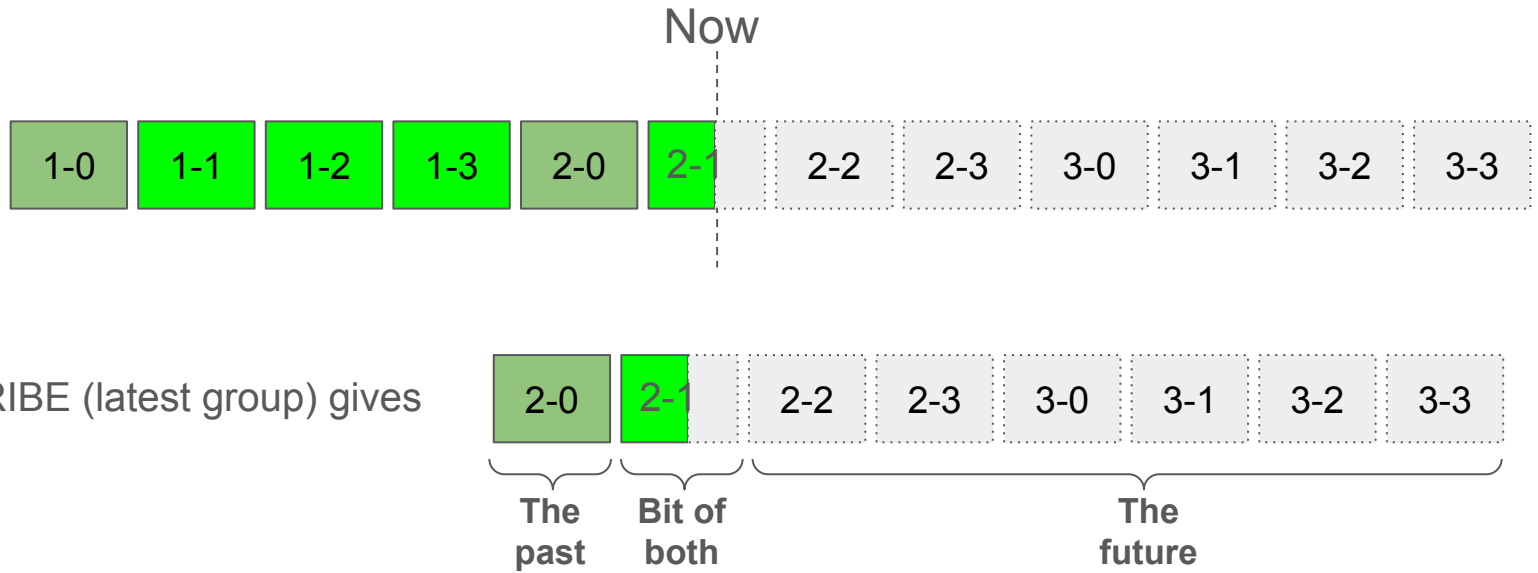
JOIN API proposal

Will Law - Akamai
Mike English - Norsk
Gwendal Simon - Synamedia

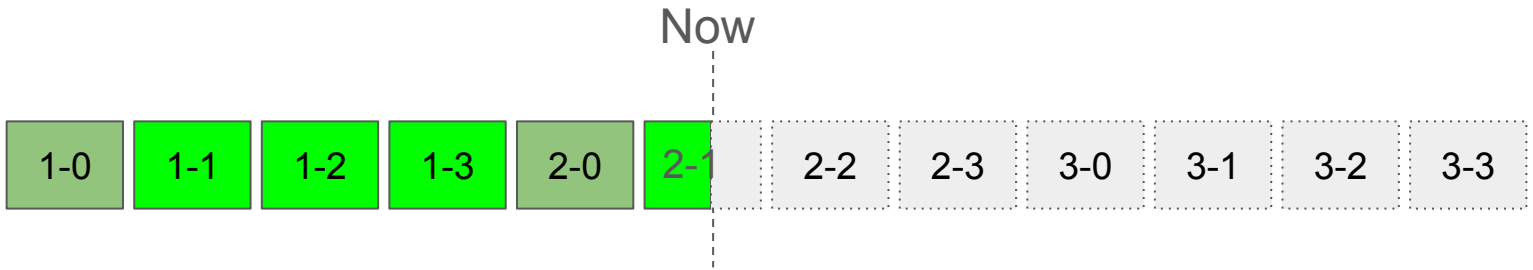
IETF #121 moq - Dublin Nov 2024

The problem with the current definition of SUBSCRIBE

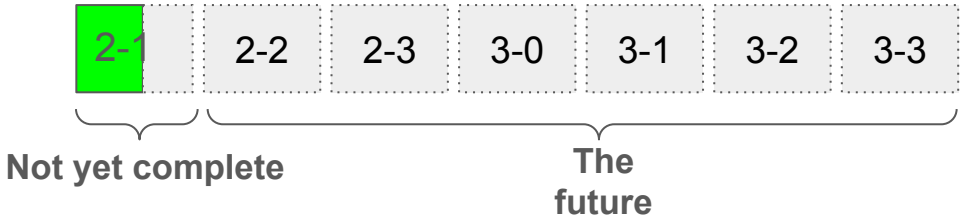


Subscribe arbitrarily allows a little bit of the past (cached objects) to be present in the response. If one group, why not two? Group ID 1-0 is no different to 2-0 yet it is excluded from the subscription.

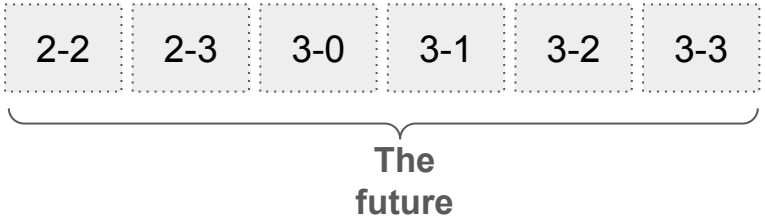
Redefine SUBSCRIBE to be future objects only



