Independent Submission
Internet-Draft
Intended status: Informational
Expires: 14 September 2023

S. Nandakumar
Cisco
C. Huitema
Private Octopus Inc.
C. Jennings
Cisco

13 March 2023

Exploration of MoQ scenarios and Data Model
draft-nandakumar-moq-scenarios-00

Abstract

This document delineates a set of key scenarios and details the requirements that they place on the MoQ data model.

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 https://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 14 September 2023.

Table of Contents

1. Introduction

When developing the data model for MoQ, we realized that different WG participants were making different assumptions about the role of streams, broadcast or emitters, and also on the delivery constraints for objects compositing different streams. This draft studies different scenarios and details their requirements.

2. Scenarios

One ambition of MoQ is to define a single QUIC based transport for multiple transmission scenarios, including streaming scenarios currently using RTMP and conferencing scenarios currently using WebRTC. Ideally, this would enable support in Content Distribution Networks for both types of scenarios.

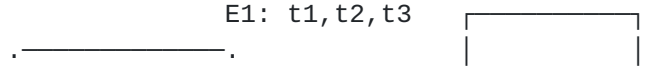
2.1. Streaming Scenarios

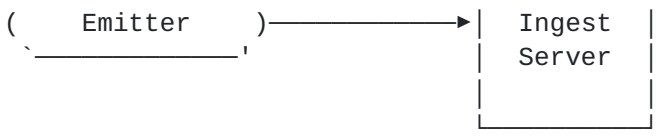
This section dicusses few scenarios for streaming use-cases. The scenarios listed are not exhaustive and doesn't intend to capture all possible applications and architectures.

Streaming scenarios typically separate "content ingestion" and "content distribution". Content is provided by one or several "emitters". Streaming scenarios typically operate with latency profile between 500 ms - 2s for live streaming use-cases.

2.2. Live Video Ingestion

In a typical live video ingestion, the broadcast client - like OBS client, publishes the video content to an ingest server under a provider domain (say twicth.com)



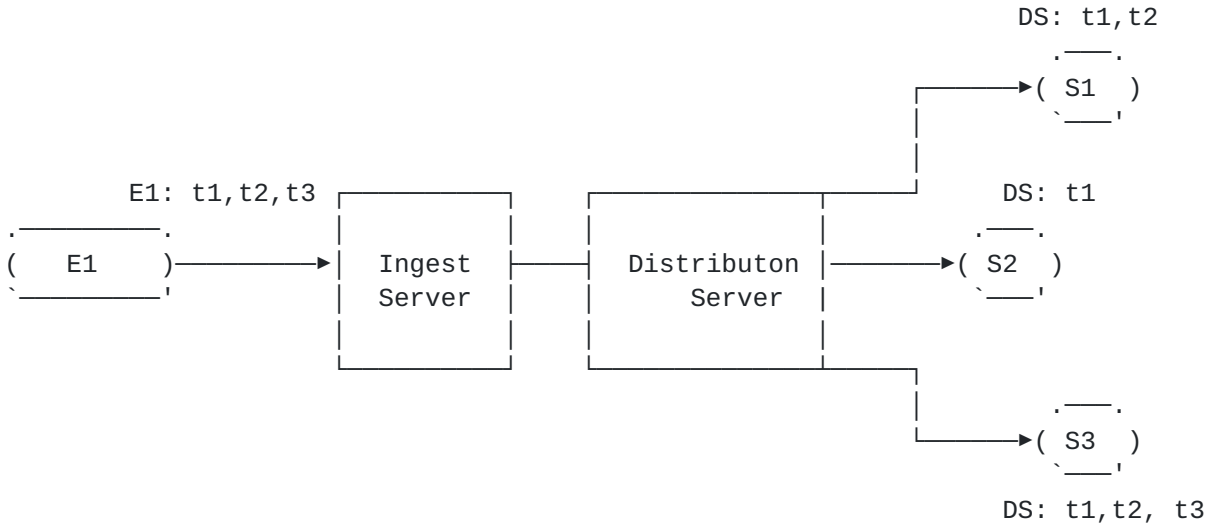


The Track IDs are scoped to the broadcast for the application under a provider domain.

2.3. Live Streaming

In a reference live streaming example shown below, the emitter live streams on or more tracks as part of the application operated under a provider domain, which gets eventually distributed to multiple clients by some form of distribution server operating under the provider domain, over a content distribution network.

