresponding attributes of each route are identical: MULTI_EXIT_DISC, NEXT_HOP.

If the aggregation occurs as part of the update process, routes with different NEXT_HOP values can be aggregated when announced through an external BGP session.

Path attributes that have different type codes can not be aggregated together. Path attributes of the same type code may be aggregated, according to the following rules:

with:

Routes that have different MULTI_EXIT_DISC attribute SHALL NOT be aggregated.

Path attributes that have different type codes can not be aggregated together. Path attributes of the same type code may be aggregated, according to the following rules:

NEXT_HOP: When aggregating routes that have different NEXT_HOP attribute, the NEXT_HOP attribute of the aggregated route SHALL identify an interface on the router that performs the aggregation.

This met with agreement.

Dimitry asked if the "Routes that have different MULTI_EXIT_DESC attribute SHALL NOT be aggregated." sentence was unnecessary since it should be a matter of local policy. Jeff replied that it has been mentioned that removing this stipulation can cause routing loops.

We are at consensus with this issue.

2.66. Complex AS Path Aggregating

Status: Consensus
Change: No
Summary: Since we have two implementations of this method, section 6.8 stays in the specification.

Discussion:

Jonathan opened this discussion with:

The part in the draft about complex AS path aggregation could/should be deleted. The current draft implies that when aggregating, for example (1st):

1 2 4 3

w/

3 4 6 5

and

5 6 7 8

then

1 2 {3 4 5 6} 7 8

...would be OK

AFAIK, all implementations aggregate by either: (2nd)putting ONLY the local AS in the path (and setting the atomic) OR (3rd)by creating 1 giant set and adding the local AS as a seq.

So he proposed we remove this to reflect current code.

Jeff replied that there is absolutely nothing wrong with doing complex aggregation, and there is no reason to remove this from the specification.

Yakov responded that:

Jonathan is certainly correct that the spec has to reflect current code. Therefore, unless there are at least two (interoperable) implementations that implement the algorithm described in "6.8 Complex AS_PATH aggregation", this section has to be removed (this is irrespective of whether there is something wrong, or nothing wrong with complex aggregation). With this in mind, if you implement this algorithm please speak up within a week. If within a week we wouldn't know that there are at least two implementations the section will be removed. And likewise, if we hear that there are at least two implementations, the section will stay.

Jeff replied:

I am also fine with removing it. I just don't think its necessary, especially if One Of These Days someone decides that running policy based on as-adjacencies would be a Nice Thing and fixes their implementation to support "complex" aggregation of paths.

That said, I am aware of no one who implements this.

As an aside, in the thread "last thought on complex aggregation" Jeff supplied:

I finally remembered what was bothering me about removing complex aggregation from the spec.

If it is removed, people who do conformance tests and some implementations may take this to mean "it shall be illegal to have an AS_PATH that contains a SEQUENCE of a particular type after a SET".

This would make a perfectly ok AS_PATH into one that isn't legal, even if no implementation currently generates it.

Jonathan replied that he thought the issue was moot since no one has

implemented this.

John replied that, although he is a bit uncomfortable in removing this from the spec, he doesn't see any harm in doing so.

With this in mind, Yakov suggested we consider the issue closed.

So we will wait a week from 10/17 to see if anyone chimes in.

Siva responded that they have implemented this functionality. So we need one more...Ben responded that they support this at Marconi as well.

So we have two implementations, the section stays in the spec. We are at consensus with this issue.

2.67. Counting AS_SET/AS_CONFED_*

Status: Consensus
Change: Yes
Summary: Move how AS_CONFED_SET & SEQUENCE affect route selection to the BGP Confederations document. Update the base draft to reflect this by changing section 9.1.2.2. Specific text is at the end of the discussion.

Discussion:

Jonathan brought up some questions on how current implementations count AS_SET and AS_CONFED_*

There were a variety of responses to this, answering his questions. Curtis pointed out that this behavior is covered in:

That's in 9.1.2.2:

a) Remove from consideration all routes which are not tied for having the smallest number of AS numbers present in their AS_PATH attributes. Note, that when counting this number, an AS_SET counts as 1, no matter how many ASs are in the set, and that, if the implementation supports [13], then AS numbers present in segments of type AS_CONFED_SEQUENCE or AS_CONFED_SET are not included in the count of AS numbers present in the AS_PATH.

Jonathan replied that this might be at odds with what Juniper does, although he was unsure, and asked for clarification.

This was discussed in the "New Issue AS path" thread.

Yakov proposed that:

The issue of route selection in the present of confederations belongs *not* to the base spec, but to the spec that describes BGP Confeds. With this in mind I would suggest to change in 9.1.2.2 from

a) Remove from consideration all routes which are not tied for having the smallest number of AS numbers present in their AS_PATH attributes. Note, that when counting this number, an AS_SET counts as 1, no matter how many ASs are in the set, and that, if the implementation supports [13], then AS numbers present in segments of type AS_CONFED_SEQUENCE or AS_CONFED_SET are not included in the count of AS numbers present in the AS_PATH.

to

a) Remove from consideration all routes which are not tied for having the smallest number of AS numbers present in their AS_PATH attributes. Note, that when counting this number, an AS_SET counts as 1, no matter how many ASs are in the set.

and ask the authors of BGP Confeds to update their document to cover how the presence of confeds impact route selection.

This met with agreement, this issue is at consensus.

2.68. Outbound Loop Detection

Status: Consensus
Change: No
Summary: The consensus is, that while this may be a useful technique, it would break existing methods, and is otherwise out-of-scope for the base draft. It was suggested that this could be addressed in the update to RFC1772.

Discussion:

Jonathan brought up that:

This paper (thanks Jeff) http://www.research.microsoft.com/scripts/pubs/view.asp?TR_ID=MSR-TR-2000-08 indicates that it is better to do the loop detection outbound as well inbound. The current draft indicates that it only needs to be done inbound. IOS does it inbound as well as outbound. GateD/Juniper does it (did it ???) only inbound.

So I propose we add: "An implementation MAY choose to not advertise routes to EBGP peers if these routes contain the AS of that peer in

the AS path." after: "If the autonomous system number appears in the AS path the route may be stored in the Adj-RIB In, but unless the router is configured to accept routes with its own AS in the AS path, the route shall not be passed to the BGP Decision Process."

If there is at least one other implementation that does outbound pruning/loop-detection.

Yakov pointed out that this is ONLY applicable to the base draft if there are multiple implementations doing this. This issue will only be considered if these implementors come forward by 10/25/02. Otherwise the issue will be considered closed.

Jeff replied that there was more at stake with this than if people had implemented it:

My suggestion is that this can stay out of the base draft. While it is true that this speeds up convergence (per the paper), it doesn't affect interoperability.

.in 4 Also, adding this specifically removes the ability from several implementations to be able to bridge a partitioned AS by permitting loops. (I'm not going to argue whether this is a Good way to do this, just that its done.)

Its also worth noting that one could produce the same resultant speed-up by detecting the loop on an outbound basis and *not* applying the min*timers to the UPDATE. Thus, this would be a case of an advertisement of NLRI being treated the same, timer-wise, as the advertisement of WD_NLRI.

I would suggest moving this suggestion for outbound loop detection in one form or another to the 1772 replacement.

Yakov agreed with this.

2.69. Appendix A - Other Documents

Over the course of this discussion, a number of issues have been raised that the group though would be better dealt with in other documents. These additional documents, and their concomitant issues will be more fully addressed when the WG turns its focus to them. These projects are:

1) Update RFC 1772: Application of the Border Gateway Protocol in the Internet. This will probably entail a complete rewrite.

2) Update Route Reflector (2796) and Confederation (3065) RFC's

regarding their impact on route selection.

3) Write a new document covering BGP Multipath. .ne 4

3. The Issues from -18 to -19

This section lists the issues discussed on the list from November 2002 to late February 2003.

3.1. Reference to RFC 1772

Status: Consensus
Change: No
Summary: Proposed changing RFC 1772 reference, since that document should be updated.

Discussion:

Jeff proposed that we reconsider referencing RFC 1772, since that document should be updated.

Yakov pointed out that this is a non-normative reference and can just be left as is.

Jeff agreed that this wasn't a big deal. We are at consensus to leave things as they are.

This was discussed in the "-18 last call comments" thread.

3.2. MUST/SHOULD Capitalization

Status: Consensus
Change: Yes
Summary: Capitalize MUST/SHOULD where appropriate.

Discussion:

Jeff brought this up, and Yakov responded asking that he point out specific instances where this is needed. Jeff said he would do so, given some time.

Yakov later replied that this would be fixed in the -19 version.

Jeff replied with a master diff showing the MUST/SHOULDs, for the entire document please see the beginning of the thread entitled: "Issues list, #2: MUST/SHOULD Capitalization"

This was discussed in the "18 last call comments" thread. This was also brought up in the "proxy: comments on draft -18" thread.

3.3. Fix Update Error Subcode 7 -- accidently removed

Status: Consensus
Change: Yes
Summary: Add error subcode 7 back in, it looks like it was inadvertently removed. Add deprecation text to Open Message Error subcode 5.

Discussion:

Jeff supplied:

Update message error subcode 7 is removed. Especially in -18, it looks like an editing mistake based on where it would fall in the editing..

Yakov mentioned that this is addressed in Appendix A.

Jeff replied:

.in 4 What I would like to see is something like this:

6 - Invalid ORIGIN Attribute 7 - [Deprecated - See Appendix A] 8 - Invalid NEXT_HOP Attribute

As it stands, 7 lies on a page boundary and looks like it got clipped by the roff.

Yakov agreed, and also said he would add similar text for Open Message Error subcode 5.

This was discussed in the "18 last call comments" thread.

3.4. Section 5.1.4 - Editorial Comment

Status: Consensus
Change: Yes
Summary: Fix "restricts" to "RESTRICTIONS"

Discussion:

Jeff proposed an editorial fix. This is agreed to.

This was discussed in the "-18 last call comments" thread.

3.5. Section 9.1 - Change "all peers" to "peers"

Status: Consensus
Change: Yes
Summary: Section 9.1 - Change "all peers" to "peers"

Discussion:

Jeff proposed:

.in 4 9.1: The output of the Decision Process is the set of routes that will be advertised to (delete all) peers; the selected routes will be stored in the local speaker's Adj-RIB-Out according to policy.

The previous wording implied that routes in the LocRib MUST be placed in the adj-rib-out.

Yakov agreed, this fix will be in the next revision.

This was discussed in the "-18 last call comments" thread.

3.6. AS Loop Detection & Implicit Withdraws

Status: Consensus
Change: Yes
Summary: Update the text to reflect the AS Loop detection should be done in the BGP decision process.

Discussion:

John brought this up, and suggested:

.in 4 I have one further comment just in case it's not perfectly obvious to everyone, which is that "ignore the UPDATE" is not strictly the action you take when receiving a looped update. Rather, you treat it as an implicit withdraw, i.e. you process it as any other update but treat the contained NLRI as unfeasible.

I was going to write that this is sufficiently clear from the spec, but I regret to say that it isn't. Here is the fourth paragraph of section 9:

.in 8 The information carried by the AS_PATH attribute is checked for AS loops. AS loop detection is done by scanning the full AS path (as specified in the AS_PATH attribute), and checking that the autonomous system number of the local system does not appear in the AS path. If the autonomous system number appears in the AS path the route may be

stored in the Adj-RIB-In, but unless the router is configured to accept routes with its own autonomous system in the AS path, the route shall not be passed to the BGP Decision Process. Operations of a router that is configured to accept routes with its own autonomous system number in the AS path are outside the scope of this document.

.in 4 I don't think this is quite right -- the decision process needs to be run if the looped routes had previously been advertised feasibly on the same session. This could be fixed by hacking the quoted paragraph, but it seems more straightforward to do it by removing the quoted paragraph and making the fix in 9.1.2 Phase 2 instead. This could be done by inserting the following between the third and fourth paragraphs of 9.1.2 Phase 2:

.in 8 If the AS_PATH attribute of a BGP route contains an AS loop, the BGP route should be excluded from the Phase 2 decision function. AS loop detection is done by scanning the full AS path (as specified in the AS_PATH attribute), and checking that the autonomous system number of the local system does not appear in the AS path. Operations of a router that is configured to accept routes with its own autonomous system number in the AS path are outside the scope of this document.

.in 4 Section 9.3, first bullet, also addresses this topic, but I don't think it's sufficient.

Yakov agreed that this was a change for the better and will include this in the next revision.

We are at consensus on this issue.

This is discussed in the "-18 last call comments" thread.

3.7. Standardize FSM Timer Descriptions

Status: Consensus
Change: Yes
Summary: Standardize the state descriptions on those listed in the discussion section of this issue.

Discussion:

Tom proposed:

.in 4 I think a standard description would serve us better instead of using the following different ways (which I take all to refer to the same entity):

delayBGP open timer BGP delay open timer BGP open delay timer delay open timer BGP delay timer

.in 4 I suggest Open Delay timer (with those capitals)

I believe that the corresponding flag is consistently referred to (apart from the capitalization) as Delay Open flag

Yakov agreed with this suggestion, no one else disagreed, we are at consensus.

This was discussed in the "BGP18-FSM-terminology" thread.

3.8. FSM MIB enumerations

Status: Consensus
Change: Yes
Summary: Move MIB references from the base spec into the MIB document.

Discussion:

Tom pointed out that:

The FSM makes several references to putting values into MIB objects and while some of the values are defined, eg FSM error or Hold Timer expired, I can find no definition of the following in any of the BGP documents, MIB or otherwise.

connect retry expired TCP disconnect administrative down collision detect closure Call Collision cease collision detected and dump connection Administrative stop

I believe an implementation needs to be told these values somewhere and that there should be a reference to that place in bgp18.

