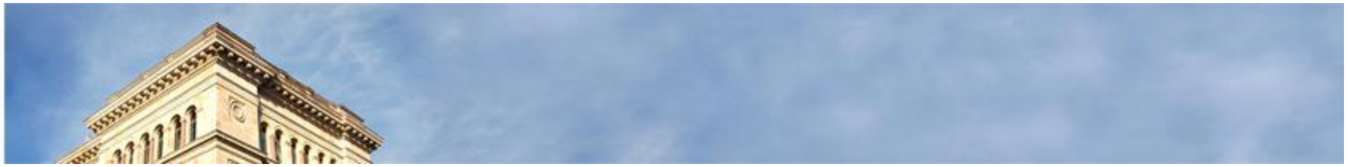
MCQUIC – A Multicast extension for QUIC

Max Franke, Jake Holland, Stefan Schmid

25.10.24

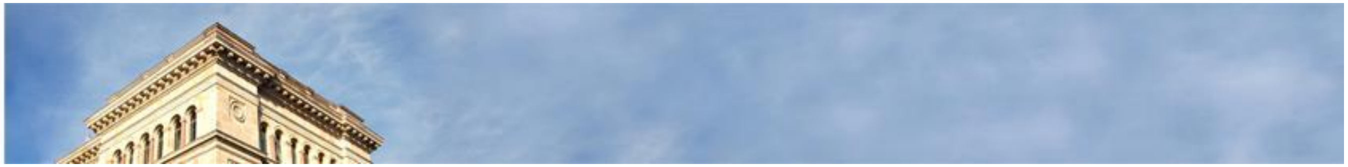


Outline

- Motivation
- Background
  - Multicast
  - QUIC
- MCQUIC Overview
- Design features
- Problems solved

Motivation - Unicast delivery is inefficient

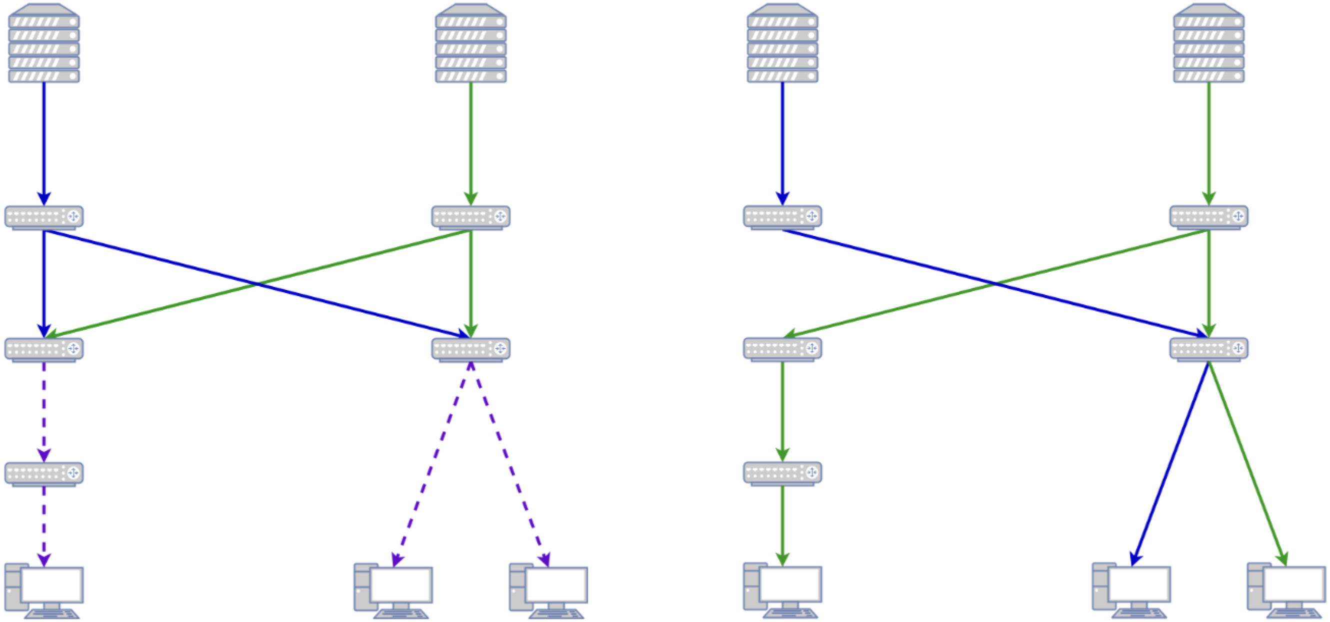
- Large mass events such as the world cup or NFL draw hundreds of millions of viewers
- Only feasible because large parts of audience watch over TV
- More and more people stream over web -> unicast delivery
- E.g. Akamais entire capacity would only be enough for ~2% of EURO final viewership
- Everyone receives same packets, so why send them multiple times? -> multicast



