IDR Workgroup
Internet-Draft
Intended status: Standards Track
Expires: 7 January 2026

A. Retana
Y. Qu
Futurewei Technologies
J. Haas
Juniper Networks
S. Chen
Huawei Technologies
J. Tantsura
Nvidia

6 July 2025

BGP over QUIC
draft-retana-idr-bgp-quic-07

Abstract

This document specifies the procedures for BGP to use QUIC as a transport protocol with a mechanism to carry Network Layer protocols (AFI/SAFI) over multiple QUIC streams to achieve high resiliency.

Status of This Memo

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

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

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

This Internet-Draft will expire on 7 January 2026.

Table of Contents

1. Introduction

The Border Gateway Protocol (BGP) [RFC4271] is the routing protocol used to exchange routing and reachability information among autonomous systems. BGP uses TCP as its transport protocol to provide reliable communication. BGP establishes peer relationships between routers using a TCP session on port 179.

The Multiprotocol Extensions for BGP-4 (MP-BGP) [RFC4760] allow BGP to carry information for multiple Network Layer protocols. However, only a single TCP connection can reach the Established state between a pair of peers [RFC4271]. As a consequence, an error related to a particular Network Layer protocol may result in the termination of the connection for all.

QUIC [RFC9000] is a UDP-based multiplexed and secure transport protocol that provides connection-oriented and stateful interaction between a client and server. It can provide low latency and encrypted transport with resilient connections.

In QUIC, application protocols exchange information using streams. Each stream is a separate unidirectional or bidirectional channel consisting of an ordered stream of bytes. Moreover, each stream has its own flow control, which limits bytes sent on a stream, together with flow control of the connection.

This document specifies the procedures for BGP to use QUIC as a transport protocol with a mechanism to carry Network Layer protocols (AFI/SAFI) over individual streams. The Network Layer protocols are identified using a combination of Address Family (AFI) and Subsequent Address Family (SAFI), as described in [RFC4760]. These per-AFI/SAFI streams (function channels) and the associated control mechanism (control channel) for the session are called "BGP channels". In one BGP over QUIC (BoQ) connection, one control channel and one or more function channels are used to carry routing information.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Terminology

BoQ, Multi-channel BGP using QUIC: Running the BGP protocol over multiple QUIC streams as defined in this document.

QUIC connection: A transport-layer connection between two endpoints using QUIC [RFC9000].

QUIC streams: A bidirectional or unidirectional bytestream provided by the QUIC transport [RFC9000].

BGP channel: An instance of a BGP protocol state machine mapped to a specific QUIC stream.

BGP control channel: A channel dedicated to transmitting the session control data, which is implemented as a bidirectional stream.

BGP function channel: A BGP per AFI/SAFI channel, which is implemented asymmetrically as unidirectional streams.

4. Summary of Operations

4.1. Establish BGP/QUIC Connection

Before two BoQ speakers start exchanging routing information, they must establish a BGP session. It is established in two phases:

* Establish a transport layer connection. TLS 1.3 is integrated with QUIC. The TLS authentication parameters used for this connection are out of the scope of this draft.

* Establish a BoQ session over this transport connection. This document specifies the details of such an operation.

QUIC connections are established as described in [RFC9000]. During connection establishment, a BoQ speaker SHOULD use UDP port TBD1 and MUST select the Application-Layer Protocol Negotiation (ALPN) [RFC7301] token "boq" in the TLS handshake. Support for other application-layer protocols MUST NOT be offered in the same

handshake. A connection MUST be closed if the ALPN token is not as indicated or if other application-layer protocols are offered in the same handshake.

[RFC4271] defines the operations for a single BGP session between two BGP speakers using TCP. This document defines the ability to carry BGP over multiple QUIC streams as "BGP channels".

On a BoQ connection, the BoQ speaker first establishes a bidirectional stream for the "BGP control channel". The control channel is used to establish a BGP peer relationship between two BoQ speakers, similar to RFC 4271. OPEN messages are exchanged on the control channel, and if the BoQ session parameters are acceptable, the peering session is established. Similar to RFC 4271, the BoQ session is terminated with a NOTIFICATION message if the parameters are unacceptable.

After establishing the control channel, each BoQ speaker may create function channels using unidirectional QUIC streams. These function channels are used to carry BGP routes for a specific AFI/SAFI. Only one function channel per AFI/SAFI can exist from one BoQ speaker to another (see Section 5.4. Unlike [RFC4271] BGP, there is no requirement for both BoQ speakers to have a symmetric and matching set of function channels.

BGP channels largely use the mechanisms of the RFC 4271 Finite State Machine (FSM) for their establishment. For the control channel carried over a bidirectional QUIC stream, the FSM is identical to the RFC 4271 FSM. However, since the function channels are unidirectional, the RFC 4271 FSM procedures cannot be carried out solely using the unidirectional channel from one BoQ speaker to another. Instead, the responding BoQ speaker must carry its replies for the unidirectional streams over the control channel and address them to a specific BGP function channel.

4.2. Establish BGP/QUIC Control Channel

After BoQ session establishment, the BoQ speakers will create the control channel. The control channel is a bidirectional QUIC stream with stream ID 0 [RFC9000]. It is created by sending a BGP OPEN message. BGP OPEN messages carry parameters such as the Autonomous System number, BGP Identifier (router-id), Hold Time, and Capabilities. These parameters are used by a BoQ speaker to decide whether a BGP session is permitted to be established.

The capabilities carried in this OPEN message for the control channel are the BoQ connection-specific parameters; i.e. those that apply to the entire connection. An example of this is the BGP Role Capability [RFC9234]. If a function-only capability - as categorized in Table 1 - is included in the OPEN message, it MUST be ignored.

The control channel uses BGP hold time procedures as specified in [RFC4271]. KEEPALIVE messages are sent periodically in the absence of other messages on the control channel. If no messages are received within the negotiated hold time on the control channel, the BoQ connection is closed with a NOTIFICATION as specified in Section 6.5 of [RFC4271]. In short, the BoQ control channel is used to establish the peering relationship and connection parameters between the two BoQ speakers, ensure connectivity over this session is verified, and further is used as the response channel for the function channels as specified in Section 4.3.

It is an error to exchange BGP routing information over the control channel. This functionality is reserved for the Per-AFI/SAFI Function channels. If BGP routes are received on the control channel, the receiving BGP speaker MUST send a BGP NOTIFICATION with a Cease code on the control channel and close the QUIC connection.

QUIC supports connection migration. However, only the client side can move. The role of the QUIC endpoints is important. For future extensibility, a new BoQ Capability indicates the configured role of the BoQ speaker: Client, Server, or Any. It is expected that the BGP configuration and QUIC roles match. The QUIC connection can be reset if they don’t. See Section Section 5.1 for details.

4.3. Establish BGP/QUIC Function Channel

Per-AFI/SAFI Function channels are used to exchange routing information. After the control channel reaches the Established state, function channels are created as unidirectional QUIC streams and advertise routes for a single AFI/SAFI using BGP UPDATE messages. Only one function channel per AFI/SAFI can exist from one BoQ speaker to another (see Section 5.4).

It is an error to try to establish Per-AFI/SAFI Function channels prior to the control channel transitioning to the Established state. Per-AFI/SAFI Function channels SHOULD NOT be permitted to transition to the Established state prior to the control channel itself entering the Established state.

BoQ speakers asymmetrically create their function channels. While it might be the typical case for there to be a symmetric set of per-AFI/SAFI function channels, one for each speaker, this is not a

requirement. For example, BGP-LS [I-D.ietf-idr-rfc7752bis] may only require that a BoQ speaker asymmetrically receive BGP-LS NLRI and may not need to send them.

A BoQ speaker that needs to advertise routes to its peer opens a unidirectional stream to its neighbor by sending an OPEN message indicating the particular AFI/SAFI to be used. The BoQ connection-wide parameters have previously been exchanged over the control channel. The function channel OPEN messages MUST contain an identical BGP Autonomous System number and BGP Identifier as the previously established control channel. It is RECOMMENDED that the BGP Hold Time value exchanged in the function channels be significantly longer than the hold time negotiated for the control channel. It is the responsibility of the hold timer for the control channel to provide connection verification for the BoQ connection. The purpose of the function channel negotiated hold time is to provide verification of the communication between the two BoQ speakers for that AFI/SAFI.

The BGP Capabilities carried on the function channel SHOULD only be those that are function-specific, as categorized in Table 1. Conflicting BoQ connection-wide parameters exchanged over the function channel MAY result in the BoQ speaker sending a NOTIFICATION message and not permitting the per-AFI/SAFI function channel to become Established.

The receiving BoQ speaker replies to those messages as defined in the [RFC4271] FSM by sending its messages (OPEN/NOTIFICATION/KEEPALIVE) addressed to the sender over the control channel.

Once the function channel has reached the Established state, BGP UPDATE messages may be sent to the remote BoQ speaker.

A single function channel for an AFI/SAFI pair results in asymmetric route advertisements. Both BoQ speakers can create a function channel to implement symmetric route advertisements.

Each function channel is created independently to naturally support multi-channel BGP. The neighbor state machines are decoupled; in case of error, it is possible to reset only one function channel (one direction of a symmetric route exchange) using a BoQ Error Message with code BoQ Chanel Reset (see Section 6). If one function channel is blocked for some reason, other channels can still progress and operate.

4.4. Channel Reset

A NOTIFICATION is sent over the control channel if the entire BGP connection needs to be reset for any reason, such as a configuration change or a network outage. Existing error messages defined by [RFC4271] and other various extensions SHOULD be used.

If the control channel is closed, the QUIC connection MUST be terminated using a CONNECTION_CLOSE frame, and an error notification (see Section 6) should be included to indicate that the connection has been terminated by BGP. If there are other open channels, they are also closed when the connection is closed.

A function channel can be reset independently without impacting any other function channels or the control channel. Please refer to Section 6.

4.5. Channel Coordination

A single QUIC stream provides ordered and reliable delivery. However, there is no guarantee of transmission and delivery orders across streams. Therefore, if specific data from one channel needs to be received before data from other channels, this requirement must be accomplished through BGP.

As defined in [RFC9000], a QUIC implementation SHOULD provide ways in which an application can indicate the relative priority of streams.

A BGP implementation utilizing QUIC as its transport protocol MUST support a prioritization mechanism for BGP streams. This is essential for ensuring that critical routing information can be transmitted with higher priority compared to non-routing information.

How to implement the supported priorities using QUIC congestion control at the connection level, stream level flow control, and packetization are out of the scope of this document.