Jeff replied that to make things easier, the MIB references will be removed from the base spec, and into the MIB document.

This was discussed in the "WG Last Call FSM MIB enumeration" thread, and the "bgp18 WG Last Call fsm MIB objects" thread.

3.9. Make "delete routes" language consistent

Status: Consensus
Change: Yes
Summary: Replace a variety of wording with "deletes all routes associated with this connection,".

Discussion:

Tom pointed out that we use a variety of language to say how we are going to delete routes in the FSM. He proposed that we instead use:

- deletes all routes associated with this connection,

This met with agreement, and will be reflected in the next version.

This was discussed in the "bgp18 WG Last Call fsm delete action" thread.

3.10. Correct OpenSent and OpenConfirm delete wording

Status: Consensus
Change: Yes
Summary: Remove delete wording from OpenSent and OpenConfirm states.

Discussion:

Venu asked why there was delete wording in the OpenSent and OpenConfirm states when a BGP speaker cannot receive routes in these states.

Jeff acknowledged that this was an error. Yakov agreed to fix the next version.

This was discussed in the "bgp18 WG Last Call fsm delete action" thread.

3.11. Incorrect next state when the delay open timer expires

Status: Consensus
Change: Yes
Summary: Fix the next state.

Discussion:

Tom pointed out that:

I believe that there is an incorrect next state when the delay open timer expires [event 12] in the Active state. The next state should be OpenSent and not OpenConfirm.

OpenConfirm is for KeepAlive processing when Open messages have been sent and received.

OpenSent is for Open sent and not yet received.

The corresponding section in Connect state I believe is correct.

Yakov agreed, and will fix this in the next revision.

This was discussed in the "bgp18 WG Last Call fsm incorrect next state"

3.12. Entering OpenConfirm / Adding "Stop OpenDelay" action

Status: Consensus
Change: Yes
Summary: Add this text:

Change 2 - Connect state event 17 (currently defined as going to Active) event 9 (stays in Connect state)

new Text:

In response to the connect retry timer expires event [Event 9], the local system: - drops the TCP connection, - restarts the connect retry timer, - stops the Open Delay timer and resets the timer to zero, - initiates a TCP connection to the other BGP peer, - continues to listen for a connection that may be initiated by the remote BGP peer, and - stays in Connect state.

If the TCP connection fails [Event17], the local system: - restarts the connect retry timer, - stops the Open Delay timer and resets value to zero, - continues to listen for a connection that may be initiated by the remote BGP peer, and - changes its state to Active.

Further discussion on Keepalives has been moved to issue 52.

Discussion:

This discussion began with Tom outlining these two points:

When the OpenConfirm state is entered from OpenSent with the receipt of a valid open [Event 18], then a KeepAlive message is sent and the timer is started.

When the OpenConfirm state is entered from Active or Connect on receipt of a valid open [Event 19], no message is sent, no timer is started. I believe this inconsistency is an error and should be corrected by adding these two actions in those two places.

Sue replied:

Just to clarify this comment: Event 19 = valid open with delay timer

running

Active = 1) awaiting TCP connection, or 2) TCP connection completed and awaiting the TCP connection with delay timer running

Case 1: - should not see Event 19 In transition from Active to Open Confirm, the connection must have a TCP connection completed. Case 1 does not have this occurring, so the transition must be avoided.

Case 2: - should see Event 19

- Open, Keepalive should be sent.

Previous text: (Action H from FSM document)

If an Open is received with the BGP Delay Open timer running, [Event 19], the local system: - clears the connect retry timer [cleared to zero), - completes the BGP initialization, - stop and clears the BGP Open Delay timer, - Sends an Open Message, - sets the hold timer to a large value (4 minutes), and - changes its state to an Open Confirm.

New text: [a New Action - N-2 : N + BGP keepalive sent]

If an Open is received with the BGP Delay Open Timer running [Event 19], the local system: - clear the connect retry timer [cleared to zero], - completes the BGP initialization, - stops and clears the BGP Open Delay timer, - Send an Open message, - Sends a Keepalive message, - If hold timer value is non-zero, - set keepalive timer - hold timer reset to negotiated value else if hold timer is zero, - reset the keepalive timer, and - reset the hold timer.

- If the value of the autonomous system field is the same as the local Autonomous system number, set the connection status to a internal connection; otherwise it is "external".

Tom and Sue discussed the OpenDelay state, and recalled that this was excluded a number of months ago as not reflecting current practice.

By way of clarification, Sue added:

1) Agree, this can occur in the Active state as well as the Connect state. Will you accept the earlier text below to be inserted both places?

Background:

The state machine for Event 19 is:

Idle Connect Active Open Open Estb Sent Confirm
== Event 19 | | | | | |
| next state |Idle | Open | Open | Open |Idle | Idle | | | confirm|
confirm| Confirm| | |
== action | V | N-2 |
N-2 | N | E-1 | E | ==

Per the State Machine.

Action v - FSM Error Action E - FSM Error, drop connection - etc, drop routes Action E-1 - FSM Error, drop connection (lots of Action N-2 (text below) Action N (text below, without sending Open)

2) Do you think that Event 19 is possible in the Open Sent state?

Please answer this separately.

Tom replied that:

.in 4 1) yes I think the same text in both Active and Connect states is a good resolution

2) complicated. As the fsm text stands, Event 19, along with a host of others, takes us back from Open Sent to Idle (I assume on the grounds this is an error condition) which seems very reasonable.

But ...in quite a few places, such as Connect state events 2, 7,8,9,10,11, 17, 18, 20 thru 27, we do not stop the OpenDelay timer when going to Idle or Active so we could then go from eg Idle with Manual start [event 1] to Connect to Open Sent all before the OpenDelay timer expires in which case event 19 can occur validly in Open Sent - obscure but possible. (This is also true with Active state and events 2, 17 and the default list at the end).

But I think this is an error, and that when exiting Connect state or Active state as listed above, we should have an additional action to stop the OpenDelay timer in which case event 19 in Open Sent becomes an error condition (again).

But but but as ever, I cannot speak with authority for implementations and so if implementations do not stop the OpenDelay timer when exiting as above, then Event 19 is valid in Open Sent state - obscure but possible (again).

My wish is to add the extra action, stop OpenDelay timer, for the events listed above in Active and Connect states in the expectation that that is what people have or should have implemented.

Tom added a response to Sue after some other threads have been discussed: .in 4

You asked if event 19 (Open with OpenDelay timer running) was possible in OpenSent state; I gave a lengthy reply (below) to the effect yes it could because the OpenDelay timer did not always get stopped but the timer should be stopped in which case the event would not happen.

Reading your responses to Siva , I see you include stopping the Open Delay timer in the action 'release all BGP resources' when going to Idle (which I missed seeing earlier in the year).

That eliminates most but not all of the possibilities I mentioned. I now believe we would need to add the action 'stop OpenDelay timer' for Connect state event 17 (currently defined as going to Active) event 9 (stays in Connect state)

in order to stop event 19 in Open Sent

Sue replied that, she thought this was at consensus, and provided the new text, which is:

Change 1: new text

Active state - event 19

If an Open is received with the Open Delay timer is running [Event 19], the local system - clears the connect retry timer (cleared to zero), - stops and clears the Open Delay timer - completes the BGP initialization, - stops and clears the Open Delay timer - sends an OPEN message, - send a Keepalive message, - if the hold timer value is non-zero, - starts the keepalive timer to initial value, - resets the hold timer to the negotiated value, else if the hold timer is zero - resets the keepalive timer (set to zero), - resets the hold timer to zero.

- changes its state to OpenConfirm.

If the value of the autonomous system field is the same as the local Autonomous System number, set the connection status to an internal connection; otherwise it is "external". Change 2 - Connect state event 17 (currently defined as going to Active) event 9 (stays in Connect state)

new Text:

In response to the connect retry timer expires event [Event 9], the

local system: - drops the TCP connection, - restarts the connect retry timer, - stops the Open Delay timer and resets the timer to zero, - initiates a TCP connection to the other BGP peer, - continues to listen for a connection that may be initiated by the remote BGP peer, and - stays in Connect state. If the TCP connection fails [Event17], the local system: - restarts the connect retry timer, - stops the Open Delay timer and resets value to zero, - continues to listen for a connection that may be initiated by the remote BGP peer, and - changes its state to Active.

Tom replied that:

.in 4 Change 2, stop Open Delay timer in Connect state events 9 and 17, fine; that is what I understand to be the real issue 12.

Change 1, event 19 in Active state, is IMHO issues 47 and 52. This is tangled because the initial paragraphs of Issue 12 in the issue list are nothing to to with stopping Open Delay timer and everything to to with sending a Keepalive message before entering Open Confirm state from Active or Connect state on event 19; which I raised and see as issue 52. Issue 47 was Siva's issue 28 and relates to a different action for Active state event 19.

I agree with change 1 in that it adds in the sending of Keepalive which I believe essential; I think Siva needs to respond concerning issue 47. (nb the stop Open Delay action is duplicated) I wonder if we should use a different character for the bullet points under the if and else clauses to make it clear where they end ie - if the hold timer .... + do this + and this else if ... + do the other + and this

But I still have an issue for Connect state event 19 where I believe, as for Active state event 19, we should send a Keepalive and start the Keepalive timer. I will pursue this as part of issue 52 if that suits you. I think the text will be the same as whatever we agree for Active state event 19.

This was discussed in the "bgp 18 WG Last Call fsm missing keepalive" thread. And also in the "Event 19 in Open Sent state was Re: bgp18 WG Last Call fsm missing keepalive" thread. This also came up in the "issues 12 - consensus & two changes - 2nd message" thread.

3.13. FSM Missing Next States

Status: Consensus
Change: Yes
Summary: Seven sub-issues spawned to resolve each of the next-state questions. See each sub-issue for specifics.

Discussion:

This began with Tom pointing out 7 places where the next state was not clear. Interlaced with his comments below is the proposed text to fix the problems and the status of the issue.

All sub-issues are at consensus.

This conversation was started in the "bgp18 WG Last Call fsm missing next state" thread.

3.13.1. FSM Missing Next States - Event 15 or 16 (Connect State)

Status: Consensus
Change: Yes
Summary: Add next state of Connect.

Discussion:

Tom pointed out that:

Connect State:

If the TCP connection succeeds [Event 15 or Event 16], the local system checks the "Delay Open Flag". If the delay Open flag is set, the local system: **enters what state

Sue proposed these changes:

1) Connect State - Event 15 or Event 16 [consensus, editorial]

note: The delay retry timer is utilized instead of the connect retry timer for the next two changes. Previous text:

If the TCP connection succeeds [Event 15 or Event 16], local system checks the "Delay Open Flag". If the delay open flag is set, the local system: - clears the connect retry timer, - sets the BGP open delay timer to initial value If the Delay Open flag is not set, the local system: - clears the connect retry timer, - clear BGP Open Delay timer (set to zero), - completes the BGP initialization, - send an Open message to its peer, - sets hold timer to a large value, and - change the state to Open Sent.

New text:

If the TCP connection succeeds [Event 15 or Event 16], local system checks the Delay Open flag prior to processing: If the Delay Open flag is set, the local system: - clears the connect retry timer, -

sets the BGP open delay timer to initial value, and - stays in the Connect state.

If the Delay Open flag is not set, the local system: - clears the connect retry timer, - clears the BGP Delay timer (sets to zero), - completes the BGP initialization, - sends an Open message to its peer, - sets the hold timer to a large value, and - changes the state to Open Sent.

Tom agreed that this was good, with the change to "Open Delay timer" as discussed in issue 7.

This conversation was started in the "bgp18 WG Last Call fsm missing next state" thread.

3.13.2. FSM Missing Next States - Event 14 (Connect State)

Status: Consensus
Change: Yes
Summary: We selected option 2 from discussion as the correct text:

2) treat it as an invalid response, reject the connection and see if a valid configured one comes within the connect timer's window.

Discussion:

Tom pointed out that:

Connect State:

If the TCP connection receives an indication that is invalid or unconfigured. [Event 14]: **enters what state

Sue proposed these alternatives:

2)Connect State - Event 14 [no consensus]

Current Text: If the TCP connection receives an indication that that is invalid or unconfigured [Event 14], - the TCP connection is rejected. At the very least this section needs more "word smithing", so I'd like to change it for more clarity at least.

I'm not sure this represents the implementations. What I'd like to do is query the implementations to see what they do if they receive a valid TCP connection with an invalid or unconfigured peer. Two options: Alternative 1: Count it as a valid response New Text: If a TCP connection is received that has an invalid format, or an unconfigured host [Event 14], the local system: - rejects the TCP

connection, - increments the connect retry counter, - performs bgp peer oscillation checks. If bgp peer oscillation checks allow for a new connection, the bgp peer - restarts the Connect retry timer with configured value, and - enters the Active state. FSM table: Idle Connect Active Open-Sent Open-Confirm Establish Event-14 === Next state Idle | Active|Active|Open-Sent|Open-Confirm|Establish| === action V | Y2 | L | Ignore | Ignore | Ignore | === Alternative 2: Reject the connection and see if valid or configured one appears within the connect timer window.

New Text: If a TCP connection is received that has an invalid format, or an unconfigured host [Event 14], the local system: - rejects the TCP connection, - and stays in the Connect state.

FSM table: Idle Connect Active Open-Sent Open-Confirm Establish Event-14 === Nxt state Idle |Connect|Active|Open-Sent|Open-Confirm|Establish| === action V | L | L | Ignore | Ignore | Ignore | ===

Sue then sent out a call to implementors to let the list know what they did with their FSMs. Tom replied that he agreed that we need to wait to see what the existing implementations do. He also suggested:

**tp need a then clause here 'if bgp peer oscillation damping does not allow for a new connection, then the local system ???'

be added before the FSM table in option 1 of the proposed text.

