TODO Working Group
Internet-Draft
Intended status: Informational
Expires: 29 January 2022

M.P. Sharabayko
M.A. Sharabayko
Haivision Network Video, GmbH

28 July 2021

Tunnelling SRT over QUIC
draft-sharabayko-srt-over-quic-00

Abstract

This document presents an approach to tunnelling SRT live streams over QUIC datagrams.

QUIC [RFC9000] is a UDP-based transport protocol providing TLS encryption, stream multiplexing, and connection migration. It was designed to become a faster alternative to the TCP protocol [RFC7323].

An Unreliable Datagram Extension to QUIC [QUIC-DATAGRAM] adds support for sending and receiving unreliable datagrams over a QUIC connection, but transfers the responsibility for multiplexing different kinds of datagrams, or flows of datagrams, to an application protocol.

SRT [SRTRFC] is a UDP-based transport protocol. Essentially, it can operate over any unreliable datagram transport. SRT provides loss recovery and stream multiplexing mechanisms. In its live streaming configuration SRT provides an end-to-end latency-aware mechanism for packet loss recovery. If SRT fails to recover a packet loss within a specified latency, then the packet is dropped to avoid blocking playback of further packets.

The Datagram Extension to QUIC could be used as an underlying transport instead of UDP. This way QUIC would provide TLS-level security, connection migration, and potentially multi-path support. It would be easier for existing network facilities to process, route, and load balance the unified QUIC traffic. SRT on its side would provide end-to-end latency tracking and latency-aware loss recovery.

Status of This Memo

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

Copyright Notice

Table of Contents

1. Introduction

1.1. SRT for Low Latency

The Secure Reliable Transport (SRT) protocol [SRTRFC] is a connection-based transport protocol that enables the secure, reliable transport of data across unpredictable networks, such as the Internet. While any data type can be transferred via SRT, it is ideal for low latency (sub-second) video streaming. SRT enables high contribution bitrates over long distance connections.

To achieve low latency streaming, SRT addresses timing issues. The characteristics of a stream from a source network can be notably changed by transmission over the public Internet, introducing delays, jitter, and packet loss. This, in turn, leads to problems with decoding, as audio and video decoders do not receive packets at the expected pace. The use of large buffers helps, but latency is increased. SRT includes a mechanism to keep a constant end-to-end latency, thus recreating the signal characteristics on the receiver side, and reducing the need for buffering.

SRT employs a listener (server) / caller (client) model. The data flow is bi-directional and independent of the connection initiation - either the sender or receiver can operate as listener or caller to initiate a connection.

The SRT protocol provides an internal multiplexing mechanism, allowing multiple SRT connections to share the same UDP port, providing access control functionality to identify the caller on the listener side. This mechanism is exactly what QUIC DATAGRAM describes as a responsibility of the application protocol.

Supporting forward error correction (FEC) and selective packet retransmission (ARQ), SRT provides the flexibility to use either of the two mechanisms or both combined, allowing for use cases ranging from the lowest possible latency to the highest possible reliability.

SRT also allows fast file transfers, and adds support for AES encryption.

1.2. QUIC for Universal Transport

The QUIC transport protocol [RFC9000] is a connection-based transport protocol built on top of UDP. It provides a workflow similar to that of TCP, but for modern fast networks.

TODO: Add more details to the current section. Write about lower connection times, faster delivery, ARQ, CC, etc.

2. Terms and Definitions

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.

SRT: The Secure Reliable Transport protocol.

3. Use Cases for SRT over QUIC

SRT itself is very close to QUIC, and provides similar transport mechanisms. However, the main focus of SRT is on low-latency live contribution and distribution. SRT is widely used by various broadcasters to enable low-latency streaming of live events. It is also used in various mobile and IoT devices to get low-latency feedback and live feeds.

QUIC is supported by CDN companies. Many facilities know how to handle and route QUIC traffic. QUIC provides certain security advantages (TLS, encrypting headers so that traffic is not distinguishable).

SRT tunnelled over QUIC allows managing live delivery mechanisms (preserving end-to-end latency and dropping "too late" data).

4. Tunnelling SRT over QUIC

The QUIC DATAGRAM frame is structured as follows:

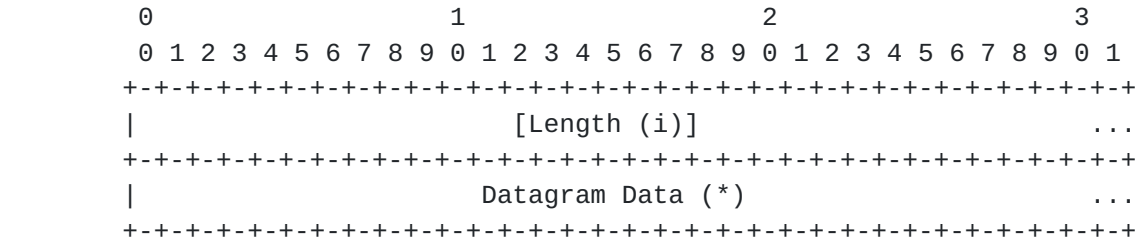

Figure 1: QUIC DATAGRAM Frame Format

Length. A variable-length integer specifying the length of the datagram in bytes.

