Media Over QUIC
Internet-Draft
Intended status: Informational
Expires: 7 November 2026

P. Gregoire
Red5
G. Simon
Synamedia

6 May 2026

MPEG-2 Transport Stream Packaging for Media Over QUIC Transport
draft-gregoire-moq-msfts-00

Abstract

This document extends the MOQT Streaming Format (MSF) by registering the "m2ts" packaging value for carrying MPEG-2 Transport Stream and M2TS source packets over Media Over QUIC Transport. It defines catalog fields for transport-stream track description and specifies receiver and relay behavior for joining, switching, and validating packetized streams.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://mondain.github.io/msfts/draft-gregoire-moq-msfts.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-gregoire-moq-msfts/.

Discussion of this document takes place on the Media Over QUIC Working Group mailing list (mailto:moq@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/moq/. Subscribe at https://www.ietf.org/mailman/listinfo/moq/.

Source for this draft and an issue tracker can be found at https://github.com/mondain/msfts.

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/.

Copyright Notice

Table of Contents

1. Introduction

Media Over QUIC Transport (MOQT) [MoQTransport] delivers named tracks as ordered groups of objects. The MOQT Streaming Format (MSF) [MSF] defines a catalog model and common streaming conventions for describing tracks delivered over MOQT. This document extends MSF by registering the "m2ts" packaging value for carrying MPEG-2 Transport Stream packets as defined by [ISO138181] and M2TS source packets that prefix each transport-stream packet with a four-octet source-packet timestamp.

The format is intended for publishers that already produce packetized MPEG-2 Transport Stream output, including contribution feeds, broadcast workflows, and systems that currently segment transport streams for HTTP-based delivery. It does not define a new elementary stream container. Instead, it preserves the packet stream and maps consecutive source packets into MOQT Objects.

This document describes version 1 of the packaging format.

2. MSF Extension

All specifications, requirements, and terminology defined in [MSF] apply to implementations of this extension unless explicitly noted otherwise in this document.

This document does not use the LOC packaging defined in [MSF]. MSF requirements that are conditioned on packaging: loc do not apply to m2ts-packaged tracks; equivalent behavior for m2ts tracks is defined in this document.

3. Conventions 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.

This document uses the conventions detailed in Section 1.3 of [RFC9000] when describing the binary encoding.

The following terms are used throughout this document:

TS packet: A 188-octet MPEG-2 Transport Stream packet as defined by [ISO138181].

M2TS source packet: A 192-octet packet consisting of a four-octet source-packet timestamp followed by a 188-octet TS packet.

Source packet: Either a TS packet or an M2TS source packet. The source-packet size is signaled by the catalog.

Access unit: A coded audio, video, or metadata unit carried by the MPEG-2 Transport Stream.

Random access point: A point in the packet stream at which a receiver can begin decoding after receiving the applicable transport-stream tables and decoder initialization.

Single-program transport stream: A transport stream whose Program Association Table lists exactly one program.

Multi-program transport stream: A transport stream whose Program Association Table lists two or more programs.

4. Scope

This specification defines:

* The "m2ts" packaging value for use in an MSF catalog.

* The mapping of consecutive TS or M2TS source packets into MOQT Objects.

* Catalog fields that describe packet size, program selection, packetization, timing, and joining behavior.

* Receiver processing rules for validating object payloads and reconstructing the packet stream.

This specification does not define:

* New MPEG-2 Transport Stream syntax.

* New audio, video, metadata, or subtitle codec signaling inside the transport stream.

* A replacement for Program Association Table, Program Map Table, PCR, PTS, DTS, continuity counter, or scrambling semantics defined by [ISO138181].

* A mandatory Adaptive Bitrate (ABR) switching model across separately encoded transport streams.

* A key management protocol.

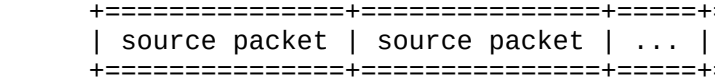
5. Media Packaging

An m2ts-packaged MOQT Track carries a single ordered packet stream. Each MOQT Object payload is a concatenation of one or more whole source packets. The publisher MUST NOT split a source packet across MOQT Objects.

When this packaging mode is used for a track, the MSF catalog packaging field MUST be present and MUST be populated with the value "m2ts".

