Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: May 6, 2020

N. Kuhn, Ed.
CNES
E. Stephan, Ed.
Orange
G. Fairhurst, Ed.
University of Aberdeen
November 3, 2019

Transport parameters for 0-RTT connections
draft-kuhn-quic-0rtt-bdp-04

Abstract

The time-to-service duration depends on both peers exchange optimization. The peer initiating the connection may not be the one which send data first. Moreover, clients may be resource-limited, behind a low bandwidth or connected to a long-RTT network and may need to adapt dynamically to improve data reception. Currently, each side has its proprietary solution to measure and to store path characteristics. Having a standard way to share these parameters should improve the adaptation to a non standard path characteristics.

QUIC v1 specification already reflects this approach. Having a symmetrical control of the optimization should reduce protocol ossification. This memo proposes the sharing of the characteristics of the path amongst the peer to reduce HTTP3 time-to-service for non default transport situation.

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 May 6, 2020.

Table of Contents

1. Introduction 2
1.1. Reducing ossification with the proposed solution 4
2. Differences between 1-RTT and 0-RTT QUIC connections establishment 5
3. An end-to-end Method 5
3.1. Description of the BDP metadata extension 5
3.2. Usage of the extension in the NewSessionTicket 6
4. Best current practice 6
5. What happens when BDP is used incorrectly? 8
6. Relevance of the solution for QUIC and other protocols . . . 9
7. Acknowledgements 9
8. IANA Considerations 9
9. Security Considerations 9
10. References 10
10.1. Normative References 10
10.2. Informative References 10
Authors' Addresses 12

1. Introduction

Some network paths experience an increased time-to-service because the default parameters controlling the initialization of the transport and congestion control are not well-suited to the path characteristics. QUIC's default congestion control is based on TCP NewReno [I-D.ietf-quic-recovery] and the recommended initial window is defined by [RFC6928]. A path with a large bandwidth delay product can therefore significantly increase the time-to-service (e.g. a path using satellite communication [IJSCN19] could observe a much longer page load time for complex pages). The 0-RTT mechanism is designed to accelerate the throughput when reconnecting to a peer where it has (recently) learned information about the path characteristics.

However, there are cases where egress acceleration like 0-RTT early_data alone does not improve the time-to-service and cases where the data transmission is symmetrical or where clients are capacity-limited: additional information can be beneficial.

As QUIC transport security is based on TLS1.3 [I-D.ietf-quic-tls], this memo describes a solution where a BDP_metadata extension is added to the NewSessionTicket of TLS1.3 [TO DO ADD REF]. The BDP_metadata informs the client about path parameters so that both the client and the server can contribute to the reduction of the time-to-service. This data is protected from in the middle-attack such as the 'early_data' extension.

1. the server learns characteristics of the path during a previous connection;

2. the server sends this information to the client at any time during the current connection, after the BDP_metadata parameters are validated;

3. the client is permitted to discard the information (when the validation period is too short, the information is found to be inconsistent with its own path characteristics measurement, for a device with limited buffer, etc.);

4. the server and the client can exploit the information to improve the time-to-service during subsequent 0-RTT connections.

The current focus of this use is QUIC. However, the method can be used with TLS1.3 over any transport (e.g., using this with TCP Fast Open [RFC7413] or DTLS [RFC6347].

This proposal follows both the approach of the extension field 'early_data' of the NewSessionTicket of TLS1.3 and its mapping in QUIC. While 'early_data' improves the egress traffic of a client, the 'BDP_metadata' provides information that can be used to improve ingress traffic towards the client. This can result in significant improvement to the quality-of-experience. For example, it enables sending measured characteristics of the path, such as the RTT, PMTU and BDP. This information can be used to adapt the initial data transmission of a 0-RTT connection. In the case of a deployment scenario with a large BDP, this can halve the page load time of a web page download [TODO ADD REF].

The proposal proposes to consider the same method for integrating TLS1.3 extension in QUIC as it is done for early_data. For the mapping of NewSessionTicket in QUIC, QUIC transports the 'early_data'

value outside the NewSessionTicket in the "initial_max_data" transport parameters (see section 4.5 of [I-D.ietf-quic-tls]).