(IP) Multicast - Overview

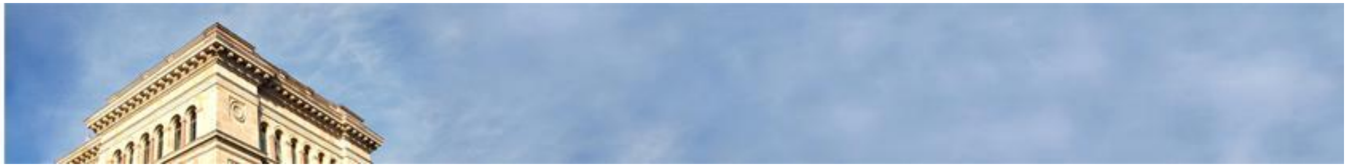
- Senders send to multicast group
- Clients join multicast group, usually using IGMP or MLD
- Network infrastructure takes care of establishing multicast tree and duplicating packets
- Each packet traverses each network segment only once
- Many receivers receive the packet
- Different solutions for routing, traditionally PIM; recently BIER
- Two major flavors, ASM and SSM; only concerned with SSM in this presentation

(IP) Multicast - Overview



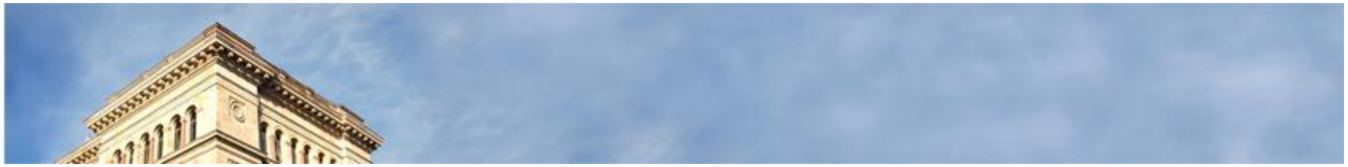
(a) ASM

(b) SSM



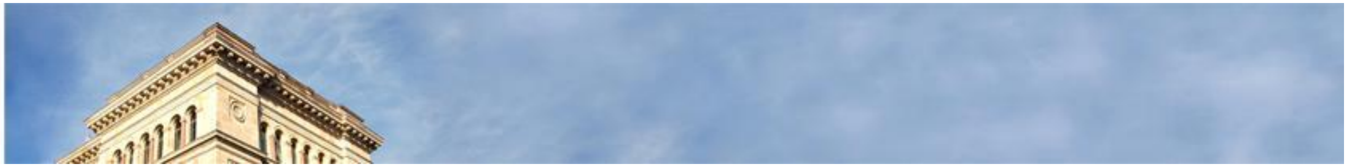
The problem(s) with multicast

- Traditional multicast deployments have several issues
- Lack of security -> packets easily injected by third parties
- Lack of privacy -> no encryption
- Lack of fallback -> most networks don’t support multicast at all
- Easily attackable -> flood router tables with state
- UDP based -> not viable path to implement in browsers



