imquic, a QUIC library for real-time media

Lorenzo Miniero   Open Media devroom @ FOSDEM   January 31, 2026

Who am I?

Lorenzo Miniero

- Ph.D @ UniNA
- Chairman @ Meetecho
- Main author of Janus

Contacts and info

- lorenzo@meetecho.com
- https://fosstodon.org/@lminiero
- https://bsky.app/profile/lminiero.it
- https://www.meetecho.com
- https://lminiero.it

Who am I?

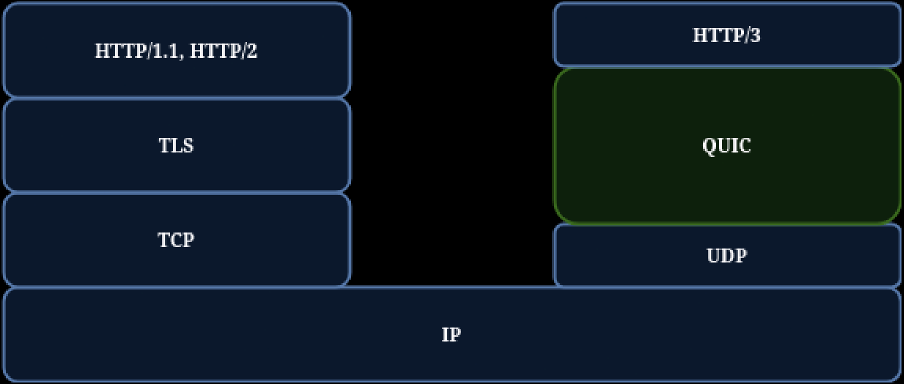
Lorenzo Miniero

- Ph.D @ UniNA
- Chairman @ Meetecho
- Main author of Janus and imquic!

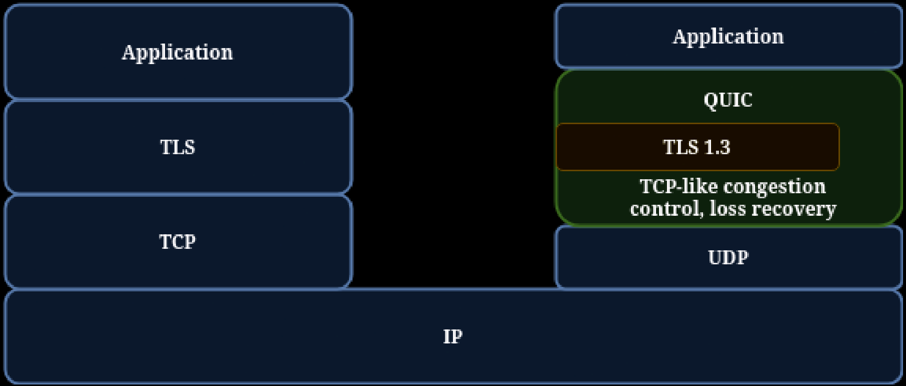
Contacts and info

- lorenzo@meetecho.com
- https://fosstodon.org/@lminiero
- https://bsky.app/profile/lminiero.it
- https://www.meetecho.com
- https://lminiero.it

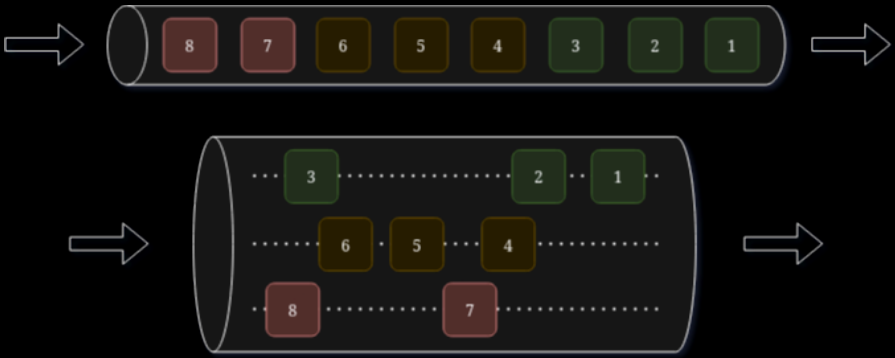
What's QUIC and why does it matter?



What’s QUIC and why does it matter?



What’s QUIC and why does it matter?



Can we use QUIC for real-time media?

- We all know about WebRTC
  - UDP based, ultralow latency
  - Conceived for conversational use cases
  - Peer-to-peer, but often used with servers in the middle
  - Supports retransmissions and congestion control
  - Natively supported in browsers (simple APIs)

Can we use QUIC for real-time media?

- We all know about WebRTC
  - UDP based, ultralow latency
  - Conceived for conversational use cases
  - Peer-to-peer, but often used with servers in the middle
  - Supports retransmissions and congestion control
  - Natively supported in browsers (simple APIs)
- What about QUIC?
  - TCP-like (e.g., HTTP/3), but UDP based
  - Optionally supports **DATAGRAM**
  - Always client/server
  - Supports retransmissions, multiple streams, congestion control
  - Can be used in browsers via WebTransport (+ WebCodecs, WebAssembly)

Having a look at RTP Over QUIC (RoQ)

- IETF is defining how to transport RTP on top of QUIC
  - https://datatracker.ietf.org/doc/draft-ietf-avtcore-rtp-over-quic/

Having a look at RTP Over QUIC (RoQ)

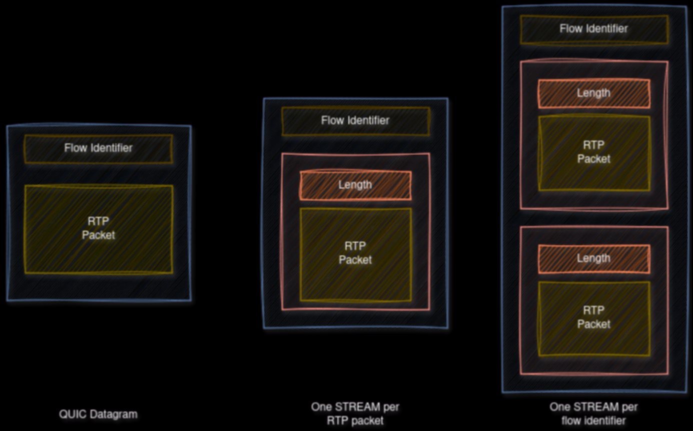
- IETF is defining how to transport RTP on top of QUIC
  - https://datatracker.ietf.org/doc/draft-ietf-avtcore-rtp-over-quic/

- Using QUIC, there are things we can (or have to) do differently
  - No need for SRTP, QUIC is already encrypted
  - Some feedback RTCP provides QUIC can already give us
  - QUIC has integrated BWE as well
  - We need framing for RTP packets (as in TCP)

Having a look at RTP Over QUIC (RoQ)

- IETF is defining how to transport RTP on top of QUIC
  - https://datatracker.ietf.org/doc/draft-ietf-avtcore-rtp-over-quic/
- Using QUIC, there are things we can (or have to) do differently
  - No need for SRTP, QUIC is already encrypted
  - Some feedback RTCP provides QUIC can already give us
  - QUIC has integrated BWE as well
  - We need framing for RTP packets (as in TCP)
- Multiplexing has interesting opportunities too
  - Can multiplex multiple sessions over the same QUIC connection (Flow ID)
  - Multiplexing can be done in different ways (DATAGRAM vs. STREAM(s))

RoQ multiplexing



https://www.meetecho.com/blog/roq-n-roll/

A step further: Media Over QUIC (MoQ)

- Low-latency media delivery solution for ingest/distribution of media
    - https://datatracker.ietf.org/group/moq/about/

A step further: Media Over QUIC (MoQ)

- Low-latency media delivery solution for ingest/distribution of media
  - https://datatracker.ietf.org/group/moq/about/
- Target is configurable latency
  - From conferencing to HLS-like to VOD (Video On Demand)
  - Of interest to both real-time services and CDNs

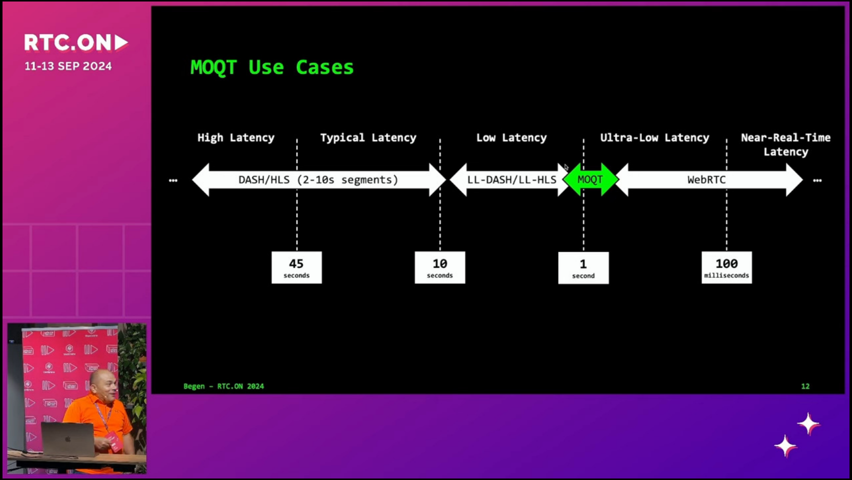
A step further: Media Over QUIC (MoQ)

