

MOQT Issues London

MoQ Interim in London, June 2026

#1655 draft-18: Tighter language around GOAWAYs on a request stream

Which are valid?

	Sender	Receiver
SUBSCRIBE	Y	Y
PUBLISH	Y	Y
PUB_NS	N	Y
SUB_NS	N	Y
SUB_TRACKS	N	Y
FETCH	N	?
TRACK_STATUS	N	N

#1653 [draft-18] REQUEST_OK in response to REQUEST_UPDATE lacks Request ID

The only way to implement -18 is to assume FIFO responses to REQUEST_UPDATE

→ Smells like HTTP/1.1

Options:

1. Assume FIFO
2. Limit to 1 outstanding REQUEST_UPDATE
3. Add a correlator (Request ID)
 - a. Only needed for _OK, REQUEST_ERROR fails everything.
 - b. Could be optional (eg the first _OK skips it)

#1634 [draft-18] What semantics do FIN or RST on a request stream carry?

In Draft < 17, there was one way to cancel a request (e.g. UNSUBSCRIBE)

Now there are 3: RST or FIN write half, STOP_SENDING read half

Are they all equivalent?

RESET and STOP_SENDING → equivalent

#1634 [draft-18] What semantics do FIN or RST on a request stream carry?

FIN

For the responder, means “Request Complete”, as long as it follows OK/ERROR

For the sender, could mean

- “I’m not going to send REQUEST_UPDATE, awaiting reply”
- Cancel

Consider TRACK_STATUS (not updatable)

Recommendation: FIN from *sender* is not a cancellation

#1635 [draft-18] Should a subscriber treat FIN/RST on a bidi stream as equivalent to PUBLISH_DONE?

P->S: PUBLISH

P->S: Object, Object, Object

P->S: FIN

We can simulate a PUBLISH_DONE or make it a session error

Recommendation: Simulate PUBLISH_DONE.

Let's pick sensible values for Status Code (FIN=SUBSCRIPTION_ENDED?
RST=INTERNAL_ERROR?) and Stream Count (maxint)

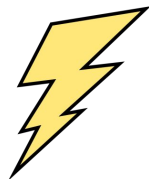
#1643 [draft-18] Is it a request or session error to FIN a response bidi stream without a control message first?

C->S: SUBSCRIBE

S->C: FIN (no REQUEST_ERROR)

This ought to be avoidable by the responder. HTTP equivalent is only a request error (malformed)

Recommendation: Request error



#1622 draft-18: Request ID in GOAWAY isn't useful

- It's not clear what to do with Request ID, given we don't have any normative language.
 - We say saying it increments by 2 for each Request
- Underlying goal of the PR is to rely on Request ID less, and GOAWAY is one of the few remaining cases
- Only added in draft-18, so if we're going to remove it, now is a great time

#1637 What does MOQT do without bidi stream credit?

- SUBSCRIBE_NAMESPACE, SUBSCRIBE_TRACKS, PUBLISH_NAMESPACE, FETCH, TRACK_STATUS clearly require retry by the requester
- We used to have REQUEST_BLOCKED, now we have QUIC STREAMS_BLOCKED
- App responsibility or MOQT responsibility? Is it an API question?
- HTTP often just opens another session. Good enough or anti-pattern?
 - Since HTTP/1.1, have discouraged multiple connections
 - Can't have request dependencies across sessions
 - Servers can't do this
- What is the expected relay behavior if a request doesn't have credit?

#1637 (cont'd) What does MOQT do without bidi stream credit?

SUBSCRIBE/PUBLISH are different.

If PUBLISH doesn't have credit, can replace with SUBSCRIBE, and vice-versa.

We have PUBLISH_BLOCKED for Namespace streams only.

What is the expected behavior on PUBLISH_BLOCKED?

- Add stream credit
- Send SUBSCRIBE – what if this is blocked?

#1651 How does GROUP_ORDER interact with PUBLISH?

- SUBSCRIBE
 - subscriber may specify the group order
 - if not specified, use publisher default
 - may not update
- PUBLISH
 - Publisher has to use its default
 - subscriber can update with PUBLISH_OK!!!!
- Proposal
 - SUBSCRIBE unchanged
 - PUBLISH: use SUBSCRIBE_TRACKS if available
 - Not allowed in PUBLISH_OK

#1650 Forward Handling: SHOULD set Forward=1 upstream is too strong

Relays SHOULD set the Forward flag to 1 when a new subscription needs to be sent upstream, regardless of the value of the Forward field from the downstream subscription.