1.1. Reducing ossification with the proposed solution

While each client and server could implement a dedicated solution to exchange and store path parameters, providing a standard method to exchange this information helps provide symmetrical control of the optimisation. This reduces protocol ossification. A client using the method is informed about path parameters: allowing both the client and the server to reduce the time-to-service for subsequent connections. This improves symmetrical transmission of data and reduces ossification of the protocol. Some advantages of the proposed solution are the following.

1. It provides symmetrical control of the optimisation: as extensions to HTTP3 envision server initiated request [I-D.ietf-quic-http] the path adaptation ought to be symmetrical and ought not to depend on policy of the peer in establishment. The QUIC transport can be used for services beyond HTTP3, including symmetrical services: where QUIC is considered as a relevant candidate for setting up proxies or tunnels [I-D.kuehlewind-quic-substrate] or for transmiting unreliable datagram services [I-D.pauly-quic-datagram]. A client device sought to be able to adapt to the path chosen by the server. A subscription where the server sends data first, it is important to dissociate the signalling (aka the initiator of the connection) from the peer that first sends application data.

2. Using the path information reduces the need for operators to deploy TCP-proxy and middleboxes, such as Performance-Enhancing Proxy (PEP) [RFC2488][RFC3135] to compensate for the characteristics of the paths: if both the client and server have learned appropriate transport parameters, they can themselves optimize the transport service by adapting the end-to-end transport protocol to the current path. As example, specific client-based adaptations can be developed, such as adapting the ACK-ratio or increasing the receive buffer size. This reduces the need to deploy middelboxes, and will result in less ossifiication along Internet paths.

3. Improve inter-operability: while each client and server can have their dedicated solution to store path parameters, having a standard way of exchanging this information helps in reducing the time-to-service when clients and servers are not provided by the same company. Both sides can independently propose optimizations to improve the ingress traffic.

2. Differences between 1-RTT and 0-RTT QUIC connections establishment

This section recalls how 1-RTT and 0-RTT operate in QUIC [I-D.ietf-quic-transport].

QUIC leverages the two handshakes of TLS1.3 [I-D.ietf-quic-tls]: The 1-RTT handshake initiates a first set of credentials. When this handshake successfully completes, the server pushes the learned information about the session to the client in an opaque session ticket (see section 4.6.1 of [RFC8446]). The information within the opaque ticket is encrypted by the server. When received, the encrypted information is stored by the client (but is not readable by the client). A session ticket can be sent at any time during the connection and a server can send several session tickets in one connection. A client wishing to establish a fast re-open of the session pushes back the (stored) opaque ticket in its 0-RTT handshake and sends early application data.

In practice, the server sends the 'ticket' in a NewSessionTicket record [I-D.ietf-quic-tls]. The structure of the NewSessionTicket includes the opaque 'ticket' and an 'extensions' field. The NewSessionTicket carries an additional field named 'early_data' that indicates to the client the maximum size of application data to insert in the 0-RTT message.

3. An end-to-end Method

QUIC encryption of transport headers prevents the adding of Performance-enhancing proxy (PEP). The BDP metadata extension may be a substitute to PEP proxy [RFC2488], [RFC3135] when time-to-service improvement requires acceleration of the refilling of client application buffers.

The BDP_metadata extension allows a cient to recall the BDP metadata previously measured by the server during the 1-RTT handshake when it initializes a 0-RTT connection. The approach enables changes to a congestion control method (e.g., tuning of the initial window for high BDP networks, as described in [I-D.irtf-iccrg-sallantin-initial-spreading]. This has been shown to improve performance both for paths with a high BDP and a more common BDP [CONEXT15][ICC16].

3.1. Description of the BDP metadata extension

This section defines an extension named "BDP_metadata" for the NewSessionTicket. This structure contains the following parameters: BDP, MTU, RTT, loss-rate.

3.2. Usage of the extension in the NewSessionTicket

At the end of a 1-RTT connection, a server can send information in a NewSessionTicket that describes the path to the client. The message includes an additional 'extensions' field named 'BDP_metadata'. The client stores this session ticket together with and the 'BDP_metadata' field.

