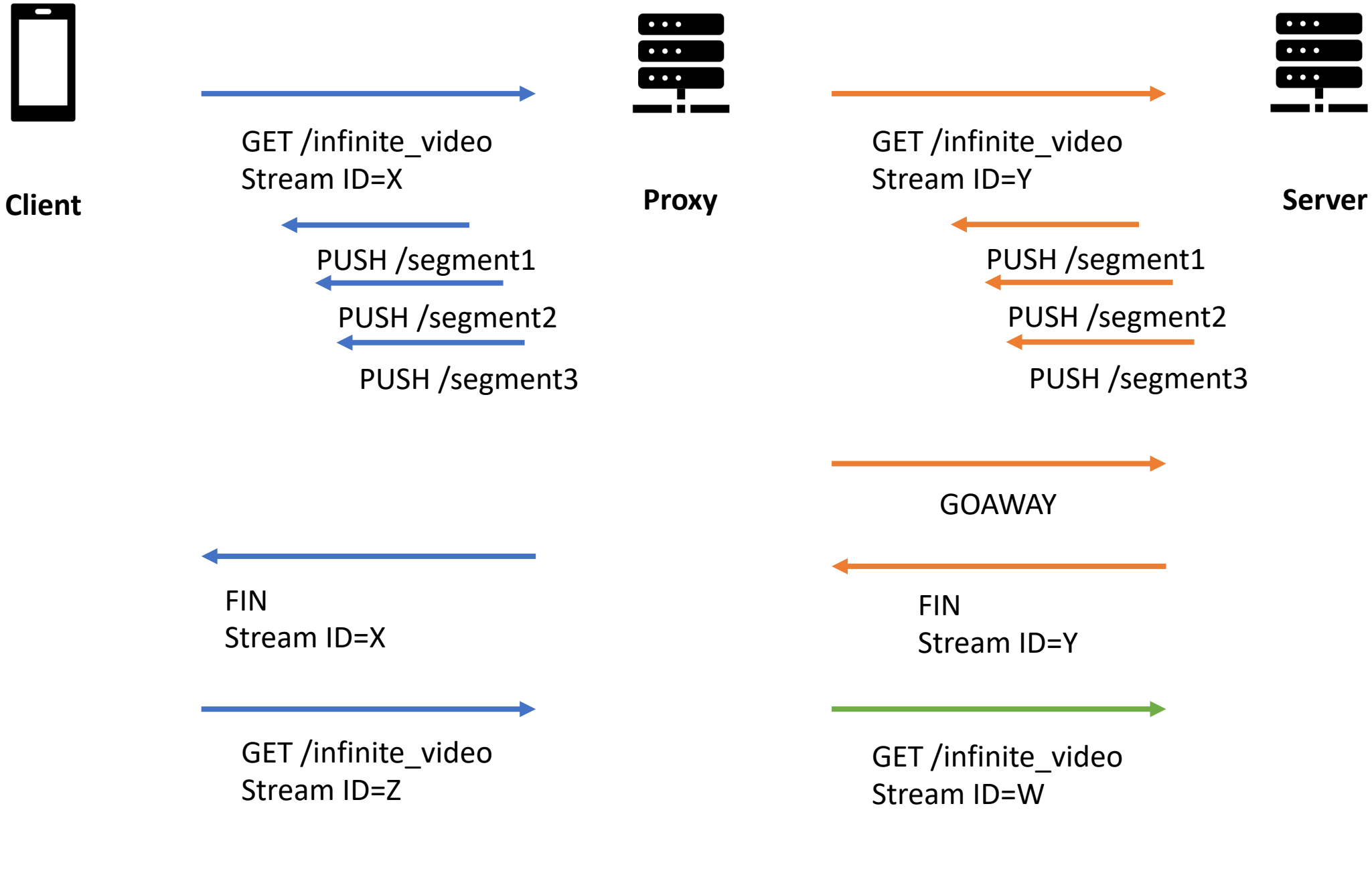
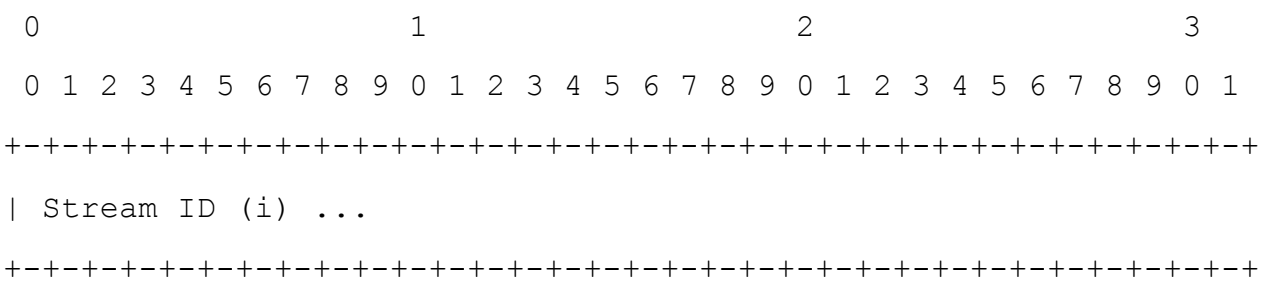
#2632 Do clients need GOAWAY?

Since clients drive request generation, clients perform a connection shutdown by not sending additional requests on the connection; responses and pushed responses associated to previous requests will continue to completion.

- What about indefinite or infinite resources? For example, a request for a live video feed.
- Not sending additional requests is a passive signal
- GOAWAY is an explicit signal that can trigger additional server logic
- Client initiated GOAWAY was part of H2 – why are we removing it?


Client
Proxy
Server
GET /infinite_video
Stream ID=X
PUSH /segment1
PUSH /segment2
PUSH /segment3
GET /infinite_video
Stream ID=Y
PUSH /segment1
PUSH /segment2
PUSH /segment3
GOAWAY
FIN
Stream ID=X
FIN
Stream ID=Y
GET /infinite_video
Stream ID=Z
GET /infinite_video
Stream ID=W

What belongs in a client initiated GOAWAY?



- Server sends QUIC Stream ID for a client-initiated bidirectional stream
- This effectively revokes previously granted stream (”request”) credits
- Options
  1. Nothing
  2. Unidirectional Stream ID
  3. Push ID
  4. Bidirectional AND Unidirectional Stream ID

Option 1: Nothing

- GOAWAY is just an explicit signal that the client wants the connection to go away

- Pros:
  - At least there's still a GOAWAY signal

- Cons:
  - Asymmetric parsing logic between client and server
  - Different semantics from H2

Option 2:
Server Initiated Unidirectional Stream ID

- Any new unidirectional streams would be rejected

- Pros:
  - Parsing is symmetric (both client and server GOAWAYs contain 1 varint)
  - Maintains H2 semantics
  - May be reusable by extensions

- Cons:
  - Maybe H2 got the semantics wrong and we want to correct them?

Option 3:
Push ID

- Any new PUSH_PROMISE or push stream with a higher ID would be cancelled

- Pros:
  - Parsing is symmetric
  - Maintains H2 semantics

- Cons:
  - Extensions that define new unidirectional stream types must define their own GOAWAY

Option 4:
Unidirectional and Bidirectional Stream ID

- This is probably what GOAWAY would look like if it were in the transport
- The server's GOAWAY would also include both Stream ID types

- Pros:
  - Fully symmetric
  - Provides functionality that can be leveraged by extensions
  - Maybe absorbable by the transport in v2?

- Cons:
  - Includes information that is not defined in H3 (Client Uni, Server Bidi)