5.1. Object Payload Format

The payload of each media Object is:



The packet size is signaled by m2tsPacketSize (Section 6.2). The payload length of every media Object on an m2ts track MUST be an integer multiple of m2tsPacketSize.

If m2tsPacketSize is 188, every source packet MUST begin with the MPEG-2 TS sync byte 0x47. If m2tsPacketSize is 192, the TS packet begins four octets after the start of each source packet and the octet at that position MUST be 0x47. A receiver MUST treat a packet that fails this validation as invalid media data for that track.

The source packets from received Objects are concatenated in ascending Group ID and Object ID order to reconstruct the packet stream. A subscriber that skips or fails to receive an Object MUST consider the reconstructed packet stream discontinuous at that point until it reaches a subsequent random access point.

5.2. Object Boundaries

Object boundaries are packaging boundaries and do not change MPEG-2 Transport Stream semantics. Continuity counters, adaptation fields, PCR, PTS, DTS, Program Specific Information (PSI), and other transport-stream syntax remain inside the source packets.

A publisher SHOULD place an independently usable random access point at the first media Object of each MOQT Group. For video, this normally means that the Group begins at or before the transport-stream packets carrying a random access point and includes the PAT and PMT packets required for program demultiplexing. Codec-level initialization data is carried inside the elementary stream packets of the first video access unit and is therefore present whenever a random access point is included.

When m2tsRandomAccess (Section 6.8) is true, the first media Object in every Group MUST begin at a random access point.

5.3. Group Numbering

For live streams, publishers SHOULD start a new MOQT Group at each point where the Group content is independently decodable without reference to prior Groups. For video, a valid Group start is any intra-coded access point at which all decoder references needed by that Group are present within the Group itself. Instantaneous Decoder Refresh (IDR) frames always satisfy this condition. A Clean Random Access (CRA) frame MAY serve as a Group start only if no subsequent Random Access Skipped Leading (RASL) pictures in that Group reference frames from a prior Group. Publishers are not required to start a new Group at every intra-coded access point: a CRA whose following RASL pictures reference only frames present earlier in the same Group may remain interior to that Group. The Object ID MUST increase by one for each Object within a Group unless MOQT delivery semantics permit gaps that are explicitly intended by the publisher.

For video-on-demand (VOD) streams, Group ID and Object ID assignment SHOULD be stable for a given asset so that relays and subscribers can cache and request repeatable ranges.

5.4. Packetization

Publishers SHOULD choose Object sizes that are large enough to amortize MOQT object overhead and small enough to avoid excessive head-of-line delay at the application layer. The number of source packets per Object can vary, but a publisher SHOULD keep it stable within a track unless adapting to network or encoder conditions.

If m2tsPacketsPerObject is present, it declares the usual number of source packets per media Object. The final Object of a Group MAY contain fewer source packets. Receivers MUST use the actual Object payload length rather than assuming every Object has the declared size.

5.5. Multi-Program Source Handling

An m2ts track SHOULD carry packets from at most one MPEG-2 program, producing a single-program transport stream. A publisher receiving a multi-program transport stream (MPTS) SHOULD produce a separate m2ts track for each program it wishes to offer, filtering the source packets so that each track contains only:

* Null packets with Packet Identifier (PID) 0x1FFF, which MAY be removed or retained at the publisher’s discretion.

* Program Association Table packets (PID 0x0000), rewritten to list only the program present in this track.

* Program Map Table packets for the selected program (whose PID is listed in the Program Association Table entry for that program).

* All packets whose PID is listed in the Program Map Table of the selected program, including the PCR_PID and the PIDs of all elementary streams.

These rules apply to unscrambled transport stream sources. Publishers filtering scrambled transport streams MUST also retain the conditional access packets required for descrambling; conditional access integration is application-specific and outside the scope of this document.

The m2tsProgramNumber field (Section 6.4) SHOULD be present on tracks derived from a multi-program source to identify the program carried.

When multiple tracks are derived from the same MPTS source, the publisher SHOULD use the MSF altGroup field if the programs are alternate renditions of the same content. Programs that are independent services SHOULD be published as separate tracks; whether to include them in the same catalog is application-specific.

5.6. PCR and Timing

The Program Clock Reference (PCR) is carried inside adaptation fields of transport-stream packets as defined by [ISO138181]. MOQT Object and Group boundaries are packaging boundaries and do not alter PCR continuity within a track.