5. Protocol Definitions

5.1. BGP Over QUIC Capability

QUIC supports connection migration. However, only the client side can move. For a BoQ speaker to take advantage of the QUIC connection migration capability, it has to be the QUIC client.

For an implementation of the BoQ defined in this document, an explicit configuration is needed to identify a BoQ speaker's role: a QUIC client, a QUIC server, or any (Don' t care). The default value can be "any"; other values MUST be explicitly configured.

A new " BGP over QUIC" capability is defined below to signal whether the BoQ speaker is a QUIC client, a QUIC server, or any (Don' t care).

```
BoQ capability:
  Code: TBD2 (to be assigned by IANA)
  Length: 1(octet)
  Value:
    0 Any
    1 Client
    2 Server
```

The BoQ Capability is a control-only capability (see Table 1), which means it SHOULD only be sent in the control channel. It MUST be ignored if received in the OPEN message of any function channel.

A BoQ session MUST be terminated if the BoQ speaker role configuration and the QUIC connection role don't match by sending a NOTIFICATION on the control channel with an error code of BGP over QUIC Message Error and a Subcode BoQ Capability Mismatch, then closing the QUIC connection with a CONNECTION_CLOSE frame and an error code of APPLICATION_ERROR. Please refer to section 19.19 in [RFC9000] . For example, if a BoQ speaker is configured as a client, but the QUIC connection comes up as a QUIC server, the QUIC connection must be terminated. The "any" configuration matches both the QUIC client and QUIC server roles.

Before initiating a QUIC connection for BGP, the BoQ role configuration MUST be checked. If a BoQ speaker is configured as a QUIC client, it MUST try to initiate the QUIC connection. If a BoQ speaker is configured as a QUIC server, it MUST wait for a QUIC connection.

The following collision avoidance procedure SHOULD be followed during QUIC connection setup:

When one BoQ speaker is configured as a client, and the other side is configured as a server, no collision will happen. If the other side initiates a QUIC connection, a QUIC CONNECTION_CLOSE frame with error code APPLICATION_ERROR MUST be sent.

When a BoQ speaker is configured as "any" or as a server, it MUST accept the QUIC connection initiated by the other BoQ speaker.

During the control channel setup, the BoQ capability MUST be checked to make sure the configured BoQ role matches the QUIC connection. When both BoQ peers are configured as "any", the session collision mechanism defined in [RFC6286] and [RFC4271] MUST be followed.

In case there is a BoQ role mismatch, for example, a BoQ speaker configured as "any" accepted a QUIC connection from a BoQ speaker configured as server, an error notification, BoQ Capability Mismatch, SHOULD be sent and the connection MUST be terminated. Please refer to Section 6 for details.

5.2. Capability Category

For existing BGP capabilities, some of of them apply to the entire connection and MUST be sent in the control channel OPEN message, such as the BGP Role defined in [RFC9234]. If such capabilities are sent in an OPEN message in a function channel, they MUST be ignored.

The following table shows the category of each capability.

Value	Name	Ref	Control/ Function
1	Multiprotocol Extensions for BGP-4	RFC2858	F
2	Route Refresh Capability for BGP-4	RFC2918	F
3	Outbound Route Filtering Capability	RFC5291	F
5	Extended Next Hop Encoding	RFC8950	F
6	BGP Extended Message	RFC8654	C/F
7	BGPsec Capability	RFC8205	C/F
8	Multiple Labels Capability	RFC8277	C - deprecated
9	BGP Role	RFC9234	C
64	Graceful Restart Capability	RFC4724	C/F

65	Support for 4-octet AS number capability	RFC6793	C/F
67	Support for Dynamic Capability (capability specific)	draft-ietf-idr-dynamic-cap	C/F
68	Multisession BGP Capability	draft-ietf-idr-bgp-multisession	Not compatible
69	ADD-PATH Capability	RFC7911	F
70	Enhanced Route Refresh Capability	RFC7313	F
71	Long-Lived Graceful Restart (LLGR) Capability	draft-uttaro-idr-bgp-persistence	C/F
72	Routing Policy Distribution	draft-ietf-idr-rpd	F
73	FQDN Capability	draft-walton-bgp-hostname-capability	C
74	BFD Capability	draft-ietf-idr-bgp-bfd-strict-mode	C
75	Software Version Capability	draft-abraitis-bgp-version-capability	C/F

Table 1: Capability Category Table

5.3. Marker for End-of-RIB

The End-of-RIB marker was defined in [RFC4724] for the purpose of BGP graceful restart, however it was recommended because the generation of such a marker was helpful for routing convergence in general.

For an implementation of this specification, a BoQ speaker SHOULD always send the End-of-RIB marker to indicate to its peer the completion of the initial routing update after the function channel is established.

The format of the End-of-RIB marker is the same as specified in [RFC4724].

5.4. Channel Collision Avoidance

A function channel for a specific Network layer protocol MUST NOT be created if one already exists.

If a BoQ speaker receives a function channel creation request for an AFI/SAFI that already exists, the local BoQ speaker SHOULD send a notification with Error Code BoQ and subcode BoQ Channel Conflict through the control channel, and upon receiving this notification the channel initiator MUST terminate the channel.

If a BoQ speaker receives a functional channel creation request for an AFI/SAFI that it doesn’t support, the local BoQ speaker SHOULD send a notification using existing subcode "Unsupported AFI/SAFI" in the OPEN Message Error BGP NOTIFICATION message.

Unless allowed via configuration, a channel collision with an existing BGP channel in the Established state causes the closing of the newly created channel.

5.5. BoQ Framing Layer

In version 1 of QUIC, BoQ messages are carried by QUIC STREAM frames. In BoQ, the control channel always uses QUIC stream 0, which is a client-initiated bidirectional stream. Function channels, which are unidirectional streams, can be client or server initiated.

Some BoQ messages, although sent in the control channel, are meant for a function channel, such as the responding OPEN message or KEEPALIVE message for a function channel. These messages need to carry the corresponding function channel/stream ID information.

There are two types of BoQ Frames: Data and Control Data.

Data frames have the following format:

```
BoQ Data Frame Format {
  Type (16) = 0,
  Length (16),
  Frame Payload (...)
}
```

Control Data Frames have the following format:

```
BoQ Control Data Frame Format {
  Type (16) = 1,
  Length (16),
  Stream ID (62),
  padding (2) = 0,
  Frame Payload (...)
}
```

Type: two octets, identifying the frame type.

Length: The two-byte unsigned integer that describes the length in bytes of the frame payload.

Stream ID: A 62-bit integer indicating the receiving stream ID of this message.

Frame Payload: BGP messages.

The following table lists the frame type to be used when BGP messages are sent in different channels.

	Control Channel	Function Channel
OPEN	Control Data	Data
UPDATE	/	Data
KEEPALIVE	Control Data	Data
NOTIFICATION	Control Data	Data
Route-Refresh	Control Data	Data (subtype 1, 2)

Table 2: BoQ Frame Type Mapping

An OPEN message sent in the control channel for the control channel creation MUST NOT contain Multiprotocol Extensions Capability (value 1) in the Capabilities. An OPEN message sent in a function channel and the responding OPEN message sent in the control channel for one AFI/SAFI MUST contain only one Multiprotocol Extensions Capability (value 1) in the Capabilities.

There is no UPDATE message sent in the control channel.

For the KEEPALIVE and NOTIFICATION messages sent in the control channel for one function channel, the BoQ Control Data frame MUST be used, and the stream ID in the frame is to indicate the the target AFI/SAFI.

Route-refresh messages are sent in the control channel and function channels. Please refer to Section 5.6 for details.

5.6. Route Refresh

BGP Route Refresh messages are defined in [RFC2918] and [RFC7313].

When two BoQ peers with the enhanced route refresh capability, wish to initiate a route refresh from one peer to another, the initiating speaker MUST send a ROUTE-REFRESH message with subtype 0 in its control channel with the stream ID set to the corresponding AFI/SAFI.

When starting a route refresh for a specific AFI/SAFI, whether initiated locally or in response to a route refresh request from the peer, a BoQ speaker MUST send a beginning of a route refresh (BoRR) message in the function channel. After the re-advertisement of the route entries it MUST send an ending of a route refresh (EoRR) message in the function channel.

If only the route refresh capability defined in [RFC2918] is supported, but not the enhanced route refresh capability defined in [RFC7313], a BoQ speaker MUST send the route refresh request message in the control channel.

6. Error Handling

OPEN message error handling is defined in section 6.2 of [RFC4271]. This document defines a new NOTIFCATION error code:

Error Code	Name
TBD	BoQ Message Error

The following error subcode is defined as well:

Subcode	Name
1	BoQ Capability Mismatch
2	BoQ Connection Reset
3	BoQ Channel Reset
4	BoQ Channel Conflict

BoQ Capability Mismatch is sent when a BoQ speaker's configured role doesn't match the QUIC connection, and the connection MUST be terminated after sending this notification. Details are described in Section 5.1.

For a BoQ speaker, BGP UPDATE messages are only sent in function channels. If there is any error detected when processing an UPDATE message, a NOTIFICATION MUST be sent in the control channel with the error subcode and function channel stream ID. The function channel SHOULD be terminated upon receiving the NOTIFICATION message. The error checking process specified in Section 6.2 of [RFC4271] SHOULD be followed.

The error handling specified in this section is applicable for a BoQ speaker implementing this document.

Any individual BGP channel can be terminated as specified in [RFC4486].

7. BoQ Finite State Machine (FSM)

The data structures and FSM described in this document are conceptual and do not have to be implemented precisely as described here, as long as the implementations support the described functionality and they exhibit the same externally visible behavior.

A BoQ implementation is expected to maintain a separate FSM for each channel. The control channel in a BoQ connection is required to reach the Established state before any function channel can be created. This means the setup of the QUIC connection and any related errors are processed by the control channel.

The BGP messages and events handled by the control and function channels vary. In general, what is specified in [RFC4271] section 8 that applies to a BGP peer connection is applicable to a BoQ channel unless explicitly specified in this document.The BoQ FSM defined in this section is to replace section 8 of [RFC4271].

RFC 4271 section 8 defines the mandatory and optional session attributes for each connection. For a BoQ implementation, some of these attributes are applicable to both the control and function channels. However some attributes only apply to the control or function channels. Also, since the function channels in BoQ are unidirectional, the FSMs are different for the function channel sending side and receiving side. The following tables list the applicability of each attribute.

Table 3 lists the mandatory attributes required for each channel.

