QUIC
Internet-Draft
Intended status: Standards Track
Expires: October 6, 2019

K. Oku
Fastly

April 04, 2019

Address-bound Token for QUIC
draft-kazuho-quic-address-bound-token-00

Abstract

This document describes a QUIC extension for an address-bound token. This token can be used for sharing address validation and congestion controller state between the same two endpoints across multiple connections and origins.

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 October 6, 2019.

Table of Contents

1. Introduction

Some, if not all of the application protocols that are built on top of QUIC [QUIC-TRANSPORT], including HTTP/3 [QUIC-HTTP], require or would require clients to establish different connections for each server name, even when those server names are hosted by the same server. This restriction introduces several drawbacks:

- Address validation is required for each connection establishment specifying a different server name, thereby restricting the amount of data that a server can initially send.

- Distribution of network bandwidth among these connections is governed by the startup phase and congestion control dynamics, which can lead to unfair distribution for short-lived connections.

- It is hard if not impossible to prioritize the transmission of some connections among others.

To resolve these issues, this document defines a QUIC transport parameter that expands the scope of the token from the server name to a union of the server name and the server's address tuple.

1.1. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Overview

When accepting a new connection, a server sends the "address_bound_token" transport parameter indicating to the client that the tokens it would send can be used for establishing future connections against the server's address tuple regardless of the server's name, and sends the tokens using the NEW_TOKEN frames.

A server can embed the identifier of the congestion controller tied to the connection within the tokens that it sends. Then, when accepting a new connection using the advertised token, the server tries to associate the new connection to the existing congestion controller by using the identifier found in the provided token. Once the server succeeds in making this association, it can skip address validation and the startup phase for the new connection, as well as using the congestion controller for distributing the network bandwidth between the old and the new connection.

Even when it is impossible to share a congestion controller among multiple connections, sharing the tokens between different server names raises the chance of the server receiving a token that has not yet expired. That improves the odds of skipping address validation and reusing the information of the path, such as the estimated round-trip time or the network bandwidth.

3. The address_bound_token Transport Parameter

A server sends the "address_bound_token" transport parameter (0xTBD) to indicate that tokens sent using the NEW_TOKEN frame are "address-bound tokens". That is, they can be used by the client for future connections established to the same server name or to the same server IP address and port.

Only the server sends the "address_bound_token" transport parameter. A client MUST NOT send this transport parameter. A server MUST treat receipt of a "address_bound_token" transport parameter as a connection error of type TRANSPORT_PARAMETER_ERROR.

The "address_bound_token" transport parameter does not carry a value; the length of the value MUST be set to zero. A client that receives this transport parameter not conforming to these requirements MUST terminate the connection with a TRANSPORT_PARAMETER_ERROR.

4. Sharing the Congestion Controller

When multiple QUIC connections share a single congestion controller, how the send window is distributed between the connections is up to the sender's discretion.

However, the use of the "address_bound_token" transport parameter MUST NOT cause any change to when the acknowledgements are sent by a connection endpoint. Similarly, while connection endpoints will forward receipts of acknowledgements and loss signals to the shared congestion controller, loss recovery logic SHOULD operate independently for each connection.

5. Security Considerations

5.1. Reflection Attack

An attacker can create a connection to obtain an address-bound token, warm up the connection, then initiate a new connection by using the token with a spoofed client address or port number. If the server skips address validation and retains the congestion window as-is, the spoofed address might receive a large amount of unsolicited data.

The impact of the attack is equivalent to the spoofed NAT rebinding attack. A server SHOULD NOT skip path validation if the source IP address of an initiating connection is different from the address for which the address-bound token was issued.

5.2. Plaintext Tokens

An address-bound token MUST NOT expose linkability between connections, for example by including the identifier of the congestion controller in cleartext. Exposing a value shared between multiple tokens that could be carried among different connections allows observers to identify the connections belonging to the same client.

6. IANA Considerations

TBD

7. References

7.1. Normative References

[QUIC-TRANSPORT]
Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-16 (work in progress), October 2018.

[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>.

7.2. Informative References

[QUIC-HTTP] Bishop, M., Ed., "Hypertext Transfer Protocol Version 3 (HTTP/3)", draft-ietf-quic-http-16 (work in progress), October 2018.

[RFC7838] Nottingham, M., McManus, P., and J. Reschke, "HTTP Alternative Services", RFC 7838, DOI 10.17487/RFC7838, April 2016, <https://www.rfc-editor.org/info/rfc7838>.

Appendix A. Design Variations

A.1. Using Alt-Svc Name as the Key

An alternative approach to using the server's address tuple as the scope of the token is to use the "host" value of the Alt-Svc [RFC7838] header field as the scope.

In such an approach, a server would send one host value for all the origins it hosts. Then, a client using the value of the host as the scope of the tokens would be able to send a token received on any of the connections that went to the server on any of the future connections that goes to the server.

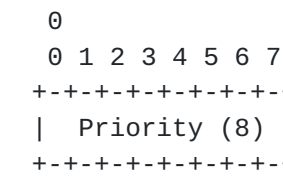
The downside of the approach is that the design works only for HTTP/3 connections being upgraded by the Alt-Svc header field.

A.2. Cross-connection Prioritization

A natural extension to the proposed scheme would be to define a way of prioritizing the connections, so that some connections can be given higher precedence than others. As an example, it would be sensible to prioritize a connection carrying real-time video stream above a connection that is transferring an update image of an operating system.

A simple way of prioritizing between the connections would be to associate a priority value to every connection that would be respected by the sender when it distributes the bandwidth among the connections.

The PRIORITY frame (type=0xTBD) indicates the priority.



The Priority field carries the priority of the connection, subtracted by one.

Each connection is assigned a priority value between 1 and 256. The initial priority is 16.

The PRIORITY frame is sent by an endpoint to encourage the receiver to assign bandwidth proportional to the suggested priority value for each connection.

The priority value carried by the PRIORITY frame is unidirectional. A client advertises its preference on how the data sent by the server should be prioritized; a server advertises its preference on how the data sent by the client should be prioritized.

Appendix B. Acknowledgements

Thanks to Eric Kinnear, Ian Swett, Jana Iyengar, Martin Thomson, Lucas Pardue for their feedback and suggestions.

A proposal exists that advocates for having a transport parameter to change the scope of a token to a list of server names: https://svs.informatik.uni-hamburg.de/publications/2019/2019-03-22-Sy-preprint-Surfing-the-Web-quicker-than-QUIC-via-a-shared-Address-Validation.pdf . The approach described in this document is different from that in the following aspects:

- The scope of the token is the union of the server name and the server's address tuple.

- The token is used also for consolidating the congestion controller.

Author's Address

Kazuho Oku
Fastly

Email: kazuhooku@gmail.com