- Low-latency media delivery solution for ingest/distribution of media
  - https://datatracker.ietf.org/group/moq/about/
- Target is configurable latency
  - From conferencing to HLS-like to VOD (Video On Demand)
  - Of interest to both real-time services and CDNs
- Publisher/Subscriber kind of approach
  - Possible roles are **`Publisher`**, **`Subscriber`** and **`PubSub`**
  - Not that far from a cascaded SFU, if you're familiar with WebRTC

A step further: Media Over QUIC (MoQ)

- Low-latency media delivery solution for ingest/distribution of media
  - https://datatracker.ietf.org/group/moq/about/
- Target is configurable latency
  - From conferencing to HLS-like to VOD (Video On Demand)
  - Of interest to both real-time services and CDNs
- Publisher/Subscriber kind of approach
  - Possible roles are **`Publisher`**, **`Subscriber`** and **`PubSub`**
  - Not that far from a cascaded SFU, if you're familiar with WebRTC
- MoQ Transport (MoQT) on top of QUIC or WebTransport
  - Encryption via QUIC, but E2EE encryption possible too
  - Independent of media formats (can transport anything media)
  - Support for relays, caching and replication points

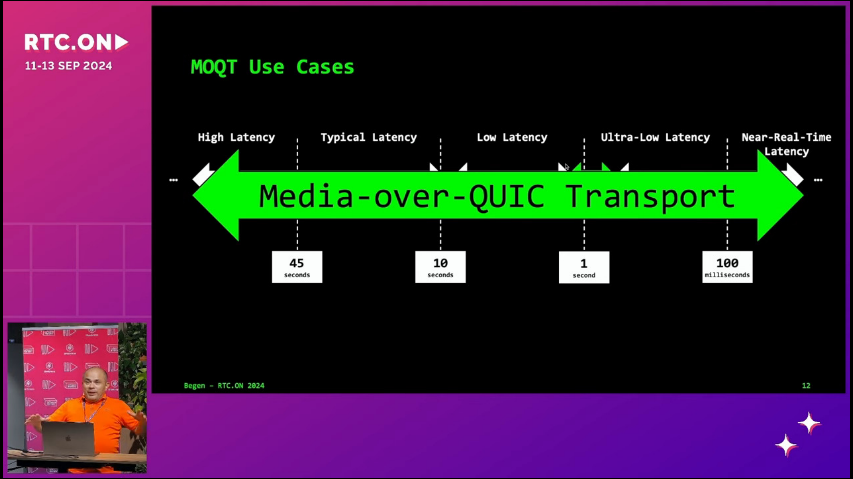
What can it be used for, then?



MOQT Use Cases

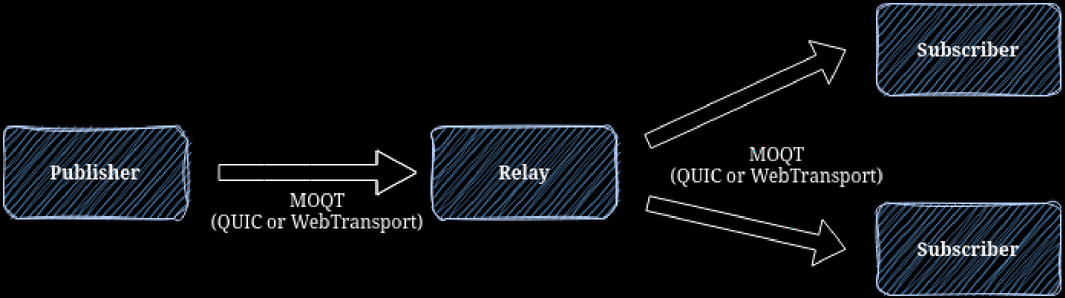
https://www.youtube.com/watch?v=3GRTV1U3bWw

What can it be used for, then?

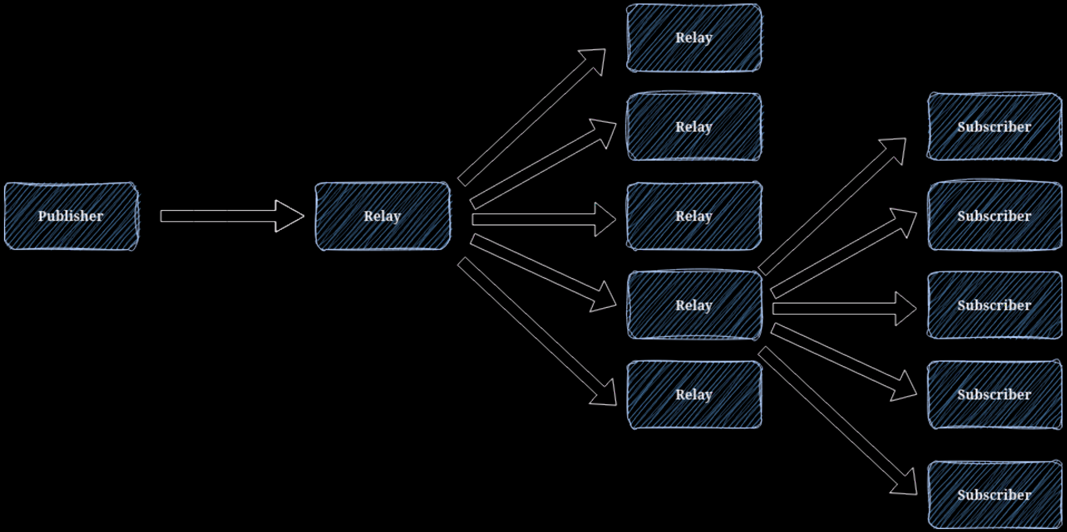


https://www.youtube.com/watch?v=3GRTV1U3bWw

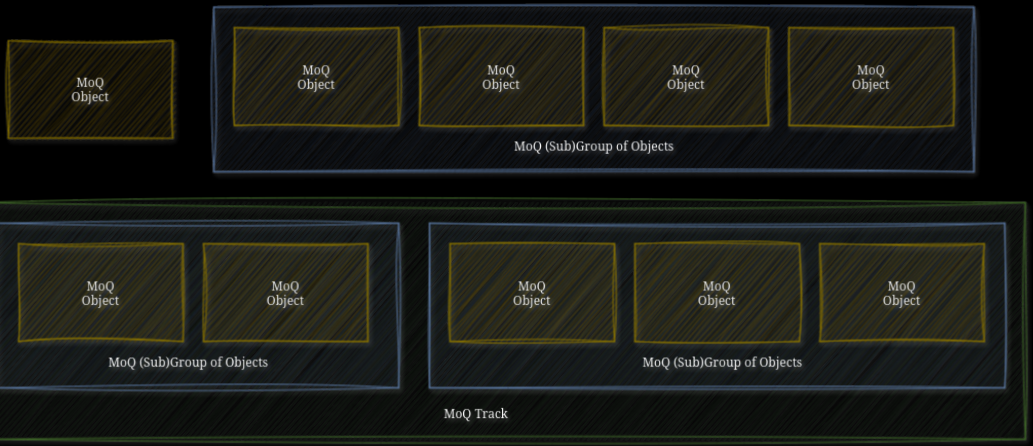
The simplest architecture diagram



Something a bit more complex



Objects, (Sub)Groups and Tracks



Implementation status

- Different implementations (sometimes different MoQT versions)
  - https://github.com/kixelated/moq (Rust/JS, pub/sub/relay)
  - https://github.com/englishm/moq-rs (Rust, pub/sub/relay)
  - https://github.com/facebookexperimental/moxygen (C++, relay)
  - https://github.com/facebookexperimental/moq-encoder-player (JS, pub/sub)
  - https://github.com/Quicr/libquicr (C++, relay/pub/sub)
  - https://github.com/kota-yata/moqtail (TypeScript, pub/sub)
  - https://github.com/moqtail/moqtail (Rust/TS, pub/sub)
  - https://github.com/meetecho/imquic (C library, pub/sub/relay)
  - ...