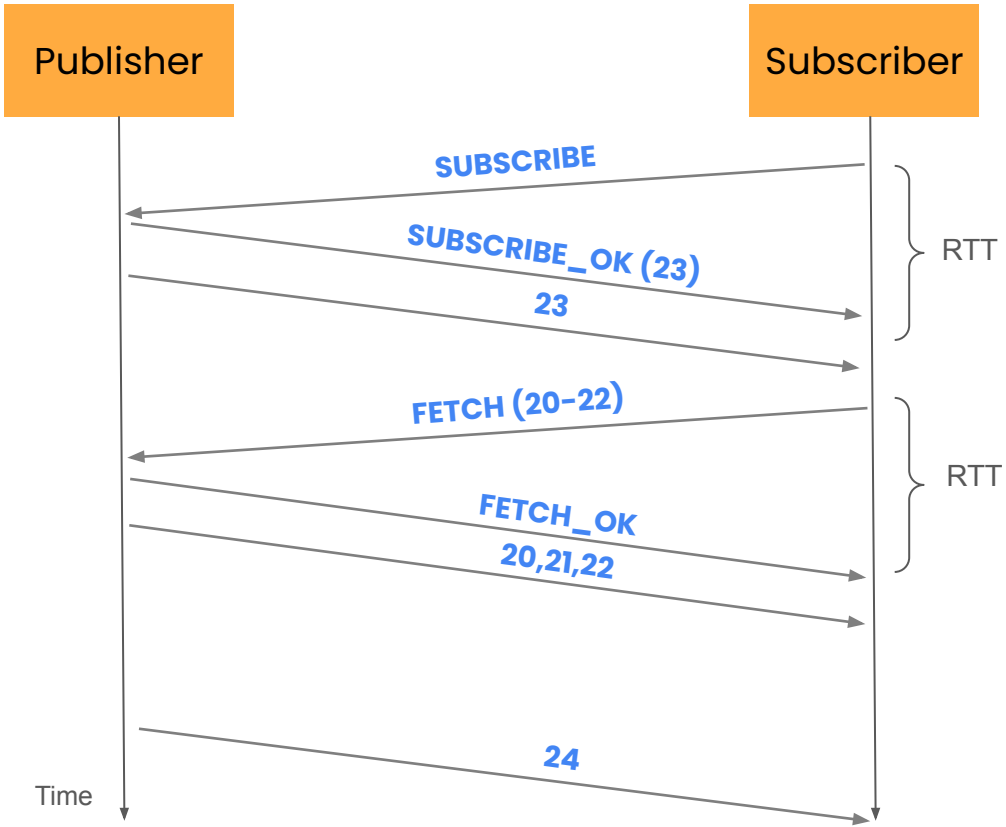
SUBSCRIBE () option 1 gives



SUBSCRIBE () option 2 gives



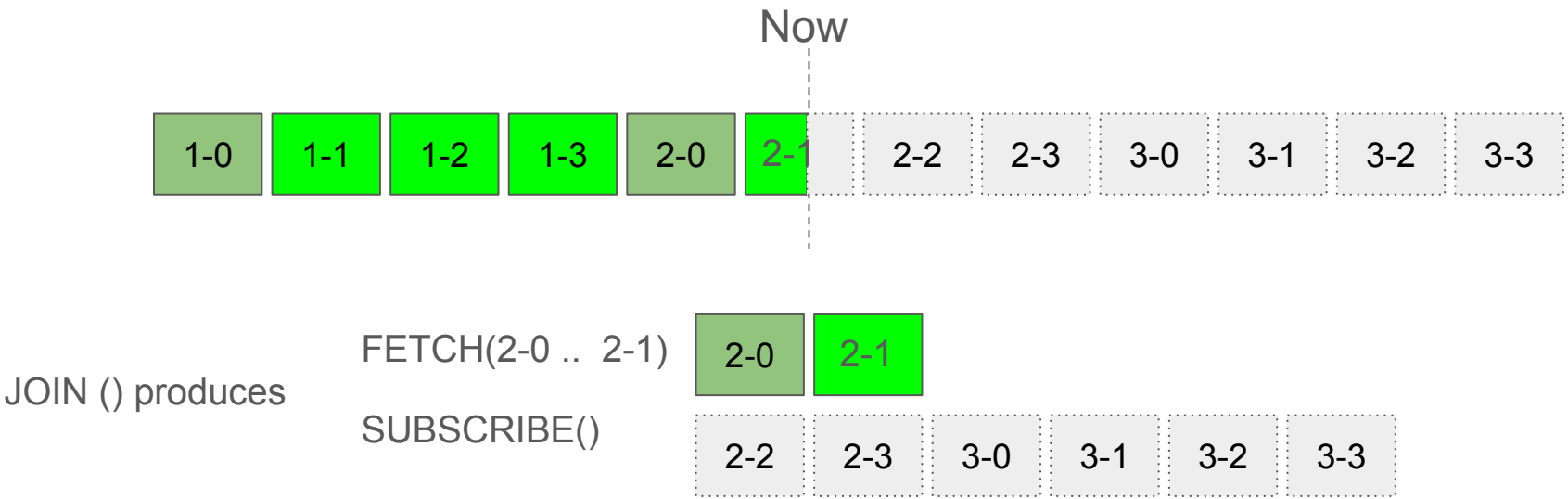
Joining a live stream using the existing APIs



At least 2 x RTT before we get our first bytes to start.

API proposal #1 - we create a new JOIN() message

- JOIN() automates the operation of joining a live stream starting at a group boundary under the condition that SUBSCRIBE only returns future objects.
- It stitches together a FETCH and SUBSCRIBE to avoid race-conditions, gaps and one RTT.



JOIN() API

JOIN (`ID, trackNameSpace, trackName, groups-behind-live, delivery-timeout, authInfo, priorityFetch, prioritySubscribe`)

`ID`: an identifier for the JOIN.This will be used as the FETCH and SUBSCRIBE ID.

`trackNameSpace` : the track name space

`trackName` : the track name

`groups-behind-live` : an integer specifying how many groups behind the live edge to begin the FETCH. 0 indicates the current group and is the default. N indicates (latest group - N).

`delivery-timeout` : the duration in milliseconds over which the relay SHOULD continue to attempt forwarding subscription Objects after they have been received.

