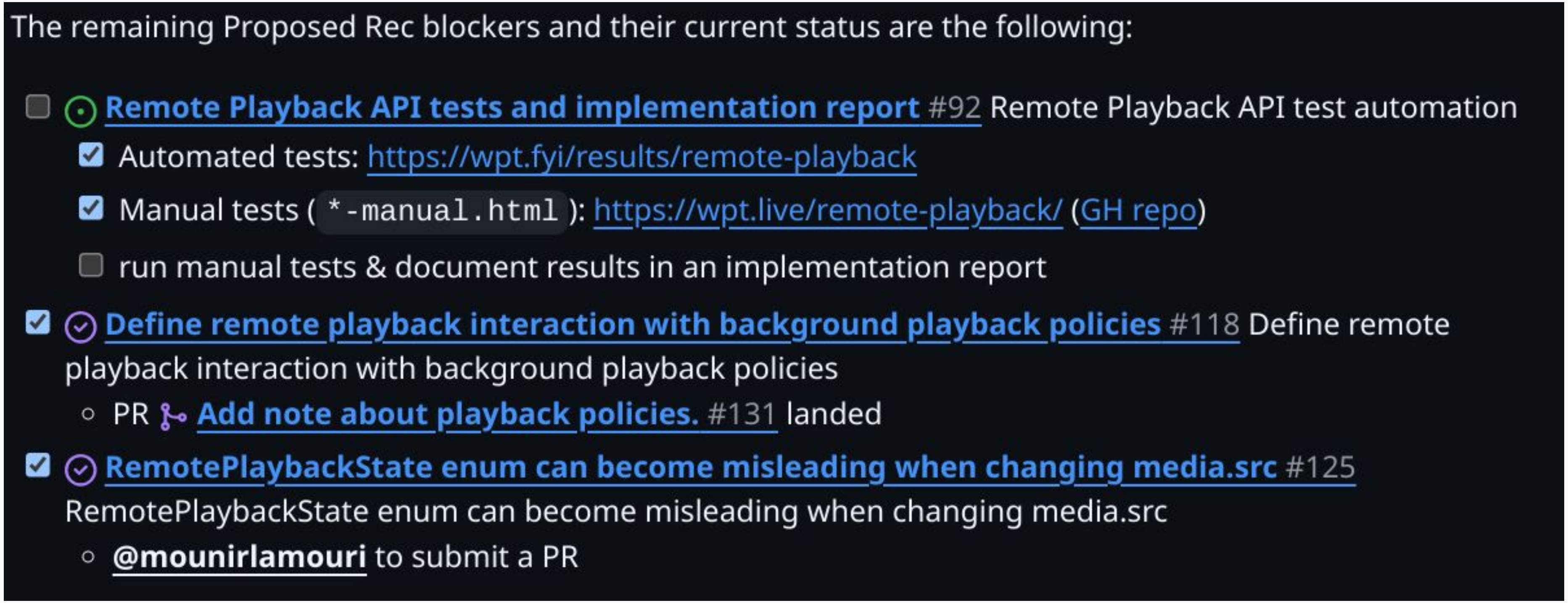
SSWG TPAC 2024

W3C Second Screen Working Group
Mark A. Foltz (mfoltz@google.com)
September 15, 2024