Implementation status

- Different implementations (sometimes different MoQT versions)
  - https://github.com/kixelated/moq (Rust/JS, pub/sub/relay)
  - https://github.com/englishm/moq-rs (Rust, pub/sub/relay)
  - https://github.com/facebookexperimental/moxygen (C++, relay)
  - https://github.com/facebookexperimental/moq-encoder-player (JS, pub/sub)
  - https://github.com/Quicr/libquicr (C++, relay/pub/sub)
  - https://github.com/kota-yata/moqtail (TypeScript, pub/sub)
  - https://github.com/moqtail/moqtail (Rust/TS, pub/sub)
  - https://github.com/meetecho/imquic (C library, pub/sub/relay)
  - ...
- More info available on MoQT wiki
  - https://github.com/moq-wg/moq-transport/wiki/Interop

There’s a new kid in town!

https://imquic.conf.meetecho.com

A QUIC look at imquic (pun intended)

- Open source QUIC library
  - https://github.com/meetecho/imquic
  - https://www.meetecho.com/blog/imquic/

A QUIC look at imquic (pun intended)

- Open source QUIC library
  - https://github.com/meetecho/imquic
  - https://www.meetecho.com/blog/imquic/
- Generic library, but with native support for a few protocols
  - Raw QUIC vs WebTransport (no full HTTP/3 support yet)
  - RTP Over QUIC (RoQ)
  - Media Over QUIC (MoQ)

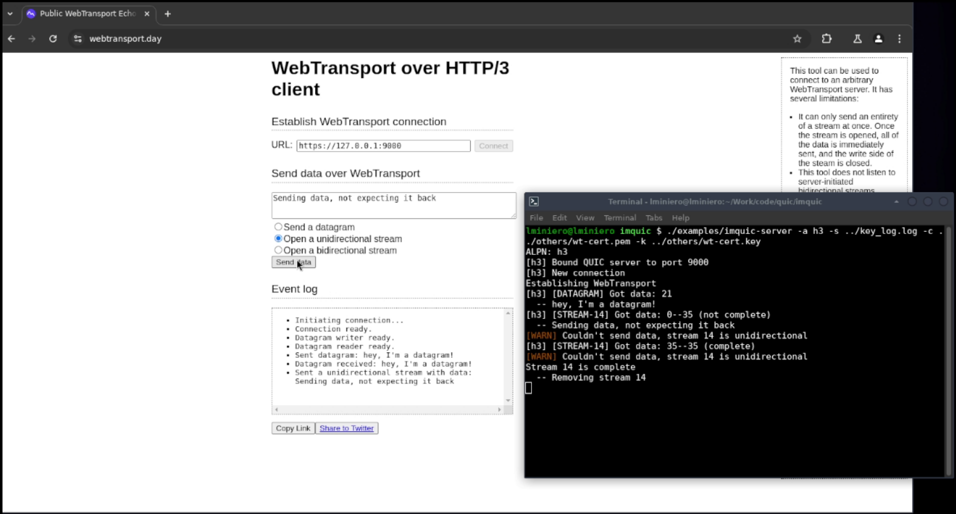
A QUIC look at imquic (pun intended)

- Open source QUIC library
  - https://github.com/meetecho/imquic
  - https://www.meetecho.com/blog/imquic/
- Generic library, but with native support for a few protocols
  - Raw QUIC vs WebTransport (no full HTTP/3 support yet)
  - RTP Over QUIC (RoQ)
  - Media Over QUIC (MoQ)
- High level APIs for servers/clients
  - Methods to proactively do things
  - Callbacks to intercept events (e.g., connection, incoming data, etc.)
  - Custom APIs for natively implemented protocols

A QUIC look at imquic (pun intended)

- Open source QUIC library
  - https://github.com/meetecho/imquic
  - https://www.meetecho.com/blog/imquic/
- Generic library, but with native support for a few protocols
  - Raw QUIC vs WebTransport (no full HTTP/3 support yet)
  - RTP Over QUIC (RoQ)
  - Media Over QUIC (MoQ)
- High level APIs for servers/clients
  - Methods to proactively do things
  - Callbacks to intercept events (e.g., connection, incoming data, etc.)
  - Custom APIs for natively implemented protocols
- Not really ready for production, yet
  - Mostly a testbed/sandbox to experiment with new protocols
  - QUIC stack in particular needs a bit of love (or replacing?)

A couple of WebTransport demos



WebTransport over HTTP/3 client

Establish WebTransport connection

URL: https://127.0.0.1:9000 Connect

Send data over WebTransport

Sending data, not expecting it back

- Send a datagram
- [✓ Open a unidirectional stream]
- Open a bidirectional stream

Send data

Event log

```
• Initiating connection...
• Connection ready.
• Datagram writer ready.
• Datagram reader ready.
• Sent datagram: hey, I'm a datagram!
• Datagram received: hey, I'm a datagram!
• Sent a unidirectional stream with data:
  Sending data, not expecting it back
```

Copy Link Share to Twitter

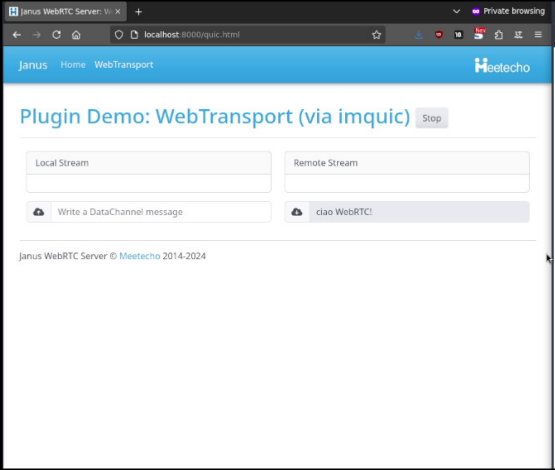
This tool can be used to connect to an arbitrary WebTransport server. It has several limitations:

- It can only send an entirety of a stream at once. Once the stream is opened, all of the data is immediately sent, and the write side of the steam is closed.
- This tool does not listen to server-initiated bidirectional streams

Terminal - lminiero@lminiero:~/Work/code/quic/imquic

```
lminiero@lminiero imquic $ ./examples/imquic-server -a h3 -s ../key_log.log -c .
./others/wt-cert.pem -k ../others/wt-cert.key
ALPN: h3
[h3] Bound QUIC server to port 9000
[h3] New connection
Establishing WebTransport
[h3] [DATAGRAM] Got data: 21
  -- hey, I'm a datagram!
[h3] [STREAM-14] Got data: 0--35 (not complete)
  -- Sending data, not expecting it back
[WARN] Couldn't send data, stream 14 is unidirectional
[h3] [STREAM-14] Got data: 35--35 (complete)
[WARN] Couldn't send data, stream 14 is unidirectional
Stream 14 is complete
  -- Removing stream 14
```

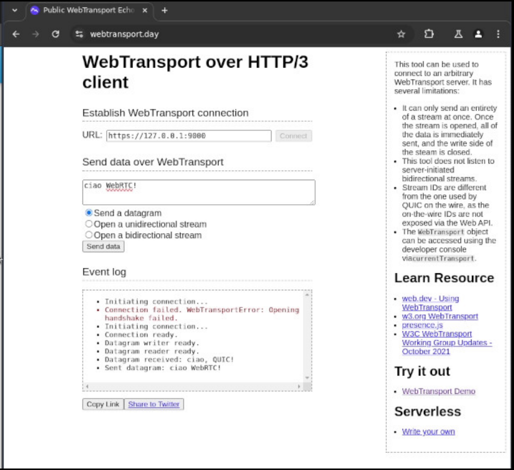
A couple of WebTransport demos



Plugin Demo: WebTransport (via imquic)

ciao WebRTC!

Janus WebRTC Server © Meetecho 2014-2024



WebTransport over HTTP/3 client

Establish WebTransport connection

URL: https://127.0.0.1:9000

Send data over WebTransport

ciao WebRTC!

[✓ Send a datagram]
Open a unidirectional stream
Open a bidirectional stream

Event log

```
• Initiating connection...
• Connection failed. WebTransportError: Opening
  handshake failed.
• Initiating connection...
• Connection ready.
• Datagram writer ready.
• Datagram reader ready.
• Datagram received: ciao, QUIC!
• Sent datagram: ciao WebRTC!
```

This tool can be used to connect to an arbitrary WebTransport server. It has several limitations:

- It can only send an entirety of a stream at once. Once the stream is opened, all of the data is immediately sent, and the write side of the steam is closed.
- This tool does not listen to server-initiated bidirectional streams.
- Stream IDs are different from the one used by QUIC on the wire, as the on-the-wire IDs are not exposed via the Web API.
- The `WebTransport` object can be accessed using the developer console via`currentTransport`.

