Network Working Group   R. Paulo
Internet-Draft   Apple Inc.
Intended status: Standards Track   March 11, 2019
Expires: September 12, 2019

Exploring QUIC Connection Migration
draft-paulo-quic-migration-00

Abstract

This document explores QUIC connection migration and suggests possible improvements.

Table of Contents

1. Introduction

This document explores QUIC connection migration in detail and suggests improvements. Familiarity with QUIC as defined in [QUIC-TRANSPORT] is necessary. For reference, QUIC considers PATH_CHALLENGE, PATH_RESPONSE, NEW_CONNECTION_ID and PADDING frames as "probing frames". All other frames are "non-probing frames"

2. Connection migration strategies

Protocols supporting multiple paths (such as MPTCP [RFC6824]) often have three modes of operation that we call failover, standby and aggregation.

QUIC currently supports only the first two.

2.1. Failover mode

In failover mode, the networking stack monitors the quality of the connection or link and, when they are not operating normally, the QUIC connection is migrated to an alternative path.

QUIC doesn't require any packets to be exchanged before migrating to a new path, so this migration is seamless. However, if the new path hasn't been validated, there's less confidence that it will work.

2.2. Standby mode

In standby mode, the networking stack establishes and validates an alternative path shortly after establishing a connection on the main path. When the link or the connection deteriorate, QUIC can switch to the alternative path.

Even though QUIC connections can migrate without validation, this mode improves migration confidence and performance, as explained below.

3. Improvements to connection migration

In contrast with failover mode, applications operating in standby mode must ensure the alternative path is alive for a significant amount of time.

The improvements described below apply only to standby mode because failover mode migrates to an alternative path quickly which doesn't give implementations enough time to determine the characteristics of the new path.

3.1. Dealing with loss and congestion

In the simplest case, when endpoints have no inflight packets, the connection migrates after the path has been validated. In this case, the endpoints reset their congestion controller and don't have to deal with packet loss. However, more commonly, connection migration happens when the main path is quickly losing quality but it's not completely unusable (i.e., not 100% packet loss) and there are plenty of packets inflight. To better deal with packet loss, endpoints SHOULD process packets from both paths for a certain amount of time. Since the ACK frame is a non-probing packet, implementations can only acknowledge packets on the new path, but this acknowledgement SHOULD include packet numbers received on the previous path during connection migration.

ACKs or other non-probing frames might be inflight to the server when the client decides to switch to an alternative path. It's possible that the ACK on the previous path is delayed and arrives at the server after a non-probing frame has been delivered on the new path. This will create a situation where the server is switching back and forth between the two paths. To avoid this problem, servers SHOULD not switch to sending on the previous path if they have received a larger packet number on the new path. Similarly, clients MAY ignore path validation requests on a path they have deemed unreliable.

If an endpoint decides to use a single congestion controller during migration, it will reset it during migration. This is acceptable as the migration will require a new congestion feedback. However, if the migration was spurious (due to an attacker or due to link quality policies), the endpoint should restore the previous congestion controller state.

3.2. Keeping the alternate path alive

Due to shorter NAT timeouts for UDP flows, QUIC connections typically exchange PING frames to maintain the NAT binding alive. However, the PING frame is considered a non-probing frame, so it can't be used to maintain a NAT binding associated with an alternative path alive.

To maintain this NAT binding alive, implementations MAY send periodic PATH_CHALLENGE frames. Since a PATH_CHALLENGE triggers a PATH_RESPONSE, the NAT binding is refreshed on both directions.

3.3. Estimating RTT

Clients can compute the RTT of an alternative path by recording the time the PATH_CHALLENGE frame was sent and measuring the RTT when the corresponding PATH_RESPONSE frame is received.

Servers can't use the client's PATH_CHALLENGE to measure the RTT, but since they will issue their own PATH_CHALLENGE, they can measure the RTT once they receive the respective PATH_RESPONSE.

In standby mode, endpoints can follow the advice on the previous section to gather more RTT samples.

3.4. Migrating without data to send

If an implementation detects the link quality is not acceptable anymore, it can send a non-probing frame, such as a PING or ACK, to migrate the connection to the new path.

This is especially useful for clients that are monitoring the link quality because they can react faster to last mile link problems.

3.5. Path validation privacy

To improve privacy, endpoints MAY randomly pad path validation packets (which include PATH_CHALLENGE and PATH_RESPONSE frames) to make it harder for observers to identify connection migrations. This might be unnecessary when the path validation frames are bundled with other non-probing frames.

3.6. Path MTU discovery

To improve performance, endpoints MAY try to discover the alternative path's MTU before connection migration happens. Endpoints can use the same method described in section 14.3 of [QUIC-TRANSPORT] but instead of using PING frames combined with PADDING frames, they can use PATH_CHALLENGE frames combined with PADDING frames.

3.7. Interaction with loss recovery

Since path probing sends QUIC packets that might not reach the server, implementations SHOULD avoid probing too many paths simultaneously to avoid issues with loss recovery, specifically with the re-ordering threshold. The issue occurs when the number of probing packets that are lost is higher than kPacketThreshold and then the client sends a packet on the main path. Once the ACK arrives, the loss recovery algorithm will assume the probing packets were lost and reduce the congestion window unnecessarily. As a more general improvement, implementations SHOULD ignore lost path probing packets with regards to congestion window reductions.

4. Security Considerations

None.

5. IANA Considerations

None.

6. Acknowledgments

Christoph Paasch and Tommy Pauly.

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-latest (work in progress).

7.2. Informative References

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[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>.

[RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, DOI 10.17487/RFC6824, January 2013, <https://www.rfc-editor.org/info/rfc6824>.

[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>.

Author's Address

Rui Paulo
Apple Inc.
One Apple Park Way
Cupertino, California 95014
United States of America

Email: rpaulo@apple.com