A publisher MUST NOT introduce a PCR discontinuity within a single MOQT Group. A publisher that introduces a PCR discontinuity between consecutive MOQT Groups MUST signal it by setting the discontinuity_indicator bit (ISO 13818-1 Section 2.4.3.5) in the adaptation field of the first TS packet carrying PCR in the new Group.

Note: The 33-bit PCR base field wraps around after approximately 26.5 hours of continuous stream time. For long-running live streams this is a normal event; receivers should handle it as a continuous timeline continuation rather than a discontinuity. Receivers that use the MSF Media Timeline [MSF] for playout timing can rely on its monotonic wall-clock abstraction independently of PCR wrap-around.

5.7. Splice Signaling

SCTE-35 splice information is carried transparently in the TS stream as splice_info_section() messages on their designated PID. Publishers MAY surface splice events via the MSF Event Timeline [MSF]. This document does not specify SCTE-35 processing.

6. Catalog

An m2ts track is described by the MSF catalog [MSF]. The catalog track name, delta update rules, variable substitution rules, authorization signaling, and common track fields are inherited from MSF.

This document defines additional fields for track objects whose packaging value is "m2ts". A parser MUST ignore fields it does not understand.

6.1. Track Object Fields

Table 1 lists the m2ts-specific fields defined within a track object.

Field	Name	Definition
M2TS packet size	m2tsPacketSize	Section 6.2
M2TS packets per Object	m2tsPacketsPerObject	Section 6.3
M2TS program number	m2tsProgramNumber	Section 6.4
M2TS PMT PID	m2tsPmtPid	Section 6.5
M2TS PCR PID	m2tsPcrPid	Section 6.6
M2TS PSI interval	m2tsPsiInterval	Section 6.7
M2TS random access	m2tsRandomAccess	Section 6.8
M2TS timestamp mode	m2tsTimestampMode	Section 6.9
M2TS SCTE-35 PID	m2tsScte35Pid	Section 6.10
Initialization data	initData	Section 6.11

Table 1

6.2. M2TS Packet Size

Required: Yes JSON Type: Number Location: Track Object

The source-packet size in octets. The value MUST be either 188 or 192. A value of 188 identifies ordinary MPEG-2 TS packets. A value of 192 identifies M2TS source packets with a four-octet timestamp prefix followed by a 188-octet TS packet.

6.3. M2TS Packets per Object

Required: Optional JSON Type: Number Location: Track Object

The usual number of source packets carried by each media Object. This field is advisory. Receivers MUST validate each Object using its actual payload length.

6.4. M2TS Program Number

Required: Optional JSON Type: Number Location: Track Object

The MPEG-2 Transport Stream program number carried by this track. When present, the track SHOULD carry packets from only that program (see Section 5.5). When absent, a track MAY carry multiple programs and subscribers MAY select a program using local policy or transport-stream signaling.

6.5. M2TS PMT PID

Required: Optional JSON Type: Number Location: Track Object

The packet identifier carrying the Program Map Table for m2tsProgramNumber. This field is advisory and does not replace the Program Association Table or Program Map Table carried in the transport stream.

6.6. M2TS PCR PID

Required: Optional JSON Type: Number Location: Track Object

The packet identifier carrying the Program Clock Reference for the program identified by m2tsProgramNumber. This field is advisory and does not replace PCR signaling in the transport stream.

6.7. M2TS PSI Interval

Required: Optional JSON Type: Number Location: Track Object

The maximum interval, in milliseconds, at which the publisher expects to repeat the Program Association Table and Program Map Table in the packet stream. When present, publishers SHOULD repeat PSI at an interval no larger than this value for live content. Subscribers MAY use this value to estimate join latency.

6.8. M2TS Random Access

Required: Optional JSON Type: Boolean Location: Track Object

When true, the first media Object in every MOQT Group begins at a random access point. When absent or false, subscribers MUST inspect the transport-stream payload to determine where decoding can begin.

6.9. M2TS Timestamp Mode

Required: Optional JSON Type: String Location: Track Object

For 192-octet source packets, this field identifies the interpretation of the four-octet source-packet timestamp. The value "arrival-time" indicates an arrival-time or emission-time stamp associated with the following TS packet. The value "opaque" indicates that the timestamp prefix is carried without specified semantics. This field MUST NOT be present when m2tsPacketSize is 188.