When the client reconnects to the same server in 0-RTT mode, it pushes back this session ticket in the ClientHello and prepares itself to receive data in the context given by the 'BDP_metadata' field. The client does not send the 'BDP_metadata' field back to the server. The server receives the session ticket and extracts the BDP context. As example, it can use this message to provide information that may allow the congestion controller to provide a throughput closer to the capacity of the path.

The path characteristics can and do change over time. The path information can therefore become invalid for use in a subsequent connection. The server MUST set the age of the ticket (see section 4.2.11.1 of [RFC8446]) to a short duration. To help ensure that the ticket is still valid, the server SHOULD also verify the IP address of the client. A server MAY update the ticket when the path characteristics of connection are known to have changed.

4. Best current practice

This section provides examples of data that could be added in the opaque session ticket field by the server. The details added by the server in the session ticket do not need to be standardized for interoperability between QUIC clients and servers because this information is opaque to the client. The presence of the "BDP_metadata" extension field in the NewSessionTicket informs the client that the server can actively take action to improve its throughput when the session will restart.

The following list describes information elements set by the server in the session ticket to accompany the signaling of field. These examples are illustrated in Figure 1 and their purpose is detailed in this section.

- A client aware of a high BDP path: Section 7.3.1 of [I-D.ietf-quic-transport] indicates that the "A client that attempts to send 0-RTT data MUST remember the transport parameters used by the server". In addition to the default transport parameters used by the server, a server that knows that the path has a large BDP can let the client adapt its parameters.

- PMTU: Knowledge of the PMTU of a previous path improves the time to service because it reduces the duration of the path validation process described in section 8.2 of [I-D.ietf-quic-transport].

- Connection RTT: The knowledge of the characteristics of a previous connection RTT can improve the throughput because a server can safely improve the slow start: e.g. using the pacing models of [I-D.irtf-iccrg-sallantin-initial-spreading] can utilise a larger IW for high RTT paths and a default IW for paths with smaller RTT. The results presented in [ICC16] show that for both files of 15 KB and 750 KB, the proposed solution reduces the time to download by approximatively 2 seconds whether the RTT is 50ms or 500ms.

- Ticket_lifetime: The server sets a shorter validity duration to avoid receiving obsolete path characteristics; (e.g., this could reduce the validity to one day).

