Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: 7 September 2022

N. Kuhn
E. Stephan
Orange
G. Fairhurst
T. Jones
University of Aberdeen
C. Huitema
Private Octopus Inc.

6 March 2022

BDP Frame Extension
draft-kuhn-quic-bdpframe-extension-00

Abstract

This draft describes the BDP Frame extension for QUIC. It enables the exchange of information related to the path characteristics between the client and the server during a connection. This information can later be exploited when a new connection is established.

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 September 2022.

Copyright Notice

Table of Contents

1. Introduction 2
1.1. Notations and terms 2
1.2. Requirements Language 4
2. BDP Frame 4
2.1. BDP Frame Format 4
2.2. Extension activation 5
3. Discussion 5
4. Acknowledgments 6
5. IANA Considerations 6
6. Security Considerations 6
7. References 6
7.1. Normative References 6
7.2. Informative References 7
Authors' Addresses 7

1. Introduction

This document proposes a method to exchange values between a client and the server in a interoperable manner:

1. For an established connection, the current RTT (current_rtt), bottleneck bandwidth (current_bb) and current client IP (current_client_ip) are stored as saved_rtt, saved_bb and saved_client_ip within a BDP_FRAME;

2. The BDP_FRAME can be sent to the client and the client can also be notified of the values of the BDP_FRAME parameters;

3. When resuming a session to the same IP address, the client is allowed to send the BDP_FRAME;

4. The server can then utilise the parameters from the BDP_FRAME in a later new connection to the same endpoint.

This method applies to any resumed QUIC session: both a saved_session

and a recon_session can be a 0-RTT QUIC connection or a 1-RTT QUIC connection.

1.1. Notations and terms

* BDP: defined below
* CWND: the congestion window used by server (maximum number of bytes allowed in flight by the CC)
* current_bb : Current estimated bottleneck bandwidth
* saved_bb: Estimated bottleneck bandwidth preserved from a previous connection
* RTT: Round-Trip Time
* current_rtt: Current RTT
* saved_rtt: RTT preserved from a previous connection
* client_ip : IP address of the client
* current_client_ip : Current IP address of the client
* saved_client_ip : IP address of the client preserved from a previous connection
* remembered BDP parameters: a combination of saved_rtt and saved_bb

[RFC6349] defines the BDP as follows: "Derived from Round-Trip Time (RTT) and network Bottleneck Bandwidth (BB), the Bandwidth-Delay Product (BDP) determines the Send and Received Socket buffer sizes required to achieve the maximum TCP Throughput." This document considers the BDP estimated by a server that includes all buffering along the network path. The estimated BDP estimated is related to the amount of bytes in flight and the measured path RTT.

A QUIC connection could use the procedure detailed in [RFC6349] to measure the BDP, but is permitted to choose another method [RFC9002] . A server might be able to utilise an other information to provide

an estimate of the BDP.

Congestion controllers, such as CUBIC or RENO, could estimate the saved_bb and current_bb values by utilizing a combination of the cwnd/flight_size and the minimum RTT. A different method could be used to estimate the same values when using a rate-based congestion controller, such as BBR [I-D.cardwell-iccrg-bbr-congestion-control]. It is important to consider whether the methods could result in over-estimating the bottleneck bandwidth, and the preserved values there ought to be used with caution.

1.2. Requirements Language

The keywords "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.

2. BDP Frame

This section describes the use of a new Frame, the BDP Frame. The BDP Frame MUST be considered by the congestion controller and its data is not be limited by flow control limits. The server and the client MAY send multiple BDP Frames in both 1-RTT and 0-RTT connections.

2.1. BDP Frame Format

A BDP Frame is formatted as shown in Figure 1.

```
BDP Frame {
  Type (i) = 0xXXX,
  Lifetime (i),
  Saved BB (i),
  Saved RTT (i),
  Saved IP length (i),
  Saved IP (...)
}
```

Figure 1: BDP Frame Format

A BDP Frame contains the following fields:

* Lifetime (extension_lifetime): The extension_lifetime is a value in milliseconds, encoded as a variable length integer. This follows the design of a NewSessionTicket of TLS [RFC8446]. This represents the validity in time of this extension.

* Saved BB (saved_bb): The saved_bb is a value in bytes, encoded as a variable length integer. The bottleneck bandwidth estimated for the previous connection by the server. Using the previous values of bytes_in_flight defined in [RFC9002] can result in overshoot of the bottleneck capacity and is not advised.