Learn Resource

- web.dev - Using WebTransport
- w3.org WebTransport
- presence.js
- W3C WebTransport Working Group Updates - October 2021

Try it out

- WebTransport Demo

Serverless

- Write your own

Prototyping RoQ

- Integrating RoQ in my test library itself
    - Leverages library core and events
    - Exposes RoQ-specific APIs to end user

Prototyping RoQ

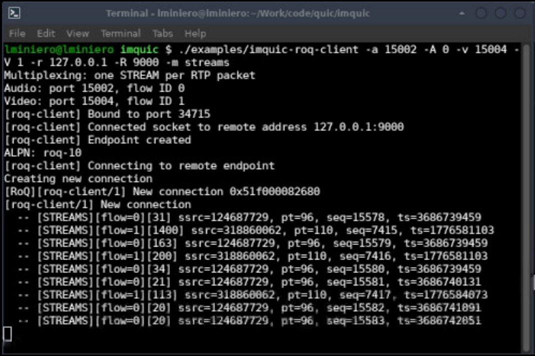
- Integrating RoQ in my test library itself
  - Leverages library core and events
  - Exposes RoQ-specific APIs to end user

- Different demo applications for testing
  - **`imquic-roq-server`**: basic RoQ server (prints headers + echo mode)
  - **`imquic-roq-client`**: basic RoQ client (injects RTP)
  - **Janus integration** (WIP): gatewaying of RoQ to/from WebRTC

Prototyping RoQ

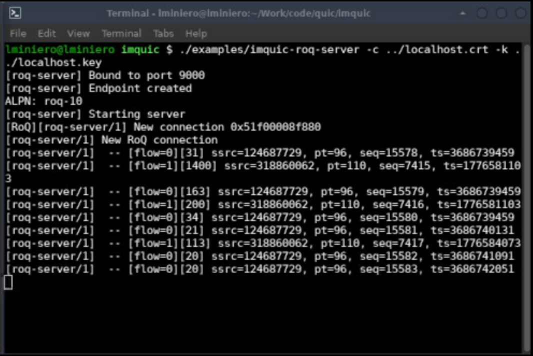
- Integrating RoQ in my test library itself
  - Leverages library core and events
  - Exposes RoQ-specific APIs to end user
- Different demo applications for testing
  - **`imquic-roq-server`**: basic RoQ server (prints headers + echo mode)
  - **`imquic-roq-client`**: basic RoQ client (injects RTP)
  - **Janus integration** (WIP): gatewaying of RoQ to/from WebRTC
- A few interop tests for validation
  - Mathis' RoQ demos (IETF Hackathon 120)

A basic client/server demo



Terminal - lminiero@lminiero:~/Work/code/quic/imquic
File Edit View Terminal Tabs Help

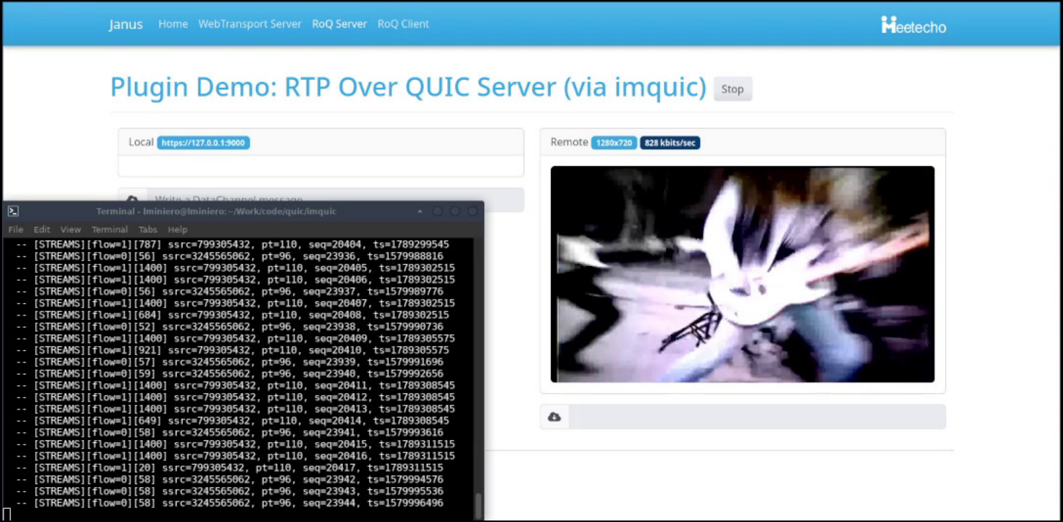
```
lminiero@lminiero imquic $ ./examples/imquic-roq-client -a 15002 -A 0 -v 15004 -
V 1 -r 127.0.0.1 -R 9000 -m streams
Multiplexing: one STREAM per RTP packet
Audio: port 15002, flow ID 0
Video: port 15004, flow ID 1
[roq-client] Bound to port 34715
[roq-client] Connected socket to remote address 127.0.0.1:9000
[roq-client] Endpoint created
ALPN: roq-10
[roq-client] Connecting to remote endpoint
Creating new connection
[RoQ][roq-client/1] New connection 0x51f000082680
[roq-client/1] New connection
  -- [STREAMS][flow=0][31] ssrc=124687729, pt=96, seq=15578, ts=3686739459
  -- [STREAMS][flow=1][1400] ssrc=318860062, pt=110, seq=7415, ts=1776581103
  -- [STREAMS][flow=0][163] ssrc=124687729, pt=96, seq=15579, ts=3686739459
  -- [STREAMS][flow=1][200] ssrc=318860062, pt=110, seq=7416, ts=1776581103
  -- [STREAMS][flow=0][34] ssrc=124687729, pt=96, seq=15580, ts=3686739459
  -- [STREAMS][flow=0][21] ssrc=124687729, pt=96, seq=15581, ts=3686740131
  -- [STREAMS][flow=1][113] ssrc=318860062, pt=110, seq=7417, ts=1776584073
  -- [STREAMS][flow=0][20] ssrc=124687729, pt=96, seq=15582, ts=3686741091
  -- [STREAMS][flow=0][20] ssrc=124687729, pt=96, seq=15583, ts=3686742051
```



Terminal - lminiero@lminiero:~/Work/code/quic/imquic
File Edit View Terminal Tabs Help

```
lminiero@lminiero imquic $ ./examples/imquic-roq-server -c ../localhost.crt -k .
./localhost.key
[roq-server] Bound to port 9000
[roq-server] Endpoint created
ALPN: roq-10
[roq-server] Starting server
[RoQ][roq-server/1] New connection 0x51f00008f880
[roq-server/1] New RoQ connection
[roq-server/1]  -- [flow=0][31] ssrc=124687729, pt=96, seq=15578, ts=3686739459
[roq-server/1]  -- [flow=1][1400] ssrc=318860062, pt=110, seq=7415, ts=177658110
3
[roq-server/1]  -- [flow=0][163] ssrc=124687729, pt=96, seq=15579, ts=3686739459
[roq-server/1]  -- [flow=1][200] ssrc=318860062, pt=110, seq=7416, ts=1776581103
[roq-server/1]  -- [flow=0][34] ssrc=124687729, pt=96, seq=15580, ts=3686739459
[roq-server/1]  -- [flow=0][21] ssrc=124687729, pt=96, seq=15581, ts=3686740131
[roq-server/1]  -- [flow=1][113] ssrc=318860062, pt=110, seq=7417, ts=1776584073
[roq-server/1]  -- [flow=0][20] ssrc=124687729, pt=96, seq=15582, ts=3686741091
[roq-server/1]  -- [flow=0][20] ssrc=124687729, pt=96, seq=15583, ts=3686742051
```

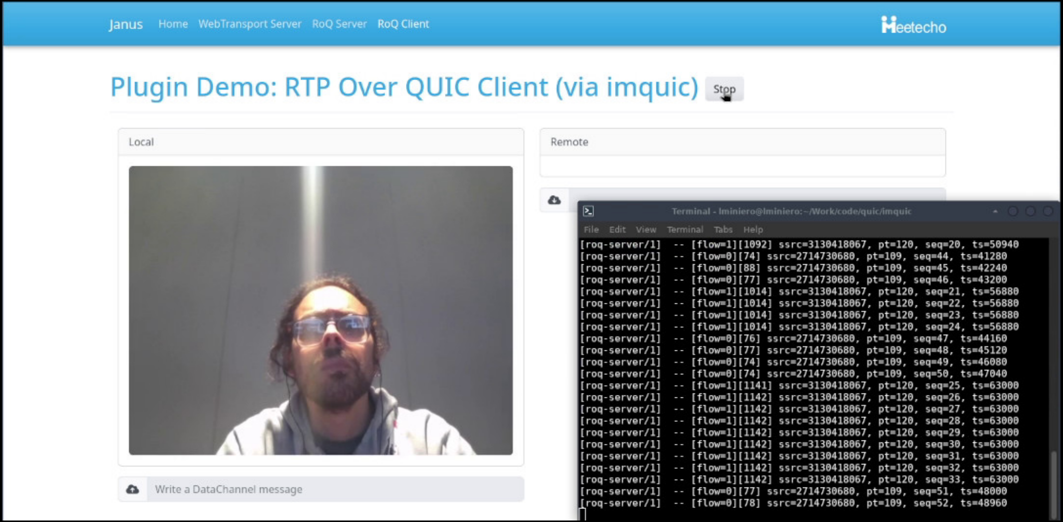
Involving Janus (and WebRTC!)