In this setup, one can imagine the ingestion and distribution as 2 separate systems operating under a given provider domain, where the track Ids used by the emitter need not match the ones referred to by the subscribers. The reason being, the distribution server sources the new tracks (possibly transcoded)



2.4. Interactive Usecases

A interactive conference typically works with the expected operating glass-to-glass latency to be around 200ms and is made up of multiplicity of participant with varying capabilities and operating under varying network conditions.

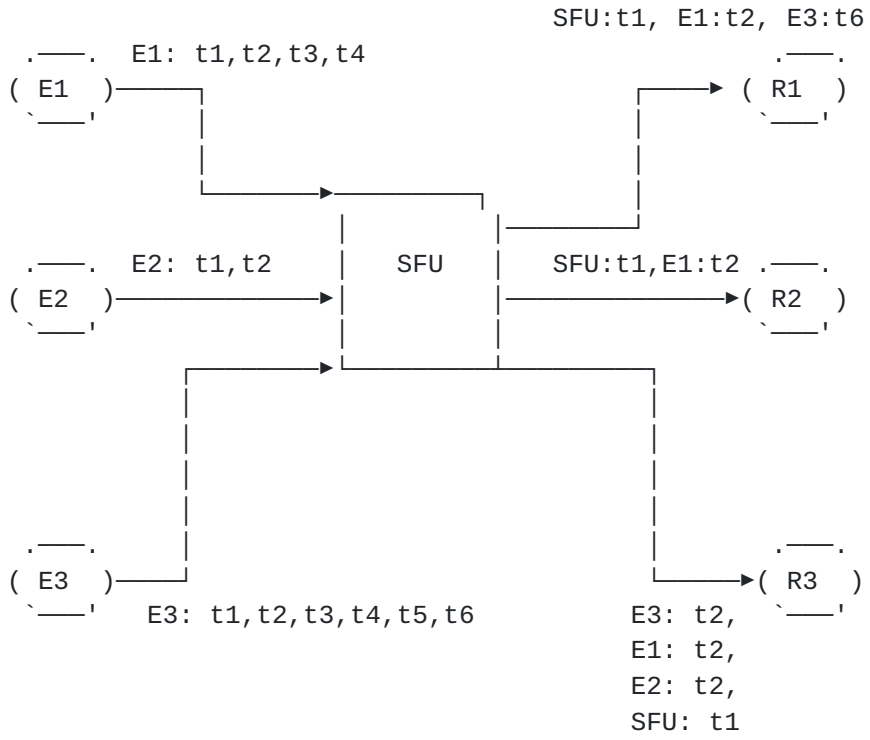
A typical conferencing session comprises of:

* Multiple emitters, publishing on multiple tracks (audio, video tracks and at different qualities)

* A media switch, sourcing tracks that represent a subset of tracks from across all the emitters. Such subset may represent tracks

representing top 5 speakers at higher qualities and lot of other tracks for rest of the emitters at lower qualities.

* Multiple receivers, with varied receiving capacity (bandwidth limited), subscribing to subset of the tracks



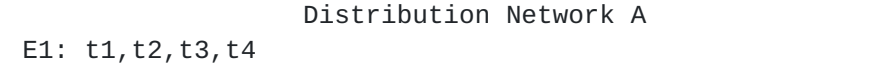
Above setup brings in following properties on the data model for the transport protocol

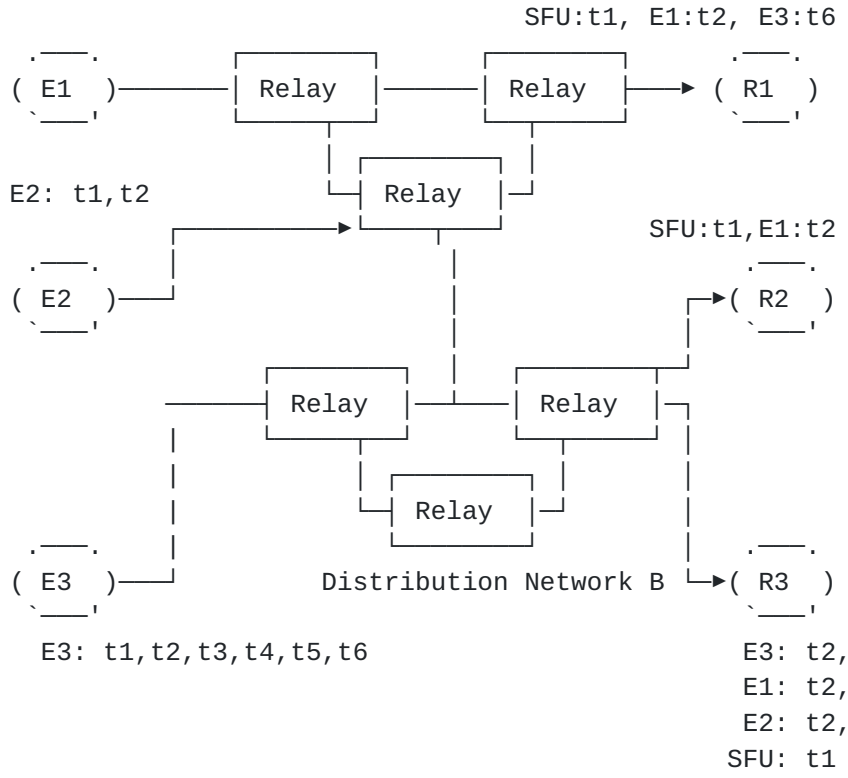
* Media Switches to source new tracks but retain media payload from the original emitters. This implies publishing new Track IDs sourced from the SFU, with object payload unchanged from the original emitters.