Remote Playback API

Remote Playback API - Proposed Recommendation

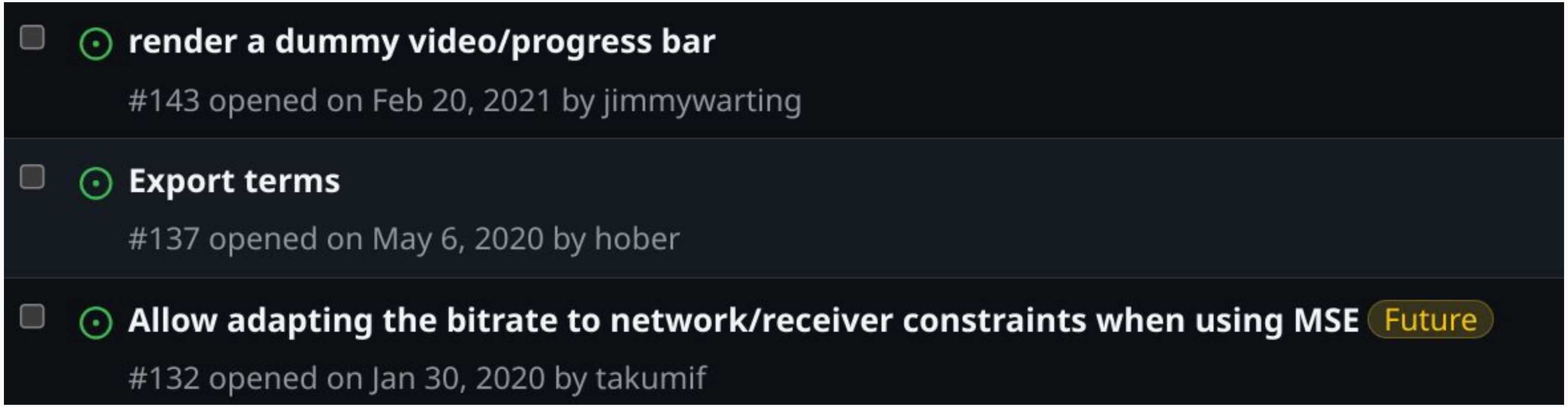
https://github.com/w3c/remote-playback/issues/130 on 1/2020



The remaining Proposed Rec blockers and their current status are the following:

- [ ] Remote Playback API tests and implementation report #92 Remote Playback API test automation
  - [x] Automated tests: https://wpt.fyi/results/remote-playback
  - [x] Manual tests (`*-manual.html`): https://wpt.live/remote-playback/ (GH repo)
  - [ ] run manual tests & document results in an implementation report
- [x] Define remote playback interaction with background playback policies #118 Define remote playback interaction with background playback policies
  - PR Add note about playback policies. #131 landed
- [x] RemotePlaybackState enum can become misleading when changing media.src #125 RemotePlaybackState enum can become misleading when changing media.src
  - @mounirlamouri to submit a PR

Remote Playback API - Proposed Recommendation


render a dummy video/progress bar
#143 opened on Feb 20, 2021 by jimmywarting

Export terms
#137 opened on May 6, 2020 by hober

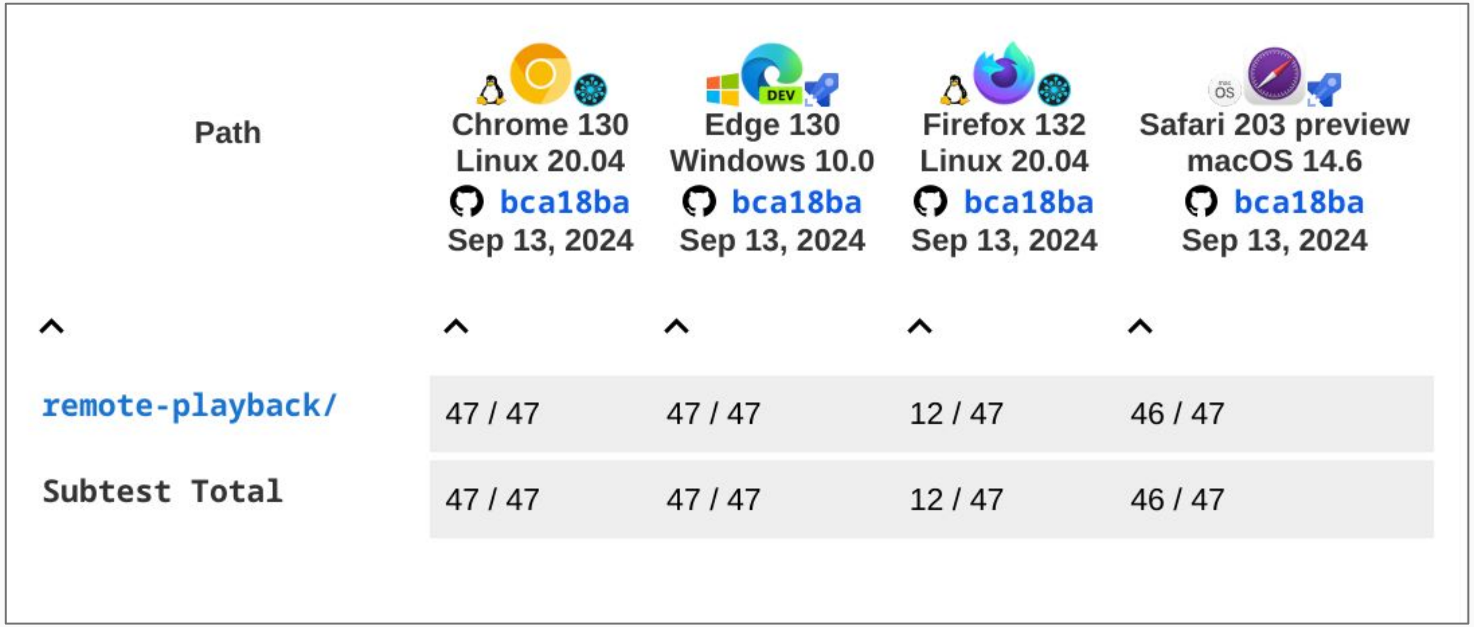
Allow adapting the bitrate to network/receiver constraints when using MSE Future
#132 opened on Jan 30, 2020 by takumif