Plugin Demo: RTP Over QUIC Server (via imquic)

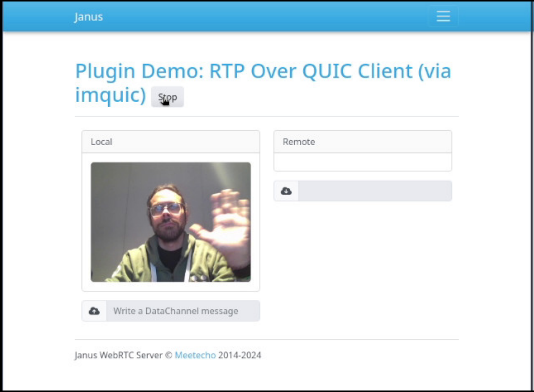
```
 -- [STREAMS][flow=1][787] ssrc=799305432, pt=110, seq=20404, ts=1789299545
 -- [STREAMS][flow=0][56] ssrc=3245565062, pt=96, seq=23936, ts=1579988816
 -- [STREAMS][flow=1][1400] ssrc=799305432, pt=110, seq=20405, ts=1789302515
 -- [STREAMS][flow=1][1400] ssrc=799305432, pt=110, seq=20406, ts=1789302515
 -- [STREAMS][flow=0][56] ssrc=3245565062, pt=96, seq=23937, ts=1579989776
 -- [STREAMS][flow=1][1400] ssrc=799305432, pt=110, seq=20407, ts=1789302515
 -- [STREAMS][flow=1][684] ssrc=799305432, pt=110, seq=20408, ts=1789302515
 -- [STREAMS][flow=0][52] ssrc=3245565062, pt=96, seq=23938, ts=1579990736
 -- [STREAMS][flow=1][1400] ssrc=799305432, pt=110, seq=20409, ts=1789305575
 -- [STREAMS][flow=1][921] ssrc=799305432, pt=110, seq=20410, ts=1789305575
 -- [STREAMS][flow=0][57] ssrc=3245565062, pt=96, seq=23939, ts=1579991696
 -- [STREAMS][flow=0][59] ssrc=3245565062, pt=96, seq=23940, ts=1579992656
 -- [STREAMS][flow=1][1400] ssrc=799305432, pt=110, seq=20411, ts=1789308545
 -- [STREAMS][flow=1][1400] ssrc=799305432, pt=110, seq=20412, ts=1789308545
 -- [STREAMS][flow=1][1400] ssrc=799305432, pt=110, seq=20413, ts=1789308545
 -- [STREAMS][flow=1][649] ssrc=799305432, pt=110, seq=20414, ts=1789308545
 -- [STREAMS][flow=0][58] ssrc=3245565062, pt=96, seq=23941, ts=1579993616
 -- [STREAMS][flow=1][1400] ssrc=799305432, pt=110, seq=20415, ts=1789311515
 -- [STREAMS][flow=1][1400] ssrc=799305432, pt=110, seq=20416, ts=1789311515
 -- [STREAMS][flow=1][20] ssrc=799305432, pt=110, seq=20417, ts=1789311515
 -- [STREAMS][flow=0][58] ssrc=3245565062, pt=96, seq=23942, ts=1579994576
 -- [STREAMS][flow=0][58] ssrc=3245565062, pt=96, seq=23943, ts=1579995536
 -- [STREAMS][flow=0][58] ssrc=3245565062, pt=96, seq=23944, ts=1579996496
```

Involving Janus (and WebRTC!)


Janus
Home
WebTransport Server
RoQ Server
RoQ Client
Meetecho
Plugin Demo: RTP Over QUIC Client (via imquic)
Stop
Local
Remote
Write a DataChannel message
Terminal - lminiero@lminiero:~/Work/code/quic/imquic
File
Edit
View
Terminal
Tabs
Help

```
[roq-server/1]  -- [flow=1][1092] ssrc=3130418067, pt=120, seq=20, ts=50940
[roq-server/1]  -- [flow=0][74] ssrc=2714730680, pt=109, seq=44, ts=41280
[roq-server/1]  -- [flow=0][88] ssrc=2714730680, pt=109, seq=45, ts=42240
[roq-server/1]  -- [flow=0][77] ssrc=2714730680, pt=109, seq=46, ts=43200
[roq-server/1]  -- [flow=1][1014] ssrc=3130418067, pt=120, seq=21, ts=56880
[roq-server/1]  -- [flow=1][1014] ssrc=3130418067, pt=120, seq=22, ts=56880
[roq-server/1]  -- [flow=1][1014] ssrc=3130418067, pt=120, seq=23, ts=56880
[roq-server/1]  -- [flow=1][1014] ssrc=3130418067, pt=120, seq=24, ts=56880
[roq-server/1]  -- [flow=0][76] ssrc=2714730680, pt=109, seq=47, ts=44160
[roq-server/1]  -- [flow=0][77] ssrc=2714730680, pt=109, seq=48, ts=45120
[roq-server/1]  -- [flow=0][74] ssrc=2714730680, pt=109, seq=49, ts=46080
[roq-server/1]  -- [flow=0][74] ssrc=2714730680, pt=109, seq=50, ts=47040
[roq-server/1]  -- [flow=1][1141] ssrc=3130418067, pt=120, seq=25, ts=63000
[roq-server/1]  -- [flow=1][1142] ssrc=3130418067, pt=120, seq=26, ts=63000
[roq-server/1]  -- [flow=1][1142] ssrc=3130418067, pt=120, seq=27, ts=63000
[roq-server/1]  -- [flow=1][1142] ssrc=3130418067, pt=120, seq=28, ts=63000
[roq-server/1]  -- [flow=1][1142] ssrc=3130418067, pt=120, seq=29, ts=63000
[roq-server/1]  -- [flow=1][1142] ssrc=3130418067, pt=120, seq=30, ts=63000
[roq-server/1]  -- [flow=1][1142] ssrc=3130418067, pt=120, seq=31, ts=63000
[roq-server/1]  -- [flow=1][1142] ssrc=3130418067, pt=120, seq=32, ts=63000
[roq-server/1]  -- [flow=1][1142] ssrc=3130418067, pt=120, seq=33, ts=63000
[roq-server/1]  -- [flow=0][77] ssrc=2714730680, pt=109, seq=51, ts=48000
[roq-server/1]  -- [flow=0][78] ssrc=2714730680, pt=109, seq=52, ts=48960
```

Involving Janus (and WebRTC!)



Janus

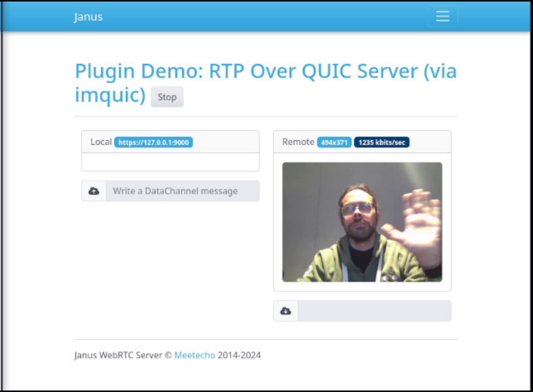
Plugin Demo: RTP Over QUIC Client (via imquic) Stop

Local

Remote

Write a DataChannel message

Janus WebRTC Server © Meetecho 2014-2024



Janus

Plugin Demo: RTP Over QUIC Server (via imquic) Stop

Local https://127.0.0.1:9000

Remote 494x371 1235 kbits/sec

Write a DataChannel message

Janus WebRTC Server © Meetecho 2014-2024

Prototyping MoQT

- Integrating MoQT (now from -11 to -16) in my test library itself
  - Leverages library core and events
  - Exposes MoQT-specific APIs to end user

[1] https://datatracker.ietf.org/doc/html/draft-afrind-moq-test/

Prototyping MoQT

- Integrating MoQT (now from -11 to -16) in my test library itself
  - Leverages library core and events
  - Exposes MoQT-specific APIs to end user
- Different demo applications for testing
  - **`imquic-moq-pub`**: basic MoQT publisher (same as Luke's **`moq-clock`**)
  - **`imquic-moq-sub`**: basic MoQT subscriber (different formats)
  - **`imquic-moq-relay`**: proof-of-concept relay
  - **`imquic-moq-test`**: a **`moq-test`**[1] implementation
  - **Janus plugin** (WIP): gatewaying of MoQT pub/sub to WebRTC (using LOC)

