Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 2, 2018

L. Pardue
BBC Research & Development

January 29, 2018

Unbound Server Push (USP) for HTTP/QUIC
draft-pardue-quic-http-unbound-server-push-00

Abstract

This document defines an HTTP semantic extension, Unbound Server Push (USP), which allows HTTP resources to be pushed without the need for a prior HTTP request. HTTP/QUIC clients opt in to this feature via an HTTP/QUIC setting.

Status of This Memo

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

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

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

This Internet-Draft will expire on August 2, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. Notational Conventions 3
2. The SETTINGS_ENABLE_UNBOUND_PUSH Parameter 3
3. Usage of Unbound Server Push 3
4. 0-RTT Considerations 4
5. Handling Multiple Clients 4
6. Security Considerations 4
7. IANA Considerations 4
7.1. Registration of SETTINGS_ENABLE_UNBOUND_PUSH parameter . 4
8. Normative References 4
Appendix A. Acknowledgements 5
Author's Address 5

1. Introduction

HTTP server push is a feature of HTTP/2 [RFC7540] and HTTP/QUIC [QUIC-HTTP] that allows a server to pre-emptively send HTTP resources to a client in association with a previous client-initiated request. This binding to a request object aligns with paradigms familiar to client and server implementations. Unbound server push, in contrast, may provide benefits for use cases where holding a request object open for long periods (long polling) is undesirable, or where a request object does not exist (unidirectional flows). (The introduction of unidirectional streams in the QUIC transport [QUIC-TRANSPORT] provides a direct expression of this message exchange pattern.)

This document defines an HTTP/QUIC protocol extension that allows a server to send an HTTP/QUIC "PUSH_PROMISE" frame on a server-initiated unidirectional stream. Endpoints opt in to the unbound server push feature using a "SETTINGS" parameter (Section 2) in accordance with Section 5.5 of [RFC7540]. This is the only behavioural change to server push as described in [QUIC-HTTP]. Unbound server push operates in addition to bound server push for any HTTP/QUIC connection.

Unbound server push should used with care. It may introduce complexities for implementations, particularly intermediaries, and it can pose challenges for presentation to the application above HTTP.

In deployments where multiple client connections are trunked by a reverse proxy onto a single upstream connection, unbound server push is effectively a mechanism for achieving application-level multicast to all downstream clients that have enabled this feature.

Authors' Note: Unbound server push is proposed as an extension to HTTP/QUIC in order to start a discussion on whether this feature should be incorporated into the core HTTP/QUIC specification document.

1.1. Notational Conventions

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.

2. The SETTINGS_ENABLE_UNBOUND_PUSH Parameter

This document adds a new HTTP/QUIC "SETTINGS" Parameter to those defined by [QUIC-HTTP] Section 5.2.5.

The new parameter is "SETTINGS_ENABLE_UNBOUND_PUSH" (type = 0xTBD). This setting can be used to enable unbound server push. The value of the parameter is an integer that MUST be 0 or 1. Any value other than 0 or 1 MUST be treated as a connection error of type "PROTOCOL_ERROR".

The initial value is 0, which indicates that unbound server push is disabled by default.

3. Usage of Unbound Server Push

Unbound server push changes only one aspect of HTTP/QUIC server push: the stream type on which an HTTP/QUIC "PUSH_PROMISE" frame can be sent. It does not prevent the conventional use of bound server push; both types MAY be used concurrently. The Push ID number space is shared across both types. Unbound server push is subject to the limits imposed by the HTTP/QUIC "MAX_PUSH_ID" frame.

An endpoint that receives the "SETTINGS_ENABLE_UNBOUND_PUSH" parameter set to a value of 0 MUST only send an HTTP/QUIC "PUSH_PROMISE" frame on an appropriate client-initiated bidirectional request stream. An endpoint that has set this parameter to 0 and had it acknowledged MUST treat the reception of an HTTP/QUIC "PUSH_PROMISE" frame on any other stream type as a connection error of type "PROTOCOL_ERROR".

A server that receives the "SETTINGS_ENABLE_UNBOUND_PUSH" parameter set to a value of 1 MAY send an HTTP/QUIC "PUSH_PROMISE" frame on a server-initiated unidirectional stream.

A client that has sent the "SETTINGS_ENABLE_UNBOUND_PUSH" parameter set to 1, and received this parameter set to a value of 1, SHOULD be ready for a server to send an HTTP/QUIC "PUSH_PROMISE" frame at any time during the connection.

4. 0-RTT Considerations

Client 0-RTT is not affected by server push configuration. There are no additional consideration to be made beyond those defined in [QUIC-HTTP].

5. Handling Multiple Clients

Unbound server push was discussed during the development of HTTP/2 [RFC7540]. The assessment was that servers that handle multiple clients within the same stack or context (such as an HTTP intermediary) may have a difficult time routing promises to the correct client. The applicability of unbound server push should be assessed and enabled where the risk of misdirected promises is determined to be acceptable.

6. Security Considerations

There are believed to be no additional considerations beyond those presented in [QUIC-HTTP].

7. IANA Considerations

7.1. Registration of SETTINGS_ENABLE_UNBOUND_PUSH parameter

This document establishes an entry for the HTTP/QUIC Settings Registry that is established by [QUIC-HTTP].

Name: "SETTINGS_ENABLE_UNBOUND_PUSH"

Code: 0xTBD

Specification: This document

8. Normative References

[QUIC-HTTP]
Bishop, M., Ed., "Hypertext Transfer Protocol (HTTP) over QUIC", draft-ietf-quic-http-09 (work in progress).

[QUIC-TRANSPORT]
Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-09 (work in progress).

[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>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <https://www.rfc-editor.org/info/rfc7540>.

[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>.

Appendix A. Acknowledgements

The author would like to thank the following for review prior to publication: Richard Bradbury.

Author's Address

Lucas Pardue
BBC Research & Development

Email: lucas.pardue@bbc.co.uk