Propose we remove the normative behavior, and replace with tradeoffs

Pro:

- upstream can prewarm the cache and reduce latency for downstream subscribers

Con:

- Possible DOS vector forcing relay to consume/cache tracks no-one wants

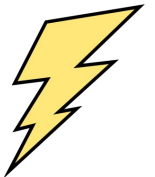
Relays will have policies that govern this behavior

#1644 Duplicate Track Properties on repeated SUBSCRIBE_OK / FETCH_OK

Every SUBSCRIBE_OK and FETCH_OK transmits a fresh copy of the Track Properties even if the subscriber already has that information

Could be bad for FETCH loops (VOD) or filling a cache with many holes, especially if properties are large

Should we have a request option like “PROPERTIES=0/1” - default=1? Or close with no action.



#1636 Empty namespace clarification

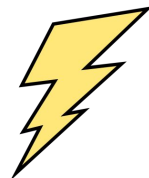
Current draft allows empty namespaces AND names

This seems odd, but is it a problem? It seems potentially confusing.

Options:

1. Status quo: Allow empty namespaces and names
2. Require Track names to be non-empty
 - a. Can still SUBSCRIBE_* to the empty namespace
3. Require Track namespace to be non-empty

Recommendation: Require Track Name to be non-empty



#1626 Version negotiation for QMUX

[#1628](#) add 4 lines of text explaining how to use MOQT with QMux

Include `moqt-qx-18` ALPN → MOQT-18 over QMux-01

Can pair with h2 if both QMux and H2-WT are supported

Aligns with current QMux and IETF 125 discussions, but a little odd

Why is the token not just `moqt-18`?

Bigger picture – Reference QMux in MOQT or make a separate short draft?

#1582 Caching and propagation of REQUEST_ERRORS

- We have a clearly defined caching model for objects (object uniqueness, MAX_CACHE_TIMEOUT)
- We also have a similar model for track properties.
- What does relay do when it receives error from the upstream?

Proposal:

- Add cache TTL to the REQUEST_ERROR.
- Do we need dedicated cache TTL, or should we use retry interval?

#1489 Should there be a Reconnect Timeout?

Time the subscriber is willing to wait for a new publisher to materialize after the old one disconnected

- Issue came from discussion on the Rendezvous Timeout PR [#1447](#)
- Not much interest recently
- **Recommendation:** Close with no action

#1453 Send Rate parameter

- We discussed this at the 3/30/2026 interim
- The concept was appealing, however Subscription for a Track doesn't map 1:1 with a QUIC connection
 - SCONE is done per QUIC connection (or path)
 - It's not clear the subscriber is in a good place to set this
 - As a result, the proposal is **no action** for now

[#1405](#) Single Object Subgroups don't need a Subgroup ID

- PR [#1608](#) said that the first Object ID in the Subgroup was the Subgroup ID
- That constrained CMAF/LOQMAF from the current design
- There is interest in **simplifying** the Object Model, but all existing proposals break existing use cases
 - It's possible that CMAF/LOQMAF and others could make reasonable adjustments to their mapping, but that raises the bar.
- The Subgroup ID can already be omitted on the wire in a single Object subgroup if the ID is the same as the Object ID

Options:

- Close the issue with no action
- Design new changes to simplify the Object model

#1352 SUBSCRIBE does not need a forward parameter if we have filters

Object Range Filters are merging in

Do we want to represent Forward = 0/1 as a filter instead of a param?

Pro: One fewer param

Con:

- One more filter type

- Forward=0/1 is somewhat baked in

Editors Recommendation: Leave as is

#1316 Possible improvements on MoQ support of VOD applications

With #1642, the following is possible:

S->P: SUBSCRIBE vod-file AbsoluteRangeFill [0-10]

P->S: Fill Fetch Stream

S->P: REQUEST_UPDATE AbsoluteRangeFill [11-20]

P->S: Fill Fetch Stream

REQUEST_UPDATE AbsoluteRangeFill [21-30]

P->S: Fill Fetch Stream

So now you can “subscribe to VOD” more or less → Good to close?