`authInfo` : a string to be used as an authToken for both the FETCH and SUBSCRIBE actions.

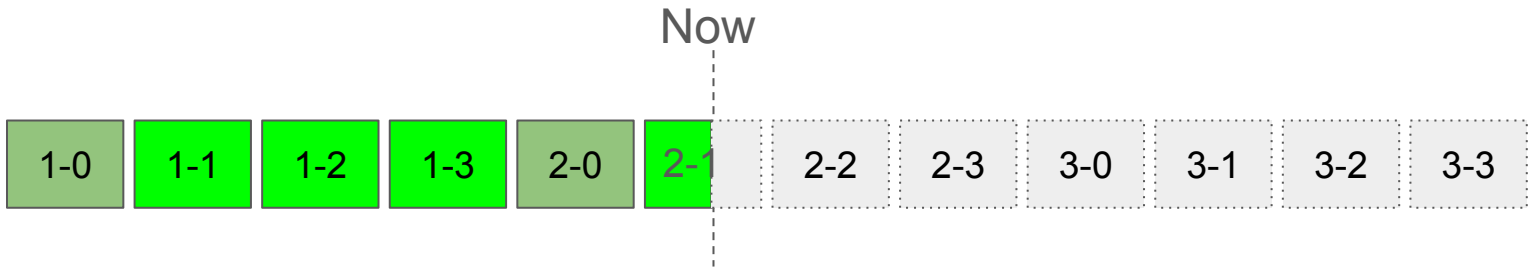
`priorityFetch` : the priority value to be assigned to the underlying FETCH

`prioritySubscribe` : the priority value to be assigned to the underlying SUBSCRIBE

FETCH_OK (..)

SUBSCRIBE_OK (..)

JOIN() with differing start values



JOIN (123, 0) produces

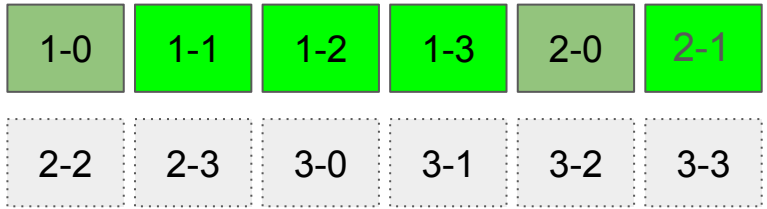
FETCH(2-0 .. 2-1)

SUBSCRIBE()

JOIN (123, 1) produces

FETCH(1-0 .. 2-1)

SUBSCRIBE()



API proposal #2 - update FETCH() to accept an existing subscription ID

Allow FETCH to reference an existing subscription ID and supply a group-offset as a substitute for the start and end properties. This gives us identical behavior to the prior API without requiring a new API surface.

```
FETCH Message {
  Type (i) = 0x16,
  Length (i),
  Subscribe ID (i),
  Track Namespace (tuple),
  Track Name Length (i),
  Track Name (..),
  Subscriber Priority (8),
  Group Order (8),
  StartGroup (i),
  StartObject (i),
  EndGroup (i),
  EndObject (i),
  Number of Parameters (i),
  Parameters (..) ...
}
```

