Workgroup: moq
Internet-Draft:
draft-hang-moq-design-space-analysis-of-moq-00
Published: 21 October 2022
Intended Status: Informational
Expires: 24 April 2023

Authors: H. Shi, Ed. Y. Cui
Huawei Technologies Tsinghua University

Design Space Analysis of MoQ

Abstract

This document investigates potential solution directions within the charter scope of MoQ WG. MoQ aims to provide low-latency, efficient media delivery solution for use cases including live streaming, gaming and video conferencing. To achieve low-latency media transfer efficiently, the network topology of relay nodes and the computation done at the relay nodes should be considered carefully. This document provides the analysis of those factors which can help the design of the MoQ protocols.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Media over QUIC aims to provide low-latency, efficient media delivery solution for use cases including live streaming, gaming, and video conferencing. The latency requirement and the transmission pattern are analyzed in [moq-req]. To scale efficiently, relay can be used to optimize the delivery performance by caching, selective dropping, etc. However, how to accomplish that remains unclear. Lots of factors of the relay and protocol design choice can affect the performance gain of leveraging relay. This document aims to provide analysis of those design choices.

2. Terminology

*Relay: An element which participates in the forwarding of the media content. Possibly support caching, selective dropping to optimize the media transmission performance.

*Producer: An endpoint which generate the media stream. Could be the original content producer (a live streaming uploader) or the re-encoder in the cloud.

*Consumer: An endpoint which receive the media stream. Could be the live stream viewer or the re-encoder in the cloud.

2.1. Requirements Language

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.

3. Design Choice 1: Static Tree Topology versus Dynamic Mesh Topology

The first question of using relay to forward the media between the producer and the consumer is the topology of relays. In traditional CDN network, each CDN site can be viewed as a relay. Those relays are organized in a tree (see Figure 1). The producer and the consumer are usually connected to the edge node of the CDN which is the leaf node in the tree. In this case, the path for media in live streaming is usually producer - edge node 1 (relay 1) - parent node 1 (relay 2) - origin node (relay 3) - parent node 2 (relay 4) - edge node 2 (relay 5) - consumer, i.e. the media need to first go up to the root of the tree, then go down to another leaf node, traversing multiple (at least 3) relays if the CDN hierarchy is deep or the producer and the consumer is highly distributed. The tree topology is simple to build since the path of the stream is fixed and the leaf node can be lightweight and deployed closely to user. The computing intensive process can be put in the much more powerful root servers.

[QUICR-arch] is similar to the tree topology of CDN with one improvement: the relay can shortcut the media transmission. If the producer and the consumer share a parent relay, the media will be forwarded in the relay instead of the root of the tree (called Origin in QUICR's term).

```
                               +----------+
                  +----------->|   Root   +-----------+
                  |            +----------+           |
                  |                                   |
             +----+-----+                        +----v-----+
      +----->| Parent-1 |                        | Parent-2 +--------+
      |      +----------+                        +----------+        |
      |                                                              |
 +----+-----+                                                   +----v--
 |  Edge-1  |                                                   |  Edge-
 +----^-----+                                                   +-------
      |                                                              |
      |                                                              |
+-----+----+                                                     +---v--
| Producer |                                                     | Consu
+----------+                                                     +------
```

Figure 1: static tree topology

Another approach is to connect the relays in a dynamic mesh instead of a static hierarchy. Alibaba's low-latency live streaming network builds on a flat CDN overlay [LiveNet]. A centralized controller collects the latency between each relay periodically and calculates the optimal path (latency-wise) for each media stream dynamically. Alibaba claims the flat topology reduce the latency by half compared to static hierarchy. An example is shown in Figure 2, the media stream is forwarded through relay 1 and relay 4, only 2 hops. If the network path between relay 1 and relay 4 are congested, relay 1 - relay 2/3 - relay 4 maybe used to provide lower-latency forwarding.

```
              +-----------------------------------------+
              |              Controller                 |
              +-----------------------------------------+
              |                                         |
              |              +---------+                |
              |      +-------> Relay-2 +---------+      |
              |      |       +----+----+ path 2  |      |
              |      |            |              |      |
+----------+  | +----+----+       |         +----v----+ |   +----------+
| Producer +--+>| Relay-1 +-------+---------> Relay-4 +-+-->| Consumer |
+----------+  | +----+----+       | path 1  +----------+ |   +----------+
              |      |            |              |      |
              |      |            |              |      |
              |      |       +----+----+         |      |
              |      +-------+ Relay-3 +---------+      |
              |              +---------+                |
              |                                         |
              +-----------------------------------------+
```