#1270 Should the publisher be able to update default publisher priority

We discussed at length in Toronto:

Default Publisher Priority specified as an Track Property can be omitted in the data plane

Updating it is racy. If you receive an object with no priority, was it sent before/after the update?

Also, you can't exactly modify a Track Property – you can override it with an Object Property. How does a late joiner know if they missed the signal?

But for low-bitrate datagrams, it might be nice?

Skip it?

#1063 Rate limiting REQUEST_UPDATE

Since the move to bidi streams, either side can send unlimited REQUEST_UPDATE messages for a given request

This class of attacks has been popular against HTTP, it will be against MOQT

We allow coalescing of REQUEST_UPDATES

Receiver can delay REQUEST_UPDATES until after processing packet/etc

PR#1613: Limits the number of outstanding REQUEST_UPDATE per request

Setup Option: MAX_REQUEST_UPDATES, default=1

#974 Announce upstream delivery timeouts

Delivery timeout is defined per-hop, with no way to express the cumulative end-to-end timeout across a multi-relay path.

Latency-sensitive subscribers (e.g., needing <1s end-to-end) can't see that an upstream relay has a much larger timeout (e.g., 5s)

Has not been an issue so far and it's a complex problem

Recommendation: Close with no action

#969 Should we move Authorization Alias to an extension?

Show of hands:

How many people have or plan to implement Auth Alias compression?

How many people have not and do not plan to implement?

Strongly inclined to remove Auth Compression

- 1) Interest is low
- 2) This is less of an issue in MOQ than it was in HTTP because subscriptions are long lived
- 3) Bidi streams reduced the effectiveness without QPACK-esque features
- 4) I wrote the [MOQPACK](#) extension:
 - a) compresses all tokens, track namespaces/names and other params
 - b) Works across QUIC streams
- 5) Having this in the core will likely draw more scrutiny during endgame

Issue #799: Auth Verification in ANNOUNCE

Issue #799 | Opened by Magnus Westerlund | Labels: Auth, Security/DoS, Needs PR

KEY POINTS

- Spec says 'receiver verifies publisher is authorized'
- End subscribers don't have keys/context to verify this
- Relays within a CDN can verify; end subscribers cannot
- AUTH_TOKEN (0x03) MAY appear on PUBLISH, SUBSCRIBE, SUB_NS, PUB_NS, REQUEST_UPDATE, SUB_TRACKS, TRACK_STATUS, FETCH
- AUTH_TOKEN is per-hop — relays don't forward it

PROPOSED RESOLUTION

AUTH_TOKEN exists on all control messages but is per-hop only. The blanket 'receiver verifies' requirement doesn't apply uniformly — different roles have different trust relationships.

Authorization Cases

Publisher -> Relay

AUTH_TOKEN on PUB_NS
verified by relay

Relay -> Subscriber

NAMESPACE inherits trust
from SUB_NS; or PUB_NS w/ AUTH

Relay -> Relay

AUTH_TOKEN on PUB_NS
CDN-internal / future work

Issue #1503: PUBLISH_NAMESPACE vs NAMESPACE

Issue #1503 | Opened by Martin Duke | Labels: Security/DoS, Needs PR

KEY POINTS

- Subscriber may receive PUBLISH_NAMESPACE for a namespace within active SUB_NS prefix
- Race condition: SUB_NS or REQUEST_OK in flight + PUBLISH_NS in flight
- PUBLISH_NAMESPACE may outlive the SUB_NS (stream close = implicit NS_DONE)
- PUBLISH_NAMESPACE can carry AUTH_TOKEN; NAMESPACE on SUB_NS stream cannot
- A relay has no reason to do this in draft-18; direct pub-sub: race possible

PROPOSED RESOLUTION

Answer is likely 'yes, it's allowed' due to the race condition. AUTH_TOKEN on PUBLISH_NAMESPACE could provide additional authorization that NAMESPACE messages on the SUB_NS stream cannot carry.