6.10. M2TS SCTE-35 PID

Required: Optional JSON Type: Number Location: Track Object

The PID carrying SCTE-35 splice_info_section() messages for this track. This field is advisory; SCTE-35 messages are also discoverable via the PMT CA/registration descriptor. When present, receivers MAY use this value to locate splice events without parsing PMT. Publishers SHOULD include this field when the track carries SCTE-35 splice signaling.

6.11. Initialization Data

Required: Optional JSON Type: String Location: Track Object

An m2ts track MAY use the MSF initData field to carry Base64 [BASE64] encoded initialization data. If present, the decoded value MUST be a sequence of whole source packets using the packet size declared by m2tsPacketSize.

Publishers SHOULD include current PAT and PMT packets in initData when those tables are not guaranteed to be available at the first Object of each Group. When PSI changes within a live track, publishers SHOULD update initData to reflect the new PAT and PMT before publishing subsequent Objects. Receivers MUST NOT assume that initData remains valid after a version change in transport-stream PSI; updated PSI in the media Objects takes precedence.

7. Catalog Examples

The following examples are non-normative.

7.1. Live 188-octet Transport Stream

```
{
  "version": 1,
  "generatedAt": 1746104606044,
  "tracks": [
    {
      "name": "program-1-ts",
      "namespace": "live.example.com/channel/1",
      "packaging": "m2ts",
      "isLive": true,
      "targetLatency": 1000,
      "role": "video",
      "mimeType": "video/mp2t",
      "bitrate": 6000000,
      "m2tsPacketSize": 188,
      "m2tsPacketsPerObject": 64,
      "m2tsProgramNumber": 1,
      "m2tsPmtPid": 256,
      "m2tsPcrPid": 257,
      "m2tsPsiInterval": 100,
      "m2tsRandomAccess": true
    }
  ]
}
```

7.2. Live 192-octet M2TS Source Packets

```
{
  "version": 1,
  "generatedAt": 1746104606044,
  "tracks": [
    {
      "name": "program-1-m2ts",
      "namespace": "contribution.example.net/feed/a",
      "packaging": "m2ts",
      "isLive": true,
      "targetLatency": 500,
      "role": "video",
      "mimeType": "video/mp2t",
      "bitrate": 12000000,
      "m2tsPacketSize": 192,
      "m2tsPacketsPerObject": 32,
      "m2tsProgramNumber": 1,
      "m2tsTimestampMode": "arrival-time",
      "m2tsRandomAccess": true
    }
  ]
}
```

7.3. VOD Transport Stream

```
{
  "version": 1,
  "tracks": [
    {
      "name": "asset-main",
      "namespace": "vod.example.com/assets/1000",
      "packaging": "m2ts",
      "isLive": false,
      "trackDuration": 632000,
      "role": "video",
      "mimeType": "video/mp2t",
      "bitrate": 4500000,
      "m2tsPacketSize": 188,
      "m2tsPacketsPerObject": 96,
      "m2tsProgramNumber": 1,
      "m2tsRandomAccess": true
    }
  ]
}
```

7.4. Multi-Program Source - Two Programs from One MPTS

This example shows a catalog for a publisher that receives a 2-program transport stream and publishes each program as a separate m2ts track. The two tracks share a namespace but are independent services; altGroup is not used because the programs carry different content.

```
{
  "version": 1,
  "generatedAt": 1746104606044,
  "tracks": [
    {
      "name": "program-1",
      "namespace": "live.example.com/mux/1",
      "packaging": "m2ts",
      "isLive": true,
      "targetLatency": 1000,
      "role": "video",
      "mimeType": "video/mp2t",
      "bitrate": 6000000,
      "m2tsPacketSize": 188,
      "m2tsPacketsPerObject": 64,
      "m2tsProgramNumber": 1,
      "m2tsPmtPid": 256,
      "m2tsPcrPid": 257,
      "m2tsPsiInterval": 100,
      "m2tsRandomAccess": true
    },
    {
      "name": "program-2",
      "namespace": "live.example.com/mux/1",
      "packaging": "m2ts",
      "isLive": true,
      "targetLatency": 1000,
      "role": "video",
      "mimeType": "video/mp2t",
      "bitrate": 4000000,
      "m2tsPacketSize": 188,
      "m2tsPacketsPerObject": 64,
      "m2tsProgramNumber": 2,
      "m2tsPmtPid": 512,
      "m2tsPcrPid": 513,
      "m2tsPsiInterval": 100,
      "m2tsRandomAccess": true
    }
  ]
}
```