Sue prodded the list saying that:

Should the peer: 1) Treat it as a valid response, and enters the active state to watch for a another TCP connection with a valid peer.

2) treat it as an invalid response, reject the connection and see if a valid configured one comes within the connect timer's window.

Without further input, I will select option 2.

Curtis replied that this was fine with him.

There has been no further disagreement, we are at consensus on this.

This conversation was started in the "bgp18 WG Last Call fsm missing

next state" thread. It was also discussed in the "BGP draft-19 - FSM input needed from developers" thread.

3.13.3. FSM Missing Next States - Event 15 or 16 (Active State)

Status: Consensus
Change: Yes
Summary: Add text listed in discussion.

Discussion:

Tom pointed out:

Active State:

A TCP connection succeeds [Event 15 or Event 16], the local system: process the TCP connection flags - If the BGP delay open flag is set: ** enters what state (I think this is an FSM error in TCP because it has not initiated a connection!)

Sue proposed these changes:

Previous text: A TCP connection succeeds [Event 15 or Event 16], the local system: process the TCP connection flags - If the BGP delay open flag is set: - clears the connect retry timer [through the following text: - and changes its state to Open Sent.

New text: If the TCP connection succeeds [Event 15 or Event 16], local system checks the "Delay Open Flag" prior to processing: If the delay open flag is set, the local system: - clears the connect retry timer, - sets the BGP open delay timer to initial value, and - stays in the Active state.

If the Delay Open flag is not set, the local system: - clears the connect retry timer, - clears the BGP Delay timer (sets to zero), - completes the BGP initialization, - sends an Open message to its peer, - sets the hold timer to a large value, and - changes the state to Open Sent.

Tom agreed with this.

This conversation was started in the "bgp18 WG Last Call fsm missing next state" thread.

3.13.4. FSM Missing Next States - Event 13-17 (TCP Connection)

Status: Consensus
Change: Yes

Summary: We selected:

Choice 2: Event 13 and Event 14 be optional, and Events 15 - 17 be mandatory.

Discussion:

Tom started this by saying that:

If the local system receives a valid TCP Indication [Event 13], the local system processes the TCP connection flags. ** enters what state

If the local system receives a TCP indication that is invalid for this connection [Event 14]: ** enters what state

Sue proposed we move this to the "fsm missing next state - Events 13-17 and the TCP connection" thread.

The response in this thread was:

4) Active State, Event 13 [no consensus] 5) Active State, Event 14 [no consensus]

The problem with this state is it is difficult to exactly specify without discussing the TCP Messages that FSM document covers. I'll query if the implementors require all of events 13-17 as mandatory.

Sue polled the implementors on the list with this query:

These events are described in section 8.1.3.

In our discussion in January through May of 2002, many implementers mapped their implementation onto the following TCP events list in 8.1.3.

Events 13 - 17

Event 13 - TCP connection indication & valid remote peer

Event 14 - TCP connection indication with invalid source or destination Event 15 - TCP connection request sent (by this peer) received an Acknowledgement

[local system sent a TCP SYN, Received a TCP SYN, ACK pair back, and Sent a TCP ACK]

Event 16 - TCP connection confirmed

[local system received a TCP SYN, sent a TCP SYN, ACK back, and received a TCP ACK]

Event 17 - TCP connections

Should we have all of these states? Which implementations support all of these Events?

The full FSM text was snipped here for brevity.

Sue prodded the list with:

Do the implementors require Events 13 - 17 in the State machine ?

Event 13 - TCP connection valid indication Event 14 - TCP connection invalid indication Event 15 - TCP connection request acknowledged Event 16 - TCP connection confirmed Event 17 - TCP connection fails

Choice 1: Events 13 - 17 are mandatory Choice 2: Event 13 and Event 14 be optional, and Events 15 - 17 be mandatory. If no one objects, we will use Choice 2.

Curtis said this was fine with him.

There has been no further disagreement, we are at consensus on this. This was started in the "bgp18 WG Last Call fsm missing next state" thread. And continued in the "fsm missing next state - Events 13-17 and the TCP connection" thread. It was also discussed in the "BGP draft-19 - FSM input needed from developers" thread.

3.13.5. FSM Missing Next States - Event 17 (Connect State)

Status: Consensus
Change: No
Summary: Closed in favor of 13.4

Discussion:

If the local system receives a TCP connection failed [Event 17] (timeout or receives connection disconnect), the local system will:
** enters what state

Sue replied with this:

.in 4 comment: In the Active state, we may already have a connection and be awaiting the Open Delay timer. The TCP disconnect or timeout could occur in this state due to the "Open Delay Timer". If the TCP Disconnect is ignored, we could have some peer oscillation.

If the we wait, then the connection retry timer needs to be kept running. The text below allows this timer. The real question is what is the status of the current implementations.

I agree, the Active state and the connect state should match.

Old Text: If the TCP connection fails (timeout or disconnection) [Event 17], the local system: - set TCP disconnect in the MIB reason code, - restart Connect retry timer (with initial value), - release all BGP resources, - Acknowledge the Drop of the TCP connection if TCP disconnect (FIN ACK), - Increment ConnectRetryCnt (connect retry count) by 1, and - performs the BGP peer oscillation damping process.

Applicable FSM State table: FSM table old: Event 17 current: Idle Connect Active Open-Sent Open-Confirm Establish === Next state Idle |Active |Idle |Active | Idle |Idle | | | | | | | === action V | Y2 | G | Ignore| Track 2nd | Track 2nd | | | | | connection | connection| ===

Alternative 1:

FSM table new:

Event 17 current: Idle Connect Active Open-Sent Open-Confirm Establish === Next state Idle |Active |Active |Active | Idle |Idle | | | | | | | === action V | G | G | Ignore| Track 2nd | Track 2nd | | | | | connection | connection| === G: The local system: - restarts the connect retry timer (at initial value), - continues to listen for a connection that may be initiated by the remote peer, and - sets its next state to Active.

New Text: (for Connect and Active state) If the TCP connection fails (timeout or disconnect) [Event 17], the local system: - restarts the connect retry timer, - continues to listen for a connection that may be initiated by the remote BGP peer, and - changes it state to Active. Alternative 2: FSM table new: Event 17 current: Idle Connect Active Open-Sent Open-Confirm Establish === Next state Idle |Idle |Idle |Active | Idle |Idle | | | | | | | === action V | Y2 | Y2 | Ignore| Track 2nd | Track 2nd | | | | | connection | connection| === Next Text: If the location system receives a TCP connection failed [Event 17], the local system will: - increment the ConnectRetryCnt

(connect retry count) by 1, - release all BGP resources associated with this connection, - perform BGP peer oscillation (if configured), and - go to Idle

Y2 - is: The local system: 1) increments the ConnectRetryCnt (connect retry count) by 1, 2) releases all BGP resources associated with this connection, and 3) performs the BGP peer oscillation damping process

if the damping process allows for a new connection, the local system: - restarts the connect retry timer (with initial value, and - goes to Idle If the damping process does not allow for a new connection, the local system - set the flags to damp the creation of a new bgp connection until a manual start occurs, and - goes to Idle.

Tom agreed with the options, and stated that he preferred option 2. Sue is also happy with option 2, if no one else chimes in.

After the issues list came out Tom responded to this issue, saying:

.in 4 I think this issue SHOULD be administratively terminated.

It relates to Connect state Event 17 (TCP connection fails) and I am credited with raising it; in fact, the issue I raised was missing next state for Active state event 17 and this has now been subsumed into 13.4 (but note that 13.4 does not explicitly say Active state - I know it should because I raised that issue too). I will ensure it does not get lost from any resolution of 13.4.

And Connect state event 17 does appear as part of issue 45 which Siva raised so I think that either way, 13.5 can go.

This conversation was started in the "bgp18 WG Last Call fsm missing next state" thread.

3.13.6. FSM Missing Next States - Event 18 (Open Confirm)

Status: Consensus
Change: Yes
Summary: This is the text:

.in 4 In the Open Confirm state, a valid Open message [Event 22] is received. The BGP Peer connection is check to see if there is a collision per section 6.8. If this connection is to be dropped due to the call collision, the local system will drop the call by:

- sending a NOTIFICATION with a CEASE, - resets the Connect timer (to zero), - releases all BGP resources (this includes stopping the Open Delay Timer and reseting it to zero), - increments the

ConnectRetryCnt by 1 (connect retry +count), and - optionally performs a BGP peer oscillation damping processing, and - enters the Idle State.

Discussion: Tom opened this with:

Open Confirm State:

If the Open messages is valid [Event 18], the collision detect function is processed per section 6.8. If this connection is to be dropped due to call collision, the local system: ** enters what state

Sue replied with:

Here's my proposed text. Please let me know what you think. I think this is an editorial change. Old text: If the open message is valid, the collision detect function is processed per section 6.8. If this connection is to be dropped due to call collision, the local system: - sends a Notification with a Cease - resets the Connect timer (to zero), - releases all BGP resources, - Drop the TCP connection (sends a TCP FIN), - increments the ConnectRetryCnt by 1 (connect retry count), and - performs an BGP peer oscillation damping process.

New text: If the open message is valid, the BGP peer connection is check to detect a collision per section 6.8. If this connection is to be dropped due to call collision, the local system: - sends a Notification with a Cease - resets the Connect timer (to zero), - releases all BGP resources, - Drop the TCP connection (sends a TCP FIN), - increments the ConnectRetryCnt by 1 (connect retry count), and - performs an BGP peer oscillation damping processing, and - enters the Idle State. notes: Collision detect impacts Open Sent, Open Confirm, and Established states.

Tom replied:

.in 4 I am still struggling with; we are in OpenConfirm so we already have received an Open from the remote peer and Event 18 is a second Open from the same peer. Perhaps my struggle is that I think in terms of two (or more) FSM for a given IP address pair so the Open Collision detection will occur when the/an- other FSM receives a valid Open in states Active/Connect/Open Sent and will generate Event 22 into this FSM so Event 18 cannot occur. But yes, if Event 18 can occur in this FSM and this connection is to be dumped, then Idle state it should be as you suggest. I have slotted in [optionally] in front of the peer oscillation damping in your text because I think it should be optional:-)

Sue replied:

this mechanism allows a single fsm to handle both. 2 fsm and 1 fsm BGP FSM seem to exist. (I queried implementors a few times on this one. So, I just put in this change to provide the flexibility.

Collision detect tends to give scrambled brains for most people.. As Dennis Ferguson said 2 years ago, that's the hardest part of the FSM.

Sue then stated that she would query implementors to see what is being done.

Sue prodded the list with:

.in 4 In the Open Confirm state, a valid Open message [Event 22] is received. The BGP Peer connection is check to see if there is a collision per section 6.8. If this connection is to be dropped due to the call collision, the local system will drop the call by:

- sending a NOTIFICATION with a CEASE, - resets the Connect timer (to zero), - releases all BGP resources (this includes stopping the Open Delay Timer and reseting it to zero), - increments the ConnectRetryCnt by 1 (connect retry +count), and - optionally performs a BGP peer oscillation damping processing, and - enters the Idle State.

Implementors need to verify if this text and the text for Event 22 allows all implementors to perform the necessary Call Collision actions. If no objects, we will use this text.

Curtis said he had no problem with this.

There has been no disagreement, we are at consensus with this.

This conversation was started in the "bgp18 WG Last Call fsm missing next state" thread. It was also discussed in the "BGP draft-19 - FSM input needed from developers" thread.

3.14. FSM - Peer Oscillation Damping

Status: Consensus
Change: Yes
Summary: Change references to peer oscillation damping to consistent phrase:
"[optionally] performs peer oscillation damping". Also remove old reference to "BGP Peer Restart Backoff Mechanisms".

Discussion:

Tom suggested we use consistent terminology to refer to peer

oscillation damping. He also pointed out a stale reference.

Yakov agreed to fix both of these.

3.15. FSM - Consistent FSM Event Names

Status: Consensus
Change: Yes
Summary: Make FSM names consistent. Specifics are in the discussion section.

Discussion:

Tom proposed that:

.in 4 The event name used in the FSM show much variation to the point sometimes where I am not clear that it is always the same event (eg where the event name is qualified by a subset of the possible causes). Assuming that it is, I propose the following changes to make the wording consistent, clear and concise for event names.

** denotes changed text using the convention /'old text'/'new text'/

8. BGP Finite State machine

Event1: Manual start Event2: Manual stop Event3: Automatic start **Event4: Manual start with passive TCP /establishment/flag/ **Event5: Automatic start with passive TCP /establishment/flag/ Event6: Automatic start with bgp_stop_flap option set **Event7: Auto//matic/ stop Event8: Idle hold timer expires Event9: Connect retry timer expires **Event10: Hold time//r/ expires Event11: Keepalive timer expires Event12: Open Delay timer expires **Event13: TCP connection valid indication **Event14: TCP connection invalid indication **Event15: TCP connection request /sent received an ACK/ acknowledged/ Event16: TCP connection confirmed Event17: TCP connection fails Event18: BGPOpen Event19: BGPOpen with *Open Delay timer running Event20: BGPHeaderErr Event21: BGPOpenMsgErr Event22: Open collision dump Event23: NotifMsgVerErr Event24: NotifMsg Event25: KeepAliveMsg Event26: UpdateMsg Event27: UpdateMsgErr

8.2.2 Finite State Machine

Connect State:

If the BGP port receives a ** valid TCP connection indication [Event 13],

If the TCP connection receives **an invalid indication [Event 14]:

If the TCP connection fails **/(timeout or disconnect)// [Event17]

Active State:

If the local system receives a **valid TCP //indication/ [Event 13], If the local system receives a TCP connection failed [Event 17] **/(timeout or receives connection disconnect)//,

Open Sent: If a connection in Open Sent is determined to be the connection that must be closed, an **/administrative collision detect/Open collision dump/ [Event 22] is signaled to the state machine. If such an **/administrative collision detect dump [Event 22]/event/ is If a TCP **//connection valid/ indication [Event 13] or TCP **//connection/ request **//acknowledged/ [Event 15] Open Confirm State: ...or receives a TCP **/Disconnect// connection fails/ [Event 17] from the

In the event of **/TCP establishment//TCP connection valid indication /[Event 13]

...the local system will **/issue a call/generate an Open/ collision dump [Event 22]. When the local system receives a **/call/open/ collision dump event [Event 22]/such an event/, the

Established State: **/disconnect from the underlying TCP/TCP connection fails/ [Event17], it:

... it will process **/a Call/an Open/ Collision dump event[Event 22].