Channel Attributes	Control Channel	Function Channel Sending	Function Channel Receiving
State	Y	Y	Y
ConnectRetryCounter	Y	N	N
ConnectRetryTimer	Y	N	N
ConnectRetryTime	Y	N	N
HoldTimer	Y	Y	Y
HoldTime	Y	Y	Y
KeepaliveTimer	Y	Y	Y
KeepaliveTime	Y	Y	Y
StreamID	N	Y	Y

Table 3: BoQ Mandatory Channel Attributes

The state channel attribute indicated the current state of the BoQ FSM. The ConnectRetryCounter indicates the number of times a BoQ speaker has tried to establish a channel with its peer. For a BoQ function channel, a new mandatory attribute StreamID is required. This attribute indicates the QUIC Stream ID used by the function channel.

The optional channel attributes are in Table 4. These optional attributes may be supported in a channel if it is indicated as "Y" for a channel in the table.

Optional Channel Attributes	Control Channel	Function Channel Sending	Function Channel Receiving
AcceptConnectionsUnconfiguredPeers	Y	N	N
AllowAutomaticStart	Y	Y	N
AllowAutomaticStop	Y	Y	N
CollisionDetectEstablishedState	Y	N	N
DampPeerOscillations	Y	N	N
DelayOpen	Y	N	N
DelayOpenTime	Y	N	N
DelayOpenTimer	Y	N	N
IdleHoldTime	Y	N	N
IdleHoldTimer	Y	N	N
PassiveQUICEstablishment	Y	N	N
SendNOTIFICATIONwithoutOPEN	Y	N	Y
TrackQUICState	Y	N	N

Table 4: BoQ Optional Channel Attributes

7.1. Events for the BoQ FSM

7.1.1. Optional Events

The Inputs to the BoQ FSM are events. Events can either be mandatory or optional. Some optional events are linked to optional channel attributes. Optional channel attributes enable several groups of FSM functionality.

The linkage between FSM functionality, events, and the optional session attributes are as described in RFC4271, Section 8.1.1. Any updates of deviations are indicated below, and the applicability is summarized in the tables above.

Group 1: Automatic Administrative Events (start/Stop)
Optional Channel Attributes: AllowAutomaticStart, AllowAutomaticStop, DampPeerOscillations, IdleHoldTime, IdleHoldTimer

Option 1: AllowAutomaticStart

Description: A BoQ channel connection can be started and stopped by administrative control. This administrative control can either be manual, based on operator intervention, or under the control of logic that is specific to a BoQ implementation. The term "automatic" refers to a start being issued to the BoQ channel FSM when such logic determines that the BoQ channel should be restarted.

The AllowAutomaticStart attribute specifies that this BoQ channel supports automatic starting of the BoQ channel.

If a BoQ implementation supports AllowAutomaticStart, a channel may be repeatedly restarted. Three other options control the rate at which the automatic restart occurs: DampPeerOscillations, IdleHoldTime, and the IdleHoldTimer.

The DampPeerOscillations option specifies that the implementation engages additional logic to damp the oscillations of BoQ channels in the face of sequences of automatic start and automatic stop. IdleHoldTime specifies the length of time the BGP peer is held in the Idle state prior to allowing the next automatic restart. The IdleHoldTimer is the timer that holds the peer in Idle state.

Values: TRUE or FALSE

Option 2: AllowAutomaticStart

Description: This BoQ channel optional attribute indicates that the channel allows "automatic" stopping of the BoQ channel. An "automatic" stop is defined as a stop under the control of implementation-specific logic. The implementation-specific logic is outside the scope of this specification.

Values: TRUE or FALSE

Option 3: DampPeerOscillations

Description: The DampPeerOscillations optional channel attribute indicates that the BoQ channel connection is using logic that damps BGP peer oscillations in the Idle State.

Values: TRUE or FALSE

Option 4: IdleHoldTime

Description: The IdleHoldTime is the value that is set in the IdleHoldTimer.

Values: Time in seconds

Option 5: IdleHoldTimer

Description: The IdleHoldTimer aids in controlling BGP peer oscillation. The IdleHoldTimer is used to keep the BGP peer in Idle for a particular duration. The IdleHoldTimer_Expires event is described in Section 7.1.3.

Values: Time in seconds

Group 2: Unconfigured Peers
Optional Channel Attributes: AcceptConnectionsUnconfiguredPeers

Option 1: AcceptConnectionsUnconfiguredPeers

Description: The BoQ FSM optionally allows the acceptance of BoQ peer connections from neighbors that are not pre-configured. The "AcceptConnectionsUnconfiguredPeers" optional channel attribute allows the FSM to support the state transitions that allow the implementation to accept or reject these unconfigured peers and is only applicable to the control channel.

The AcceptConnectionsUnconfiguredPeers has security implications. Please refer to the BGP Vulnerabilities document [RFC4272] for details.

Values: TRUE or FALSE

Group 3: QUIC processing
Optional Channel Attributes: PassiveQUICEstablishment, TrackQUICState

Option 1: PassiveQUICEstablishment

Description: This option indicates that the BoQ FSM will

passively wait for the remote BGP peer to establish the BGP QUIC connection. As specified in Section 5.1, a BoQ speaker's role can be a QUIC client, a QUIC server, or any. If a BoQ speaker is configured as a QUIC client, the PassiveQUICEstablishment MUST be set to FALSE; when a BoQ speaker is configured as a QUIC server, the PassiveQUICEstablishment MUST be set to TRUE.

Values: TRUE or FALSE

Option 2: TrackQUICState

Description: The BoQ FSM normally tracks the end result of a QUIC connection attempt rather than individual QUIC messages. Optionally, the BoQ FSM can support additional interaction with the QUIC connection negotiation. The interaction with the QUIC events may increase the amount of logging the BGP peer connection requires and the number of BoQ FSM changes.

Values: TRUE or FALSE

Group 4: BGP Message Processing

Optional Session Attributes: DelayOpen, DelayOpenTime, DelayOpenTimer, SendNOTIFICATIONwithoutOPEN, CollisionDetectEstablishedState

Option 1: DelayOpen

Description: The DelayOpen optional channel attribute allows implementations to be configured to delay sending an OPEN message for a specific time period (DelayOpenTime). The delay allows the remote BGP Peer time to send the first OPEN message.

Values: TRUE or FALSE

Option 2: DelayOpenTime

Description: The DelayOpenTime is the initial value set in the DelayOpenTimer.

Values: Time in seconds

Option 3: DelayOpenTimer

Description: The DelayOpenTimer optional channel attribute is used to delay the sending of an OPEN message on a channel. The DelayOpenTimer_Expires event (Event 12) is described in Section 7.1.3.

Values: Time in seconds

Option 4: SendNOTIFICATIONwithoutOPEN

Description: The SendNOTIFICATIONwithoutOPEN allows a peer to send a NOTIFICATION without first sending an OPEN message. Without this optional channel attribute, the BGP connection assumes that an OPEN message must be sent by a peer prior to the peer sending a NOTIFICATION message.

Values: TRUE or FALSE

Option 5: CollisionDetectEstablishedState

Description: This optional channel attribute indicates that this BGP connection processes collisions in the Established state.

Values: TRUE or FALSE

7.1.2. Administrative Events

An administrative event is an event in which the operator interface and BGP Policy engine signal the BoQ FSM to start or stop the BGP state machine. The basic start and stop indications are augmented by optional connection attributes that signal a certain type of start or stop mechanism to the BoQ FSM.

Note that only Event 1 (ManualStart) and Event 2 (ManualStop) are mandatory administrative events. All other administrative events are optional (Events 3-8). Each event below has a name, definition, status (mandatory or optional), and the optional channel attributes that SHOULD be set at each stage. When generating Event 1 through Event 8 for the BoQ FSM, the conditions specified in the "Optional Attribute Status" section are verified. If any of these conditions are not satisfied, then the local system should log an FSM error.

The settings of optional channel attributes may be implicit in some implementations, and therefore may not be set explicitly by an external operator action. The administrative states described below may also be implicit in some implementations and not directly configurable by an external operator.

Event 1: ManualStart

Definition: Local system administrator manually starts a BoQ channel. For the control channel, this event indicates the start of the QUIC connection and the control channel. For a function channel, it is to start an unidirectional channel to the BoQ peer.

Status: Mandatory

Optional Attribute Status: For the control channel, the PassiveQUICEstablishment attribute SHOULD be set to FALSE.

Event 2: ManualStop
Definition: Local system administrator manually stops a BoQ channel. For the control channel, this event indicates the end of the BoQ connection.

Status: Mandatory

Optional Attribute Status: No interaction with any optional attributes.

Event 3: AutomaticStart
Definition: Local system automatically starts a BoQ channel. For the control channel, this event indicates the start of the QUIC connection and the control channel. For a function channel, it is to start an unidirectional channel to the BoQ peer.

Status: Optional

Optional Attribute Status: 1) The AllowAutomaticStart attribute SHOULD be set to TRUE if this event occurs.

2) If the PassiveQUICEstablishment optional session attribute is supported, it SHOULD be set to FALSE.

3) If the DampPeerOscillations optional session attribute is supported, it SHOULD be set to FALSE when this event occurs.

Event 4: ManualStart_with_PassiveQUICEstablishment
Definition: Local system administrator manually starts the peer connection, but has PassiveQUICEstablishment enabled. The PassiveQUICEstablishment optional attribute indicates that the peer will listen prior to establishing the connection. This event only applies to the control channel.

Status: Optional

Optional Attribute Status: 1) The PassiveQUICEstablishment attribute SHOULD be set to TRUE if this event occurs.

2) The DampPeerOscillations attribute SHOULD be set to FALSE when this event occurs.

Event 5: AutomaticStart_with_PassiveQUICEstablishment

Definition: Local system automatically starts the BGP connection with the PassiveQUICEstablishment enabled. The PassiveQUICEstablishment optional attribute indicates that the peer will listen prior to establishing a connection. This event only applies to the control channel.

Status: Optional

Optional Attribute Status: 1) The AllowAutomaticStart attribute SHOULD be set to TRUE.

2) The PassiveTcpEstablishment attribute SHOULD be set to TRUE.

3) If the DampPeerOscillations attribute is supported, the DampPeerOscillations SHOULD be set to FALSE.

Event 6: AutomaticStart_with_DampPeerOscillations

Definition: Local system automatically starts the BGP peer connection with peer oscillation damping enabled. The exact method of damping persistent peer oscillations is determined by the implementation and is outside the scope of this document.

Status: Optional

Optional Attribute Status: 1) The AllowAutomaticStart attribute SHOULD be set to TRUE.

2) The DampPeerOscillations attribute SHOULD be set to TRUE.

3) The PassiveQUICEstablishment attribute SHOULD be set to FALSE.