Datagram Data. The bytes of the datagram to be delivered.

The structure of the SRT packet is shown in Figure 2. For SRT over QUIC tunnelling the full SRT packet is placed inside the Datagram Data field.

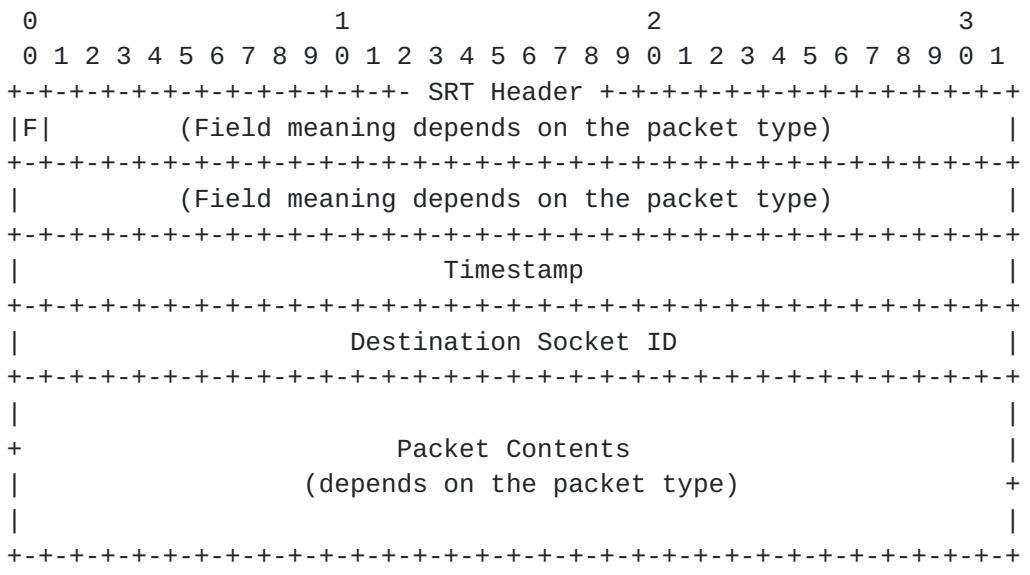

Figure 2: SRT Packet Structure

F: 1 bit. Packet Type Flag. A control packet has this flag set to "1". A data packet has this flag set to "0".

Timestamp: 32 bits. The timestamp of the packet, in microseconds. The value is relative to the time the SRT connection was established. Depending on the transmission mode, the field stores the packet send time or the packet origin time.

Destination Socket ID: 32 bits. A fixed-width field providing the SRT socket ID to which a packet should be dispatched. The field may have the special value "0" when the packet is a connection request.

Packet Contents. Packet Contents as per packet type.

4.1. Overhead

An SRT data packet has a 16-byte header, which adds to the payload of a QUIC packet.

For example, let us consider a payload size of 1128 bytes (six 188-byte MPEG-TS packets). For a 20 Mbps stream, knowing that each data packet gets an additional 16 bytes of overhead, SRT would provide an overhead of only ~280 kbits/s (or 1.4%).

Increasing the size of the payload, e.g., to 1316 bytes (seven 188-byte MPEG-TS packets), SRT overhead at 20 Mbps would be ~240 kbits/s (or 1.2%).

An SRT receiver also sends a full ACK packet every 10 ms. The size of the ACK packet is 44 bytes. This traffic goes in the opposite direction: from the payload receiver to the payload sender. The payload sender responds to every ACK packet with a corresponding 16-byte ACKACK packet. This gives an additional 1600 bytes per second, which may be considered negligible.

4.2. Packet Integrity

SRT does not provide mechanisms to verify the integrity of packets, nor to distinguish a packet from a continuous data stream. SRT assumes that the underlying transport protocol delivers a single and undamaged packet to SRT. Therefore, the underlying transport MUST provide a mechanism for SRT to send and receive exactly one packet.

One SRT packet MUST be sent over exactly one QUIC Datagram frame.

4.3. Connection Establishment

QUIC has a fast and secure crypto handshake based on TLS. A client connects to a server, and it can verify the server based on its certificate. A new client connection takes two times the RTT to be established. If a client connects to a known server, then it can try to establish a faster 0-RTT connection.