*Parameters in red can be inferred by the existing subscription ID

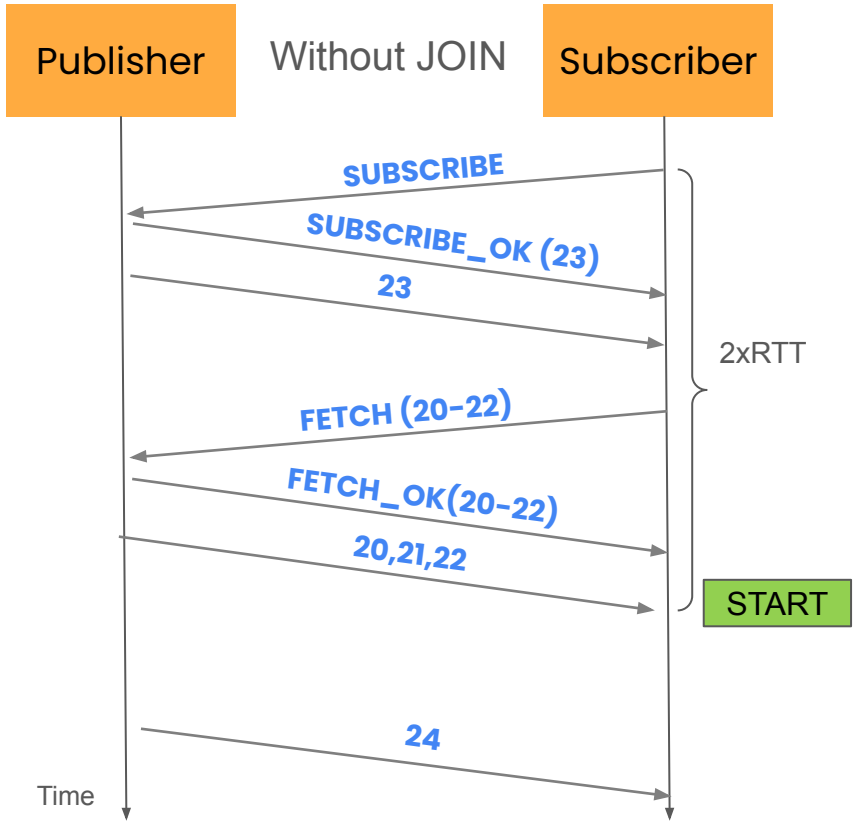
```
FETCH Message {
  Type (i) = 0x16,
  Length (i),
  Subscriber Priority (8),
  SubscriptionID-to-join
  Start-group-offset-from-live
}
```

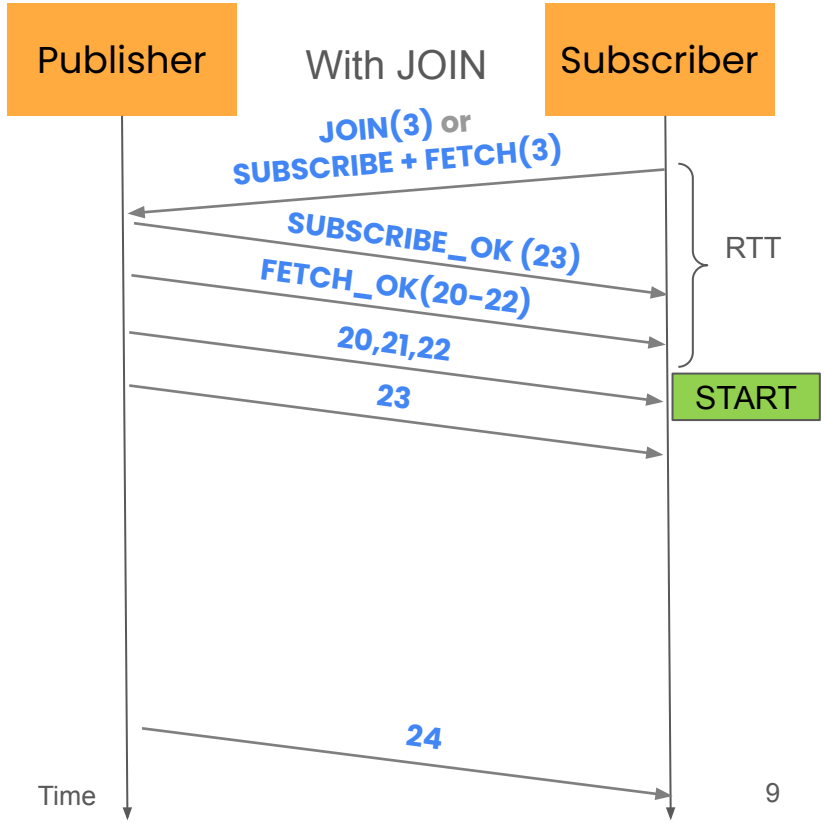
FETCH with new JOIN option

This FETCH can be sent immediately after a subscribe, without waiting for the SUSBCRIBE_OK. For example:

SUBSCRIBE(123, “example.com”,”myvideo”,2)
FETCH(123, 2,1)