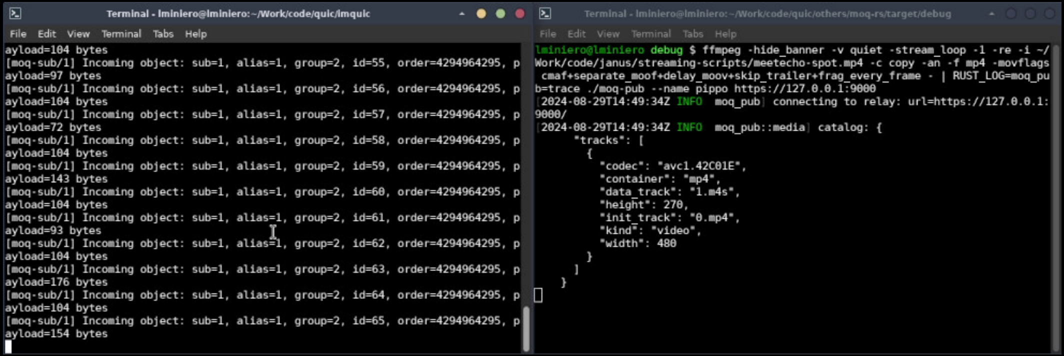
[1] https://datatracker.ietf.org/doc/html/draft-afrind-moq-test/

Prototyping MoQT

- Integrating MoQT (now from -11 to -16) in my test library itself
  - Leverages library core and events
  - Exposes MoQT-specific APIs to end user
- Different demo applications for testing
  - **`imquic-moq-pub`**: basic MoQT publisher (same as Luke's **`moq-clock`**)
  - **`imquic-moq-sub`**: basic MoQT subscriber (different formats)
  - **`imquic-moq-relay`**: proof-of-concept relay
  - **`imquic-moq-test`**: a **`moq-test`**[1] implementation
  - **Janus plugin** (WIP): gatewaying of MoQT pub/sub to WebRTC (using LOC)
- Many interop tests with different implementations
  - Regular tests during IETF Hackathon sessions
  - Active **#moq** channel on **quicdev** Slack

[1] https://datatracker.ietf.org/doc/html/draft-afrind-moq-test/

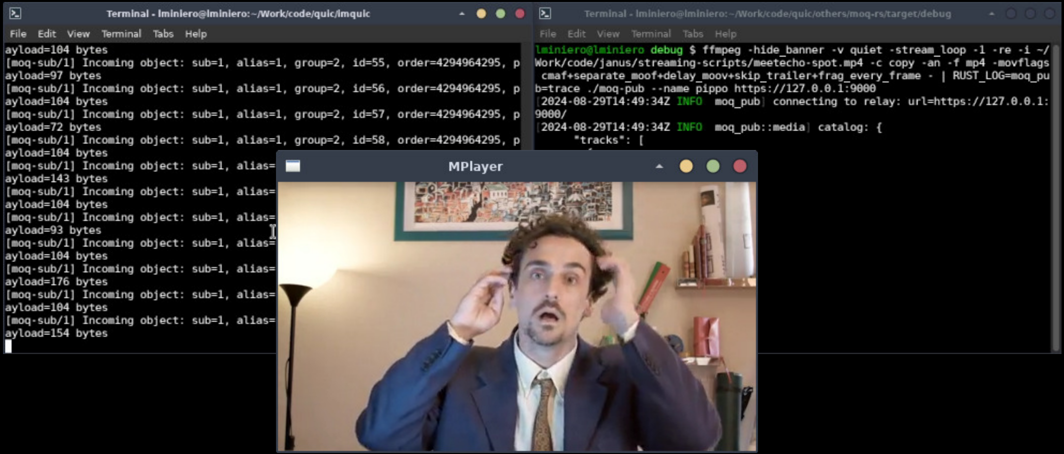
moq-rs + imquic


Terminal - lminiero@lminiero:~/Work/code/quic/imquic
File Edit View Terminal Tabs Help
Terminal - lminiero@lminiero:~/Work/code/quic/others/moq-rs/target/debug
File Edit View Terminal Tabs Help

```
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=55, order=4294964295, p
ayload=97 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=56, order=4294964295, p
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=57, order=4294964295, p
ayload=72 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=58, order=4294964295, p
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=59, order=4294964295, p
ayload=143 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=60, order=4294964295, p
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=61, order=4294964295, p
ayload=93 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=62, order=4294964295, p
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=63, order=4294964295, p
ayload=176 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=64, order=4294964295, p
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=65, order=4294964295, p
ayload=154 bytes
```

```
lminiero@lminiero debug $ ffmpeg -hide_banner -v quiet -stream_loop -1 -re -i ~/
Work/code/janus/streaming-scripts/meetecho-spot.mp4 -c copy -an -f mp4 -movflags
 cmaf+separate_moof+delay_moov+skip_trailer+frag_every_frame - | RUST_LOG=moq_pu
b=trace ./moq-pub --name pippo https://127.0.0.1:9000
[2024-08-29T14:49:34Z INFO  moq_pub] connecting to relay: url=https://127.0.0.1:
9000/
[2024-08-29T14:49:34Z INFO  moq_pub::media] catalog: {
      "tracks": [
        {
          "codec": "avc1.42C01E",
          "container": "mp4",
          "data_track": "1.m4s",
          "height": 270,
          "init_track": "0.mp4",
          "kind": "video",
          "width": 480
        }
      ]
    }
```

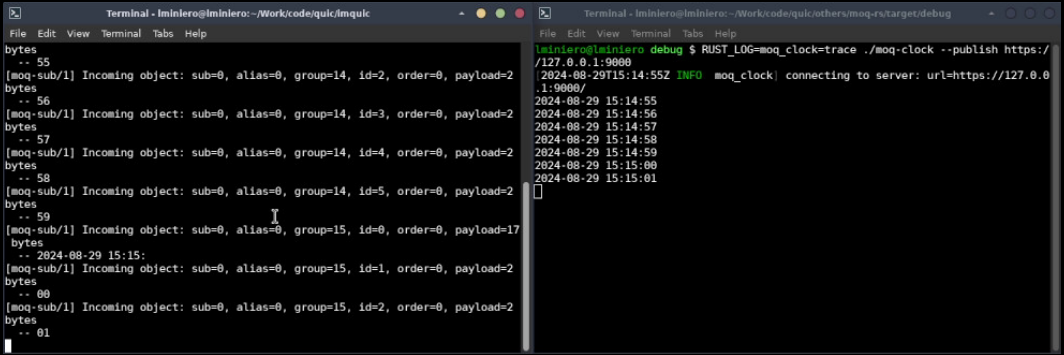
moq-rs + imquic


Terminal - lminiero@lminiero:~/Work/code/quic/imquic
File Edit View Terminal Tabs Help
Terminal - lminiero@lminiero:~/Work/code/quic/others/moq-rs/target/debug
File Edit View Terminal Tabs Help
MPlayer

```
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=55, order=4294964295, p
ayload=97 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=56, order=4294964295, p
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=57, order=4294964295, p
ayload=72 bytes
[moq-sub/1] Incoming object: sub=1, alias=1, group=2, id=58, order=4294964295, p
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=
ayload=143 bytes
[moq-sub/1] Incoming object: sub=1, alias=
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=
ayload=93 bytes
[moq-sub/1] Incoming object: sub=1, alias=
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=
ayload=176 bytes
[moq-sub/1] Incoming object: sub=1, alias=
ayload=104 bytes
[moq-sub/1] Incoming object: sub=1, alias=
ayload=154 bytes
```

```
lminiero@lminiero debug $ ffmpeg -hide_banner -v quiet -stream_loop -1 -re -i ~/
Work/code/janus/streaming-scripts/meetecho-spot.mp4 -c copy -an -f mp4 -movflags
 cmaf+separate_moof+delay_moov+skip_trailer+frag_every_frame - | RUST_LOG=moq_pu
b=trace ./moq-pub --name pippo https://127.0.0.1:9000
[2024-08-29T14:49:34Z INFO  moq_pub] connecting to relay: url=https://127.0.0.1:
9000/
[2024-08-29T14:49:34Z INFO  moq_pub::media] catalog: {
      "tracks": [
```

moq-rs + imquic



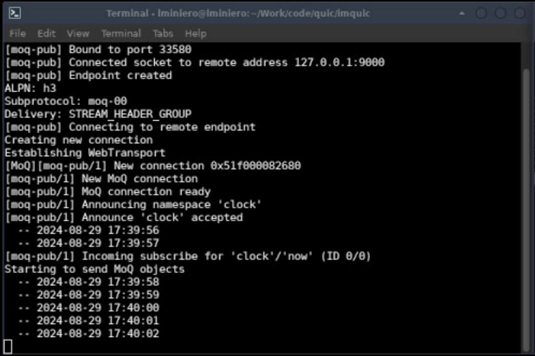
Terminal - lminiero@lminiero:~/Work/code/quic/imquic
File Edit View Terminal Tabs Help

```
bytes
  -- 55
[moq-sub/1] Incoming object: sub=0, alias=0, group=14, id=2, order=0, payload=2
bytes
  -- 56
[moq-sub/1] Incoming object: sub=0, alias=0, group=14, id=3, order=0, payload=2
bytes
  -- 57
[moq-sub/1] Incoming object: sub=0, alias=0, group=14, id=4, order=0, payload=2
bytes
  -- 58
[moq-sub/1] Incoming object: sub=0, alias=0, group=14, id=5, order=0, payload=2
bytes
  -- 59
[moq-sub/1] Incoming object: sub=0, alias=0, group=15, id=0, order=0, payload=17
 bytes
  -- 2024-08-29 15:15:
[moq-sub/1] Incoming object: sub=0, alias=0, group=15, id=1, order=0, payload=2
bytes
  -- 00
[moq-sub/1] Incoming object: sub=0, alias=0, group=15, id=2, order=0, payload=2
bytes
  -- 01
```