Event 7: AutomaticStart_with_DampPeerOscillations_and_PassiveQUICE stablishment

Definition: Local system automatically starts the BGP peer connection with peer oscillation damping enabled and PassiveQUICEstablishment enabled. The exact method of damping persistent peer oscillations is determined by the implementation and is outside the scope of this document. This event only applies to the control channel.

Status: Optional

Optional Attribute Status: 1) The AllowAutomaticStart attribute SHOULD be set to TRUE.

2) The DampPeerOscillations attribute SHOULD be set to TRUE.

3) The PassiveQUICEstablishment attribute SHOULD be set to TRUE.

Event 8: AutomaticStop
Definition: Local system automatically stops a BoQ channel. For the control channel, this event indicates the end of the BoQ connection.

An example of an automatic stop event for a function channel is exceeding the number of prefixes for a given peer and the local system automatically disconnecting the peer.

Status: Optional

Optional Attribute Status: 1) The The AllowAutomaticStop attribute SHOULD be TRUE.

7.1.3. Timer Events

Event 9: ConnectRetryTimer_Expires
Definition: An event generated when the ConnectRetryTimer expires.

Status: Mandatory

Event 10: HoldTimer_Expires
Definition: An event generated when the HoldTimer expires.

Status: Mandatory

Event 11: KeepaliveTimer_Expires
Definition: An event generated when the KeepaliveTimer expires.

Status: Mandatory

Event 12: DelayOpenTimer_Expires
Definition: An event generated when the DelayOpenTimer expires.

Status: Optional

Optional Attribute Status: If this even occurs,

1) DelayOpen attribute SHOULD be set to TRUE,

2) DelayOpenTime attribute SHOULD be supported,

3) DelayOpenTimer SHOULD be supported.

Event 13: IdleHoldTimer_Expires
Definition: An event generated when the IdleHoldTimer expires, indicating that the BoQ connection has completed waiting for the back-off period to prevent BGP peer oscillation.

The IdleHoldTimer is only used when the persistent peer oscillation damping function is enabled by setting the DampPeerOscillations optional attribute to TRUE.

Implementations not implementing the persistent peer oscillation damping function may not have the IdleHoldTimer.

Status: Optional

Optional Attribute Status: If this even occurs,

1) DampPeerOscillations attribute SHOULD be set to TRUE.

2) IdleHoldTimer SHOULD have just expired.

7.1.4. QUIC Connection-Based Events

Event 14: QUICConnection_Valid
Definition: Event indicating the local system reception of a QUIC connection request with a valid source IP address, UDP port, destination IP address, and UDP Port. The definition of invalid source and invalid destination IP address is determined by the implementation.

BoQ’s destination port SHOULD be port TDB1 (Section 4.1).

A QUIC connection request is denoted by the local system receiving the QUIC Initial packet [RFC9000]. This event only applies to the control channel.

Status: Optional

Optional Attribute Status: The TrackQUICState attribute SHOULD be set to TRUE if this event occurs.

Event 15: QUIC_CR_Invalid
Definition: Event indicating the local system reception of a QUIC connection request with either an invalid source address or port number, or an invalid destination address or port number.

BoQ’s destination port SHOULD be port TDB1 (Section 4.1).

A QUIC connection request is denoted by the local system receiving the QUIC Initial packet [RFC9000]. This event only applies to the control channel.

Status: Optional

Optional Attribute Status: The TrackQUICState attribute SHOULD be set to TRUE if this event occurs.

Event 16: QUIC_CR_Acked
Definition: Event indicating the local system's request to establish a QUIC connection to the remote peer has been completed.

The local system's QUIC connection request has completed and a HANDSHAKE_DONE frame is received [RFC9000]. This event only applies to the control channel.

Status: Mandatory

Event 17: QUICConnectionConfirmed
Definition: Event indicating that the local system has received a confirmation that the QUIC connection has been established by the remote site.

The HANDSHAKE_DONE frame has been acknowledged [RFC9000]. This event only applies to the control channel.

Status: Mandatory

Event 18: QUICConnectionFails
Definition: Event indicating that the local system has received a QUIC connection failure notice. This event only applies to the control channel.

Status: Mandatory

7.1.5. QUIC Stream-Based Events

Event 29: QUICStreamOpen
Definition: Event indicating that the local system's request to open a QUIC stream to the remote peer has been completed.

Status: Mandatory

Event 30: QUICStreamClose
Definition: Event indicating that the local system has received a

QUIC stream close notice. The event needs to include a StreamID to identify the stream closed.

Status: Mandatory

7.1.6. BGP Message-Based Events

Event 19: BGPOpen
Definition: An event is generated when a valid OPEN message has been received.

Status: Mandatory

Optional Attribute Status: 1) The DelayOpen optional attribute SHOULD be set to FALSE.

2) The DelayOpenTimer SHOULD not be running.

Event 20: BGPOpen with DelayOpenTimer running
Definition: An event is generated when 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.

Status: Optional

Optional Attribute Status: 1) The DelayOpen attribute SHOULD be set to TRUE.

2) The DelayOpenTimer SHOULD be running.

Event 21: BGPHeaderErr
Definition: An event is generated when a received BGP message header is not valid.

Status: Mandatory

Event 22: BGPOpenMsgErr
Definition: An event is generated when an OPEN message has been received with errors.

Status: Mandatory

Event 23: OpenCollisionDump
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 be disconnected. See Section 5.4 in this document and Section 6.8 in [RFC4271] for more information on collision detection.

Event 23 is an administrative action generated by implementation logic that determines whether this connection needs to be dropped per the rules in Section 5.4 in this document and Section 6.8 in [RFC4271].

Status: Optional

Optional Attribute Status: If the state machine is to process this event in the Established state,

1) CollisionDetectEstablishedState optional attribute SHOULD be set to TRUE.

Please note: The OpenCollisionDump event can occur in Idle, Connect, Active, OpenSent, and OpenConfirm without any optional attributes being set.

Event 24: NotifMsgVerErr
Definition: An event is generated when a NOTIFICATION message with "version error" is received.

Status: Mandatory

Event 25: NotifMsg
Definition: An event is generated when a NOTIFICATION message is received and the error code is anything but "version error".

Status: Mandatory

Event 26: KeepAliveMsg
Definition: An event is generated when a KEEPALIVE message is received.

Status: Mandatory

Event 27: UpdateMsg
Definition: An event is generated when a valid UPDATE message is received.

Status: Mandatory

Event 28: UpdateMsgErr
Definition: An event is generated when an invalid UPDATE message

is received.

Status: Mandatory

7.2. FSM Definition

BoQ implementations are expected to maintain a separate FSM for each BoQ channel. As described later in this section, a BoQ implementation will have one FSM for the control channel, plus one FSM for each direction of a function channel. For example, two BoQ speakers configured to advertise IPv6 routes in both directions will have 3 active FSMs: one for the control channel, and one for each direction (send and receive) of the IPv6 route exchange. One BoQ FSM corresponds to one QUIC stream connection.

A BoQ implementation MUST connect to and listen on UDP port TBD1 for incoming connections in addition to trying to connect to peers. 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 may exist for the same configured peering. This is referred to as a connection collision (see Section 6.8 of [RFC4271]).

7.2.1. Terms ’active’ and ’passive’

There is only one active side and one passive side to any one QUIC connection. In BGP over TCP, it doesn’t matter which end is active and which is passive. Differently in BoQ, the distinction is significant and they correspond to the QUIC client and server roles. Refer to Section 5.1 for more details.

7.2.2. FSM and Collision Detection

There is one control channel FSM per BoQ connection. When a connection collision occurs prior to determining what peer a connection is associated with, there may be two connections for one peer. Connection collisions are resolved using the specification in Section 6.8 of [RFC4271]. After the connection collision is resolved, the FSM for the connection that is closed SHOULD be disposed.

7.2.3. FSM and Optional Channel Attributes

Optional Channel Attributes specify either attributes that act as flags (TRUE or FALSE) or optional timers. For optional attributes that act as flags, if the optional channel attribute can be set to TRUE on the system, the corresponding BoQ FSM actions must be supported. For example, if the following options can be set in a BoQ implementation: AutoStart and PassiveQUICEstablishment, then Events 3, 4 and 5 must be supported. If an optional channel attribute cannot be set to TRUE, the events supporting that set of options do not have to be supported.

Each of the optional timers (DelayOpenTimer and IdleHoldTimer) has a group of attributes that are:

- flag indicating support,

- Time set in Timer,

- Timer.

The two optional timers show this format:

DelayOpenTimer: DelayOpen, DelayOpenTime, DelayOpenTimer

IdleHoldTimer: DampPeerOscillations, IdleHoldTime, IdleHoldTimer

If the flag indicating support for an optional timer (DelayOpen or DampPeerOscillations) cannot be set to TRUE, the timers and events supporting that option do not have to be supported.

7.2.4. FSM Event Numbers

Same as [RFC4271], the Event numbers (1-30) utilized in this state machine description aid in specifying the behavior of the BGP state machine. Note event 29 and 30 are defined in this document. Implementations MAY use these numbers to provide network management information. The exact form of an FSM or the FSM events are specific to each implementation.

7.3. BoQ Finite State Machine

7.3.1. Control Channel FSM

Idle State:
Initially, the control channel FSM is in the Idle state.

In this state, the control channel FSM refuses all incoming BGP connections for this peer. No resources are allocated to the peer. In response to a ManualStart event (Event 1) or an AutomaticStart event (Event 3), the local system:

* initializes all BGP resources for the peer connection,

* sets ConnectRetryCounter to zero,

* starts the ConnectRetryTimer with the initial value,

* initiates a QUIC connection to the other BGP peer if the local system is configured as BoQ client or any,

* listens for a connection that may be initiated by the remote BGP peer, and

* changes its state to Connect.

The ManualStop event (Event 2) and AutomaticStop (Event 8) event are ignored in the Idle state.

In response to a ManualStart_with_PassiveQUICEstablishment event (Event 4) or AutomaticStart_with_PassiveQUICEstablishment event (Event 5), the local system:

* initializes all BGP resources for the peer connection,

* sets the ConnectRetryCounter to zero,

* starts the ConnectRetryTimer with the initial value,

* listens for a connection that may be initiated by the remote peer, and

* changes its state to Active.

The exact value of the ConnectRetryTimer is a local matter, but it SHOULD be sufficiently large to allow QUIC initialization.

If the DampPeerOscillations attribute is set to TRUE, the following three additional events may occur within the Idle state:

* AutomaticStart_with_DampPeerOscillations (Event 6),

* AutomaticStart_with_DampPeerOscillations_and_ PassiveQUICEstablishment (Event 7),

* IdleHoldTimer_Expires (Event 13).