Notes: Event 4 title brought in line with text Event 5 title brought in line with text Event 7 title brought in line with text Event 13 title shortened to be closer to text, text brought in line Event 14 title shortened to be closer to text, text brought in line Event 15 title brought in line with text Event 17 text brought in line with title (text often introduces qualifying conditions that are too restrictive) Event 22 text brought in line with title

Sue replied:

I will accept the text you proposed for the Event names. I will update the FSM text to include your changes. We'll consider issue 15 in consensus. I've fixed the text.

So we are at consensus here.

This is discussed in the thread: "bgp18 WG Last Call fsm event names." It was also discussed in the "Issue 15 - Consistent FSM

Event Names" thread.

3.16. Many Editorial Comments

Status: Consensus
Change: Yes
Summary: Many editorial suggestions, and what we are doing with them are listed below. Some issues have been broken out separately where there is a longer discussion on them.

Discussion:

Alex began this by presenting comments from an anonymous reviewer, unless otherwise noted, responses are from Yakov:

> Almost all of these are simple clarifications. > > Section 1, page 5: IGP definition - it's not clear from this > definition whether IBGP would be considered an IGP? any suggestion on how to improve the definition to clarify this issue would be appreciated.

There was some further discussion on this and it was decided that people reading this document ought to know what an IGP is.

> Section 3, page 7, para 4: Does RFC 1772 still represent the > *planned* use of BGP? Or the actual use? Or something > different from actual use?

Perhaps we should just take out references to 1772.

Further discussion seemed to indicate that this reference should stay.

> Section 3, page 8, para 3 - "The hosts executing..." This > paragraph seems obsolete.

I'll take it out.

With regard to this, Siva asked if some route optimization vendors rely on this. Since this wasn't resolved, it is discussed further in issue 17.

> Section 4.1, page 11 - Length is in network byte order.

all the encodings are in network byte order. This applies not just to the BGP spec, but to other protocols as well.

This comment was made about a number of fields. It was later agreed that a reference would be made to this at the beginning of the

document.

> Section 4.2, page 12 - Hold Time - what does a value of zero > indicate?

if you read section 4.4 then you'll find that: If the negotiated Hold Time interval is zero, then periodic KEEPALIVE messages MUST NOT be sent.

> Section 4.2, page 13 - BGP Identifier - network byte order? > "IP address" -> "IPv4 address"

I'll put at the beginning a sentence saying that in the context of this document the term "IP address" means an IP Version 4 address.

> Section 4.3, Page 14, para1, sentence 2 - "path attribute" -> > "path attributes"

fixed.

> Section 4.3,Page 17, NEXT_HOP: "IP address" -> "IPv4 address" > Specify that this is 4 octets. > Reference here to multi-protocol extensions for IPv6 > nexthop?

no.

> RFC 2283 is unclear whether NEXT_HOP should always be > included when using multiprotocol extensions. Clarify > this here?

It is already clarified in 2283bis.

> Section 4.3, Page 17/18 - MED and LocalPref: > "non-negative" -> "unsigned" for consistency with > elsewhere. (non-negative might imply values > 2^31 > cannot be used). fixed. > Section 4.3, Page 19 - Prefix: "IP address" -> "IPv4 address" > Prefix: "enough trailing bits to" -> "the minimum number > of trailing bits needed to"

fixed.

> Section 4.4, Page 20: - "BGP does not use any TCP-based keep-alive > mechanism to determine if peers are reachable". Is it worth noting > that TCP may still timeout the connection even if TCP keepalives are > turned off?

the text is fine as it is.

> Section 4.4, Page 20: > KEEPALIVE message consists" -> "A KEEPALIVE message consists" fixed.

> Section 5, Page 23: "The same attribute can not appear more than > once with the Path Attributes field...". Does this mean the same > attribute type, or the same attribute type and value?

the former (the same attribute type).

> Section 5.1 "The usage of each BGP path attributes .." -> > attribute

fixed.

> Section 5.1.3 "IP address" -> "IPv4 address" > > "A BGP speaker must never advertise an address of a peer to that > peer as a NEXT_HOP, for a route that the speaker is originating." > suggest replace this text with: > "A route originated by a BGP speaker must never be advertised to a > peer using an address of that peer as NEXT_HOP"

fixed.

> Section 5.1.4: "A BGP speaker MUST IMPLEMENT a mechanism ... which > allows the MULTI_EXIT_DISC to be removed from a route." Might > want to say that this is dangerous unless you received the route > from an EBGP peer?

think we should keep the text as is.

> Section 5.1.5: "If it [LOCAL_PREF] is contained in an UPDATE > message that is received from an external peer, then this > attribute MUST be ignored by the receiving speaker, except for the > case of BGP Confederations [RFC3065]." > - "ignored" might be taken to mean that you don't process it for > decision, but that you propagate it to internal peers. I might > write "silently removed" or something similar.

I think the text is ok as is.

> Section 5.1.5, para 2. "set of AS" -> "set of ASs"

fixed.

> Section 6.3: wrt NEXT_HOP semantic correctness: should we check > that a NEXT_HOP is not a multicast or broadcast address?

I'll add to the definition of NEXT_HOP that it is a unicast address.

> Section 6.3, page 32, para 7: "peer than sent" -> "peer that > sent"

fixed.

> Section 6.3: "if any attribute appears more than once" - does this > mean the same attribute type, or the same attribute type and > value?

the former.

> Section 6.8 "Comparing BGP identifiers is done by treating them as > (4-octet-long) unsigned integers". Need to convert to host byte > order before comparing.

fixed.

> Section 6.8, item 2: "closes BGP connection" -> "closes the BGP > connection"; "accepts BGP connection" -> "accepts the BGP connection".

fixed.

> Section 9.1.2.2: item (c): in the explanation of neighborAS(n), it > is unclear for IBGP connections how to determine "the neighbor AS > from which the other IBGP speaker learned the route". If this is > really the leftmost entry in the AS path (or the local AS if the > path is empty), the spec should explicitly say so.

fixed.

> Section 9.1.2.2, page 63, paragraph starting "If a MULTI_EXIT_DISC > attribute is removed before..." The first sentence is pretty > nearly incomprehensible.

This topic has some more discussion surrounding what text we should use to clarify this issue. This is followed up in issue 18.

> Section 9.1.2.2 (d) > "d) If at least one of the candidate routes was received from > an external peer in a neighboring autonomous system, remove > from consideration all routes which were received from > internal peers." > For consistency with (c) and clarity, this might be reworded: > "d) If any of the candidate routes was learned via EBGP, > remove from consideration all routes which were learned by > IBGP."

fixed.

> Section 9.1.2.2 (e) > "cost (n) is better than cost (m)" > Given the definition of cost, it might be clearer to say > "cost (n) is lower than cost (m)"

fixed.

> Section 9.1.2.2 (g) > "neighbor address" has not been defined.

I'll replace "neighbor address" with "peer address".

> Section 9.2.2.2, Page 70 (AGGREGATOR) - "All AGGREGATOR attributes > of all routes to be aggregated should be ignored." > > Perhaps "ignored" is ambiguous here, and it's not clear > whether should is a SHOULD. Suggest: > > "Any AGGREGATOR attributes from the routes to be aggregated > MUST NOT be included in the aggregated route." fixed.

> Section 9.3 - shouldn't this subsection be moved to the discussion
> of Phase 1 or Phase 2 of the decision process? Or at least move it
> before Section 9.2.

I think it is fine where it is now.

> Appendix E, para 2: IP precedence has been deprecated. Delete > this paragraph, or replace with appropriate diffserv codepoint.

deleted.

> Security Considerations: > "BGP supports the ability to authenticate BGP messages by using > BGP authentication." > This sentence should be removed, and the Authentication > Information parameter has been deprecated.

Please see the recent e-mail exchange on the Security Considerations

See issue 19 for more on the Security Considerations section of the draft.

These topics were discussed in the "proxy: more comments on the draft -18" thread.

3.17. Section 3, Page 8, Paragraph 3 - Obsolete?

Status: Consensus
Change: Yes
Summary: Leave the current definition of BGP Speaker, and normalize the text to use "BGP Speaker" instead of router.

Discussion:

This issue was spawned from the discussions in issue 16, specifically:

Anonymous reviewer:

> Section 3, page 8, para 3 - "The hosts executing..." This > paragraph seems obsolete.

Yakov:

I'll take it out.

With regard to this, Siva asked if some route optimization vendors rely on this.

Jeff replied:

To provide context, this paragraph currently reads:

: The hosts executing BGP need not be routers. A non-routing host : could exchange routing information with routers via EGP [RFC904] : or even an interior routing protocol. That non-routing host could : then use BGP to exchange routing information with a border router : in another Autonomous System. The implications and applications of : this architecture are for further study. .in 4 There are several deployed entities that could be considered to "exploit" this paragraph. Route collectors, route servers, bandwidth shapers and other optimizers. However, the original text may be showing its age a little bit.

Perhaps the following might be a bit more appropriate:

"The hosts executing BGP need not be routers. A non-routing host may exchange routing information with a BGP speaker for reasons that are outside the scope of this document."

I would also propose adding to the same paragraph (but could be persuaded to drop it since it is *logically* redundant): "These non-routing hosts should exercise great care not to insert themselves into the forwarding path if they re-announce BGP routes."

Yakov replied:

Since operations of non-routing host are outside the scope of the document, and since the document doesn't preclude non-routing hosts to run BGP, I would prefer just to take the following paragraph out, and not to add any new text.

The hosts executing BGP need not be routers. A non-routing host could exchange routing information with routers via EGP [RFC904] or even an interior routing protocol. That non-routing host could then

use BGP to exchange routing information with a border router in another Autonomous System. The implications and applications of this architecture are for further study.

Jeff replied that this was ok, and instead suggested:

At the beginning of the document, we define: BGP speaker A router that implements BGP.

This (potentially) restricts a speaker to being a router. Additionally, several spots in the text where we probably should say "BGP speaker", we use router.

Yakov agreed to add this definition.

Jeff replied that there still was a problem with this definition being too limiting. The discussion meandered off list for a couple of exchanges and these additional definitions were proposed:

First Jeff proposed this:

"A router that implements the BGP protocol. Non-routing hosts that also implement BGP are out of scope of this document."

Then Andrew replied, that we should make sure the definition does not opt out entirely from making sure that non-routing hosts are interoperable:

BGP Speaker .in 7 A router that implements the BGP protocol. The internal behavior of non-routing hosts that also implement BGP are out of scope of this document. However, in their interactions with routers, non-routing hosts must behave as if they were routers.

And Jeff replied:

BGP Speaker .in 7 A router that implements the BGP protocol. The internal behavior of non-routing hosts that also implement BGP are out of scope of this document. However, in their interactions with BGP speaking routers, non-routing hosts that implement BGP should be indistinguishable from a router on the wire. .in 4

(or something like that - s/on the wire/ with whatever sounds best.)

IOW, look like bgp on the wire - what you do internally is out of scope.

Yakov replied, that we should keep the current definition, since it is clear that non-routing hosts are outside of the scope. Jeff

responded that he is ok with that if we normalize the use of "BGP Speaker" instead of "BGP router" in the document. Yakov agreed to this, we are at consensus on this.

This was discussed in the "proxy: more comments on draft -18" thread. And in the "Issues list, #17: Section 3, Page 8, Paragraph 3 - Obsolete?" thread. And also, the "issue 17 - final resolution" thread.

3.18. MED Removal Text

Status: Consensus
Change: Yes
Summary: Use text at the end of the discussion.

Discussion:

This issue is spawned from issue 16.

An anonymous reviewer pointed out:

> Section 9.1.2.2, page 63, paragraph starting "If a MULTI_EXIT_DISC
> attribute is removed before..." The first sentence is pretty
nearly > incomprehensible.

Yakov replied:

here is my attempt to clarify this:

If a MULTI_EXIT_DISC attribute is removed before re-advertising a route into IBGP, then (prior to the removal) the MULTI_EXIT_DISC attribute may only be considered in the comparison of EBGP learned routes; the attribute is then removed, and then the remaining EBGP learned routes may be compared to the remaining IBGP learned routes, without considering the MULTI_EXIT_DISC attribute for those EBGP learned routes whose MULTI_EXIT_DISC attribute will be removed before advertising these routes to IBGP.

Any further suggestions on how to improve this would be appreciated.

Siva replied:

How about this:

If a MULTI_EXIT_DISC attribute is removed before re-advertising a route into IBGP, then comparison based on the MULT_EXIT_DISC attribute may (MUST?) be performed only among the EBGP learned routes. This comparison MUST be performed before the removal of the

MULTI_EXIT_DISC attribute. The MULT_EXIT_DISC attribute must then be removed from those EBGP routes where such removal is required and which are still eligible. This is followed by comparison with IBGP learned routes.