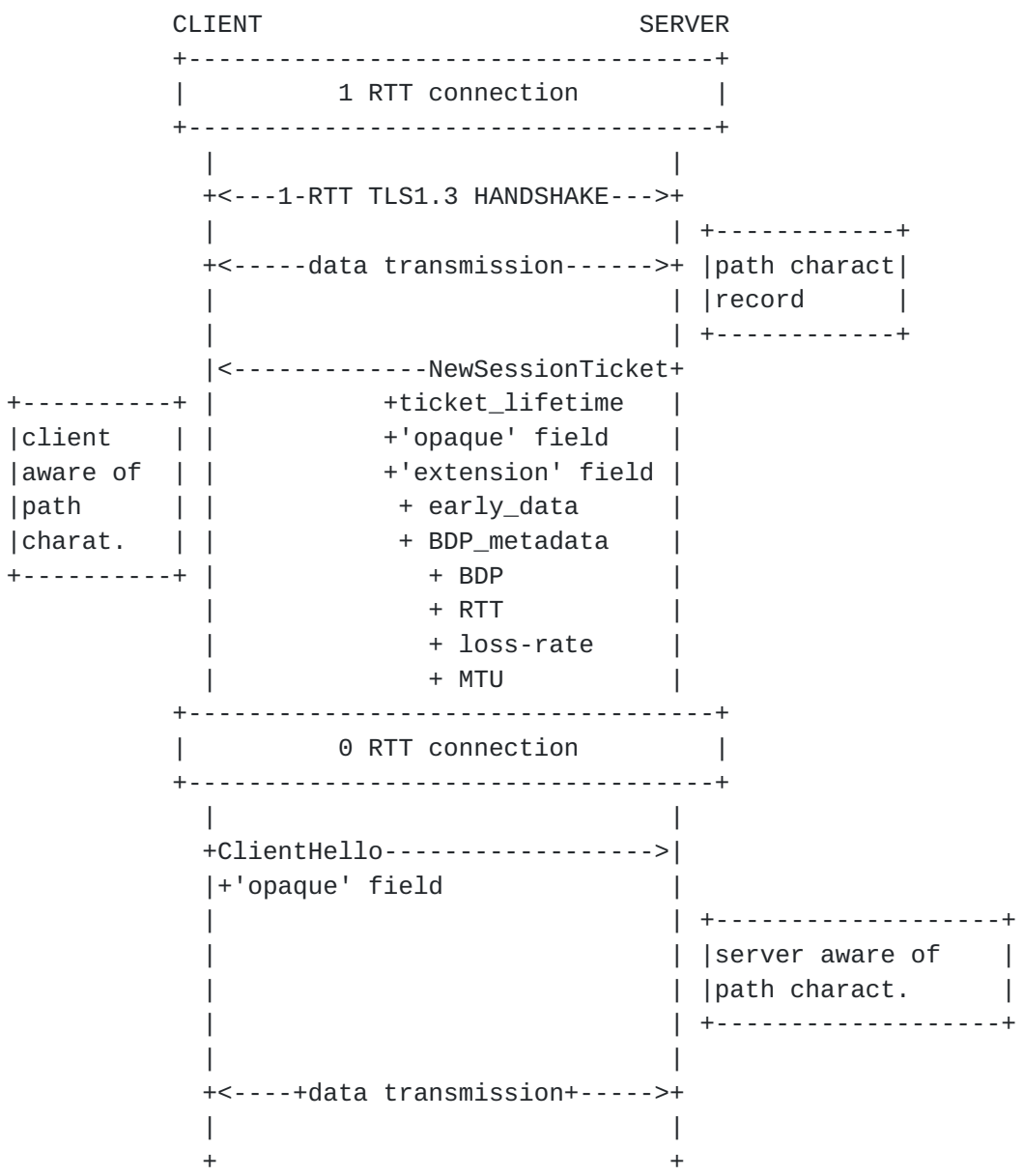

Figure 1: Example of opaque ticket content

5. What happens when BDP is used incorrectly?

This section discusses the impact of a server activating the 'BDP_metadata' field when the network underneath actually has a small BDP. This could happen when the server BDP estimate was incorrect, when a client has multiple paths to choose from and uses the ticket on a different path to which it was requested, or when the path characteristics have changed significantly.

Incorrectly exploiting the BDP_metadata could result in pre-assigning additional resources (e.g. transport buffer space) that later fails

to be used. Many endpoints implementations do not statically pre-assign buffer space, so increasing the limit does not have an impact when the resource is unused. Some cases could be resource-limited.

The server could adapt the initial window because it expects a high BDP path, when the actual BDP is significantly smaller. This issue can be mitigated when packets are paced from the server over the associated RTT, since the server would receive an acknowledgment after the actual RTT period, and before it has used the complete initial window.

6. Relevance of the solution for QUIC and other protocols

The QUIC framework would allow solutions to have been proposed. As an example, the NEW_TOKEN frame could be used to send the path characteristics information to the client. However, this would require specifying its content, consistently with QUIC transport parameters, so that any client can exploit the information transmitted by any server in a standard way. Moreover, the NEW_TOKEN frame is not symmetrical (Clients MUST NOT send NEW_TOKEN frames) does not enable the support of a symmetrical control of the optimisation.

The proposed solution has been proposed with QUIC standardization in mind, but is applicable to any transport under TLS1.3.

7. Acknowledgements

The authors would like to thank Gabriel Montenegro, Patrick McManus, Ian Swett, Igor Lubashev and Christian Huitema for their fruitful comments on earlier versions of this document.

8. IANA Considerations

TBD: text is required to register the extension BDP_metadata field.

9. Security Considerations

The security is provided by the 1-RTT phase. The measure of BDP is made during a previous connection. The exchange and the information are protected both by the TLS encryption and the NewSessionTicket (see section 4.6.1 of [RFC8446]).