* Media Switches to propogate subset of tracks as-is from the emitters to the subscribers. This implies Track IDs to be unchanged between the emitters and the receivers.

* Subscribers to explictily request multiple appropriate qualities and dynamically move between the qualtiies during the course of the session

Another topology for the interactive use-case is to use multiple distribution networks for delivering the media, with thus media switching functionality running across disrtibution networks and also moving these media functions to the core distribution network as shown below





Such a topology needs to meet all the properties listed in the homogenous topology setup, however having multiple distribution networks and relying on the distribution networks to carryout the media delivery, brings in further requirements towards a data model that enables tracks to be uniquely identifiable across the distribution networks and not just within a single distribution network.

3. Scenario differences

We find that scenarios differs in multiple ways. In the previous sections we detail the obvious differences, such as different network topologies or different latency targets, but other factors also come in play.

3.1. Interval between access points

In the streaming scenarios, there is an important emphasis on resynchronization, characterized by a short distance between "access points". This can be used for features like fast-forward or rewinding, which are common in non-real-time streaming. For real-time streaming experiences such as watching a sport event, frequent access points allow "channel surfers" to quickly join the broadcast and enjoy the experience. The interval between these access points will often be just a few seconds.

In video encoding, each access point is mapped to a fully encoded frame that can be used as reference for the "group of blocks". The

encoding of these reference frames is typically much larger than the differential encoding of the following frames. This creates a peak of traffic at the beginning of the group. This peak is much easier to absorb in streaming applications that tolerate higher latencies than interactive video conferences. In practice, many real time conferences tend to use much longer groups, resulting in higher compression ratios and smoother bandwidth consumption along with a way to request the start of a new group when needed. Other real time conferences tend to use very short groups and just wait for the next group when needed.

Of course, having longer blocks create other issues. Realtime conferences also need to accomodate the occasional occasional late comer, or the disconnected user who want to resynchronize after a network event. This drives a need for synchronization "between access points". For example, rather than waiting for 30 seconds before connecting, the user might quickly download the "key" frames of the past 30 seconds and replay them in order to "synchronize" the video decoder.

3.2. Intervals and congestion

It is possible to use groups as units of congestion control. When the sending strategy is understoud, the objects in the group can be assigned sequence numbers and drop priorities that capture the encoding dependencies, such that:

* an object can only have dependencies with other objects in the same group,

* an object can only have dependencies with other objects with lower sequence numbers,

* an object can only have dependencies with other objects with lower or equal drop priorities.

This simple rules enable real-time congestion control decisions at relays and other nodes. The main drawback is that if a packet with a given drop priority is actually dropped, all objects with higher sequence numbers and higher or equal drop priorities in the same group must be dropped. If the group duration is long, this means that the quality of experience may be lowered for a long time after a brief congestion. If the group duration is short, this can produce a jarring effect in which the quality of experience drops perdiodically at the tail of the group.

4. Handling Scalable Video Codecs

Some video codecs have a complex structure. Consider an application using both temporal layering and spatial layering. It would send for example:

* an object representing the 30 fps frame at 720p

* an object representing the spatial enhancement of that frame to 1080p

* an object representing the 60 fps frame at 720p

* an object representing the spatial enhancement of that 60 fps frame to 1080p

The encoding of the 30 fps frame depends on the previous 30 fps frames, but not on any 60 fps frame. The encoding of the 60 fps depends on the previous 30 fps frames, and possibly also on the previous 60 fps frames (there are options). The encoding of the spatial enhancement depends on the corresponding 720p frames, and also on the previous 1080p enhancements. Add a couple of layers, and the expression of dependencies can be very complex. The AV1 documentation for example provides schematics of a video stream with 3 frame rate options at 15, 30 and 60 fps, and two definition options, with a complex graph of dependencies. Other video encodings have similar provisions. They may differ in details, but there are constants: if some object is dropped, then all objects that have a dependency on it are useless.