Figure 2: dynamic mesh topology

4. Design Choice 2: Stateless HTTP versus Stateful pub/sub

Traditionally the CDN are using HTTP to support live streaming. The media stream is broken into a series of chunks which are mapped to HTTP resources. In this way, the HTTP stack and infrastructure is reused. Since HTTP is stateless, each relay only need to act as an HTTP server/client. There is no need to store the relationship between different HTTP flows hence the relay is easier to implement. However, such a stateless HTTP server will suffer from the delay of the chunk because each relay need to download the chunk first before it can serve the chunk to the downstream node as an HTTP server. The delay will be stacked along with the relay chain. Reducing the chunk size can reduce the delay, but the number of chunk will increase thus brings higher burden of management and signalling.

Using pub/sub as the metaphor requires that the relay node keeps track of the mapping between the publisher and subscribers, i.e. the subscription information state. A packet receive from the publisher can be duplicated and forwarded to subscribers immediately without any delay, forming a relay chain for packets instead of chunks. HTTP can be modified to send partial chunks before a full chunk is received, then the downstream HTTP stream will bind with the upstream one, essentially brings back the subscription information state.

Another way to eliminate the state in the relay node is to encode the state information on the data-plane. When the producer sends out the media, it tags the subscriber information in each packet or flow. The relay forwards the packet or flow based on the tag. The relay node need to be preloaded the tag forwarding rule. Luckily this tag forwarding rule is related to the topology which is rather static comparing to the highly dynamic media stream subscriber information.

5. Design Choice 3: QUIC hop-by-hop versus end-to-end

The media flow sending from the producer to the consumer will go through several relays. The media content will be encrypted using QUIC encryption as requested in charter. But whether the relay node will terminate the QUIC connection remains open. There are following two options to implement the MoQ protocol stack.

The first option is to running the entire MoQ protocol inside QUIC encryption, including the media metadata which is needed by relay (see Figure 3). Thus the relay has to terminate the QUIC connection, decrypting the QUIC payload. This will require each relay node hold a valid CA certificate and run the CA verification process. Just like what the CDN node does nowadays.

```
        Media (Metadata + Content)
   ---------------------------------------------
       Protocol header  |  Protocol payload          <-------- MoQ
   ---------------------------------------------
                      QUIC                           <-------- Transport
```

Figure 3: MoQ running over QUIC, like HTTP

The second option is to only encrypt the media content using QUIC encryption but leave the metadata to other mechanism (see Figure 4). In this way, the QUIC connection is from producer to consumer. The relay does not need to decrypt the QUIC, saving the computing power. As the charter put it: "Even when media content is end-to-end encrypted, the relays can access metadata. Hence a new mechanism to convey the metadata to the relay is needed, similar to SDP for RTP, or m3u8 file for HLS.

```
     Media metadata      |  Media content            <-----\
-------------------------|-----------------------           \
    Protocol header      |  Protocol payload         <------- MoQ
-------------------------|-----------------------           /
        Other            |    QUIC                   <-----/
```

Figure 4: MoQ using QUIC for media, other for metadata, like WebRTC

6. Security Considerations

When the metadata is not carried inside the QUIC payload, it should be protected from unauthorized third-part access to to protect the privacy. Relay should be authenticated to access the metadata.

7. References

7.1. 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>.

[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>.

7.2. Informative References

[LiveNet] "LiveNet - A Low-Latency Video Transport Network", October 2022, <https://dl.acm.org/doi/abs/10.1145/3544216.3544236>.

[moq-req] Gruessing, J. and S. Dawkins, "draft-gruessing-moq-requirements-02", October 2022, <https://datatracker.ietf.org/doc/draft-gruessing-moq-requirements/>.

[QUICR-arch] Jennings, C. and S. Nandakumar, "QuicR - Media Delivery Protocol over QUIC", October 2022, <https://datatracker.ietf.org/doc/draft-jennings-moq-quicr-arch/>.

Authors' Addresses

Hang Shi (editor)
Huawei Technologies

China

Email: shihang9@huawei.com

Yong Cui
Tsinghua University
China

Email: cuiyong@tsinghua.edu.cn