#143 - Developer scenario; not directly related to Remote Playback API

#137 - Fixed by PR #151?

#132 - PR #248 landed in OSP. Marked as “Future”

Implementation Report - Automated Tests



Path	Chrome 130 Linux 20.04 bca18ba Sep 13, 2024	Edge 130 Windows 10.0 bca18ba Sep 13, 2024	Firefox 132 Linux 20.04 bca18ba Sep 13, 2024	Safari 203 preview macOS 14.6 bca18ba Sep 13, 2024
remote-playback/	47 / 47	47 / 47	12 / 47	46 / 47
Subtest Total	47 / 47	47 / 47	12 / 47	46 / 47

https://wpt.fyi/results/?label=master&label=experimental&aligned&q=remote-playback%2F

Implementation Report - Manual Tests

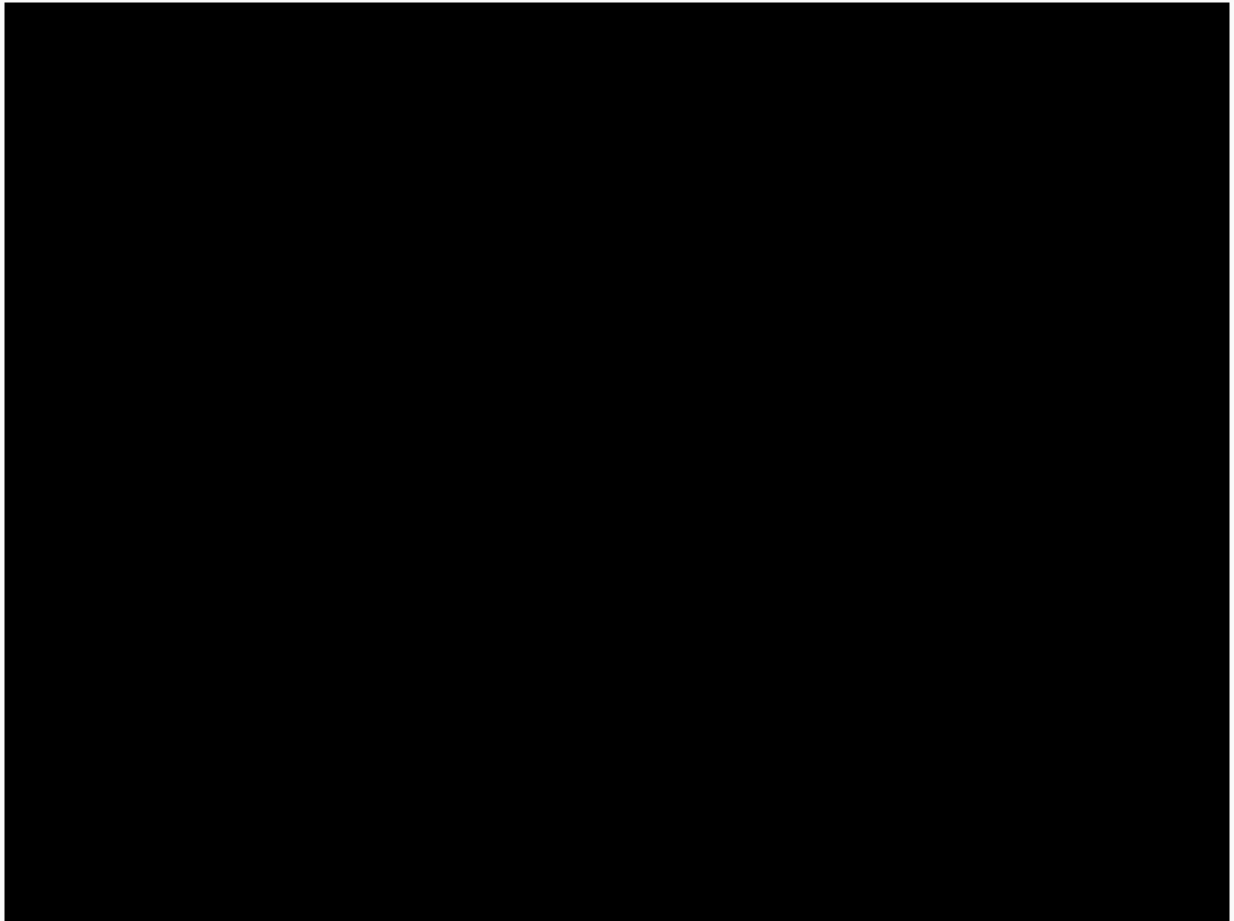
https://wpt.live/remote-playback/

- event-handlers-manual.html
- prompt-and-cancel-selection-manual.html
- prompt-and-select-device-manual.html
- prompt-and-watch-availability-no-device-manual.html
- prompt-and-watch-availability-with-device-manual.html
- remote-video-control-pausing-manual.html
- remote-video-control-seek-manual.html
- remote-video-playback-manual.html
- state-attribute-changes-when-selecting-device-manual.html

On Chrome Desktop:

- “Pick a Device” does nothing
- Or it’s skipped (failing test)

prompt-and-select-device-manual.html



Presentation API

Open Pull Requests