I think this reflects our objectives, which is:

a) If MED is to be removed, compare EBGP routes based on the MED

b) Then remove the MED

c) Then do comparison with IBGP routes

Andrew suggested:

If a router is configured to remove a MULTI_EXIT_DISC attribute from a route learned from EBGP, before re-advertising it into IBGP the router MUST compare the route with other EBGP-learned routes before removing the MULTI_EXIT_DISC. Once this comparison is complete, the MED may be removed, and any remaining routes can be compared with IBGP routes to determine the best route.

Yakov replied:

Here is the text that will go in the next version of the draft:

If a MULTI_EXIT_DISC attribute is removed before re-advertising a route into IBGP, then comparison based on the MULT_EXIT_DISC attribute MAY be performed only among the EBGP learned routes. This comparison MUST be performed before the removal of the MULTI_EXIT_DISC attribute. The MULT_EXIT_DISC attribute is then removed from those EBGP routes where such removal is required and which are still eligible. This is followed by comparison with IBGP learned routes.

Matthew responded to this with:

I think this new text is ambiguous.

>Here is the text that will go in the next version of the draft: > > If a MULTI_EXIT_DISC attribute is removed before re-advertising a > route into IBGP, then comparison based on the MULT_EXIT_DISC > attribute MAY be performed only among the EBGP learned routes.

.in 4 This could be taken to mean either that the comparison may be performed, and if it's performed it must be performed only between EBGP learned routes, or that the comparison must be performed, but it may be performed only between EBGP learned routes.

> This comparison MUST be performed before the removal of the > MULTI_EXIT_DISC attribute.

.in 4 If doing the comparison is optional, then I think that this sentence should read "If the comparison is performed, then it MUST be perfo..."

> The MULT_EXIT_DISC attribute is then > removed from those EBGP routes where such removal is required and > which are still eligible. This is followed by comparison with > IBGP learned routes.

<snip>

I think that it is desirable for an operator to be able to turn off MED processing entirely (including turning off all MED based comparisons), so I would suggest the following text:

.in 5 If a MULTI_EXIT_DISC attribute is removed before re-advertising a route into IBGP, comparison based on the received MULTI_EXIT_DISC attribute MAY be performed. If an implementation chooses to perform this comparison, then the comparison MUST be performed only among EBGP learned routes, and it MUST be performed before the removal of the MULTI_EXIT_DISC attribute.

Curtis replied to Yakov's message:

.in 4 Looks good to me.

I see no need to change "This comparison MUST be performed before the removal of the MULTI_EXIT_DISC attribute". There is no implication that MULTI_EXIT_DISC must be removed and the first sentence clearly indicates that doing so is not required therefore no ambiguity. Adding a "If a MULTI_EXIT_DISC attribute is removed" to the second sentence would be redundant.

After some further discussion we have reached full consensus with:

.in 4 If a MULTI_EXIT_DISC attribute is removed before re-advertising a route into IBGP, then comparison based on the received EBGP MULTI_EXIT_DISC attribute MAY still be performed. If an implementation chooses to remove MULTI_EXIT_DISC, then the optional comparison on MULTI_EXIT_DISC if performed at all MUST be performed only among EBGP learned routes. The best EBGP learned route may then be compared with IBGP learned routes after the removal of the MULTI_EXIT_DISC attribute. If MULTI_EXIT_DISC is removed from a subset of EBGP learned routes and the selected "best" EBGP learned route will not have MULTI_EXIT_DISC removed, then the MULTI_EXIT_DISC must be used in the comparison with IBGP learned routes. For IBGP

learned routes the MULTI_EXIT_DISC MUST be used in route comparisons which reach this step in the decision process.

This is discussed in the "proxy: more comments on draft 18" thread. And in the "issue 18" thread.

3.19. Security Considerations

Status: Consensus
Change: Yes
Summary: Fix Security Considerations section to include mandatory MD5 auth and advance security considerations draft along with the base draft.

Discussion:

Yakov started this discussion by proposing text which would require TCP MD5 authentication for BGP implementations. This is to bring the spec in line with an IETF requirement that authentication be available.

After some discussion the plan is to advance draft-ietf-idr-bgp-vuln-00.txt as Informational along with the base BGP specification. This draft will serve as the security analysis section of the base spec.

This is discussed in the "revised Security Considerations section" thread.

3.20. Peer Oscillation Damping

Status: Consensus
Change: No
Summary: Keep the Peer Oscillation Damping reference in the specification.

Discussion:

This began when Siva proposed:

.in 4 Since this feature is going to be added in a new draft, and its addition will change the operation of the state machine, can we remove all mention of it in the state machine ? As part of this removal, can we also remove the IdleHold Timer from the FSM since it is not useful in the absence of peer oscillation damping ?

The draft that describes this procedure can then describe the change in the state machine required to do this.

Sue replied that:

The reason we should not remove the peer oscillation damping from the state machine:

1) Deployed implementations support peer oscillation damping 2) Hooks for the additions in the FSM cannot be added later.

These hooks are optional and do not need to be implemented.

Siva replied:

I understand. I am not trying to object to peer oscillation damping, I think it is a good idea and we have included it in our implementation as well. I was suggesting that instead of a partial description in this draft, it be completely described in the draft on peer oscillation damping.

However, I do see your point, and unless there are any objections from others, I think we have consensus on this issue.

This was discussed in the "Response to FSM input - Comments 1-10" thread: Comment #1.

3.21. Session Attributes - IdleHold Timer

Status: Consensus
Change: Yes
Summary: Add the text in the discussion section.

Discussion:

This discussion began with Siva asking:

.in 4 Why have a Hold Timer and a Hold Time ? Can we replace this with just Hold timer ?

Can we also add the following session attributes:

a) DelayBgpOpenTimer b) IdleHold Timer (in case we choose not to remove this from the base FSM)

Can we also add the following flag to the session attributes: a) DelayOpen Flag

After some discussion we have this text on the table:

Event8: Idle hold timer expires

Definition: An Event generated when the Idle Hold Timer expires. The Idle Hold Timer is only used when the persistent peer oscillation damping function is enabled.

% Implementations not implemented persistent % peer oscillations damping functions may not % have the Idle Hold Timer. Sue replied:

I will accept the new text for the following total text: Event8: Idle hold timer expires

Definition: An event generated when the Idle Hold Timer .in 24 expires indicating that the session has completed a back-off period to prevent bgp peer oscillation.

The Idle Hold Timer is only used when the persistent peer oscillation damping function is enabled.

Implementations not implementing the persistent peer oscillation damping functions may not have the Idle Hold Timer.

Status: Optional

We are at consensus with this.

Tom added a couple of minor edits, correcting the spelling of "persistent" in the third paragraph, and pointing out that:

.in 4 oscillation damping functions may not have the Idle Hold ** function ** (because we only have function not functions in the previous sentence) Timer.

Sue added the edits.

Siva also liked the way this issue has turned out.

This was discussed in the "Response to FSM input - Comments 1-10" thread: Comment #2. And in the "Draft 19 - issue #21" thread, alternately the "Draft 19 - Issue 21" thread.

3.22. Specify New Attributes (Accept Connections/Peer Oscillation Damping)

Status: Consensus
Change: Yes
Summary: Add the text in the discussion section to section 8.0.

Discussion:

This began with Siva proposing:

Can we call these out as well:

* Accept Connections from unconfigured peers (Enabled/Disabled) * Peer Oscillation Dampening (Enabled/Disabled) (In case we choose not to remove it from base spec)

After some discussion we have this text on the table:

The following will be added to 8.0 Optional parameters that may be supported either per connection or per implementation:

1) Delay Open flag 2) Delay Open Timer 3) Perform automatic start flag 4) Passive TCP establishment flag 5) BGP stop_peer_flag flag 6) Idle Hold timer 7) Perform automatic stop flag 8) Perform Collision detect in Establish mode flag

Sue accepted these changes.

Tom added this correction for item 2 in Sue's text:

2) Delay Open Timer

** Open Delay timer ** (for which we have consensus in Issue list v2 item 7)

Siva asked, and Sue accepted these additional changes:

9) accept connections from un-configured peers 5) BGP stop_peer_flap flag

We are at consensus on this.

This was discussed in the "Response to FSM input - Comments 1-10" thread: Comment #3. This was also discussed in the "BGP Draft 19 - Close open items 22" thread.

3.23. Event1/Event2 Clean Up

Status: Consensus
Change: Yes
Summary: Use "Local system administrator" in both sections.

Discussion:

Siva proposed that we clean up the text for these Events by selecting either "Administrator" or "Local system" but not both.

Sue proposed text using "Local system administrator" that was agreed on.

This was discussed in the "Response to FSM input - Comments 1-10" thread: Comment #4.

3.24. Events 3, 5, 6 & 7 Give Examples

Status: Consensus
Change: No
Summary: Leave the examples out.

Discussion:

This began with Siva proposing we add examples for these event states. Sue believes this is largely out-of-scope, but did agree to move the example of "automatic stop" to the event description section. She asked for proposed text for additional examples.

Sue replied that she has made the following changes, and asked if these worked for Siva.

New text: Event7: Automatic stop

Definition: Local system automatically stops the BGP connection.

An example of an automatic stop event is exceeding the number of prefixes for a given peer and the local system automatically disconnecting the peer.

Status: Optional depending on local system

Siva thought this for Event 7 was fine.

Sue replied to the list, saying that, previously examples had caused dissension, and asked if there was a strong feeling either way.

Siva proposed this text for Events 3, 5 & 6:

Event 3: Examples of this event are: When a connection is terminated during exchange of Open messages due to version failure

Event 5: Examples of this event are: Similar to Event 3

Event 6: Examples of this event are: Similar to Event 3 and b) When a Idle Hold timer expires (within local limit)

Sue replied to this:

I'm going to leave the examples out of events 3, 4, 6 since I've not heard any strong input on the mail list **and** I had strong comments on prior versions of the draft. I'd like to declare that issue 24 has consensus.

Siva agreed, we are at consensus on this issue.

This was discussed in the "Response to FSM input - Comments 1-10" thread: Comment #5. This was also in the "Issue 25" thread, and the "Issue 25 - this is really issue 24" threads. This is also in the "Draft 19 - Issue 24" thread.

3.25. Event 4 & 5 Session Initiation Text

Status: Consensus
Change: No
Summary: Leave the text as is.

Discussion:

This began with Siva wanting to change:

Definition: Local system automatically starts the BGP session with the passive flag enabled. The passive flag indicates that the peer will listen prior to establishing a connection.

to:

The passive flag indicates that the state machine will wait for specified peer to initiate a connection with the local system. If this does not happen within a specific time (hold time), the local system will then also attempt to initiate connection with the specified peer.

Sue replied:

The text in 8.2.1.1 indicates the definition of the passive flag. 6a) ========== My understanding of your text is that you want to replace in both sets of text:

"The passive flag indicates the peer will listen prior to establishing a connection".

with:

"The passive flag indicates that the state machine will wait for the specified peer to initiate a connection with a local system.

The problem with this sentence is that in the "unconfigured" case the phrase "specified" peer is confusing. I think the original text is clearer.

6b) ========== If this does not happen within a specific time (hold time), the local system will then also attempt to initiate (a) connection with the specified peer. My comments: Again, the "specified peer" term is confusing. Also, the 2nd half of the statement mixes the actions of the state machine with the events. I believe this muddies the text instead of clarifying it.

Siva and Sue later agreed to leave the text the same because of the Unconfigured + passive TCP connection + Delay Open situation.

This was discussed in the "Response to FSM input - Comments 1-10" thread: Comment #6.

3.26. Event 4 & 5 - bgp_stop_flap option

Status: Consensus
Change: Yes
Summary: Add new event below.

Discussion:

This began with Siva asking:

Won't a variant of this with bgp_stop_flap option set be required ? We can also achieve the same by using the bgp_stop-Flap option as a flag that is provided as an input to the state machine.

Siva later clarified this to include:

We already have Event 3 - Automatic Start Event 5 - Automatic start with bgp_stop_flap option set To make things consistent, shouldn't we either a) Add 3 new events : .in 24 1) Manual start with bgp_stop flap option set 2) Manual start with passive TCP establishment and bgp_stop_flap option set 3) Automatic start with passive TCP establishment and bgp_stop_flap option set

or b) Remove Event 6, and rely on a flag to tell us wither peer flap damping is to be performed for the session or not.

Sue said she preferred option A. And stated that #1 & #2 are infeasible, but that we need to add #3.

Tom replied:

.in 4 But if we add an event, then we must add and agree on actions for all six existing states so I think to say that adding a new event settle things might be naive.

If we do add 3) Automatic start with passive TCP establishment and bgp_stop_flap option set

which I understand is Sue's resolution, then for Idle state the actions are straightforward but for the other five, is the event completely ignored? If so, does it mean that the passive flag and the bgp_stop_flap option are ignored and we carry on as if we were when we were started which may have been without them. Or is the fact of starting ignored but the flags remain set and so color the effect of other events? Needs defining.

Jeff replied to this, quoting the existing draft:

The start events [Event 1, 3-6] are ignored in connect state.

The start events [Event1, 3-6] are ignored in the Active state.

The Start events [Event1, 3-6] are ignored in the OpenSent state.

Any start event [Event1, 3-6] is ignored in the OpenConfirm state.

Any start event (Event 1, 3-6) is ignored in the Established state.