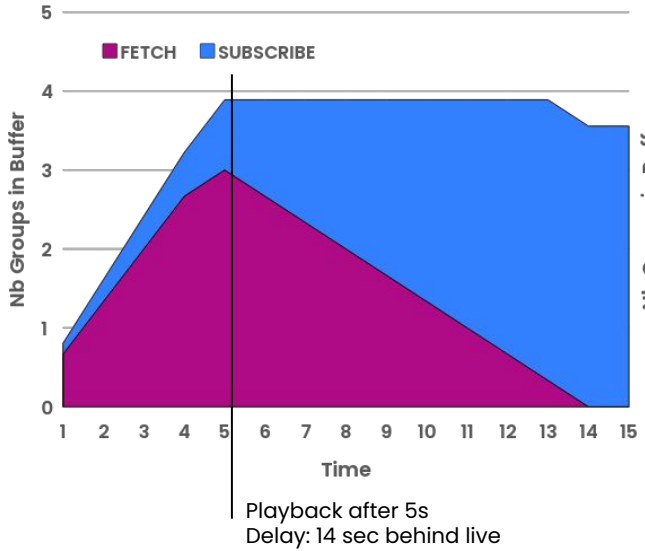
Benefits of JOIN - reduces start time by 1 x RTT



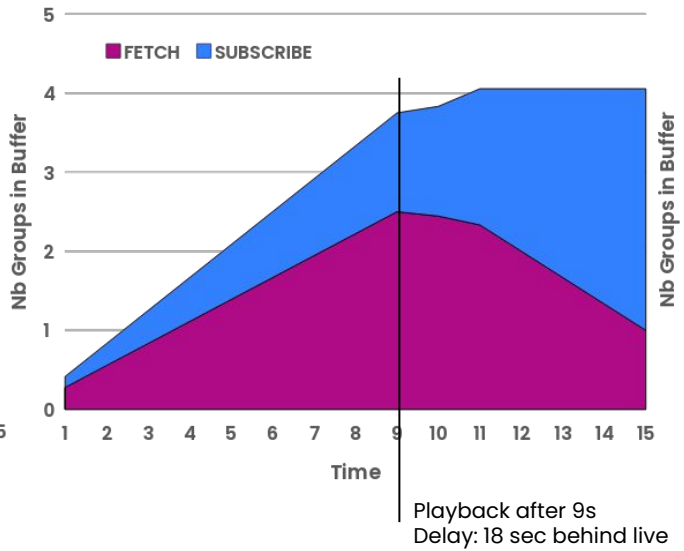


Many Flavors of FETCH & SUBSCRIBE

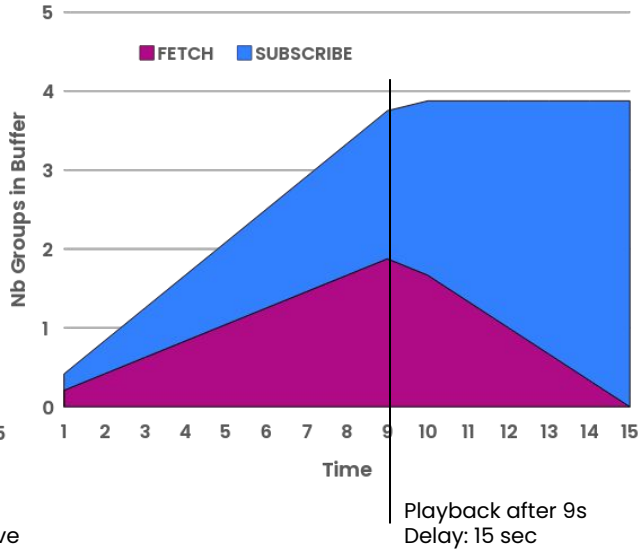
FETCH 3 Groups HD Priority 1



FETCH 3 Groups UHD Priority 1



FETCH 2 Groups UHD Priority 2



Startup when 4 groups in buffer
SUBSCRIBE UHD Priority 2
BW 15 Mbps – HD 5 Mbps – UHD 12 Mbps
Group length 3 seconds