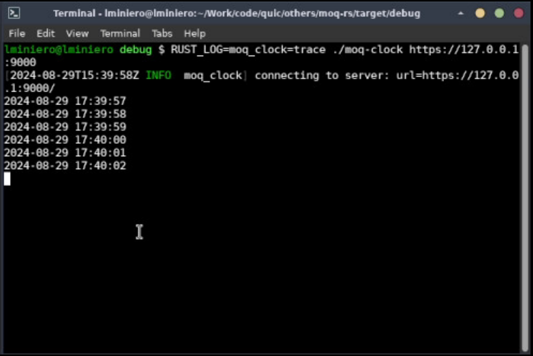
Terminal - lminiero@lminiero:~/Work/code/quic/others/moq-rs/target/debug
File Edit View Terminal Tabs Help

```
lminiero@lminiero debug $ RUST_LOG=moq_clock=trace ./moq-clock --publish https:/
/127.0.0.1:9000
[2024-08-29T15:14:55Z INFO  moq_clock] connecting to server: url=https://127.0.0
.1:9000/
2024-08-29 15:14:55
2024-08-29 15:14:56
2024-08-29 15:14:57
2024-08-29 15:14:58
2024-08-29 15:14:59
2024-08-29 15:15:00
2024-08-29 15:15:01
```

moq-rs + imquic

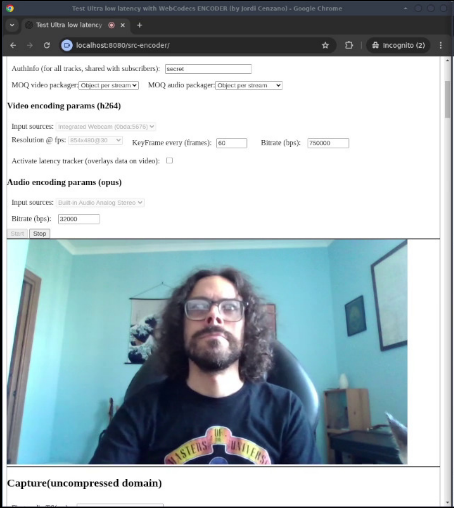

Terminal - lminiero@lminiero:~/Work/code/quic/imquic
File Edit View Terminal Tabs Help

```
[moq-pub] Bound to port 33580
[moq-pub] Connected socket to remote address 127.0.0.1:9000
[moq-pub] Endpoint created
ALPN: h3
Subprotocol: moq-00
Delivery: STREAM_HEADER_GROUP
[moq-pub] Connecting to remote endpoint
Creating new connection
Establishing WebTransport
[MoQ][moq-pub/1] New connection 0x51f000082680
[moq-pub/1] New MoQ connection
[moq-pub/1] MoQ connection ready
[moq-pub/1] Announcing namespace 'clock'
[moq-pub/1] Announce 'clock' accepted
  -- 2024-08-29 17:39:56
  -- 2024-08-29 17:39:57
[moq-pub/1] Incoming subscribe for 'clock'/'now' (ID 0/0)
Starting to send MoQ objects
  -- 2024-08-29 17:39:58
  -- 2024-08-29 17:39:59
  -- 2024-08-29 17:40:00
  -- 2024-08-29 17:40:01
  -- 2024-08-29 17:40:02
```


Terminal - lminiero@lminiero:~/Work/code/quic/others/moq-rs/target/debug
File Edit View Terminal Tabs Help

```
lminiero@lminiero debug $ RUST_LOG=moq_clock=trace ./moq-clock https://127.0.0.1
:9000
[2024-08-29T15:39:58Z INFO  moq_clock] connecting to server: url=https://127.0.0
.1:9000/
2024-08-29 17:39:57
2024-08-29 17:39:58
2024-08-29 17:39:59
2024-08-29 17:40:00
2024-08-29 17:40:01
2024-08-29 17:40:02
```

moxygen + moq-encoder-player (Meta)



Test Ultra low latency with WebCodecs ENCODER (by Jordi Cenzano) - Google Chrome

localhost:8080/src-encoder/

AuthInfo (for all tracks, shared with subscribers): secret

MOQ video packager: Object per stream   MOQ audio packager: Object per stream

Video encoding params (h264)

Input sources: Integrated Webcam (0bda:5676)

Resolution @ fps: 854x480@30   KeyFrame every (frames): 60   Bitrate (bps): 750000

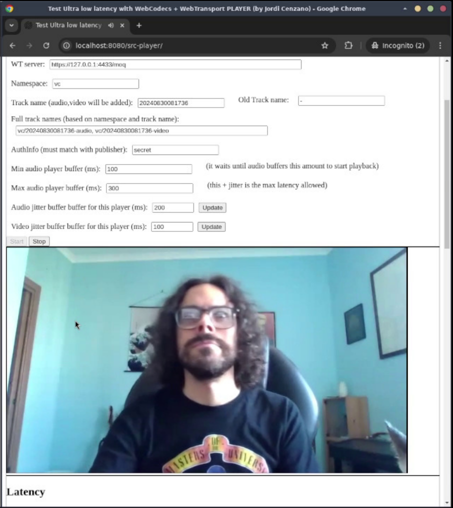
Activate latency tracker (overlays data on video): ☐

Audio encoding params (opus)

Input sources: Built-in Audio Analog Stereo

Bitrate (bps): 32000

Capture(uncompressed domain)



Test Ultra low latency with WebCodecs + WebTransport PLAYER (by Jordi Cenzano) - Google Chrome

localhost:8080/src-player/

WT server: https://127.0.0.1:4433/moq

Namespace: vc

Track name (audio,video will be added): 20240830081736   Old Track name: -

Full track names (based on namespace and track name): vc/20240830081736-audio, vc/20240830081736-video

AuthInfo (must match with publisher): secret

Min audio player buffer (ms): 100   (it waits until audio buffers this amount to start playback)

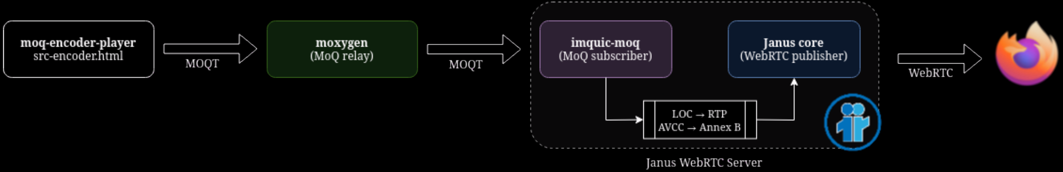
Max audio player buffer (ms): 300   (this + jitter is the max latency allowed)

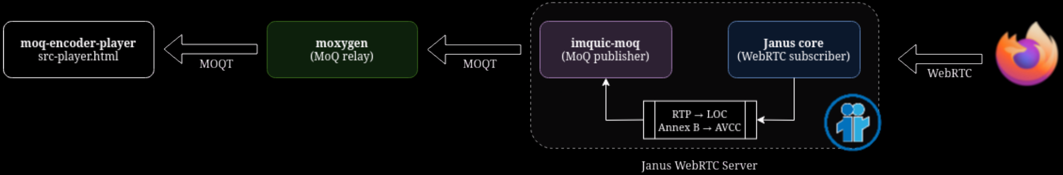
Audio jitter buffer buffer for this player (ms): 200

Video jitter buffer buffer for this player (ms): 100

Latency

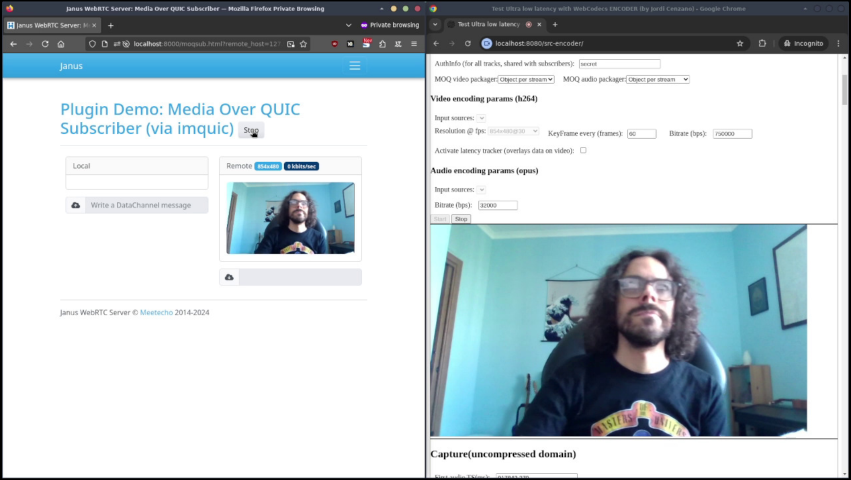
MoQT + WebRTC (via Janus/imquic)





https://www.meetecho.com/blog/moq-webrtc/

MoQT + WebRTC (via Janus/imquic)