- PresentationRequest.getAvailability() could always return a new Promise
  - Addressed by PR #525
- Missing tasks in parallel steps in Presentation API
  - PR #524 updates spec to always use the “global task queue” to resolve Promises and fire events
  - Uses the “presentation task source”

Open Screen Protocol

Implementation Update (Open Screen Library)

- https://chromium.googlesource.com/openscreen/+/refs/heads/main/osp/
- ✅ QUIC implementation completely rewritten
- ✅ Client and server support for TLS 1.3 certificates
- ✅ Agent certificate generation
- ✅ Several updates to align protocol implementation with spec
- ✅ Many cleanups, simplifications and bug-fixes
- 🔁 Mutual authentication protocol (SPAKE2 + PSK)

👏 👏 👏Many thanks to Wei Wang (Intel)
for contributing nearly all of this work.

Open Screen Protocol 1.0 Spec status (updated 2024-09-15)

Number “v1” open issues 14 => 8 over 2024

Label/Category	Number	With PR
v1-spec	1	1
security-tracker	4	2.5
privacy-tracker	0	n/a
meta	3	n/a
Total	**8**	**3.5 (out of 5)**

Issues Affecting Implementation

- Certificate structure
  - TLS SNI requirement is incompatible with TLS SNI definition (#275)
  - Agent Certificate has a circular dependency on itself (#276)
- Add guidelines on how CBOR messages are mapped onto streams (#334)