Upon receiving these 3 events, the local system will use these events to prevent peer oscillations. The method of preventing persistent peer oscillation is outside the scope of this document.

Any other event (Events 9-12, 15-30) received in the Idle state does not cause change in the state of the local system.

Connect State:

In this state, the control channel FSM is waiting for the QUIC connection to be completed.

The start events (Events 1, 3-7) are ignored in the Connect state.

In response to a ManualStop event (Event 2), the local system:

* drops the QUIC connection,

* releases all BGP resources,

* sets ConnectRetryCounter to zero,

* stops the ConnectRetryTimer and sets ConnectRetryTimer to zero, and

* changes its state to Idle.

In response to the ConnectRetryTimer_Expires event (Event 9), the local system:

* drops the QUIC connection,

* restarts the ConnectRetryTimer,

* stops the DelayOpenTimer and resets the timer to zero,

* initiates a QUIC connection to the other BGP peer,

* continues to listen for a connection that may be initiated by the remote BGP peer, and

* stays in the Connect state.

If the DelayOpenTimer_Expires event (Event 12) occurs in the Connect state, the local system:

* sends an OPEN message to its peer,

* sets the HoldTimer to a large value, and

* changes its state to OpenSent.

If the control channel FSM receives a QUICConnection_Valid event (Event 14), the QUIC connection is processed, and the connection remains in the Connect state.

If the control channel FSM receives a QUIC_CR_Invalid event (Event 15), the local system rejects the QUIC connection, and the connection remains in the Connect state.

If the QUIC connection succeeds (Event 16, Event 17 or Event 29), the local system checks the DelayOpen attribute prior to processing. If the DelayOpen attribute is set to TRUE, the local system:

* stops the ConnectRetryTimer (if running) and sets the ConnectRetryTimer to zero,

* sets the DelayOpenTimer to the initial value, and

* stays in the Connect state.

If the DelayOpen attribute is set to FALSE, the local system:

* stops the ConnectRetryTimer (if running) and sets the ConnectRetryTimer to zero,

* completes BGP initialization

* sends an OPEN message to its peer,

* sets the HoldTimer to a large value, and

* changes its state to OpenSent.

A HoldTimer value of 4 minutes is suggested.

If the QUIC connection fails (Event 18), the local system checks the DelayOpenTimer. If the DelayOpenTimer is running, the local system:

* restarts the ConnectRetryTimer with the initial value,

* stops the DelayOpenTimer and resets its 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 DelayOpenTimer is not running, the local system:

* stops the ConnectRetryTimer to zero,

* drops the QUIC connection,

* releases all BGP resources, and

* changes its state to Idle.

If the QUIC stream fails (Event 30) with StreamID 0, the local system:

* stops the ConnectRetryTimer to zero,

* drops the QUIC connection,

* releases all BGP resources, and

* changes its state to Idle.

If an OPEN message is received while the DelayOpenTimer is running (Event 20), the local system:

* stops the ConnectRetryTimer (if running) and sets the ConnectRetryTimer to zero,

* completes the BGP initialization,

* stops and clears the DelayOpenTimer (sets the value to zero),

* sends an OPEN message,

* sends a KEEPALIVE message,

* if the HoldTimer initial value is non-zero,

  - starts the KeepaliveTimer with the initial value and

  - resets the HoldTimer to the negotiated value,

* else, if the HoldTimer initial value is zero,

- resets the KeepaliveTimer and

- resets the HoldTimer value to zero,

* and 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 will be "external".

If BGP message header checking (Event 21) or OPEN message checking detects an error (Event 22), the local system:

* (optionally) If the SendNOTIFICATIONwithoutOPEN attribute is set to TRUE, then the local system first sends a NOTIFICATION message with the appropriate error code, and then

* stops the ConnectRetryTimer (if running) and sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

If a NOTIFICATION message is received with a version error (Event 24), the local system checks the DelayOpenTimer. If the DelayOpenTimer is running, the local system:

* stops the ConnectRetryTimer (if running) and sets the ConnectRetryTimer to zero,

* stops and resets the DelayOpenTimer (sets to zero),

* releases all BGP resources,

* drops the QUIC connection, and

* changes its state to Idle.

If the DelayOpenTimer is not running, the local system:

* stops the ConnectRetryTimer and sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* performs peer oscillation damping if the DampPeerOscillations attribute is set to True, and

* changes its state to Idle.

In response to any other events (Events 8, 10-11, 13, 19, 23, 25-28), the local system:

* if the ConnectRetryTimer is running, stops and resets the ConnectRetryTimer (sets to zero),

* if the DelayOpenTimer is running, stops and resets the DelayOpenTimer (sets to zero),

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* performs peer oscillation damping if the DampPeerOscillations attribute is set to True, and

* changes its state to Idle.

Active State:

In this state, the control channel FSM is trying to acquire a peer by listening for, and accepting, a QUIC connection.

The start events (Events 1, 3-7) are ignored in the Active state.

In response to a ManualStop event (Event 2), the local system:

* (Optionally)If the SendNOTIFICATIONwithoutOPEN session attribute is set, the local system sends a NOTIFICATION with a Cease,

* releases all BGP resources including stopping the DelayOpenTimer,

* drops the QUIC connection,

* sets ConnectRetryCounter to zero,

* stops the ConnectRetryTimer and sets the ConnectRetryTimer to zero, and

* changes its state to Idle.

In response to a ConnectRetryTimer_Expires event (Event 9), the local system:

* restarts the ConnectRetryTimer (with initial value),

* initiates a QUIC connection to the other BGP peer,

* continues to listen for a QUIC connection that may be initiated by a remote BGP peer, and

* changes its state to Connect.

If the local system receives a DelayOpenTimer_Expires event (Event 12), the local system:

* sets the ConnectRetryTimer to zero,

* stops and clears the DelayOpenTimer (set to zero),

* completes the BGP initialization,

* sends the OPEN message to its remote peer,

* sets its hold timer to a large value, and

* changes its state to OpenSent.

A HoldTimer value of 4 minutes is also suggested for this state transition.

If the local system receives a QUICConnection_Valid event (Event 14), the local system processes the QUIC connection flags and stays in the Active state.

If the local system receives a QUIC_CR_Invalid event (Event 15), the local system rejects the QUIC connection and stays in the Active State.

In response to the success of a QUIC connection (Event 16, Event 17 or Event 29), the local system checks the DelayOpen optional attribute prior to processing.

If the DelayOpen attribute is set to TRUE, the local system:

- stops the ConnectRetryTimer and sets the ConnectRetryTimer to zero,
- sets the DelayOpenTimer to the initial value (DelayOpenTime), and
- stays in the Active state.

If the DelayOpen attribute is set to FALSE, the local system:

- sets the ConnectRetryTimer to zero,
- completes the BGP initialization,
- sends the OPEN message to its peer,
- sets its HoldTimer to a large value, and
- changes its state to OpenSent.

A HoldTimer value of 4 minutes is suggested as a "large value" for the HoldTimer.

If the local system receives a QUICConnectionFails event (Event 18), the local system:

* restarts the ConnectRetryTimer (with the initial value),
* stops and clears the DelayOpenTimer (sets the value to zero),
* releases all BGP resource,
* increments the ConnectRetryCounter by 1,
* optionally performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and
* changes its state to Idle.

If an OPEN message is received and the DelayOpenTimer is running (Event 20), the local system:

* stops the ConnectRetryTimer (if running) and sets the ConnectRetryTimer to zero,

* stops and clears the DelayOpenTimer (sets to zero),

* completes the BGP initialization,

* sends an OPEN message,

* sends a KEEPALIVE message,

* if the HoldTimer value is non-zero:

  - starts the KeepaliveTimer to initial value,

  - resets the HoldTimer to the negotiated value,

* else if the HoldTimer is zero:

  - resets the KeepaliveTimer (set to zero),

  - resets the HoldTimer to zero, and

* 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 will be external.

If BGP message header checking (Event 21) or OPEN message checking detects an error (Event 22) (see Section 6), the local system:

* (optionally) sends a NOTIFICATION message with the appropriate error code if the SendNOTIFICATIONwithoutOPEN attribute is set to TRUE,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

If a NOTIFICATION message is received with a version error (Event 24), the local system checks the DelayOpenTimer. If the DelayOpenTimer is running, the local system:

* stops the ConnectRetryTimer (if running) and sets the

* ConnectRetryTimer to zero,

* stops and resets the DelayOpenTimer (sets to zero),

* releases all BGP resources,

* drops the QUIC connection, and

* changes its state to Idle.

If the DelayOpenTimer is not running, the local system:

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

In response to any other event (Events 8, 10-11, 13, 19, 23, 25-28, 30), the local system:

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by one,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

OpenSent State:

In this state, the control channel FSM waits for an OPEN message from its peer.

The start events (Events 1, 3-7) are ignored in the OpenSent state.

If a ManualStop event (Event 2) is issued in the OpenSent state, the local system:

* sends the NOTIFICATION with a Cease,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* sets the ConnectRetryCounter to zero, and

* changes its state to Idle.

If an AutomaticStop event (Event 8) is issued in the OpenSent state, the local system:

* sends the NOTIFICATION with a Cease,

* sets the ConnectRetryTimer to zero,

* releases all the BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

If the HoldTimer_Expires (Event 10), the local system:

* sends a NOTIFICATION message with the error code Hold Timer Expired,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

If a QUICConnection_Valid (Event 14), QUIC_CR_Acked (Event 16), or a QUICConnectionConfirmed event (Event 17) is received, a second QUIC connection may be in progress. This second QUIC connection is tracked per Connection Collision processing (Section 6.8 of [RFC4271]) until an OPEN message is received.

A QUIC Connection Request for an Invalid port (QUIC_CR_Invalid (Event 15)) is ignored.

If a QUICStreamOpen (Event 29) is received, and the stream ID is 0, the local system stays in OpenSent state. If the steam ID is not 0, the local system:

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection, and

* changes its state to Idle.

If a QUICConnectionFails event (Event 18) is received, the local system:

* closes the BGP connection,

* restarts the ConnectRetryTimer,

* continues to listen for a connection that may be initiated by the remote BGP peer, and

* changes its state to Active.

If a QUICStreamClose event (Event 30) with StreamID 0 is received, the local system:

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection, and

* changes its state to Idle.

When an OPEN message is received (Event 19), all fields are checked for correctness. If there are no errors in the OPEN message (Event 19), the local system:

* resets the DelayOpenTimer to zero,

* sets the BGP ConnectRetryTimer to zero,

* sends a KEEPALIVE message, and

* sets a KeepaliveTimer,

* sets the HoldTimer according to the negotiated value,

* changes its state to OpenConfirm.