Of course, we could encode these dependencies as properties of the object being sent, stating for example that "object 17 can only be decoded if objects 16, 11 and 7 are available." However, this approach leads to a lot of complexity in relays. We believe that a linear approach is preferable, using attributes of objects like delivery order or priorities.

4.1. Application choice for ordering

The conversion from dependency graph to linear ordering is not unique. The simple graph in our example could be ordered either "frame rate first" versus "definition first". If the application chooses frame rate first, the policy is expressed as "in case of congestion, drop the spatial enhancement objects first, and if that is not enough drop the 60 fps frames". If the application chooses "definition first", the policy becomes "drop the 60 fps frames and their corresponding 1080p enhancement first, and if that is not enough also drop the 1080p enhancement of the 30 fps frames".

More complex graphs will allow for more complex policies, maybe for example "15 fps at 720p as a minimum, but try to ensure at least 30fps, then try to ensure 1080p, and if there is bandwidth available forward 60 fps at 1080p". Such linearization requires choices, and the choices should be made by the application, based on the user experience requirements of the application.

The relays will not understand all the variation of what the media is

but the applications will need a way to indicate to the relays the information they will need to correctly order which data is sent first.

4.2. Linear ordering using priorities

We propose to express dependencies using a combination of object number and object priority.

Let's consider our example of an encoding providing both spatial enhancement and frame rate enhancement options, and suppose that the application has expressed a preference for frame rate. We can express that policy as follow:

* the frames are ordered first by time and when the time is the same by resolution. This determines the "object number" property.

* the frame priority will be set to 1 for the 720p 30 fps frame, 2 for the 720p 60 fps frames, and 3 for all the enhancement frames.

If the application did instead express a preference for definition, object numbers will be assigned in the same way, but the priorities will be different:

* the frame priority will be set to 1 for the 720p 30 fps I frames and 2 for the 720p 30 fps P and B frames, 3 and 4 for the 1080p enhancements of the 60 fps frames, and 5 and 6 for the 60 fps frames and their enhancements.

Object numbers and priorities will be set by the publisher of the track, and will not be modified by the relays.

4.3. Relay behavior

In case of congestion, the relay will use the priorities to selectively drop the "least important" objects:

* if congestion is noticed, the relay will drop first the lesser priority layer. In our example, that would mean the objects marked at priority 6. The relay will drop all objects marked at that priority, from the first dropped object to the end of the group.

* if congestion persists despite dropping a first layer, the relay will start dropping the next layer, in our example the objects marked at priority 5.

* if congestion still persist after dropping all but the highest priority layer, the relay will have to close the group, and start relaying the next group.

When dropping objects within the same priority:

* higher object numbers in the same group, which are later in the group, are "less important" and more likely to be dropped than objects in the same group with a lower object number. Objects in a previous group are "less important" than objects in the current group and MAY be dropped ahead of objects in the current group.

The specification above assumes that the relay can detect the onset of congestion, and has a way to drop objects. There are several ways to achieve that result, such as sending all objects of a group in a single QUIC stream and making explicit action at the time of relaying, or mapping separate priority layers into different QUIC streams and marking these streams with different priorities. The exact solution will have to be defined in a draft that specifies transport priorities.

5. High Loss Networks

Web conferencing systems are used on networks with well over 20% packet loss and when this happens, it is often on connections with a relatively large round trip times. In these situtation, forward error correction or redundant transmitions are used to provide a reasonable user experience. Often video is turned off in. There are multiple machine learning based audio codecs in development that targeting a 2 to 3 Kbps rate.

This can result in scenarios where very small audio objects are sent at a rate of several hundreds packets per second with a high network loss rate.

6. Security and Privacy Considerations

This document provides an abstract analysis of MoQ scenarios, but does not detail any security considerations.

7. IANA Considerations

This document makes no request of IANA.

8. Acknowledgments

The IETF MoQ mailing lists and discussion groups.

Authors' Addresses

Suhas Nandakumar
Cisco
Email: snandaku@cisco.com

Christian Huitema
Private Octopus Inc.

Email: huitema@huitema.net

Cullen Jennings
Cisco
Email: fluffy@iii.ca