#275: TLS SNI requirement

- Problem: The TLS handshake (ClientHello) has a Server Name Indication to help the server to choose a certificate.
- TLS implementations/conventions require the SNI to consist of LDH labels (Letters/Digits/Hyphens).
- The current spec uses `<fp>._openscreen_.udp._local`
- This is both circular (if used as CN) and invalid
- SNIs allow the same IP:port to serve multiple agent instances
  - Useful in restricted or proxied network environments
  - Or to allow one server to multiplex several agents, e.g. one per screen

#275: TLS SNI requirement

- Solution 1: Define a custom SNI for our use case
    - Requires specification by the IETF TLS working group
    - Implementations may not accept it anyway
- Solution 2: Use the DNS-SD Instance Name (e.g. BigTV.local)
    - Requires agents reachable by the same IP:port to have unique names
    - Ties SNI to a concept from DNS-SD
- Solution 3: Use <sn>.openscreen.udp
    - <sn> is a certificate serial number and should be unique per certificate per agent
    - <sn> is a 64-bit integer

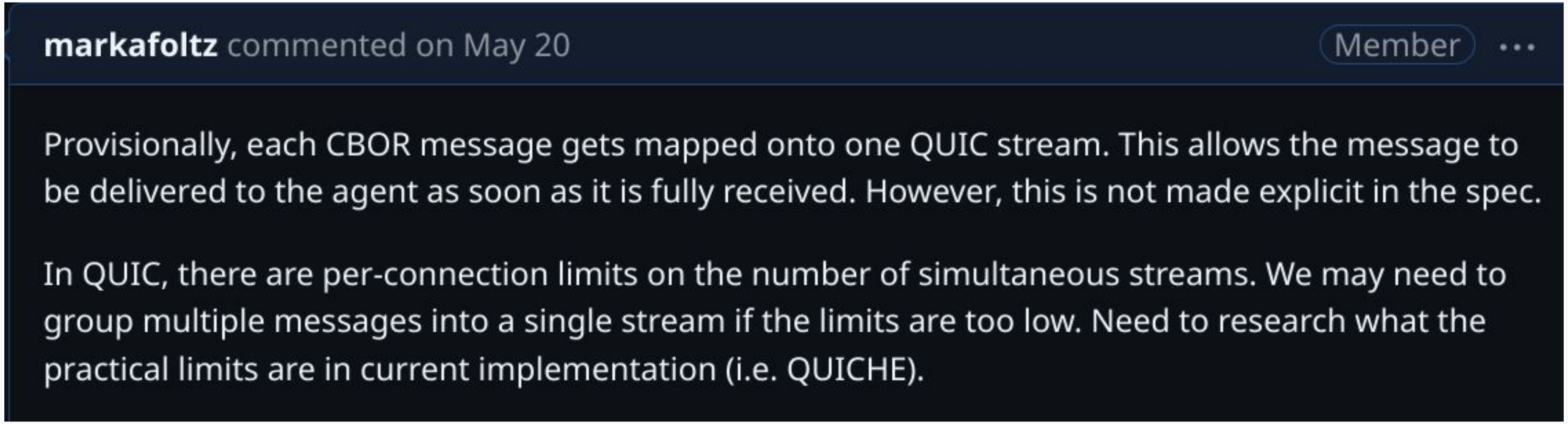
#275: TLS SNI requirement

- Proposal: Use Hex(<sn>).<Instance Name>
    - Example: 0afed352a.BigTV.local
- This does not affect the DNS-SD Instance Name
    - Do does not impact service browsing
- Does not require uniqueness of DNS-SD Instance Names
- Meets the rules for host names, so implementations should be OK
- If discovery is not through DNS-SD, then <Instance Name> is still required

#276 Agent Certificate circular dependency

- This has an open PR #332
- However, the PR also changed the SNI to match, requiring resolution of #275
- The PR can be updated and landed to resolve both #275 and #276
- Need to define what happens if the SNI does not match the CN in the certificate
- SN may need stronger guarantee of uniqueness as it is now the de facto “agent identifier”