And elaborated, saying that:

.in 4 "ignore" means do nothing. This means don't twiddle with the flags. :-)

The text that was finally agreed on is:

Event 7: Automatic start with bgp_stop flap option set and passive TCP establishment option set Definition: Local system automatically starts the .in 24 BGP peer connection with peer oscillation damping enabled and passive TCP establishment enabled. The exact method of damping persistent peer oscillations is left up to the implementation, and is outside the scope of this document.

Status: Optional, used only if the bgp peer has .in 24 enabled bgp peer oscillation damping with following optional flags settings below.

Optional attributes: 1) Perform automatic start flag SHOULD be set 2) BGP stop_peer_flap flag SHOULD be set I've re-ordered the Timer events to keep the text changes down to a minimum.

action 9 - connect retry timer action 10 - Hold Timer expires action 11 - Keepalive timer expires action 13 - Open Delay timer expires action 14 - Idle Hold timer expires

All other events are incremented by 1

This was discussed in the "Response to FSM input - Comments 1-10" thread: Comment #7.

3.27. Event 5 Clarification

Status: Consensus
Change: No
Summary: Leave the text as is.

Discussion:

This began when Siva asked that in event 5:

.in 4 Is it correct that this event will occur only when we want to restart a connection (after it had been terminated due to some reason beside administrative action) that we had accepted from an unconfigured peer ?

Sue replied:

.in 4 The automatic start function is an implementation specific mechanism. This text does not seek to restrict it in any fashion.

Siva said that although he felt his original clarification would be more useful to new implementors he is ok with the text as is.

This was discussed in the "Response to FSM input - Comments 1-10" thread: Comment #8.

3.28. Timer Events Definition - Make Consistent

Status: Consensus
Change: Yes
Summary: Change text to use "generate" across the board.

Discussion:

Can we use similar language for Events 8-12 to make them consistent?

It was agreed that we will use "generate" i.e.:

Event 8: An event generated when the Idle Hold timer expires. Event

9: An event generated when the ConnectRetry timer expires. Event 10: An event generated when the Hold timer expires. Event 11: An event generated when the Keepalive timer expires Event 12: An event generated when the Delay BGP Open timer expires. This is at consensus.

This was discussed in the "Response to FSM input - Comments 1-10" thread Comment #9.

3.29. Event 8 - Clean Up

Status: Consensus
Change: Yes
Summary: Clean up first sentence. New text below.

Discussion:

Siva began this by asking if we could clean up the wording of Event 8.

After some discussion with Sue we are at this change for the first sentence:

An event triggered by the expiry of the Idle Hold timer, indicating that the session has completed waiting for a back-off period to prevent bgp peer oscillation.

This was discussed in the "Response to FSM input - Comments 1-10" thread: Comment #10.

3.30. Hold Timer - Split?

Status: Consensus
Change: No
Summary: Keep the hold timer text as is.

Discussion:

Siva proposed that since:

.in 4 We use the hold timer for two purposes

* Waiting for an open message (with a default value of 240 seconds) * Waiting for Keepalives (with a default value of 90 seconds)

Can we use two different timers (or at least call them two different timer events) ?

Sue replied that this is not how it is implemented currently. Siva replied that we have two conceptually different timers, but that it would certainly work to only have one, since only one needs to be running at any given time.

Tom agreed that we can keep things as is.

This was discussed in the "Comments 11-20" thread: Comment #11.

3.31. OpenDelay Timer Definition

Status: Consensus
Change: Yes - See issue 28
Summary: This is fixed by the fixing of issue 28.

Discussion:

This began with Siva's request that we add something to Event 12 to specify what to do when the timer expires. This seems to have been addressed in issue 28.

This was discussed in the "Comments 11-20" thread: Comment #12.

3.32. Definition of TCP Connection Accept (Event 13)

Status: Consensus
Change: Yes
Summary: Change "Definition" text as indicated below.

Discussion:

Siva proposed that we change text from referring to "TCP connection request" to "receiving a TCP connection". This led to this proposed text:

Definition: Event indicating the reception of a TCP connection request with a valid source IP address and TCP port, and valid destination IP address and TCP Port. The definition of invalid source address and port and invalid destination address is left to the implementation.

This met with agreement.

This thread also discussed the idea of filtering the incoming address/port. It was decided that this was implementation dependent.

This was discussed in the "Comments 11-20" thread: Comment #13.

3.33. Event 13 & 14 - Valid Addresses & Ports

Status: Consensus
Change: Yes
Summary: See text at the end of the discussion.

Discussion:

With regard to Event 13 & 14, Siva raised questions about: 1) What does it mean to validate a port, and 2) Should we state what we consider an invalid IP address to be?

Sue replied that this is local policy and is implementation dependent. Siva agreed regarding the source port & IP address, but disagreed about the destination port. He argued that we need to know the destination port for interoperability.

Sue asked Siva to provide some text.

After a long lull, Sue replied with:

I would like to keep the current text of "Should" in the following text "BGP's destination port SHOULD be port 179 as defined by IANA."

Should indicates that it normally should be 179. If an implementation allows for an alternative TCP port, it is still valid as the "MUST" is not indicated.

There have been no further comments on this, the chairs have decided to close it.

This was discussed in the "Comments 11-20" thread: Comment #14. This was also in the "BGP-19: Issue 33" thread.

3.34. Event 17 - TCP Connection Fails to TCP Connection Termination

Status: Consensus
Change: Yes
Summary: Change the text to "fails."

Discussion:

This began with Siva observing:

.in 4 This event can occur even when the transport connection is closed by the other end. Since this does not reflect a 'failure ', can we change the event name to

% Event17: TCP connection termination

Sue replied that:

Discussion: It both terminates from the remote site and can "timeout" - fail. Suggestions? I can use "disconnect", what do you think.

Siva replied that this was a minor issue, and on further reflection, either "fails" or "disconnect" would be acceptable.

Sue replied that she has accepted Siva's comments, and the text will be changed to "fails".

This was discussed in the "Comments 11-20" thread: Comment #15. This was also discussed in the "BGP-19: Issue 34-35, 40-48" thread.

3.35. Making Definition Style Consistent

Status: Consensus
Change: Yes
Summary: Adopt consistent style for the definition of events.

Discussion:

This started with Siva asking if we could make the definition style consistent across events. Sue replied to this with text for 13-17, Siva clarified that he was talking more about 18-21, and proposed text.

We are agreed on the text for 13-17:

Event13: TCP connection indication and valid remote peer

Definition: Event indicating the local system reception .in 24 of a TCP connection request with a valid source IP address and TCP port, and valid destination IP address and TCP Port. The definition of invalid source, and invalid destination IP address is left to the implementation.

BGP's destination port SHOULD be port 179 as defined by IANA.

TCP connection request is denoted by the local system receiving a TCP SYN.

Status: Mandatory (Optional)

Event14: RCV TCP connection indication with invalid source or destination Definition: Event indicating the local system reception

of a TCP connection request with either an invalid source address or port number or an invalid destination address or port number. BGP destination port number SHOULD be 179 as defined by IANA.

Again, a TCP connection request denoted by local system receiving a TCP SYN. Status: Mandatory (Optional) Event15: TCP connection request sent received an ACK.

Definition: Event indicating the Local system's request to establish a TCP connection to the remote peer. The local system's TCP session sent a TCP SYN, and received a TCP SYN, ACK pair of messages, and Sent a TCP ACK. Status: Mandatory Event16: TCP connection confirmed Definition: Event indicates that the local system receiving a confirmation that the TCP connection has been established by the remote site.

The remote peer's TCP engine sent a TCP SYN. The local peer's TCP engine sent a SYN, ACK pair, and now has received a final ACK. Status: Mandatory Event17: TCP connection fails Definition: Event indicates that the local system has received a TCP connection failure notice.

The remote BGP peer's TCP machine could have sent a FIN. The local peer would respond with a FIN-ACK. Another alternative is that the local peer indicated a timeout in the TCP session and downed the connection. Status: Mandatory

Siva proposed these changes for 18-21:

Event18: BGPOpen Definition: An event indicating that a valid Open message has been received.

with Event18: BGPOpen Definition: An event is generated when a valid Open message has been received.

Event19: BGPOpen with BGP Delay Open Timer running Definition: An event indicating that a valid Open message has been successful established for a peer that is currently delaying the sending of an BGP Open message.

with

Event19: BGPOpen with BGP Open Delay Timer running Definition: An event is generated when a valid Open message has been received for a peer that is currently delaying the sending of a BGP Open message.

Editorial Note: "Delay Open Timer" replaced with "Open Delay Timer" per issue 7. Event20: BGPHeaderErr Definition: BGP message header is

not valid.

with

Event20: BGPHeaderErr Definition: An event is generated when a received BGP message header is not valid.

Event21: BGPOpenMsgErr Definition: An BGP Open message has been received with errors.

with Event21: BGPOpenMsgErr Definition: An event is generated when BGP Open message with errors has been received.

Sue replied that she accepted Siva's comments, so we are at consensus here.

This was discussed in the "Comments 11-20" thread: Comment #16. This also came up in the "BGP-19: Issue 34-35, 40-48" thread.

3.36. Event 19 - Definition Cleanup

Status: Consensus
Change: Yes
Summary: Replace definition for Event 19 with the text in the discussion.

Discussion:

Siva proposed we replace:

.in 4 Definition: An event indicating that a valid Open Message has been successful established for a peer that is currently delaying the sending of an BGP Open message.

with:

.in 4 Definition: An event indicating that a valid OPEN Message has been received for a peer that has a successfully established transport connection and is currently delaying the sending of a BGP open message

in Event 19. Sue agreed to the changes.

This was discussed in the "Comments 11-20" thread: Comment #17.

3.37. Event 22 - Cleanup

Status: Consensus
Change: Yes
Summary: Replace Event 22 definition with the text from the discussion.

Discussion:

Siva began with observing:

Event22: Open collision discard Definition: An event generated administratively when a connection Collision has been detected while processing an incoming Open message. This connection has been

Isn't this event 'automatically' generated, since it is a system generated event ?

Sue replied that:

response: How this generated is implementation specific. The "administratively" is to cover policy.

Siva also proposed an editorial fix with:

Event 22 is an administrative could occur if FSM is implemented as two

The word event is missing. How about

Event 22 is an automatic event that could occur if FSM is implemented as two

Sue replied with this rewritten text:

Event22: Open collision dump Definition: An event generated administratively when a connection collision has been detected while processing an incoming OPEN message and this connection has been selected to disconnected. See Section 6.8 for more information on collision detection.

Event22 is an administrative based only implementation specific policy. This Event may occur if the FSM is implemented as two linked state machines.

Siva agreed with this new text.

This was discussed in the "Comments 11-20" thread: Comment #18.

3.38. FSM Description - ConnectRetry Count

Status: Consensus
Change: No
Summary: Leave the counter text alone, since it is used in peer oscillation and will be in the MIB.

Discussion:

Siva opened with this question:

The Connect Retry count is updated by the FSM but never used. In the absence of peer oscillation damping, will this be used to stop connection establishment attempts after a certain maximum number ?

<Sue> Yes, this is either implementation specific or is it based on the peer oscillation damping draft. </Sue>

Can we include the use of this counter in some place ?

<Sue> Connect retry counter 1) Will be utilized by the peer oscillation damping draft. 2) Will be included in bgp-4-mibv2-xx. I just check and I didn't find it.

Do you still want text in the main? </Sue>

To which Siva replied that he believes we can leave the main text alone.

This was discussed in the "Comments 11-20" thread: Comment #19.

3.39. Handling Event 7 (Auto Stop) to Idle State processing

Status: Consensus
Change: Yes
Summary: Fix the text as indicated in the discussion.

Discussion:

Siva began with:

.in 4 The handling of Event 7 is missing from the Idle State processing. Can we add this ? How about replacing

An manual stop event (Event2) is ignored in the Idle state.

with

Manual stop (Event 2) and Auto stop (Event 7) events are ignored in the Idle state

Sue replied that she would add the text.

This was discussed in the "Comments 11-20" thread: Comment #20.

3.40. Clearing the Connection Retry Timer

Status: Consensus
Change: No
Summary: Leave things alone, since it is better to be redundant than to let something slip through.

Discussion:

Siva opened with the observation:

.in 4 There are a few sections where the FSM draft states that the Connection Retry timer needs to be reset, whereas the connect retry timer had been cleared prior to entering that state. We can remove these instructions to clear the connect retry timer.

List of places where the connect retry timer need not be cleared

a) Handling of Event 19 in the Connect State b) Handling of Events 12 in the Active State c) All cases where it is referred to in the OpenSent, OpenConfirm and Established states

Sue replied:

Comment: 1) Does it hurt to have the connect retry timer cleared at these points, since it has already been cleared.

I felt it eased the implementations to allow the action routines to be shared across as many states as possible. You can see this a bit more actively.

Tom replied to this:

.in 4 I propose we leave it in and close this issue.

1) To take out an action as redundant you need to be supremely confident that it really cannot make a difference. I am not (supremely confident); rather, the more I look at the FSM, the more places I find where actions are missing, as I have posted to the list, from obscure yet possible sequences of events and timing. And there is an outstanding issue of mine which flagged seven places

where the next state was missing and so I think it impossible for any one to be confident that any particular action is redundant until that is cleared up and that is proving complex in some cases. So, play safe, keep them in.

2) The argument for removing them is that the number of possible distinct action lists is increased. True - it will mean that an implementor will have to code more code when first implementing BGP.

For me this is no contest; keeping it safe at the possible cost of redundancy outweighs the one-off cost of additional implementation.

So keep the actions in and close the issue.

Jeff replied that he agreed with Tom on this.

Siva concurred, that this approach was acceptable.

Unless someone objects, this issue is at consensus.