Janus WebRTC Server: Media Over QUIC Subscriber — Mozilla Firefox Private Browsing

localhost:8000/moqsub.html?remote_host=127

Janus

Plugin Demo: Media Over QUIC Subscriber (via imquic) Stop

Local

Write a DataChannel message

Remote 854x480 0 kbits/sec

Janus WebRTC Server © Meetecho 2014-2024

Test Ultra low latency with WebCodecs ENCODER (by Jordi Cenzano) - Google Chrome

localhost:8080/src-encoder/

AuthInfo (for all tracks, shared with subscribers): secret

MOQ video packager: Object per stream MOQ audio packager: Object per stream

Video encoding params (h264)

Input sources:

Resolution @ fps: 854x480@30 KeyFrame every (frames): 60 Bitrate (bps): 750000

Activate latency tracker (overlays data on video): ☐

Audio encoding params (opus)

Input sources:

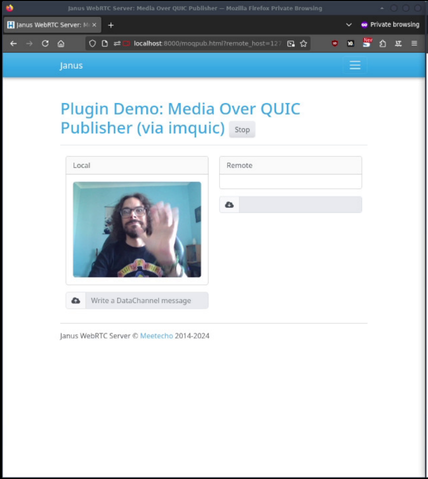
Bitrate (bps): 32000

Start Stop

Capture(uncompressed domain)

https://www.meetecho.com/blog/moq-webrtc/

MoQT + WebRTC (via Janus/imquic)

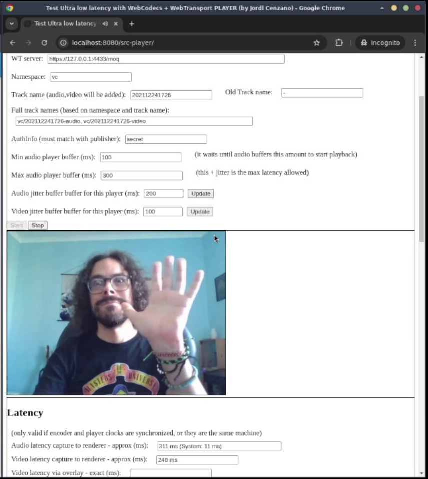


Janus WebRTC Server: Media Over QUIC Publisher — Mozilla Firefox Private Browsing

localhost:8000/moqpub.html?remote_host=127

Plugin Demo: Media Over QUIC Publisher (via imquic)

Janus WebRTC Server © Meetecho 2014-2024



Test Ultra low latency with WebCodecs + WebTransport PLAYER (by Jordi Cenzano) - Google Chrome

localhost:8080/src-player/

WT server: https://127.0.0.1:4433/moq

Namespace: vc

Track name (audio,video will be added): 202112241726 Old Track name: -

Full track names (based on namespace and track name): vc/202112241726-audio, vc/202112241726-video

AuthInfo (must match with publisher): secret

Min audio player buffer (ms): 100 (it waits until audio buffers this amount to start playback)

Max audio player buffer (ms): 300 (this + jitter is the max latency allowed)

Audio jitter buffer buffer for this player (ms): 200

Video jitter buffer buffer for this player (ms): 100

Latency

(only valid if encoder and player clocks are synchronized, or they are the same machine)

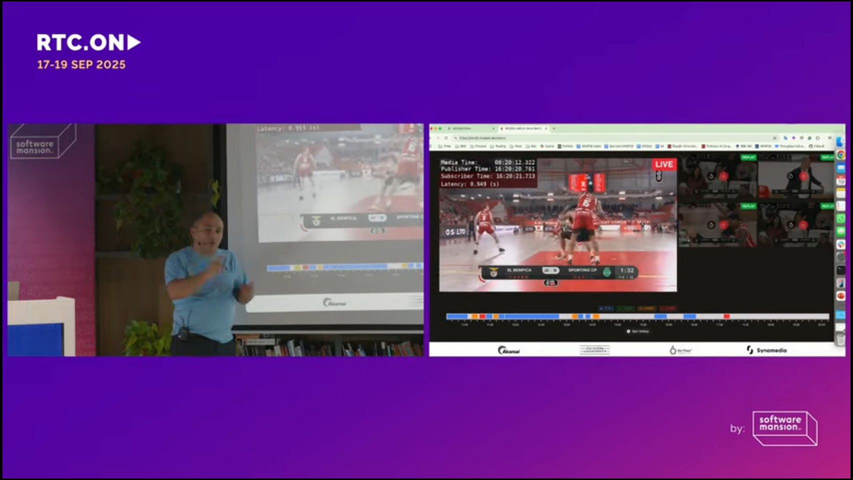
Audio latency capture to renderer - approx (ms): 311 ms (System: 11 ms)

Video latency capture to renderer - approx (ms): 248 ms

Video latency via overlay - exact (ms):

https://www.meetecho.com/blog/moq-webrtc/

A demo like this, but with SSC Napoli = 😎



```
Media Time:      00:20:12.322
Publisher Time:  16:20:20.761
Subscriber Time: 16:20:21.713
Latency: 0.949 (s)
```

https://www.youtube.com/watch?v=l4dwczI5Bis

Next steps

- A **LOT** still to do
  - QUIC stack itself needs some (maybe too many?) enhancements (e.g., CC)
  - Evaluating ngtcp2 for QUIC itself (imquic for high level API and RoQ/MoQ)
  - Release Janus integration (new plugin, RoQ forwarders, Streaming plugin)

Next steps

- A **LOT** still to do
    - QUIC stack itself needs some (maybe too many?) enhancements (e.g., CC)
    - Evaluating ngtcp2 for QUIC itself (imquic for high level API and RoQ/MoQ)
    - Release Janus integration (new plugin, RoQ forwarders, Streaming plugin)
- Keep up to date with MoQT (and maybe RoQ too)
    - Working group is *very* active
    - Specification changes often, thanks to feedback from implementations

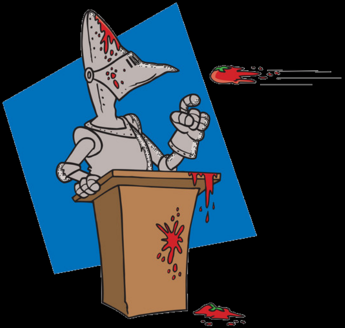
Next steps

- A **LOT** still to do
  - QUIC stack itself needs some (maybe too many?) enhancements (e.g., CC)
  - Evaluating ngtcp2 for QUIC itself (imquic for high level API and RoQ/MoQ)
  - Release Janus integration (new plugin, RoQ forwarders, Streaming plugin)
- Keep up to date with MoQT (and maybe RoQ too)
  - Working group is *very* active
  - Specification changes often, thanks to feedback from implementations
- Start working on media
  - Functional tests on transport are helpful, but not that fun
  - Time to play more with LOC, WARP, WebCodecs, and stuff like that!

Next steps

- A **LOT** still to do
    - QUIC stack itself needs some (maybe too many?) enhancements (e.g., CC)
    - Evaluating ngtcp2 for QUIC itself (imquic for high level API and RoQ/MoQ)
    - Release Janus integration (new plugin, RoQ forwarders, Streaming plugin)
- Keep up to date with MoQT (and maybe RoQ too)
    - Working group is *very* active
    - Specification changes often, thanks to feedback from implementations
- Start working on media
    - Functional tests on transport are helpful, but not that fun
    - Time to play more with LOC, WARP, WebCodecs, and stuff like that!
- Testing testing testing!
    - Interop (for everything, from QUIC to MoQT) helps, but is not enough
    - Play with it, have fun, and break things!

Thanks! Questions? Comments?



Contacts

- https://fosstodon.org/@lminiero
- https://bsky.app/profile/lminiero.it
- https://www.meetecho.com/blog/
- https://lminiero.it