#334 CBOR Message <=> QUIC Stream Mapping

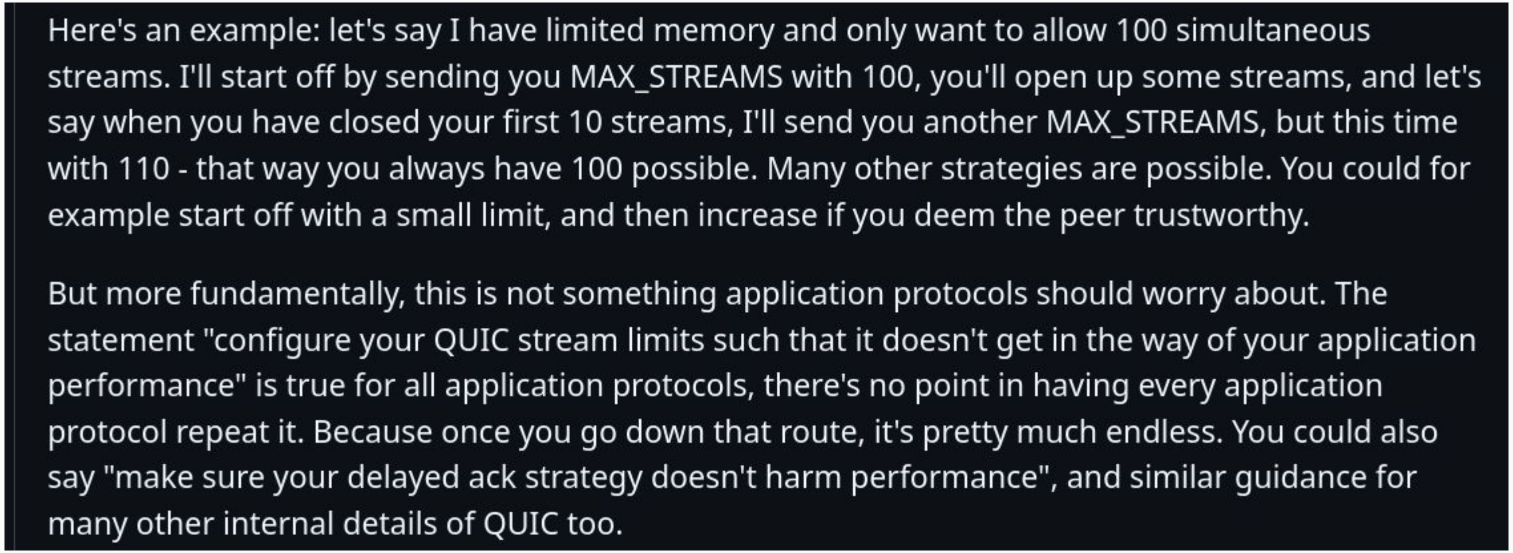

markafoltz commented on May 20

Provisionally, each CBOR message gets mapped onto one QUIC stream. This allows the message to be delivered to the agent as soon as it is fully received. However, this is not made explicit in the spec.

In QUIC, there are per-connection limits on the number of simultaneous streams. We may need to group multiple messages into a single stream if the limits are too low. Need to research what the practical limits are in current implementation (i.e. QUICHE).

#334 CBOR Message ⇐> QUIC Stream Mapping

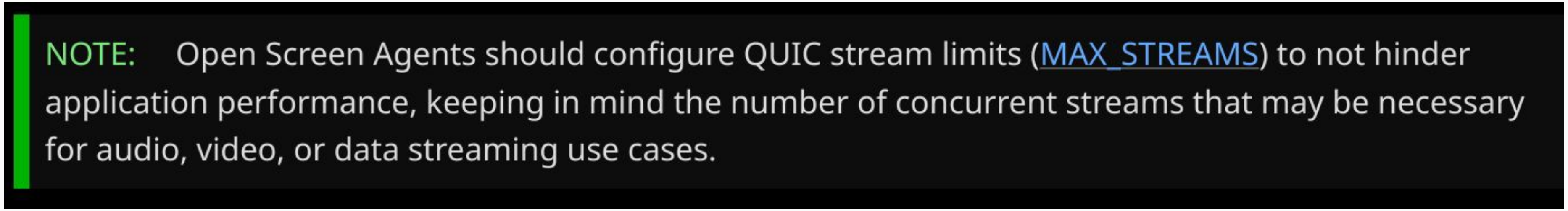
https://github.com/w3c/openscreenprotocol/pull/336