Subscriber

Relay

SUBSCRIBE_NAMESPACE(prefix) ->

<- NAMESPACE (on SUB_NS stream)

<- PUBLISH_NAMESPACE (control stream)

Race: SUB_NS in flight while
PUBLISH_NAMESPACE also in flight

NAMESPACE has no params (no AUTH)
PUBLISH_NAMESPACE can carry AUTH_TOKEN

Proposed Spec Changes

PR: [#1656](#)

KEY POINTS

- SUBSCRIBE_NAMESPACE section: add that sending SUB_NS implies trust in the relay for that namespace prefix
- PUBLISH/PUBLISH_NAMESPACE: soften 'receiver verifies' to 'verification is application-specific'
- Security Considerations: document when SUB_NS is safe to use (i.e., when app trusts relay for the prefix)
- Note: relay-to-relay auth is out of scope for this document

Spec Sections to Update

SUBSCRIBE_NAMESPACE

Add: implies trust in relay
for the given prefix

PUBLISH / PUB_NS

Soften: 'receiver verifies'
-> app-specific

Security Considerations

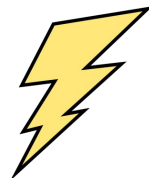
When SUB_NS is safe;
trust model rationale

#899 Do we need to allow multiple ranges in a FETCH request

Now that we have filters, you can filter to a list of subgroups or object ID within a subgroup

Is that sufficient?

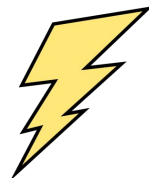
Recommendation: Close with No Action



#881 Renumber Everything

Yes we plan to renumber all the message, enum, data plane, error code points to be sane instead of swiss cheese

Will re-park this until closer to done and bring a proposal



[#869](#) Difficult to limit resource consumption of a Subscription

[PR#1591](#) is an initial proposal to add Subscription flow control

- Provides limits on total streams and total bytes sent on a Subscription
- Uses control messages on bidi stream, RESET_STREAM_AT not required
- Alan's [PR #11](#) is an improved write-up, but conceptually the same

The working group has not expressed enthusiasm in this proposal

- If we allowed doing SUBSCRIBE delivery in the distant past, flow control becomes more necessary.

Recommendation: Close / move to an extension?

#475 Add a timestamp extension

Use Cases:

- Time alignment of Tracks (ie: Audio and Video or Video.Q1 and Video.Q2)
 - Including Tracks that didn't start at the same time
- More accurate Delivery Timeout and Cache Expiry

Time sync across multiple publishers is a difficult to impossible problem

- However, this is only for delivery, so $O(Xms)$ is sufficiently accurate

Proposed ABR mechanisms, such as DTS ([slides](#)) require Group alignment

- Could use timestamp if available, but Group alignment is enough for a single original video source.

#475 (cont'd) Add a timestamp extension

Possible Proposal:

- Add a Track and Object Property for 'Absolute Timestamp'

- Add an Object Property for 'Relative Timestamp' since Track start

- Add a 'TimeSyncGroupID' to Subscriptions to keep them in sync

If the Track 'Absolute Timestamp' matches, then assume they're synced

The Track 'Absolute Timestamp' for non-synced Tracks could match

But the odds are extremely low

#273 Nothing good comes out of a reason phrase

Pro:

Significantly improves the developer experience

Con:

Everything of value should be structured

Everything with no value should be removed

The sentiment on the issue is they should stay at least “for now” → we’re hopefully getting close to the end - let’s make a call

Some arguing for full removal before we ship, some partial (remove from requests but keep in GOAWAY)

#1658 Add Error Payload to REQUEST_ERROR for structured binary data

Privacy Pass needs to return a binary blob in a REQUEST_ERROR

What is the intersection with Reason Phrase?

- a) Keep Both
- b) Remove Reason Phrase
 - i) Nothing stops an implementation from putting a reason phrase in Error Payload?

#1507 FEATURE REQUEST: Mechanism to get the sender's bitrate

The publisher (congestion controller) knows/limits the maximum send rate.
The subscriber wants to know and increase this limit for ABR.

1. Subscriber requests their estimated bitrate.
 - Publisher periodically reports the maximum *connection* bitrate.
2. Subscriber requests increasing their estimated bitrate to X. (higher rendition)
 - Publisher periodically tests higher bitrates while under X.
 - Optimized stacks can transmit redundant copies, otherwise PADDING.
3. Both are per-hop *only* and implementation specific.
 - Should be negotiated per-hop in SETUP (or extension).
 - Alternatives exist if not supported.