* Saved RTT (saved_rtt): The saved_rtt is a value in milliseconds, encoded as a variable length integer. This could be set to the minimum RTT (min_rtt). The saved_rtt can be set to the min_rtt. NOTE: The min_rtt defined in [RFC9002], does not track a decreasing RTT: therefore the min_rtt reported might be larger than the actual minimum RTT measured during the 1-RTT connection.

* Saved IP length (saved_ip_length) : The length of the IP address in octets is set to either 4 (IPv4) or 16 (IPv6).

* Saved IP (saved_client_ip) : The saved_client_ip could be set to the IP address of the client.

2.2. Extension activation

The client can accept the transmission of BDP Frames from the server by using the enable_bdp transport extension.

enable_bdp (0xTBD): in the 1-RTT connection, the client indicates to the server that it wishes to receive BDP extension Frames for improving ingress of 0-RTT connection. The default value is 0. Values larger than 3 are invalid, and receipt of these values MUST be

treated as a connection error of type TRANSPORT_PARAMETER_ERROR.

* 0: Default value. If the client does not send this parameter, the server considers that the client does not support or does not wish to activate the BDP extension.

* 1: The client indicates to the server that it wishes to receive BDP Frame and activates the ingress optimization for the 0-RTT connection.

* 2: The client indicates that it does not wish to receive BDP Frames but activates ingress optimization.

* 3: The client indicates that it wishes to receive BDP Frames, but does not activate ingress optimization.

This Transport Parameter is encoded as described in Section 18 of [RFC9000].

3. Discussion

With the BDP Frame extension, the client has the choice of accepting the reuse of the previous parameters or not.

The BDP metadata parameters are measured by the server during a previous connection. The BDP extension is protected by the mechanism that protects the exchange of the 0-RTT transport parameters. For version 1 of QUIC, the BDP extension is protected using the mechanism that already protects the "initial_max_data" parameter. This is defined in sections 4.5 to 4.7 of [RFC9001]. This provides a way for the server to verify that the parameters proposed by the client are the same as those that the server sent to the client during the previous connection.

The server SHOULD NOT trust the client. Indeed, even if 0-RTT packets containing the BDP Frame are encrypted, a client could modify the values within the extension and encrypt the 0-RTT packet. Authentication mechanisms might not guarantee that the values are safe. It is not an easy operation for a client to modify

authenticated or encrypted data without this being detected by a server. Modification could be realized by malicious clients. One way to avoid this is for a server to also store the saved_rtt and saved_bb parameters.

4. Acknowledgments

The authors would like to thank Gabriel Montenegro, Patrick McManus, Ian Swett, Igor Lubashev, Robin Marx, Roland Bless and Franklin Simo for their fruitful comments on earlier versions of this document.

5. IANA Considerations

TBD: Text is required to register the BDP Frame and the enable_bdp transport parameter. Parameters are registered using the procedure defined in [RFC9000].

6. Security Considerations

Security considerations are discussed in Section 3.

7. References

7.1. 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>.

[RFC6349] Constantine, B., Forget, G., Geib, R., and R. Schrage, "Framework for TCP Throughput Testing", RFC 6349, DOI 10.17487/RFC6349, August 2011, <https://www.rfc-editor.org/info/rfc6349>.

[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>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[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>.

[RFC9001] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <https://www.rfc-editor.org/info/rfc9001>.

[RFC9002] Iyengar, J., Ed. and I. Swett, Ed., "QUIC Loss Detection and Congestion Control", RFC 9002, DOI 10.17487/RFC9002, May 2021, <https://www.rfc-editor.org/info/rfc9002>.

7.2. Informative References

[I-D.cardwell-iccrg-bbr-congestion-control] Cardwell, N., Cheng, Y., Yeganeh, S. H., Swett, I., and V. Jacobson, "BBR Congestion Control", Work in Progress, Internet-Draft, draft-cardwell-iccrg-bbr-congestion-control-01, 7 November 2021, <https://www.ietf.org/archive/id/draft-cardwell-iccrg-bbr-congestion-control-01.txt>.

Authors' Addresses

Nicolas Kuhn
Email: nicolas.kuhn.ietf@gmail.com

Emile Stephan
Orange
Email: emile.stephan@orange.com

Godred Fairhurst

University of Aberdeen
Department of Engineering
Fraser Noble Building
Aberdeen
Email: gorry@erg.abdn.ac.uk

Tom Jones
University of Aberdeen
Department of Engineering
Fraser Noble Building
Aberdeen
Email: tom@erg.abdn.ac.uk

Christian Huitema
Private Octopus Inc.
Email: huitema@huitema.net