This was discussed in the "Comments 21-30" thread: Comment #21. This is also discussed in the "BGP-draft-19: Issue 40 Clear Connect retry timer" thread.

3.41. Handling of Event 14 in the Connect State

Status: Consensus
Change: Yes
Summary: Make event 14 optional.

Discussion:

Siva opened the discussion with:

> If the transport connection receives an indication > that is invalid or unconfigured. [Event 14]: > - the TCP connection is rejected.

I don't understand how we would get this event while in this state.

Sue replied:

See my earlier comments (1-10) on the connection state. It happens in implementations which track the TCP state more closely. I suggest that Event 14 become optional.

Sue also suggested we fold this into the discussion about events 13-17, which is tracked in issue 13.4.

Sue proposed:

My resolution: Let event 14 be optional. Not all BGP implementations support it.

And asked if this let us reach consensus on this issue.

Siva agreed with this, we are at consensus on this.

This was discussed in the "Comments 21-30" thread: Comment #22. This was also brought up in the "BGP-19: Issue 34-35, 40-48" thread.

3.42. Handling events 20, 21 in the Connect State and Active State

Status: Consensus
Change: Yes
Summary: Use the text Tom proposed in the discussion section.

Discussion:

Siva began this with:

We need to consider the case where we receive events 20 (message header error) and 21 (Open message error) when the delay timer is running.

Since the connection has been established at this point, we need to send a Notification message and then terminate the connection.

To which Sue replied:

Alternative comments:

1) We have not sent an Open statement. 2) Why do we have to send an Notification? I see no justification for it.

Suggestion: Do you have implementations that send notification? Do you know of others that don't.

Jeff saw this as indicative of an issue with section 4.2 the way it is currently written:

>From section 4.2 of -18: .in 4 4.2 OPEN Message Format

.in 7 After a TCP is established, the first message sent by each side is an OPEN message. If the OPEN message is acceptable, a KEEPALIVE message confirming the OPEN is sent back. Once the OPEN is confirmed, UPDATE, KEEPALIVE, and NOTIFICATION messages may be

exchanged.

This text implies that NOTIFICATIONs can only be sent once we have sent an open and then a keepalive, generally meaning we're in the Established state.

Anyone suggestions for modifying the wording?

Section 6.1 (Message header error) is one situation that implies that a NOTIFICATION can be sent without sending even an OPEN message. Note that since the base FSM implies that we send an OPEN message immediately when we have a completed transport connection, we SHOULD be in at least OpenSent. However, the DelayOpen timer means that we MAY send a NOTIFICATION when we are in the Connect state.

GateD, at least, will not send a NOTIFICATION without first sending an OPEN.

We need to pick one: You can send NOTIFICATIONS before OPEN or before OPEN if the OpenDelay timer is running. However, we MUST fix the text above.

Tom opined:

.in 4 A NOTIFICATION without a preceding OPEN is rather hard to interpret; it is the OPEN that gives the recipient what it needs to know about its potential peer (Version, AS number, ID, options etc) so it makes sense to send an OPEN even if it is followed by a NOTIFICATION to say goodbye :-(as opposed to a KEEPALIVE which says hello:-).

But as ever, what is implemented?

Yakov suggested these modifications to the text to resolve this:

.in 4 1. Delete the last sentence in the above paragraph

or

2. Delete "and NOTIFICATION" in the last sentence in the above paragraph

Jeff replied that he preferred the first option, and that the second could be interpreted as NOTIFICATIONs not being legal, when, in fact, they may.

So the text on the table to resolve this is:

4.2 OPEN Message Format

.in 7 After a TCP is established, the first message sent by each side is an OPEN message. If the OPEN message is acceptable, a KEEPALIVE message confirming the OPEN is sent back.

However, this does not entirely clear up the original point about the FSM. If we receive an error in Connect/Active, do we send a NOTIFY? Do we preface it with an OPEN, so that OPEN/NOTIFY are sent in immediate succession?

Sue replied:

.in 4 I suggest we don't send a "NOTIFICATION" when Event 20 or Event 21 is received in Connect or Active state.

Tom responded to this issue with:

.in 4 Issue 42 queries whether or not we can send a NOTIFICATION when we have not successfully exchanged OPENs. I propose we should, following the suggestions of Jeff and Yakov.

As Yakov suggested, this requires the removal of the second sentence, first paragraph, of 4.2 which implies a NOTIFICATION can only be sent after a successful exchange of OPENs. I think this fits best with the other references to the uses of NOTIFICATION in the draft.

In terms of the FSM, it means that in Connect and Active states, on receipt of events 20 or 21, we should send a NOTIFICATION so that the last section starting

In response to any other event.............

is replaced by (and noting we have agreed to drop references to MIB actions)

If the BGP message header checking or OPEN message checking detect an error (see Section 6.2) [Events 20 or 21], the local system: - sends a NOTIFICATION message with the appropriate error code, - resets the connect retry timer (sets to zero), - releases all BGP resources, - drops the TCP connection - increments the ConnectRetryCnt (connect retry count) by 1, - [optionally] performs peer oscillation damping - and goes to the Idle state. In response to any other event (Events 7-8, 10-11,18, 22-27), the local system: - resets the connect retry timer (sets to zero), - releases all BGP resources, - drops the TCP connection, - increments the ConnectRetryCnt (connect retry count) by one, - [optionally] performs peer oscillation damping, - and goes to the Idle state

.in 4 (Note that this text is not quite watertight. Suppose we are in Active state, having been started with CRT running, receive an SYN (event 13), send SYN-ACK and then get a malformed message (events 20/21). We have not yet received an ACK and so should not send anything over TCP; I would expect TCP to buffer this awaiting the ACK except we then take down the TCP connection - or try to; I don't know what happens next but regard it as sufficiently obscure not to be concerned).

(My other concern is greater; why do we now not send NOTIFICATIONs for other events; in Open Sent, Open Confirm or Established, we send one for the 'default event list' so what makes events 20 and 21 in Active and Connect so special? I can justify the absence of a NOTIFICATION for events 7, 8, 10, 11, 18, 22 since there is no evidence of a TCP connection to send it on; but events 23-27 in Active or Connect say we have received an erroneous message, the TCP connection is there so why not send a NOTIFICATION? Event7: Automatic stop Event8: Idle hold timer expires Event10: Hold timer expires Event11: Keepalive timer expires Event18: BGPOpen Event22: Open collision dump Event23: NotifMsgVerErr Event24: NotifMsg Event25: KeepAliveMsg Event26: UpdateMsg Event27: UpdateMsgErr

Sue accepted Tom's text, so barring any objections, we are at consensus on this.

This was discussed in the "Comments 21-30" thread: Comment #23. This was also brought up in the "BGP-19: Issue 34-35, 40-48" thread, and the "Draft bgp19 - issue #42 NOTIFICATION before OPEN" thread.

3.43. Handling the default events in the Connect state

Status: No Consensus
Change: Potentially
Summary: Add text at the end of the discussion.

Discussion:

Siva opened this with:

.in 4 The Open Delay timer [original: BGP Delay Open Timers) needs to be cleared if it is running.

How about adding this:

% - If the ConnectRetry Timer is running % - Clear the Connect Retry timer % - Otherwise % - Clear the Open Delay timer [original: BGP Delay Open Timer]

Sue replied that:

By the default you mean the text:

In response to any other events[Events 7-8, 10-11, 18, 20-27], the local system:

"resets" to me implies stops and clears. I think the text is clear than the text above. ------------ Is this the replacement text you imply above: - resets the connect retry timer (sets to zero), - clears the Open Delay timer [original: BGP Delay timer] (sets to zero), - increments the ConnectRetryCnt (connect retry count) by 1, - [optionally] performs bgp peer oscillation damping, and - goes to Idle text:

Editor's note: various incarnations of "Open Delay timer" have been replaced with "Open Delay timer". See issue 7.

Sue replied that she accepted Siva's changes with these editorial changes:

old text: - resets the connect retry timer (sets to zero) - clears the open delay timer

new text: - if the connect retry timer is running, clear the connect retry timer (set to zero). - if the open delay timer is running, clear the open delay timer (set to zero).

Since the substantive changes have been accepted, unless someone objects, this issue is at consensus.

This was discussed in the "Comments 21-30" thread: Comment #24. This was also brought up in the "BGP-19: Issue 34-35, 40-48"

3.44. Handling Event 23 in Connect and OpenSent

Status: Consensus
Change: Yes
Summary: Adopt text at the end of the discussion section.

Discussion:

This began with Siva saying:

.in 4 This is currently being handled in the default event processing section. However, we do not need to go through the peer oscillation damping process in this case. Can we change the wordings to reflect this, or move this out of peer oscillation damping processing ?

Sue replied:

1) There is no default event handling process in the text, you will need to specify the text.

2) The state table below (hares-statemt-03.txt) states shows the changes

```
-------------
Event 23
states:
current Idle Connect Active Open-Sent Open-Cnf Establish
-------------------------------------------------
next state Idle Idle Idle Idle Idle Idle
-------------------------------------------------
action V D D Y Y T =================================================
```

V - Indicate FSM errors and ignore. D - 1) resets the connect retry timer (sets to zero), 2) drops the TCP connection, 2) releases all BGP resources, 3) increments the ConnectRetryCnt (connect retry count) by 1, 4) [optionally] performs the bgp peer oscillation damping, and Goes to Idle state. Y 1) resets the connect retry timer (sets to zero), 2) Drops the TCP connection, 3) releases all BGP resources, 4) [optionally]

In an exchange between Siva and Sue, this came up:

Siva:

"Default event handling" was perhaps a poor choice of words.

What I meant is this

.in 4 Event 23 (Notify Message Version error) only indicates a version mismatch. By going through action sequence D, we will be performing peer oscillation damping. Should we perform damping, since this is not really a cause for persistent oscillation ?

Also, since we have a distinct event to indicate a version error event, can include text indicating that version negotiation processing should take place upon receipt of this event ?

Sue:

Yes, we can change the "D" in state machine to a "y".

The issue is what if Connect state occurs and there is not a TCP connection. Should an OPEN with wrong version be accepted? If the

Open Delay flag is off, the connection state should not be getting an Open. The "D" action below works for "open delay flag off".

The "y" action you suggest can occur if the open delay timer is on.

If this is the issue, please confirm.

We could say: if open delay flag is on -> y action if open delay flag is off -> D action

Please let me know if this is the concern, and suggest text.

Prior to this exchange, this issue was at consensus. The only thing that is firm in this exchange is changing "D" to "y". There seems to be some open discussion still, so we'll reopen it.

After some discussion, this is the text we have settled on:

If a NOTIFICATION message is received with a version error[Event24], the local system checks the Open Delay timer. If the Open Delay timer is running, the local system: - resets the connect retry timer (sets to zero), - stops and reset the Open Delay timer (sets to zero, - releases all BGP resources, - drops the TCP connection, - changes its state to Idle. If the Open Delay timer is not running, the local system: - resets the connect retry timer (sets to zero), - releases all BGP resources, - drops the TCP connection, - increments the ConnectRetryCnt (connect retry count) by 1, - optionally performs peer oscillation damping, and - changes its state to Idle.

N.B. This is now event 24 (see issue 26).

We are at consensus with this.

This was discussed in the "Comments 21-30" thread: Comment #25. This was also brought up in the "BGP-19: Issue 34-35, 40-48" thread.

3.45. Event 17 in the Connect state

Status: Consensus
Change: Yes
Summary: Adopt text at the end of the discussion section.

Discussion:

This began with Siva asking:

.in 4 If the transport connection fails (timeout or transport disconnect) [Event17], the local system: - changes its state to

Active.

If the transport connection fails when the Open Delay timer [original: BGP Open Delay timer] is running, should we still be going into the Active state ?

Sue replied referring to the discussion tracked in issue 13.4.

Jeff responded that:

.in 4 In this particular case, I think the issue is separate from the issues for events 13-17 since this isn't particular to how deep the BGP implementation meddles in the TCP implementation.

If we are in the Connect state, because we have an incoming transport connection that has completed, but we have the OpenDelay timer running and the transport connection is closed, we can simply drop into Active after resetting the ConnectRetry timer and clearing the OpenDelay timer (if set/exists). In the case of an unconfigured peer, we can discard the FSM instance.

Tom replied that he agreed with this.

Tom then proposed this text:

If the TCP connection fails[Event 17] and the Open Delay timer is running, the local system: - restarts the connect retry timer, - clears the Open Delay timer - continues to listen for a connection that may be initiated by the remote BGP peer, and - changes its state to Active.

If the TCP connection fails [Event17] and the Open Delay timer is not running, the local system: - drops the TCP connection, - releases all BGP resources, - sets ConnectRetryCnt (the connect retry count) to zero - resets the connect retry timer (sets to zero), and - goes to Idle state.

to replace If the TCP connection fails (timeout or disconnect) [Event17], the local system: - restarts the connect retry timer, - continues to listen for a connection that may be initiated by the remote BGP peer, and - changes its state to Active.

Sue agreed to change the text to reflect the comments.

Jeff brought out a couple of other concerns, and Tom replied:

> If the TCP connection fails [Event17] and the Open Delay > timer is not running, the local system: > - drops the TCP connection, > -

releases all BGP resources,

.in 4 There are no resources to release while in the connect state. (Unless we're using this as shorthand for something else - I forget.)

Tom:

.in 4 I was unsure about this action. It is present for Active state event 17 which is why I put it in, it does include sub-actions such as clear Open Delay timer (not running), clear Connect Retry timer (could be running) so I think it right to play safe and include it.