Here's an example: let's say I have limited memory and only want to allow 100 simultaneous streams. I'll start off by sending you MAX_STREAMS with 100, you'll open up some streams, and let's say when you have closed your first 10 streams, I'll send you another MAX_STREAMS, but this time with 110 - that way you always have 100 possible. Many other strategies are possible. You could for example start off with a small limit, and then increase if you deem the peer trustworthy.

But more fundamentally, this is not something application protocols should worry about. The statement "configure your QUIC stream limits such that it doesn't get in the way of your application performance" is true for all application protocols, there's no point in having every application protocol repeat it. Because once you go down that route, it's pretty much endless. You could also say "make sure your delayed ack strategy doesn't harm performance", and similar guidance for many other internal details of QUIC too.

#334 CBOR Message ⇐> QUIC Stream Mapping

Proposal: Update PR to add an implementation note to adjust QUIC parameters for good performance on their device.


NOTE: Open Screen Agents should configure QUIC stream limits (MAX_STREAMS) to not hinder application performance, keeping in mind the number of concurrent streams that may be necessary for audio, video, or data streaming use cases.

Splitting the OSP specification?

The OSP spec has two main parts:

1. Finding, connecting, authenticating between agents (**Network Protocol**)
2. Data exchange to support the relevant APIs (**Application Protocol**)

It would be simpler to write this as two different specs so we could support alternative mechanisms for #1.

https://github.com/w3c/openscreenprotocol/issues/321

Why split the OSP specification?

There are multiple ways to create an authenticated connection.

- QUIC + TLS + SPAKE-2 (what OSP requires now)
- Matter + Certificates + QUIC?
- WebTransport?
- RtcDataChannel, HTTP/3, etc.

It’s possible to exchange OSP messages with all of them.

WebTransport/RtcDataChannel/http

These are interesting because they are already part of the Web, so any browser that supports them would be able to exchange OSP messages.

The messaging part of OSP can be implemented in script; cbor-web is one JS parser for CBOR.

To support encoding/decoding media, you would need WebCodecs.

Possible split

	Network	Application
API and Non-functional Requirements	✅	✅
Discovery with mDNS	✅	
Transport and Metadata with QUIC	✅	✅
Message Delivery using CBOR and QUIC streams	✅	✅
Authentication	✅	
Presentation API		✅
Remote Playback		✅
Streaming		✅
Security and Privacy, etc.	✅	✅

Split Drafts - #344 and #347

	Network	Application
API and Non-functional Requirements	✅	✅
Discovery with mDNS	✅	
Transport and Metadata with QUIC	✅	✅
Message Delivery using CBOR and QUIC streams	✅	
Authentication	✅	
Presentation API		✅
Remote Playback		✅
Streaming		✅
Security and Privacy, etc.	✅	✅

Application Protocol - Transport Requirements (#342)