QUIC

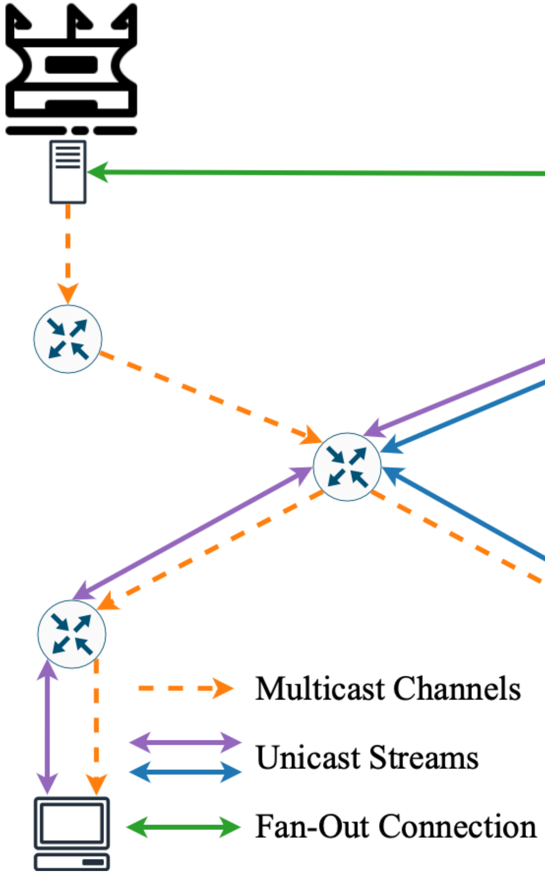
- QUIC is a UDP based reliable transport protocol, provides similar guarantees as TCP
- Specified in RFC 9000 (among others); with upgradeability in mind
- Includes TLS for encryption
- Uses variable bit-length encoding for most header fields for efficiency
- Uses UDP packets allowing easier implementations and deployment
  - Prevents compatibility issues with middleware (routers, switches etc.) that stifled SCTP
  - Can be implemented entirely in user space
- Along with QUIC, new version of HTTP (HTTP/3) was developed and introduced
- QUIC quickly gained support by most big vendors and developers

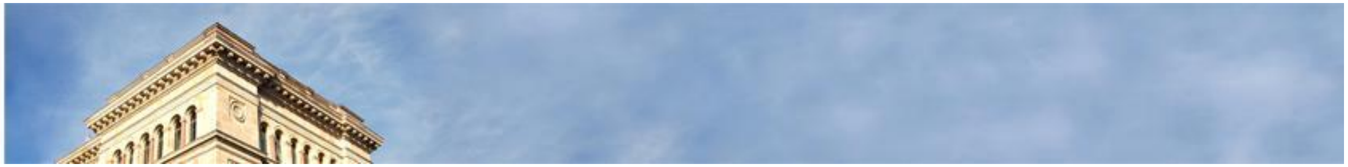


QUIC – Important concepts

- Transport parameters to negotiate connection properties during handshake
- Different Frame types for different purposes, e.g. reliable or unreliable data etc.
- Packets are protected through AEAD
    - I.e. header can be readable by the network while payload is encrypted
    - Whole packet is integrity and authenticity checked
- Arbitrary number of streams in a connection, each with own flow control
    - Can be used to create back pressure and prevent HoL blocking