The BDP information the server will received is protected in the opaque session ticket. The 'BDP_metadata' field is visible by the client only. An client that does not trust the server transport adaptation ignores any session ticket associated to a 'BDP_metadata' field.

The server does not have to honour all the received requests (e.g. when it is resource-limited).

10. References

10.1. 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>.

10.2. Informative References

[CONEXT15] Li, Q., Dong, M., and P. Godfrey, "Halfback: Running Short Flows Quickly and Safely", ACM CoNEXT , 2015.

[I-D.ietf-quic-http] Bishop, M., "Hypertext Transfer Protocol Version 3 (HTTP/3)", draft-ietf-quic-http-23 (work in progress), September 2019.

[I-D.ietf-quic-recovery] Iyengar, J. and I. Swett, "QUIC Loss Detection and Congestion Control", draft-ietf-quic-recovery-23 (work in progress), September 2019.

[I-D.ietf-quic-tls] Thomson, M. and S. Turner, "Using TLS to Secure QUIC", draft-ietf-quic-tls-23 (work in progress), September 2019.

[I-D.ietf-quic-transport] Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-23 (work in progress), September 2019.

[I-D.ietf-tls-ticketrequests] Pauly, T., Schinazi, D., and C. Wood, "TLS Ticket Requests", draft-ietf-tls-ticketrequests-03 (work in progress), October 2019.

[I-D.irtf-iccrg-sallantin-initial-spreading] Sallantin, R., Baudoin, C., Arnal, F., Dubois, E., Chaput, E., and A. Beylot, "Safe increase of the TCP's Initial Window Using Initial Spreading", draft-irtf-iccrg-sallantin-initial-spreading-00 (work in progress), January 2014.

[I-D.kuehlewind-quic-substrate]
Kuehlewind, M., Sarker, Z., Fossati, T., and L. Pardue, "Use Cases and Requirements for QUIC as a Substrate", draft-kuehlewind-quic-substrate-01 (work in progress), July 2019.

[I-D.pauly-quic-datagram]
Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable Datagram Extension to QUIC", draft-pauly-quic-datagram-04 (work in progress), October 2019.

[ICC16] Sallantin, R., Baudoin, C., Chaput, E., Arnal, F., Dubois, E., and A-L. Beylot, "Reducing web latency through TCP IW: Be smart", IEEE ICC , 2016.

[ICCRG100]
Kuhn, N., "MPTCP and BBR performance over Internet satellite paths", IETF ICCRG 100, 2017.

[IJSCN19] Thomas, L., Dubois, E., Kuhn, N., and E. Lochin, "Google QUIC performance over a public SATCOM access", International Journal of Satellite Communications and Networking , 2019.

[NCT13] Pirovano, A. and F. Garcia, "A new survey on improving TCP performances over geostationary satellite link", Network and Communication Technologies , 2013.

[RFC2488] Allman, M., Glover, D., and L. Sanchez, "Enhancing TCP Over Satellite Channels using Standard Mechanisms", BCP 28, RFC 2488, DOI 10.17487/RFC2488, January 1999, <https://www.rfc-editor.org/info/rfc2488>.

[RFC3135] Border, J., Kojo, M., Griner, J., Montenegro, G., and Z. Shelby, "Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations", RFC 3135, DOI 10.17487/RFC3135, June 2001, <https://www.rfc-editor.org/info/rfc3135>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[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>.

[RFC6928] Chu, J., Dukkipati, N., Cheng, Y., and M. Mathis, "Increasing TCP's Initial Window", RFC 6928, DOI 10.17487/RFC6928, April 2013, <https://www.rfc-editor.org/info/rfc6928>.

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <https://www.rfc-editor.org/info/rfc7413>.

[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>.

Authors' Addresses

Nicolas Kuhn (editor)
CNES

Email: nicolas.kuhn@cnes.fr

Emile Stephan (editor)
Orange

Email: emile.stephan@orange.com

Gorry Fairhurst (editor)
University of Aberdeen

Email: gorry@erg.abdn.ac.uk