Once a QUIC connection is established, QUIC datagrams can be sent in both directions. SRT can use this QUIC datagram tunnelling to establish one or many connections on its own. Each SRT connection would take two times the RTT for handshaking.

The first handshake round of SRT is intended to get an initial response from the server, identifying handshake procedure version and getting a cookie from the server (listener) to mitigate potential DoS attacks. Apart from that, no viable data is exchanged during this first "induction" handshake phase.

If an SRT connection is established over an established and verified QUIC connection, the SRT connection time could be reduced to a single RTT interval if only the "conclusion" handshaking phase is performed. However SRT does not currently provide this functionality, thus appropriate modifications are required.

4.4. Bidirectional Transmission

Both QUIC and SRT allow bidirectional transmission of a payload over a single SRT connection. Even with a payload sent in one direction, some control packets are still sent in the opposite direction over the same connection.

4.5. Congestion Control

QUIC as a transport mechanism can apply congestion control. It is worth noting, however, the specifics of live streaming compared to file-based transmissions. The payload is not available right away, therefore using regular bandwidth probing mechanisms to increase or decrease the sending rate would not work.

The current sending rate is provided by SRT, which in turn receives the payload from a live source and can add some overhead when retransmission of lost packets is performed.

However, QUIC can use congestion control to detect congestion and throughput bottlenecks, and prevent sending above a certain limit. SRT MUST handle such cases by eventually dropping packets, which can no longer be recovered.

It would make sense for a QUIC connection to provide this throughput limitation value back to SRT. In that case SRT could use the number to make clever and transport-aware retransmission decisions.

It is also possible for QUIC to not apply any congestion control, relying on SRT. However, SRT does not reduce the sending rate below the input rate. If the bitrate of the original stream already exceeds network throughput, SRT would still try to deliver it, maintaining congestion and eventually breaking the SRT connection.

It is worth mentioning that in a live streaming scenario it may be beneficial to move congestion control mechanisms outside of the protocol towards the encoder (payload producer), implementing a network adaptive encoding based on the telemetry provided by the SRT and QUIC protocols.

4.6. Pacing

SRT uses ACK/ACKACK packet pairs to measure RTT on a link, and to track latency and clock drift. It also uses packet pair probing to estimate connection bandwidth, although in live configurations such estimates are informative only.

Buffering and pacing of SRT packets by QUIC SHOULD be done with the understanding that this would interfere with the corresponding SRT mechanisms. Alternatively, SRT may implement a pacer on top of QUIC's congestion control and probing mechanisms to abstract the complexity associated with live streaming use cases.

4.7. Connection Mitigation

QUIC utilizes Connection UUID to distinguish between connections (compared to the IP:Port scheme used by UDP and TCP). This enables already established connections to be handed over seamlessly across network interfaces without requiring a new handshake/negotiation. SRT may expand on this to enable network bonded delivery workflows to switch between optimal transports without a latency hit.

4.8. Datagram vs H3 Datagram

As an alternative to using the QUIC Datagram extension, it might be possible to consider the H3 Datagram version in order to be compatible with more existing load balancers.

5. Security Considerations

TODO

6. IANA Considerations

TODO

Acknowledgments

It is worth acknowledging the participation of the following people in the project discussions: Ying Yin (Google), Ian Swett (Google), Victor Vasiliev (Google), Kazuko Oku (Fastly), Marc Cymontkowski (Haivision), Nikos Kyriopoulos (Haivision), Jake Weissman (Facebook), Jordi Cenzano (Facebook), Alan Frindell (Facebook), Jeongseok Kim (SK Telecom), Joonwoong Kim (SK Telecom).

Quicly library [QUICLY] from Fastly was chosen to provide a QUIC datagram transport layer for SRT over QUIC PoC. We would like to thank Kazuho Oku (Fastly) for his help.

References

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>.

[RFC7323] Borman, D., Braden, B., Jacobson, V., and R. Scheffenegger, Ed., "TCP Extensions for High Performance", RFC 7323, DOI 10.17487/RFC7323, September 2014, <https://www.rfc-editor.org/info/rfc7323>.

[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>.

[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>.

Informative References

[QUIC-DATAGRAM] Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable Datagram Extension to QUIC", n.d., <https://datatracker.ietf.org/doc/draft-ietf-quic-datagram/>.

[QUICLY] "QUIC protocol implementation quicly for H2O server", July 2021, <https://github.com/h2o/quicly>.

[SRTRFC] Sharabayko, M., Sharabayko, M., Dube, J., Kim, J., and J. Kim, "The SRT Protocol", December 2019, <https://datatracker.ietf.org/doc/draft-sharabayko-srt/>.

Authors' Addresses

Maxim Sharabayko
Haivision Network Video, GmbH

Email: maxsharabayko@haivision.com

Maria Sharabayko
Haivision Network Video, GmbH

Email: msharabayko@haivision.com