7.5. ABR Alternate Renditions - Two Bitrate Tracks

This example shows a catalog for a live channel published at two bitrates as alternate renditions. Both tracks are in the same altGroup; video tracks MUST align Group boundaries at identical presentation positions. The tracks use different PID assignments: a subscriber switching between them MUST re-parse PAT and PMT on the new track before routing packets to a decoder.

```
{
  "version": 1,
  "generatedAt": 1746104606044,
  "tracks": [
    {
      "name": "video-high",
      "namespace": "live.example.com/channel/1",
      "packaging": "m2ts",
      "isLive": true,
      "targetLatency": 1000,
      "role": "video",
      "mimeType": "video/mp2t",
      "bitrate": 6000000,
      "altGroup": 1,
      "m2tsPacketSize": 188,
      "m2tsPacketsPerObject": 64,
      "m2tsProgramNumber": 1,
      "m2tsPmtPid": 256,
      "m2tsPcrPid": 257,
      "m2tsPsiInterval": 100,
      "m2tsRandomAccess": true
    },
    {
      "name": "video-low",
      "namespace": "live.example.com/channel/1",
      "packaging": "m2ts",
      "isLive": true,
      "targetLatency": 1000,
      "role": "video",
      "mimeType": "video/mp2t",
      "bitrate": 2000000,
      "altGroup": 1,
      "m2tsPacketSize": 188,
      "m2tsPacketsPerObject": 64,
      "m2tsProgramNumber": 1,
      "m2tsPmtPid": 512,
      "m2tsPcrPid": 513,
      "m2tsPsiInterval": 100,
      "m2tsRandomAccess": true
    }
  ]
}
```

8. Subscriber Processing

A subscriber obtains the catalog using the MSF catalog workflow and subscribes to one or more m2ts tracks. For each received media Object, the subscriber:

1. Validates that the payload length is a non-zero integer multiple of m2tsPacketSize.

2. Validates the TS sync byte position for each source packet.

3. Reconstructs the packet stream by appending the source packets in MOQT object order.

4. Applies normal MPEG-2 Transport Stream demultiplexing, timing recovery, and decoder initialization.

If validation fails, the subscriber SHOULD discard the invalid Object and treat the reconstructed packet stream as discontinuous. A subscriber MAY continue processing at the next Object, but it SHOULD wait for a random access point before presenting decoded media.

When joining a live track, a subscriber SHOULD start at the newest Group whose first Object is available when m2tsRandomAccess is true. Otherwise, a subscriber SHOULD select a starting Group far enough back to encompass at least one complete PSI repetition cycle before its target presentation time; when m2tsPsiInterval is declared, that value bounds the maximum look-back interval needed. A subscriber MAY use the MSF Media Timeline [MSF] to resolve this time bound to a concrete MOQT Group location for use with a Joining FETCH [MoQTransport]. A subscriber MUST NOT begin media presentation until it has received a valid PAT and PMT for the track.

9. Relay Processing

MOQT relays are not required to parse MPEG-2 Transport Stream syntax. A relay can cache, forward, and prioritize m2ts Objects using MOQT namespace, track, Group ID, Object ID, and delivery metadata.

Relays MAY discard older Groups according to MOQT cache policy. For live content, when m2tsRandomAccess is true, relays that retain partial Groups SHOULD retain the first Object of each Group; by definition, publishers are required to populate that Object with a random access point together with the PAT and PMT packets needed by joining subscribers.

10. Switching and Alternate Renditions

Multiple m2ts tracks can be advertised as alternatives using the MSF altGroup field. Video tracks in the same alternate group MUST place Group boundaries at identical presentation positions; other tracks SHOULD align their Group boundaries to the same positions where possible. All tracks in the alternate group SHOULD set m2tsRandomAccess to true. This ensures that a subscriber can switch

between alternate video tracks at any Group boundary without encountering a misaligned access point. A subscriber SHOULD switch between alternate m2ts tracks only at Group boundaries or at transport-stream random access points that it can independently decode.