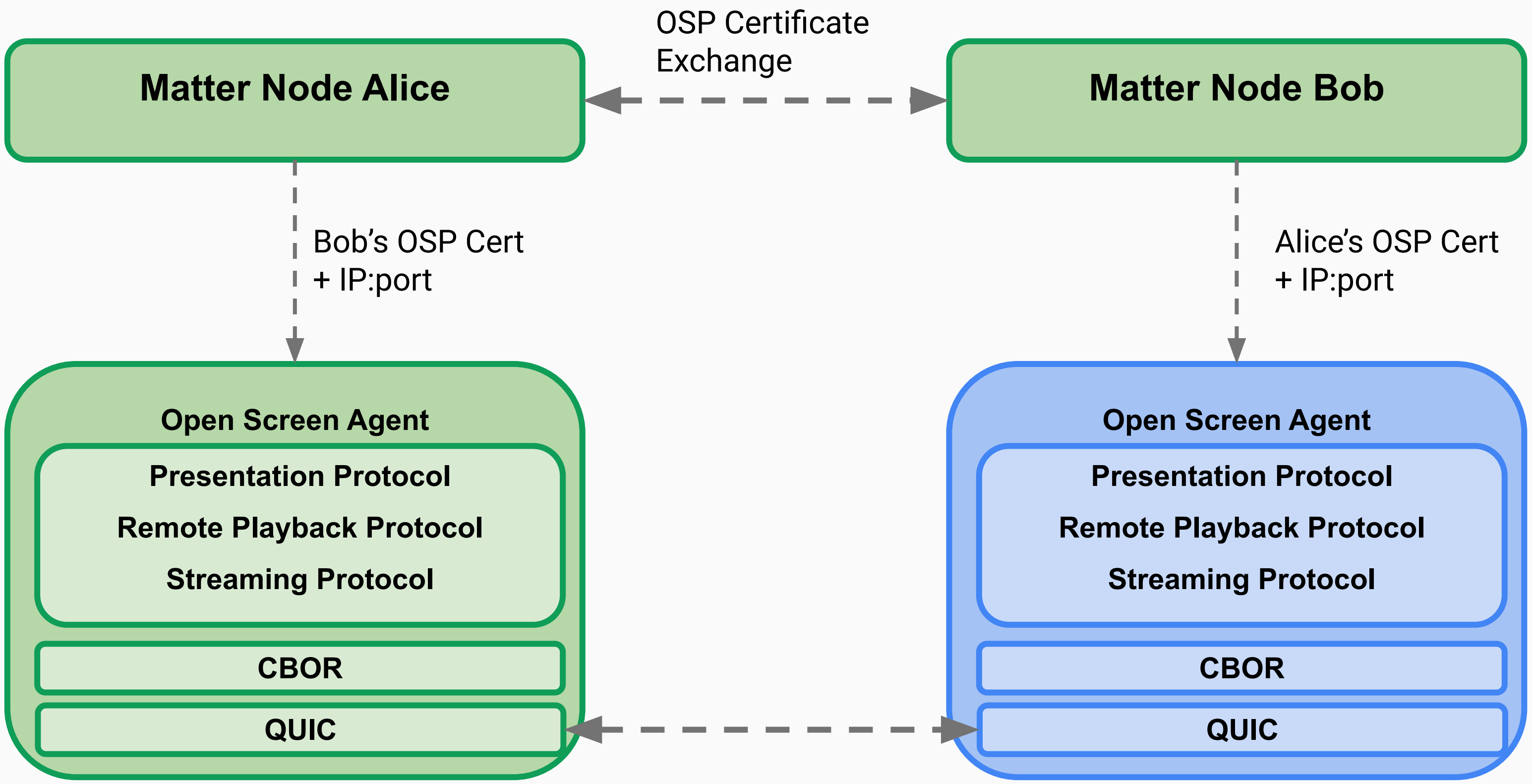
Need to describe an abstract transport for application messages

- Confidential and secure from MITM attacks
  - Authentication context must be reusable across connections
- Connection oriented (vs connectionless)
- Stream oriented (vs message oriented)
  - Existing spec embeds messages in QUIC streams
- Low packet latency (for e.g. lip sync or gaming)
- Low connection latency
- Keep-alive

QUIC vs Matter Transport

	QUIC	Matter
Confidential + Secure	✅	✅
Connection-oriented	✅	✅
Stream-oriented	✅	✖
Packet latency	✅	❓
Connection latency	✅	❓
Keep-alive	✖	❓

Mapping onto Matter (bootstrapping)



Recommendations for SSWG (Updated)

1. 🔁 Land PRs for security-tracker issues in OSP.
2. 🔁 Land PRs for v1-spec issues.
3. ✅ FInish "spec split" if supported by group and "freeze" previous document.
4. Prototype application protocol using existing Web APIs for validation. (Discovery mechanism is lacking.)
5. Potentially bring network protocol to IETF Dispatch.

End