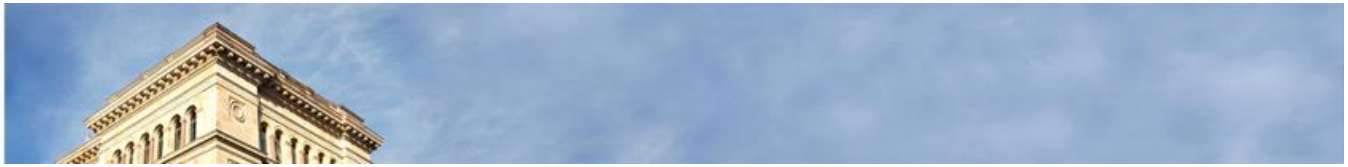
MCQUIC Overview



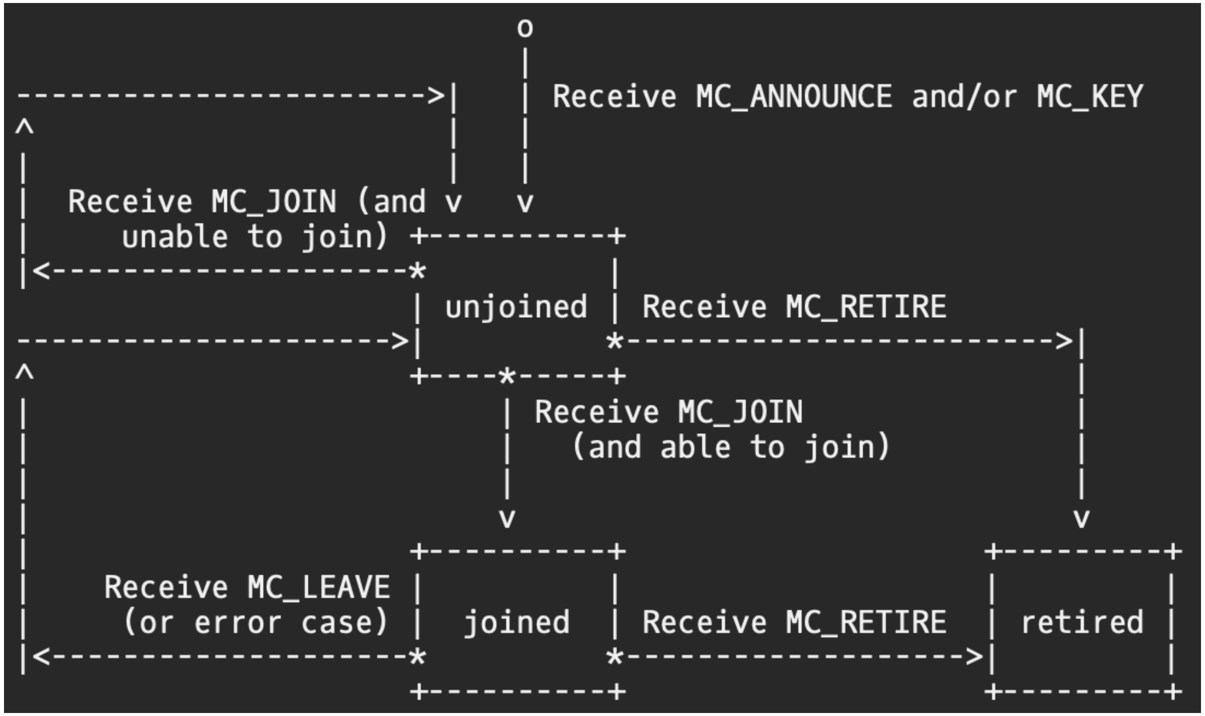


MCQUIC – Overview

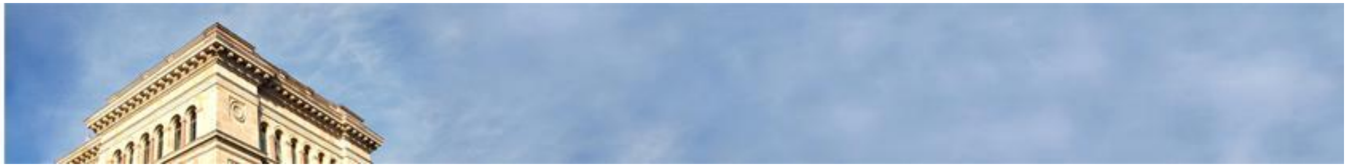
- Addition of Multicast channels to Unicast streams
- New transport parameters to negotiate multicast limits
- Server driven, Client executed
  - Server instruct client to join/leave multicast, client decides to do so
- New frames for control of joining/leaving multicast channels
- Automatic fallback to unicast; Transparent to application
- Client sets limits for congestion and flow control
- Actual multicast delivery is independent, can be done through any technology, e.g. PIM, BIER etc.