If the negotiated hold time value is zero, then the HoldTimer and KeepaliveTimer are not started. If the value of the Autonomous System field is the same as the local Autonomous System number, then the connection is an "internal" connection; otherwise, it is an "external" connection.

If the BGP message header checking (Event 21) or OPEN message checking detects an error (Event 22), the local system:

* sends a NOTIFICATION message with the appropriate error code,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is TRUE, and

* changes its state to Idle.

Collision detection mechanisms need to be applied when a valid BGP OPEN message is received (Event 19 or Event 20). Please refer to Section 5.4 in this document and Section 6.8 [RFC4271] of for the details of the comparison. A CollisionDetectDump event occurs when the BGP implementation determines, by means outside the scope of this document, that a connection collision has occurred.

If a connection in the OpenSent state is determined to be the connection that must be closed, an OpenCollisionDump (Event 23) is signaled to the state machine. If such an event is received in the OpenSent state, the local system:

* sends a NOTIFICATION with a Cease,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

If a NOTIFICATION message is received with a version error (Event 24), the local system:

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection, and

* changes its state to Idle.

In response to any other event (Events 9, 11-13, 20, 25-28), the local system:

* sends the NOTIFICATION with the Error Code Finite State Machine Error,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

OpenConfirm State:
The control channel FSM waits for a KEEPALIVE or NOTIFICATION message.

Any start event (Events 1, 3-7) is ignored in the OpenConfirm state.

In response to a ManualStop event (Event 2) initiated by the operator, the local system:

* sends the NOTIFICATION message with a Cease,

* releases all BGP resources,

* drops the QUIC connection,

* sets the ConnectRetryCounter to zero,

* sets the ConnectRetryTimer to zero, and

* changes its state to Idle.

In response to the AutomaticStop event initiated by the system (Event 8), the local system:

* sends the NOTIFICATION message with a Cease,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

If the HoldTimer_Expires event (Event 10) occurs before a KEEPALIVE message is received, the local system:

* sends the NOTIFICATION message with the Error Code Hold Timer Expired,
* sets the ConnectRetryTimer to zero,
* releases all BGP resources,
* drops the QUIC connection,
* increments the ConnectRetryCounter by 1,
* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and
* changes its state to Idle.

If the local system receives a KeepaliveTimer_Expires event (Event 11), the local system:

* sends a KEEPALIVE message,
* restarts the KeepaliveTimer, and
* remains in the OpenConfirm state.

In the event of a QUICConnection_Valid event (Event 14), the success of a QUIC connection (Event 16 or Event 17), or QUICStreamOpen (Event 29) while in OpenConfirm, the local system needs to track the second connection.

If a QUIC connection is attempted with an invalid port (Event 15), the local system will ignore the second connection attempt.

If the local system receives a QUICConnectionFails event (Event 18) from the underlying QUIC or a NOTIFICATION message (Event 25), the local system:

* sets the ConnectRetryTimer to zero,
* releases all BGP resources,
* drops the QUIC connection,
* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

If the local system receives a NOTIFICATION message with a version error (NotifMsgVerErr (Event 24)), or a QUICStream_Close event (Event 30) with StreamID 0, the local system:

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection, and

* changes its state to Idle.

If the local system receives a valid OPEN message (BGPOpen (Event 19)), the collision detect function is processed per Section 5.4 in this document and Section 6.8 [RFC4271]. If this connection is to be dropped due to connection collision, the local system:

* sends a NOTIFICATION with a Cease,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection (send CONNECTION_CLOSE frame),

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

If an OPEN message is received, all fields are checked for correctness. If the BGP message header checking (BGPHeaderErr (Event 21)) or OPEN message checking detects an error (BGPOpenMsgErr (Event 22)), the local system:

* sends a NOTIFICATION message with the appropriate error code,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

If, during the processing of another OPEN message, the BGP implementation determines, by a means outside the scope of this document, that a connection collision has occurred and this connection is to be closed, the local system will issue an OpenCollisionDump event (Event 23). When the local system receives an OpenCollisionDump event (Event 23), the local system:

* sends a NOTIFICATION with a Cease,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

If the local system receives a KEEPALIVE message (KeepAliveMsg (Event 26)), the local system:

* restarts the HoldTimer and

* changes its state to Established.

In response to any other event (Events 9, 12-13, 20, 27-28, 30), the local system:

* sends a NOTIFICATION with a code of Finite State Machine Error,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

Established State:

In this state, the control channel FSM can exchange NOTIFICATION, KEEPALIVE and ROUTEREFRESH messages with its peer. There is no UPDATE message in the control channel.

Any Start event (Events 1, 3-7, 27-29) is ignored in the Established state.

In response to a ManualStop event (initiated by an operator) (Event 2), the local system:

* sends the NOTIFICATION message with a Cease,

* sets the ConnectRetryTimer to zero,

* releases BGP resources,

* drops the QUIC connection,

* sets the ConnectRetryCounter to zero, and

* changes its state to Idle.

In response to an AutomaticStop event (Event 8), the local system:

* sends a NOTIFICATION with a Cease,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

The following text from [RFC4271] is not applicable to BoQ control channel FSM as there is no UPDATE message in the control channel.

> One reason for an AutomaticStop event is: A BGP receives an UPDATE messages with a number of prefixes for a given peer such that the total prefixes received exceeds the maximum number of prefixes configured. The local system automatically disconnects the peer.

If the HoldTimer_Expires event occurs (Event 10), the local system:

* sends a NOTIFICATION message with the Error Code Hold Timer Expired,
* sets the ConnectRetryTimer to zero,
* releases all BGP resources,
* drops the QUIC connection,
* increments the ConnectRetryCounter by 1,
* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and
* changes its state to Idle.

If the KeepaliveTimer_Expires event occurs (Event 11), the local system:

* sends a KEEPALIVE message, and
* restarts its KeepaliveTimer, unless the negotiated HoldTime value is zero.

Each time the local system sends a KEEPALIVE message, it restarts its KeepaliveTimer, unless the negotiated HoldTime value is zero.

A QUICConnection_Valid (Event 14), received for a valid port, will cause the second connection to be tracked.

An invalid QUIC connection (QUIC_CR_Invalid event (Event 15)) will be ignored.

If the Graceful Restart Capability [RFC4724] with one or more AFIs/SAFIs has not been received for the session, then in response to an indication that a QUIC connection is successfully established (Event 16 or Event 17), the second connection SHALL be tracked until it sends an OPEN message.

However, if the Graceful Restart Capability with one or more AFIs/SAFIs has been received for the session, then in response to Event 16 or Event 17 the local system:

* retains all routes associated with this connection according to section 4.2 of [RFC4724].

* releases all other BGP resources,

* drops the QUIC connection associated with the ESTABLISHED session,

* initializes all BGP resources for the peer connection, other than those required in order to retain routes according to section 4.2 of [RFC4724],

* sets ConnectRetryCounter to zero,

* starts the ConnectRetryTimer with the initial value, and

* changes its state to Connect.

If a valid OPEN message (BGPOpen (Event 19)) is received, and if the CollisionDetectEstablishedState optional attribute is TRUE, the OPEN message will be checked to see if it collides (Section 6.8 in [RFC4271]) with any other connection. If the BGP implementation determines that this connection needs to be terminated, it will process an OpenCollisionDump event (Event 23). If this connection needs to be terminated, the local system:

* sends a NOTIFICATION with a Cease,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations is set to TRUE, and

* changes its state to Idle.

If the local system receives a NOTIFICATION message (Event 24 or Event 25) or a QUICConnectionFails (Event 18) from the underlying QUIC, and the Graceful Restart capability with one or more AFIs/SAFIs has not been received for the session, the local system:

* sets the ConnectRetryTimer to zero,

* releases all the BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* changes its state to Idle.

However, if the local system receives a QUICConnectionFails (Event 18) from the underlying QUIC, and the Graceful Restart Capability with one or more AFIs/SAFIs has been received for the session, the local system:

* sets the ConnectRetryTimer to zero,

* retains all routes associated with this connection according to section 4.2 of [RFC4724],

* releases all other BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1, and

* changes its state to Idle.

If the local system receives a QUICStream_Close event (Event 30) with StreamID 0, the local system:

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection, and

* changes its state to Idle.

When a KEEPALIVE message (Event 26) is received, whether it is destined to the control channel or a function channel, the local system:

* restarts its HoldTimer, if the negotiated HoldTime value is non-zero, and

* remains in the Established state.

If the local system receives an UPDATE message, it SHOULD be ignored.

In response to any other event (Events 9, 12-13, 20-22), the local system:

* sends a NOTIFICATION message with the Error Code Finite State Machine Error,

* sets the ConnectRetryTimer to zero,

* releases all BGP resources,

* drops the QUIC connection,

* increments the ConnectRetryCounter by 1,

* (optionally) performs peer oscillation damping if the DampPeerOscillations attribute is set to TRUE, and

* changes its state to Idle.

7.3.2. Function Channel FSM

Function channels can only be created after the control channel has reached Established state. QUIC connection related events (Event 14-18) MUST be handled exclusively by the control channel. However, a function channel may receive QUIC stream based events (Event 29 and 30).

Function channels operate in a unidirectional manner. For each AFI/SAFI, there may exist one or two FSMs: one function channel sending FSM and one function channel receiving FSM.

The following events SHOULD NOT be received in a function channel:

* ManualStart_with_PassiveQUICEstablishment event (Event 4)

* AutomaticStart_with_PassiveQUICEstablishment event (Event 5)

* AutomaticStart_with_DampPeerOscillations_and_PassiveQUICEstablishment (Event 7)

* ConnectRetryTimer_Expires (Event 9)

* DelyOpenTimer_Expires (Event 12)

* QUICConnection_Valid (Event 14)

* QUIC_CR_Invalid (Event 15)

* QUIC_CR_Acked (Event 16)

* QUICConnectionConfirmed (Event 17)

* QUICConnectionFails (Event 18)

* BGPOpen with DelyOpenTimer running (Event 20)

If any of the above events are received, they SHOULD be ignored.

In BoQ, an AFI/SAFI can be started/reset/deleted independently. When an AFI/SAFI is deleted, the corresponding FSMs SHOULD be deleted as well. Compared with the FSM defined in [RFC4271], there is a new state called "Terminating" for a Function channel FSM. Once a FSM enters the Terminating state, it means it should be deleted.

7.3.2.1. Function Channel Sending FSM

Connect State:
Function channel sending FSM starts from connect state.

The events (Events 1, 3-7, 9, 12-18, 20) are ignored in the Connect state.