This document does not require continuity counter values or PID assignments to match across alternate tracks. Receivers MUST treat a switch between tracks as a packet-stream discontinuity unless application-specific signaling establishes stronger continuity.

A receiver MUST treat a switch between alternate tracks as a PCR discontinuity and MUST re-initialize its system time clock (STC) recovery using the first PCR value received on the new track as the initial reference. In addition to the Group boundary alignment requirements above, publishers providing alternate tracks SHOULD align presentation timestamps at Group boundaries across tracks to enable seamless presentation switching at the application layer. Because PID assignments need not match across alternate tracks, a receiver MUST re-parse the PAT and PMT of the new track after every track switch before routing elementary-stream packets to a decoder.

11. Content Protection

This packaging format preserves any scrambling or conditional access information present in the MPEG-2 Transport Stream. Transport-stream scrambling is opaque to MOQT relays and to this specification.

Object-level encryption MAY be applied using a mechanism such as MoQ Secure Objects [SecureObjects] when signaled by the catalog. When object-level encryption is used, source packet validation is performed after successful decryption.

12. Authorization

Authorization requirements can be advertised using MSF catalog authorization fields. For example, a publisher can use Common Access Token signaling [C4M], Privacy Pass authorization [PrivacyPassAuth], or an application defined authorization scheme.

13. Security Considerations

The security considerations of MOQT [MoQTransport], MSF [MSF], MPEG-2 Transport Stream [ISO138181], and any object encryption scheme apply.

Receivers need to treat transport-stream syntax as untrusted input. Invalid packet sizes, invalid sync bytes, malformed PSI, inconsistent continuity counters, excessive table repetition, and timestamp discontinuities can cause decoder failures or resource exhaustion if not bounded by implementation policy.

Catalog metadata is also untrusted input. Subscribers MUST validate packet sizes, payload lengths, Base64 values, PIDs, program numbers, and object ordering before using the values to allocate memory or configure decoders.

Object-level encryption protects MOQT Object payloads but does not hide MOQT namespace, track name, Group ID, Object ID, object size, or delivery timing from authorized relays. Applications that require confidentiality for media payloads SHOULD use an object encryption scheme in addition to transport security.

14. IANA Considerations

This document has no IANA actions.

If MSF establishes an IANA registry for packaging values, this document requests registration of the value "m2ts" with this document as the reference.

15. References

15.1. Normative References

[BASE64] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/rfc/rfc4648>.

[ISO138181] ISO/IEC, "Information technology - Generic coding of moving pictures and associated audio information: Systems", ISO/IEC 13818-1, 2023.

[MoQTransport] Nandakumar, S., Vasiliev, V., Swett, I., and A. Frindell, "Media over QUIC Transport", Work in Progress, Internet-Draft, draft-ietf-moq-transport-17, 2 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-moq-transport-17>.

[MSF] Law, W., "MOQT Streaming Format", Work in Progress, Internet-Draft, draft-ietf-moq-msf-00, 19 January 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-moq-msf-00>.

[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/rfc/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/rfc/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/rfc/rfc9000>.

15.2. Informative References

[C4M] Law, W., Lemmons, C., Simon, G., and S. Nandakumar, "Authentication scheme for MOQT using Common Access Tokens", Work in Progress, Internet-Draft, draft-ietf-moq-c4m-00, 19 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-moq-c4m-00>.

[PrivacyPassAuth] Nandakumar, S., Jennings, C. F., and T. Meunier, "Privacy Pass Authentication for Media over QUIC (MoQ)", Work in Progress, Internet-Draft, draft-ietf-moq-privacy-pass-auth-02, 2 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-moq-privacy-pass-auth-02>.

[SecureObjects] Jennings, C. F., Nandakumar, S., and R. Barnes, "End-to-End Secure Objects for Media over QUIC Transport", Work in Progress, Internet-Draft, draft-ietf-moq-secure-objects-00, 2 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-moq-secure-objects-00>.

Appendix A. Acknowledgments

This document follows the repository and draft structure used by the MOQT Streaming Format work.

Authors' Addresses

Paul Gregoire
Red5
Email: paul@red5.net

Gwendal Simon
Synamedia
Email: gsimon@synamedia.com