Jeff:

> - sets ConnectRetryCnt (the connect retry count) to zero

I'm forgetting if this action is consistent with everything else. I don't have a current copy of the FSM and I don't trust -18 to be current enough. :-) This said, why do we go to zero? I could see not incrementing it and letting the normal decay process deal with it. The same would apply for the above.

Tom:

.in 4 Again, I was unsure about this so put it in and waited for comment. I have a chart of 27 events and 6 states in which I have colored in the connect retry and peer oscillation damping actions and it looks like measles; I could not divine the underlying logic. Incrementing the connect retry count would make as much if not more sense to me. (It is zeroed for Manual Stop).

But the action '[optionally] perform peer oscillation damping' is yet more erratic (eg for event 10 - Hold Timer expired - it is performed exiting Connect, Active, Established but not Open Confirm or Open Sent) so I left it out. Again, it might make more sense put it in.

Sue replied to this:

The connect state could have a few resources (minimum peer footprint) as the FSM goes from Idle to Connected state. While this amount of BGP resources is not as much as the final amount, it still needs to get released.

2nd - I think the ConnectRetry count should be removed; Thanks for catching that.

Please confirm that part #1 is OK with you so we can put issue 45 into consensus state.

Sue accepted Tom's solution, for the following text:

If the TCP connection fails [Event18], the local system checks the Open Delay Timer. If the Open Delay timer is running, the local system: - restarts the connect retry timer, - stops the Open Delay timer and resets value to zero, - continues to listen for a connection that may be initiated by the remote BGP peer, and - changes its state to Active. If the open Delay timer is not running, the local system: - resets the connect retry timer (sets to zero), and - Drops the TCP connection, - Releases all BGP resources, - and goes to Idle State.

N.B. This is now event 18 (see issue 26).

We are at consensus with this.

This was discussed in the "Comments 21-30" thread: Comment #26. This was also brought up in the "BGP-19: Issue 34-35, 40-48" thread.

3.46. Handling of Event 17 in Active state

Status: Consensus
Change: No
Summary: See issue 13.4, this issue closed in favor of that one.

Discussion:

This began with Siva saying:

We should now move into Idle state. Can we add

% - Goes to Idle state

Sue replied that she thought this should be bundled in with the issue tracked in 13.4. Since no one objected, this issue has been closed in favor of that one.

This was discussed in the "Comments 21-30" thread: Comment #27.

3.47. Handling of Event 19 in Active state

Status: Consensus
Change: Yes
Summary: Add the new text in the discussion section.

Discussion:

This began with Siva suggesting:

> - Set the Hold timer to a large value (4 minutes), Since OPEN messages have been exchanged, can we change this to - If the negotiated Hold time is not 0, set the Hold time to - the negotiated value

Sue replied that:

The text in Active and Open Sent needs to be the same. The text in Open Sent is: - sets the Hold timer according to the negotiated value (see section 4.2), and

Which text do you prefer?

Sue replied that this text would be added to the next draft:

New text

- if the hold timer value is non-zero, - starts the keepalive timer to initial value, - resets the hold timer to the negotiated value, - else if the hold timer is zero - resets the keepalive timer (set to zero), - resets the hold timer to zero.

This seems to address Siva's concerns, this issue is at consensus, if there are objections, we can reopen it.

This was discussed in the "Comments 21-30" thread: Comment #28. This was also brought up in the "BGP-19: Issue 34-35, 40-48" thread.

3.48. Handling of Event 2 in Active state

Status: Consensus
Change: Yes
Summary: Update the draft with the text at the end of the discussion section.

Discussion:

Siva opened with:

> A manual stop event[Event2], the local system: > - Sends a notification with a Cease, > - drops the Transport connection

These two actions are possible only if a transport connection had already been established. How about changing the text to

% - If a transport connection had been successfully established % - Send a Notification with a Cease % - Drop the Transport Connection

Sue counter suggested:

A manual stop event [Event 2], the local system - Drop the TCP connection, - Release all BGP resources, - resets the connection retry timer [sets to zero], - goes to Idle.

Jeff replied:

I'm rather confused. Under exactly what circumstances can we be in the Active state and have an active TCP connection at all? Ditto for having any BGP resources?

Going to Idle is fine.

Tom offered this example:

eg start with passive flag, TCP SYN received, SYN-ACK sent, ACK received, Delay Open flag set and there we are. Most events are now possible either from a well-implemented remote peer or a badly implemented remote peer.

Sue asked if there were any additional comments, if not, the text will be:

A manual stop event[Event2], the local system: - Sends a NOTIFICATION with a Cease, - releases all BGP resources including - stopping the Open delay timer - drops the TCP connection, - sets ConnectRetryCnt (connect retry count) to zero - resets the connect retry timer (sets to zero), - changes its state to Idle.

There have been no additional comments, we will use the text Sue proposed.

This was discussed in the "Comments 21-30" thread: Comment #29. This was also brought up in the "BGP-19: Issue 34-35, 40-48" thread.

3.49. Default Event handling in Active state

Status: Consensus
Change: No
Summary: No routes in active.

Discussion:

Siva began with:

To ensure consistency with E2 handling, can we add

% - If any BGP Routes exist, delete the routes

Sue replied:

Comment: Yakov and Jeff noted, there are no routes in Active state.

Since there were no responses disagreeing, we'll consider this closed unless someone wants to open it back up.

This was discussed in the "Comments 21-30" thread: Comment #30.

3.50. Clearing Hold timer in OpenSent, OpenConfirm and Established State

Status: Consensus
Change: No
Summary: This issue is addressed in the "Clear BGP resources"

Discussion:

This began with Siva stating:

.in 4 In all event handling where we go to Idle state, we need to clear the Hold Timer as well.

Sue replied that:

issue resolve one way last Jan - March Clearing of keep alive timer included in Clear BGP resources

No response to this yet, but since this seems to be resolved it is at consensus unless someone objects.

This was discussed in the "Comments 30-36" thread: Comment #31.

3.51. Clearing Keepalive timer in OpenConfirm and Established State

Status: Consensus
Change: No
Summary: This issue is addressed in the "Clear BGP resources"

Discussion:

This began with Siva stating:

.in 4 In all event handling where we go to Idle state, we need to clear the Keepalive Timer as well.

Sue replied that:

issue resolve one way last Jan - March Clearing of keep alive timer included in Clear BGP resources

No response to this yet, but since this seems to be resolved it is at consensus unless someone objects.

This was discussed in the "Comments 30-36" thread: Comment #32.

3.52. Handling Event 18 in the OpenSent state (Keepalive Timer)

Status: Consensus
Change: Yes
Summary: Make the event optional.

Discussion:

This began with Siva asking:

.in 4 Why do we start the Keepalive timer at this stage ? Isn't it sufficient to do so when we move into Established state ?

Sue replied:

An earlier comment from Tom (and you) requested the following:

<--Open [Open sent state]

Open--> [Event 18]

<---Open <---Keepalive [Action from Event 18 in Open Sent] [Open Confirm] Keepalive -> [Event 25] [established]

What do implementations do? We'll have to query implementations.

Jeff added:

I'm assuming the second OPEN going from right to left is a typo. If it isn't, thats a FSM error to the peer on the left.

Theoretically, an implementation that utilizes its keepalive timer to send the first keepalive to transition to Established is still interoperable. However:

o Keepalives can be disabled by negotiating hold time of zero o We really shouldn't need to restart the Keepalive timer. If there is a delay in the keepalive that transitions from OpenConfirm to

Established, its due to the transport connection. It should be reliable and it *should* get through. If it doesn't, there's other problems and the hold timer for the peer on the right should do the Right Thing and drop the connection.

> What do implementations do? We'll have to query implementations.

.in 4 GateD at least waits to enter the Established state prior to starting the KeepAlive timer.

Tom also added:

.in 4 My comment was that if we do not send a KeepAlive (and start the KeepAlive timer), on exiting from Active with Event 19 to OpenConfirm then we never will and the connection will die. Open Confirm state means valid Open received so we must send a KeepAlive to acknowledge the Open (as pointed out in Jeff's other posting) and we never do it in OpenConfirm state itself (unless the KeepAlive timer expires which it cannot because we have not started it).

So for me, OpenSent state Event 18 was and is correct, sending the KeepAlive without which the connection goes no further and Active state Event 19 needs to be brought into line.

To say that the timer is started when entering Established state is fine except for a slight problem; we have no way in this FSM of defining actions that are taken on entering a state, only actions to be taken on leaving another state so that is why the KeepAlive actions need to be where they are (or are not in the case of Active state Event 19).

Sue replied, asking more implementors to chime in on what they do for this part of the FSM.

Curtis replied that we should:

.in 4 Make it optional. Timing out in open or open-sent has never been much of an issue, so whether one or three keepalive get sent shouldn't be a hot topic.

Sue said that this was fine, and she would work on text specifying optional.

Jeff replied regarding GateD's behavior:

.in 4 GateD will start its keepalive timer while in this state, so multiple keepalives will be sent.

As someone previously said, this is a "yawn" issue. But to choose one way or the other, we may potentially make someone in non-compliance.

From the closure of issue 12, we have this text, which discusses Keepalives to consider in relation to the other keepalive issue here:

Change 1: new text

Active state - event 19

If an Open is received with the Open Delay timer is running [Event 19], the local system - clears the connect retry timer (cleared to zero), - stops and clears the Open Delay timer - completes the BGP initialization, - stops and clears the Open Delay timer - sends an OPEN message, - send a Keepalive message, - if the hold timer value is non-zero, - starts the keepalive timer to initial value, - resets the hold timer to the negotiated value, else if the hold timer is zero - resets the keepalive timer (set to zero), - resets the hold timer to zero.

- changes its state to OpenConfirm.

.in 7 If the value of the autonomous system field is the same as the local Autonomous System number, set the connection status to an internal connection; otherwise it is "external".

Since there were no more comments, this is at consensus.

This was discussed in the "Comments 30-36" thread: Comment #33. And in the "BGP-draft-19: Issue 52 - Event 18 in OpenSent State (Keepalive timer set)" thread.

3.53. Established State MIB

Status: Consensus
Change: No
Summary: MIB references pulled in favor of having them in the MIB document. See issue 8.

Discussion:

This began with Siva asking:

.in 4 Some event handling in the Established state do not set the MIB Reason when handling an event that causes an error. Can we add this ?

Sue replied that we have pulled the MIB wording from the FSM. See issue 8.

This was discussed in the "Comments 30-36" thread: Comment #34.

3.54. State impact of not supporting Optional Events

Status: Consensus
Change: Yes
Summary: Add the text at the end of the discussion section.

Discussion:

Siva stated that:

.in 4 For the events whose status is optional, can we state the impact of not supporting them (in terms of any interoperability issues). I understand that most of the optional events will not have such an impact; but a clarification statement for the optional events would benefit new implementors.

Sue responded:

Much of the support of optional parameters depends on policy. I could put a short note about the optional events and parameters as part of 8.1.5 or 8.2.1.3

I think it fits better in 8.1.5. Optional: Events: 3-8, 12, 13-14[my suggestion] 19, 22 Timers: Idle Hold Timer Open Delay Timer

Required flags for optional parameters: Open Delay Flag BGP Stop Flap

Sue said she would try to work up more if it is agreed that this is on the right track.

Sue provided this text to clarify the behavior associated with Optional Attributes:

8.2.1.3 FSM and Optional Attributes

Optional Attributes specify either flags that augment the normal processing of the BGP FSM, or optional timers. If a Optional attribute can be set on a system, the Events and the BGP FSM actions must be support. For example, if the following options can be set in a BGP implementation: AutoStart and Passive TCP connection Establishment flag, then the events 3, 4 and 5 must be supported.

If an Optional attribute cannot be set (that is declared always off

logically), the events supporting that set of options do not have to be supported.

This was discussed in the "Comments 30-36" thread: Comment #35.

3.55. New DelayOpen State

Status: Consensus
Change: No
Summary: We've chosen not to reopen the debate about adding a DelayOpen State to the FSM.

Discussion:

Siva began with asking:

.in 4 Is delaying the sending of an OPEN message a standard industry practice ?

Also, in the FSM, this has been handled by practically implementing a sub-state each, within the CONNECT and ACTIVE states. Won't the FSM look more simple if we just had a new DelayOpen state that we could move into ?

Sue responded that this was something we have tried to do before, but that it spawned some degree of rabid response on both sides. Given our current mandate to stick with what is implemented, it is probably best not to reopen this debate.

Unless someone badly wants to reopen this debate, the issue is at Consensus.

This was discussed in the "Comments 21-30" thread: Comment #22.
This was discussed in the "Comments 21-30" thread: Comment #26.
This was discussed in the "Comments 30-36" thread: Comment #36.

3.56. Clarify what is covered in the base document

Status: Consensus
Change: Yes
Summary: Add the text at the end of the discussion to clarify what is documented where with regard to BGP and its extensions.

Discussion:

This grew out of a discussion on how to use BGP Identifiers in an IPv6-only environment. In that discussion it became clear that the way the documents are currently structured it is not clear to new

readers that extension specifications can and do specify behavior that supersedes the behavior specified in the base spec. To that end it was agreed that this text should be added:

.in 5 This document specifies the base behavior of the BGP protocol. This behavior can and is modified by extension specifications. When the protocol is extended the new behavior is fully documented in the extension specifications.

This was discussed in the "Next-Hop in IPv6 only environments" thread.

4. Security Considerations

This document is an informational document that discusses the changes made in revising the BGP-4 specification. There are no security considerations applicable to this document.

5. Normative References

[RFC1771] Rekhter, Y. and T. Li, "A Border Gateway Protocol 4 (BGP-4)", RFC 1771, March 1995.

Author's Address

Andrew S. Lange
Alcatel-Lucent

Email: andrew.lange@alcatel-lucent.com