The local system:

* initializes the function channel BGP resources for the peer connection,

* initiates a QUIC stream to the other BGP peer by sending an OPEN message using a unidirectional QUIC stream,

* sets the HoldTimer to a large value,

* record the StreamID, and

* changes its state to OpenSent.

In response to a ManualStop event (Event 2), or a QUICStreamClose event (Event 30) is received and the StreamID matches the function channel's StreamID, the local system:

* releases all BGP resources for the channel,

* changes its state to Terminating.

If a NOTIFICATION message (Event 25) or a NOTIFICATION message is received with a version error (Event 24) in the control channel destined to the function channel, the local system:

* release all BGP resources for the function channel,

* changes its state to Terminating.

In response to any other events (Events 8, 10-11, 13), the local system:

* releases all BGP resources for the channel,

* changes its state to Terminating.

Any other event (Event 14-23, 26-29) received in the Connect state does not cause change in the state of the local system.

OpenSent State:

A BoQ speaker waits for an OPEN message from its peer in the control channel with destination StreamID matches its own StreamID.

The start events (Events 1, 3-7, 9, 12-18, 20, 29) are ignored in the OpenSent state.

If a ManualStop event (Event 2), an AutomaticStop event (Event 8), or a QUICStreamClose event (Event 30) is issued in the OpenSent state, the local system:

* sends the NOTIFICATION with a Cease,

* releases all BGP resources for this channel,

* closes the stream,

* changes its state to Terminating.

If the HoldTimer_Expires event (Event 10) is received in the OpenSent state, the local system:

* sends the NOTIFICATION with the error code Hold Timer Expired,

* releases all BGP resources for this channel,

* closes the stream,

* changes its state to Terminating.

When an OPEN message from its peer is received, it’s checked for correctness. In case of error, the local BoQ speaker:

* sends a NOTFICATION message,

* closes the channel/QUIC stream,

* releases the corresponding BGP resources for the channel,

* and changes its state to Terminating.

When an OPEN message (Event 19) is received and there is no error in the received OPEN message, the local system:

* sends a KEEPALIVE message,

* sets a KeepAlive timer,

* sets the HoldTimer, and

* changes its state to OpenConfirm.

If the negotiated hold time value is zero, then the HoldTimer and KeepaliveTimer are not started.

If the BGP message header checking (Event 21) or OPEN message checking detects an error (Event 22), or an OpenCollisionDump (Event 23) is received, the local system:

* sends a NOTFICATION message with the appropriate error code,

* closes the channel/QUIC stream,

* releases the corresponding BGP resources for the channel,

* changes its state to Terminating.

If a NOTIFICATION message (Event 25) or a NOTIFICATION message with version error (Event 24) is received in the control channel with matching StreamID, the local system:

* releases the related BGP resources for the channel,

* closes the stream, and

* changes its state to Terminating.

In response to any other event (Events 11, 26-28), the local system:

* sends the NOTIFICATION with the Error Code Finite State Machine Error,

* releases related BGP resources,

* drops the QUIC stream connection/channel,

* changes its state to Terminating.

OpenConfirm State:
In this state, BoQ waits for a KEEPALIVE or NOTIFICATION message.

Any start event (Events 1, 3-7, 12-20, 29) is ignored in the OpenConfirm state.

In response to a ManualStop event (Event 2) initiated by the operator, the local system:

* sends the NOTIFICATION message with a Cease,

* releases all BGP resources for the channel,

* closes the stream, and

* changes its state to Terminating.

In response to the AutomaticStop event initiated by the system (Event 8), the local system:

* sends the NOTIFICATION message with a Cease,

* releases all BGP resources,

* drops the stream/channel connection,

* changes its state to Terminating.

If the HoldTimer_Expires event (Event 10) occurs before a KEEPALIVE message is received, the local system:

* sends the NOTIFICATION message with the Error Code Hold Timer Expired,

* releases related BGP resources,

* drops the stream/channel connection,

* changes its state to Terminating.

If the local system receives a KeepaliveTimer_Expires event (Event 11), the local system:

* sends a KEEPALIVE message,

* restarts the KeepaliveTimer, and

* remains in the OpenConfirmed state.

If the local system receives a QUICStreamClose event (Event 30), a NOTIFICATION message (Event 25), or a NOTIFICATION message with a version error (NotifMsgVerErr (Event 24))the local system:

* releases all BGP resources for the channel,

* closes the channel connection,

* changes its state to Terminating.

If an OPEN message is received in the control channel for this function channel, all fields are checked for correctness. If the BGP message header checking (BGPHeaderErr (Event 21)), OPEN message checking detects an error (BGPOpenMsgErr (Event 22)), or a OpenCollisionDump event (Event 23) is received, the local system:

* sends a NOTIFICATION message with the appropriate error code,

* releases all BGP resources for the channel,

* drops the stream connection,

* changes its state to Idle.

If the local system receives a KEEPALIVE message (KeepAliveMsg (Event 26)) from the control channel with matching stream ID, the local system:

* restarts the HoldTimer and

* changes its state to Established.

In response to any other event (Events 9, 27-28), the local system:

* sends a NOTIFICATION with a code of Finite State Machine Error,

* releases all BGP resources for the channel,

* drops the stream connection,

* changes its state to Idle.

Established State:
When the sending function channel reaches established state, it sends UPDATE, NOTIFCATION and KEEPALIVE messages to its peer.

Any Start event (Events 1, 3-7, 12, 14-20, 29) is ignored in the Established state.

In response to a ManualStop event (initiated by an operator)(Event 2), an AutomaticStop event (Event 8) or a QUICStreamClose (Event 30), the local system:

* sends the NOTIFICATION message with a Cease,

* deletes all routes associated with this connection,

* releases related BGP resources for the channel,

* drops the stream/channel connection,

* changes its state to Terminating.

If the HoldTimer_Expires event occurs (Event 10), the local system:

* sends a NOTIFICATION message with the Error Code Hold Timer Expired,

* releases related BGP resources,

* drops the stream/channel connection,

* closes the stream, and

* changes its state to Terminating.

If the KeepaliveTimer_Expires event occurs (Event 11), the local system:

* sends a KEEPALIVE message, and

* restarts its KeepaliveTimer, unless the negotiated HoldTime value is zero.

If the local system receives a NOTIFICATION message (Event 24 or 25) from the control channel with matching StreamID or a QUICStreamClose event (Event 30), the local system:

* deletes all routes associated with this AFI/SAFI,

* releases the related BGP resources for this channel,

* drops the stream/channel connection,

* closes the stream, and

* changes its state to Terminating.

If the local system receives a KEEPALIVE message (Event 26) in the control channel with matching StreamID, the local system:

* restarts its HoldTimer, if the negotiated HoldTime value is non-zero, and

* remains in the Established state.

A function channel is unidirectional, it SHOULD NOT receive any UPDATE message from the control channel with matching StreamID. In case an UPDATE message is received, it SHOULD be ignored.

In response to any other event (Events 9, 13, 21-23, 27-28), the local system:

* sends a NOTIFICATION with a code of Finite State Machine Error,

* releases all BGP resources for the channel,

* drops the stream connection,

* changes its state to Terminating.

Terminating State:
This is the final state of the BoQ Sending FSM. In this state, the FSM has completed its task.

7.3.2.2. Function Channel Receiving FSM

In BoQ, function channels are unidirectional, hence after the control channel reaches established state, as the receiving side a BoQ speaker doesn’t know what function channels will be created by its peer. A BoQ implementation is suggested to have a central process or thread to handle BGP packets received from its peer in function channels that don’t have receiving FSMs yet, we can call this the packet dispatcher.

This dispatcher SHOULD only handle OPEN messages received from its BoQ peer, any other BGP messages SHOULD be ignored. When an OPEN message is received by the dispatcher, and there is no receiving FSM created for the function channel, a function Chanel receiving FSM SHOULD be created to handle packets from the stream. For example, when an OPEN packet for IPv6 unicast is received on stream #2, and no receiving FSM for IPv6 unicast exists, a receiving FSM SHOULD be created for IPv6 unicast, and the StreamID should be set to 2. If a receiving FSM for IPv6 unicast with different StreamID already exists, a NOTIFICATION with BoQ error code, BoQ channel conflict subcode SHOULD be sent in the control channel (see Section 5.4).

Compared with the FSM defined in [RFC4271] Section 8, a function channel receiving FSM starts from the "active" state.

Active State:
In this state, the function channel receiving FSM process the received OPEN message (Event 19), if there is no error in the OPEN message, the local system:

* sends an OPEN message in the control channel as an acknowledgement with the StreamID,

* sends a KEEPALIVE message in the control channel,

* if the HoldTimer initial value is non-zero,

  - starts the KeepaliveTimer with the initial value and

  - resets the HoldTimer to the negotiated value,

* else, if the HoldTimer initial value is zero,

  - resets the KeepaliveTimer and

  - resets the HoldTimer value to zero,

* and changes its state to OpenConfirm.

If BGP message header checking (Event 21) or OPEN message checking detects an error (Event 22), the local system:

* (optionally) If the SendNOTIFICATIONwithoutOPEN attribute is set to TRUE, then the local system first sends a NOTIFICATION message with the appropriate error code, and then

* releases all related BGP resources, and

* changes its state to Terminating.

If he ManualStop event (Event 2), AutomaticStop event (Event 8), a NOTIFICATION message is received with a version error (Event 24), a NOTIFICATION message (Event 25) or a QUICStreamCLose event (Event 30) is received, the local system:

* releases all related BGP resources for the channel, and

* changes its state to Terminating.

In response to any other events (Events 9-11, 13, 23-28), the local system:

* sends a NOTIFICATION with a code of Finite State Machine Error,

* releases all related BGP resources, and

* changes its state to Terminating.

Any other events (Events 1, 3-7, 12, 14-20, 29) received are ignored in the Active state.

OpenConfirm State:
In the state, the receiving FSM wait for a KEEPALIVE or NOTIFICATION message.

Any start event (Events 1, 3-7, 14-18, 20, 29) is ignored in the OpenConfirm state.

In response to a ManualStop event (Event 2) initiated by the operator or the AutomaticStop event initiated by the system (Event 8), the local system:

* sends the NOTIFICATION message with a Cease in the control channel,

* releases all related BGP resources,

* closes the stream, and

* changes its state to Terminating.

If the HoldTimer_Expires event (Event 10) occurs before a KEEPALIVE message is received, the local system:

* sends the NOTIFICATION message with the Error Code Hold Timer Expired,

* releases all related BGP resources,

* closes the stream, and

* changes its state to Terminating.