MCQUIC Overview

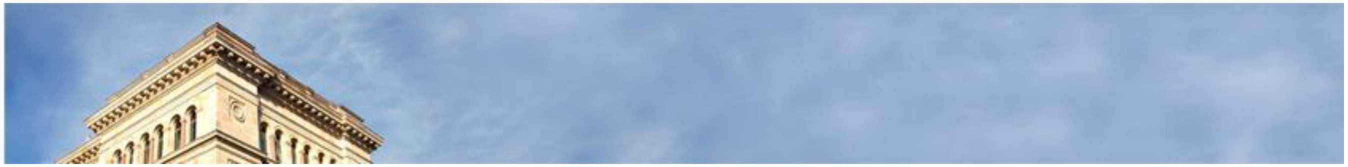


```
                            o
                            |
----------------------->|   | Receive MC_ANNOUNCE and/or MC_KEY
^                       |   |
|                       |   |
|  Receive MC_JOIN (and v   v
|     unable to join) +----------+
|<--------------------*          |
                      | unjoined | Receive MC_RETIRE
--------------------->|          *------------------------>|
^                     +----*-----+                         |
|                          | Receive MC_JOIN               |
|                          |   (and able to join)          |
|                          |                               |
|                          v                               v
|                     +----------+                    +---------+
|    Receive MC_LEAVE |          |                    |         |
|     (or error case) |  joined  | Receive MC_RETIRE  | retired |
|<--------------------*          *------------------->|         |
                      +----------+                    +---------+
```



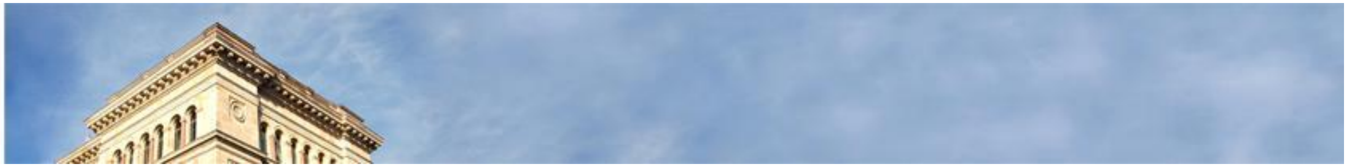
MCQUIC Design - 1

- Use QUIC unicast connections as anchor for multicast
  - Client declares multicast support and limits (e.g. max rate)
  - Server picks fitting SSM channel(s) and tells client to join them
  - Client attempts join and receives data packets over multicast
- Unicast connection is used for integrity/ crypto keys
  - Each packet sent over multicast has integrity frame (i.e. checksum) associated with it
  - Multicast packets are encrypted



MCQUIC – Problems solved

- Feasible way to get multicast into browsers via QUIC
- Encryption for packets (though weak as all receivers can decrypt)
- Integrity prevents injection of packets by attackers
- Reliability means application layer does not have to worry about FEC
- Since it is still just a QUIC connection, applications need little changes
- Pushes large data packets to multicast while only keeping integrity frames over unicast



MCQUIC Design - 2

- Client ACKs each packet over unicast
  - Server can choose to retransmit lost packets over unicast or (if many clients lost it) over multicast again
  - Guarantees reliability
- Multicast only works from server to client
- Multicast packets are also part of the QUIC connection, client does not need to differentiate

Questions?

25.10.24