If the local system receives a KeepaliveTimer_Expires event (Event 11), the local system:

* sends a KEEPALIVE message in the control channel,

* restarts the KeepaliveTimer, and

* remains in the OpenConfirmed state.

If the local system receives a NOTIFICATION message (Event 25), a NOTIFICATION message with a version error (NotifMsgVerErr (Event 24)), or a QUICStreamCLose event (Event 30) is received, the local system:

* releases all related BGP resources,

* closes the stream, and

* changes its state to Terminating.

If a valid OPEN message is received (Event 19) on the same stream, it SHOULD be ignored.

If an OPEN message is received with error, (BGPHeaderErr (Event 21)or BGPOpenMsgErr (Event 22)), the local system:

* sends a NOTIFICATION message with the appropriate error code in the control channel,

* releases all related BGP resources,

* closes the stream, and

* changes its state to Terminating.

If the local system receives a KEEPALIVE message (KeepAliveMsg (Event 26)), the local system:

* restarts the HoldTimer and

* changes its state to Established.

In response to any other event (Events 9, 12-13, 23, 27-28), the local system:

* sends a NOTIFICATION with a code of Finite State Machine Error,

* releases all related BGP resources,

* closes the stream, and

* changes its state to Terminating.

Established State:
In the Established state, a function channel receiving FSM can receive UPDATE, NOTIFICATION, and KEEPALIVE messages from its peer.

Any Start event (Events 1, 3-7, 14-20, 29) is ignored in the Established state.

In response to a ManualStop event (initiated by an operator) (Event 2), or an AutomaticStop event (Event 8), the local system:

* sends the NOTIFICATION message with a Cease in the control channel,

* deletes all routes associated with this connection,

* releases all related BGP resources,

* closes the stream, and

* changes its state to Terminating.

If the HoldTimer_Expires event occurs (Event 10), the local system:

* sends a NOTIFICATION message with the Error Code Hold Timer Expired in the control channel,

* releases all related BGP resources,

* closes the stream, and

* changes its state to Terminating.

If the KeepaliveTimer_Expires event occurs (Event 11), the local system:

* sends a KEEPALIVE message in the control channel, and

* restarts its KeepaliveTimer, unless the negotiated HoldTime value is zero.

If the local system receives a NOTIFICATION message (Event 24 or Event 25), or a QUICStreamCLose event (Event 30) is received, the local system:

* deletes all routes associated with this connection,

* releases all related BGP resources,

* closes the stream, and

* changes its state to Terminating.

If the local system receives a KEEPALIVE message (Event 26), the local system:

* restarts its HoldTimer, if the negotiated HoldTime value is non-zero, and

* remains in the Established state.

If the local system receives an UPDATE message (Event 27), the local system:

* processes the message,

* restarts its HoldTimer, if the negotiated HoldTime value is non-zero, and

* remains in the Established state.

If an UPDATE message is received with error (Event 28), the local system:

* sends a NOTIFICATION message with an Update error,

* deletes all routes associated with this connection,

* releases all related BGP resources,

* closes the stream, and

* changes its state to Terminating.

In response to any other event (Events 9, 12-13, 21-23), the local system:

* sends a NOTIFICATION message with the Error Code Finite State Machine Error,

* deletes all routes associated with this connection,

* releases all related BGP resources,

* closes the stream, and

* changes its state to Terminating.

8. Operational Considerations

8.1. Using Multi-Channel BGP over QUIC

The decision to use BoQ instead of the TCP-based mechanism defined in [RFC4271] is an operational decision and out of the scope of this document. An implementation MUST provide a configuration mechanism to enable BoQ on a per-peer basis.

Connectivity problems (e.g., blocking UDP) can result in a failure to establish a QUIC connection; BGP speakers SHOULD attempt to establish a TCP-based BGP session in this case.

8.2. BGP Multi-Channel Prioritization

One of the drawbacks of a single BGP session is that control plane messages for all supported Network Layer protocols use the same connection, which may cause resource contention.

QUIC [RFC9000] does not provide a mechanism for exchanging prioritization information. Instead, it recommends that implementations provide ways for an application to indicate the relative priority of streams, in this case, mapped to BGP channels. An operator should prioritize BGP channels (streams) that carry critical control plane information if the functionality is available. The definition of this functionality and the determination of the importance of a BGP session are both outside the scope of this document.

9. Security Considerations

This document replaces the transport protocol layer of BGP from TCP to QUIC. It does not modify the basic protocol specifications of BGP, and therefore does not introduce new security risks to the basic BGP protocol. The non-TCP-related considerations of [RFC4271], [RFC4272], and [RFC7454] apply to the specification in this document.

BoQ enhances transport-layer security for BGP sessions, refer to [RFC7454]:

(1) Supports QUIC server identity authentication.

(2) (Optional) Supports QUIC client identity authentication.

(3) Confidentiality protection of BGP messages is supported. All BGP messages are encrypted for transmission.

(4) Supports integrity protection for BGP messages.

The use of a specific UDP port number and an ALPN token protects a BoQ speaker from attempts to establish an unexpected BGP session. Additionally, all packets directed to UDP port TBD on the local device and sourced from an address not known or permitted to become a BGP neighbor SHOULD be discarded.

With BGP multi-channel support using QUIC streams, it separates the control plane traffic over multiple channels. The effect of a session-based vulnerability is reduced; only a single channel is affected and not the whole connection. The result is increased resiliency.

On the other hand, a high number of BGP channels may result in higher resource utilization and the risk of depletion. Also, more channels may imply additional configuration and operational complexity.

10. IANA Considerations

10.1. UDP Port for BoQ

IANA is requested to assign a UDP port (TBD1) from the "Service Name and Transport Protocol Port Number Registry" as follows:

Service Name	boq
Port Number	TBD1
Transport Protocol	udp
Description	BGP over QUIC
Assignee	IETF
Contact	IDR WG
Registration Data	TBD
Reference	this document
Unauthorized Use Reported	idr@ietf.org

Table 5: Port Number Registry

10.2. Registration of the BGP4 Identification String

This document creates a new registration for the identification of BGP [RFC4271] in the "TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs" registry.

The "boq" string identifies BGP-4 [RFC4271] over QUIC:

Protocol: Multi-Channel BGP over QUIC

Identification Sequence: 0x62 0x6f 0x71 ("boq")

Specification: This document

10.3. BGP Over QUIC Capability

IANA is asked to assign a new Capability code [RFC5492] for the BGP over QUIC Capability Section 5.1 as follows:

Value	TBD2
Description	BoQ Capability
Reference	[This Document]
Change Controller	IETF

Table 6: BoQ Capability Registration

10.4. Error Code

This document defines a new NOTIFICATION error code and related subcodes related to the BoQ procedures. IANA is asked to assign a new error code from the "BGP Error (Notification) Codes" registry with the name "BGP over QUIC Message Error", referencing this document.

IANA is asked to create a new registry for the error subcodes as follows:

Under "Border Gateway Protocol (BGP) Parameters",
under "BGP Error Subcodes":
Registry: "BGP over QUIC Message Error subcodes"
Reference: this document
Registration Procedure(s): Values 0-127 Standards Action,
values 128-255 First Come First Served

Value	Name	Reference
0	Reserved	[this document]
1	BoQ Capability Mismatch	[this document]
2	BoQ Connection Reset	[this document]
3	BoQ Channel Reset	[this document]
4	BoQ Channel Conflict	[this document]
5-255	Unassigned	

Table 7: BGP over QUIC Message Error subcodes

11. Acknowledgement

This document references the text and procedures defined in [I-D.ietf-idr-bgp-multisession], and we are grateful for their contributions.

The authors would like to thank xx for review and comments.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

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

[RFC4486] Chen, E. and V. Gillet, "Subcodes for BGP Cease Notification Message", RFC 4486, DOI 10.17487/RFC4486, April 2006, <https://www.rfc-editor.org/info/rfc4486>.

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

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

[RFC6286] Chen, E. and J. Yuan, "Autonomous-System-Wide Unique BGP Identifier for BGP-4", RFC 6286, DOI 10.17487/RFC6286, June 2011, <https://www.rfc-editor.org/info/rfc6286>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <https://www.rfc-editor.org/info/rfc7301>.

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

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

12.2. Informative References

[I-D.ietf-idr-bgp-multisession] Scudder, J., Appanna, C., and I. Varlashkin, "Multisession BGP", Work in Progress, Internet-Draft, draft-ietf-idr-bgp-multisession-07, 13 September 2012, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-bgp-multisession-07>.

[I-D.ietf-idr-rfc7752bis] Talaulikar, K., "Distribution of Link-State and Traffic Engineering Information Using BGP", Work in Progress, Internet-Draft, draft-ietf-idr-rfc7752bis-17, 25 August 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-rfc7752bis-17>.

[RFC2918] Chen, E., "Route Refresh Capability for BGP-4", RFC 2918, DOI 10.17487/RFC2918, September 2000, <https://www.rfc-editor.org/info/rfc2918>.

[RFC4272] Murphy, S., "BGP Security Vulnerabilities Analysis", RFC 4272, DOI 10.17487/RFC4272, January 2006, <https://www.rfc-editor.org/info/rfc4272>.

[RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <https://www.rfc-editor.org/info/rfc4760>.

[RFC7313] Patel, K., Chen, E., and B. Venkatachalapathy, "Enhanced Route Refresh Capability for BGP-4", RFC 7313, DOI 10.17487/RFC7313, July 2014, <https://www.rfc-editor.org/info/rfc7313>.

[RFC7454] Durand, J., Pepelnjak, I., and G. Doering, "BGP Operations and Security", BCP 194, RFC 7454, DOI 10.17487/RFC7454, February 2015, <https://www.rfc-editor.org/info/rfc7454>.

[RFC9234] Azimov, A., Bogomazov, E., Bush, R., Patel, K., and K. Sriram, "Route Leak Prevention and Detection Using Roles in UPDATE and OPEN Messages", RFC 9234, DOI 10.17487/RFC9234, May 2022, <https://www.rfc-editor.org/info/rfc9234>.

Authors' Addresses

Alvaro Retana
Futurewei Technologies
United States of America
Email: aretana@futurewei.com

Yingzhen Qu
Futurewei Technologies
United States of America
Email: yingzhen.ietf@gmail.com

Jeffrey Haas
Juniper Networks
1133 Innovation Way
Sunnyvale, CA 94089
United States of America
Email: jhaas@juniper.net

Shuanglong Chen
Huawei Technologies
No.156 Beiqing Rd.
Beijing
100095
China
Email: chenshuanglong@huawei.com

Jeff Tantsura
Nvidia
United States of America
Email: jefftant.ietf@gmail.com