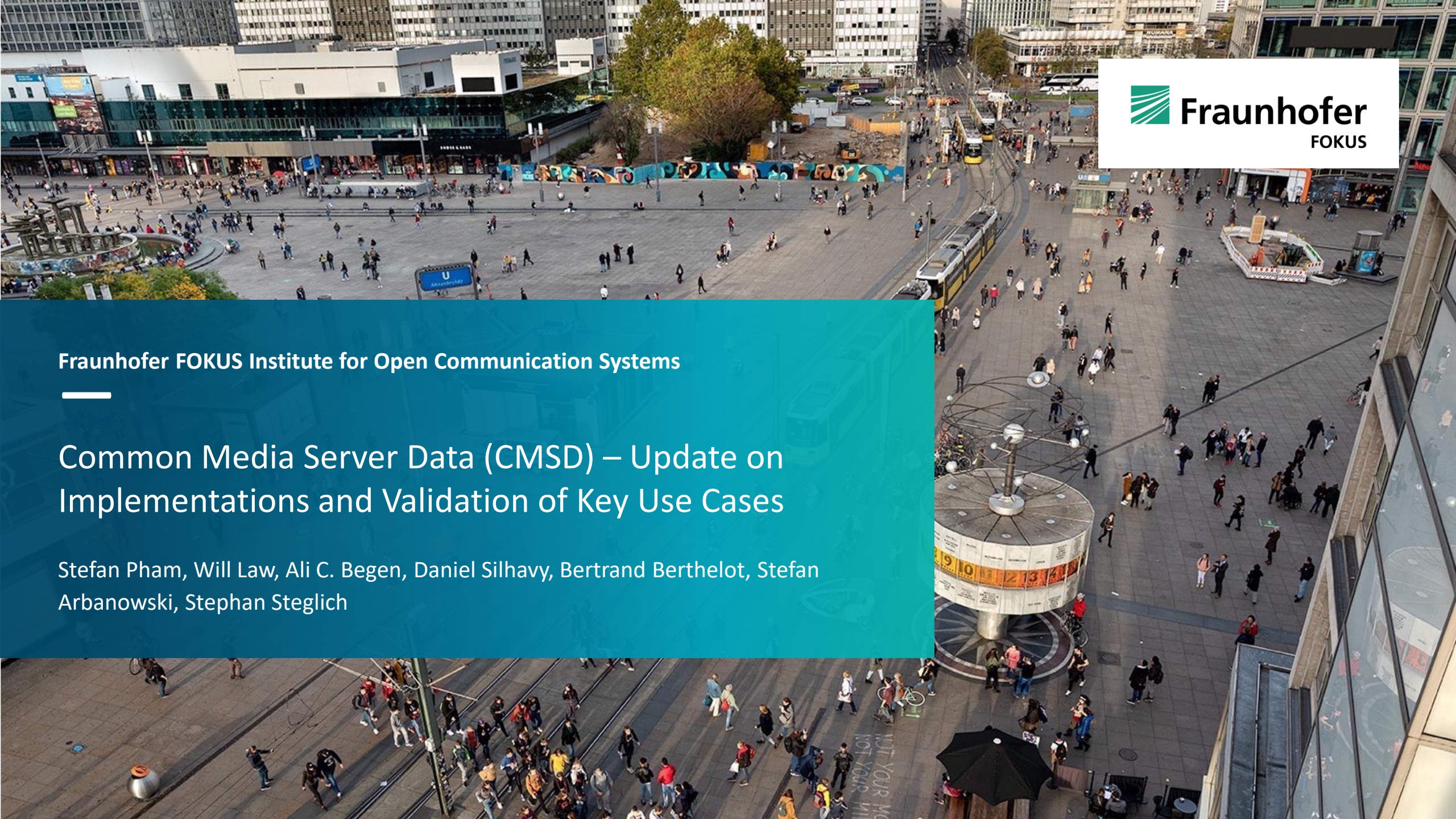


Fraunhofer FOKUS

Fraunhofer FOKUS Institute for Open Communication Systems

Common Media Server Data (CMSD) – Update on Implementations and Validation of Key Use Cases

Stefan Pham, Will Law, Ali C. Begen, Daniel Silhavy, Bertrand Berthelot, Stefan Arbanowski, Stephan Steglich

CMSD - Common Media Server Data

- Published as CTA-5006 in November 2022
- uniform method for media servers to exchange data with each media object response
- The aim is to enhance distribution efficiency, performance, and ultimately, the user experience

Open source Implementations:

- added to dash.js
- Origin implementations from Unified Streaming and National University of Singapore/Ozyegin University

CMSD parameters

- at: Availability Time
- du: Duress
- br: Encoded bitrate
- etp: Estimated throughput
- ht: Held time
- n: Intermediary identifier
- mb: Max suggested bitrate
- nor: Next object response
- nrr: Next range response
- d: Object duration
- ot: Object type
- rd: Response delay
- rtt: Round trip time
- su: Startup
- st: Stream Type
- sf: Streaming format
- v: Version

CMCD – Common Media Client Data

- Published as CTA-5004 in September 2020 out of CTA WAVE group
- Media player clients can convey information to Content Delivery Networks (CDNs) with each object request
  - useful in log analysis, QoS monitoring and delivery optimization
  - improve the quality of service offered by CDNs
- Transport via
  - custom HTTP request header,
  - HTTP query argument,
  - As a JSON object independent of the HTTP object request
- Integrated with dash.js -> see Blog Post

CMCD parameters

- bl: Buffer length
- br: Encoded bitrate
- bs: Buffer starvation
- cid: Content ID
- d: Object duration
- dl: Deadline
- mtp: Measured throughput
- nor: Next object request
- nrr: Next range request
- ot: Object type
- pr: Playback rate
- rtp: Requested maximum throughput
- sf: Streaming format
- sid: Session ID
- st: Stream Type
- su: Startup
- tb: top bitrate

How CMSD can data be transmitted

Simply via **Response Headers.**

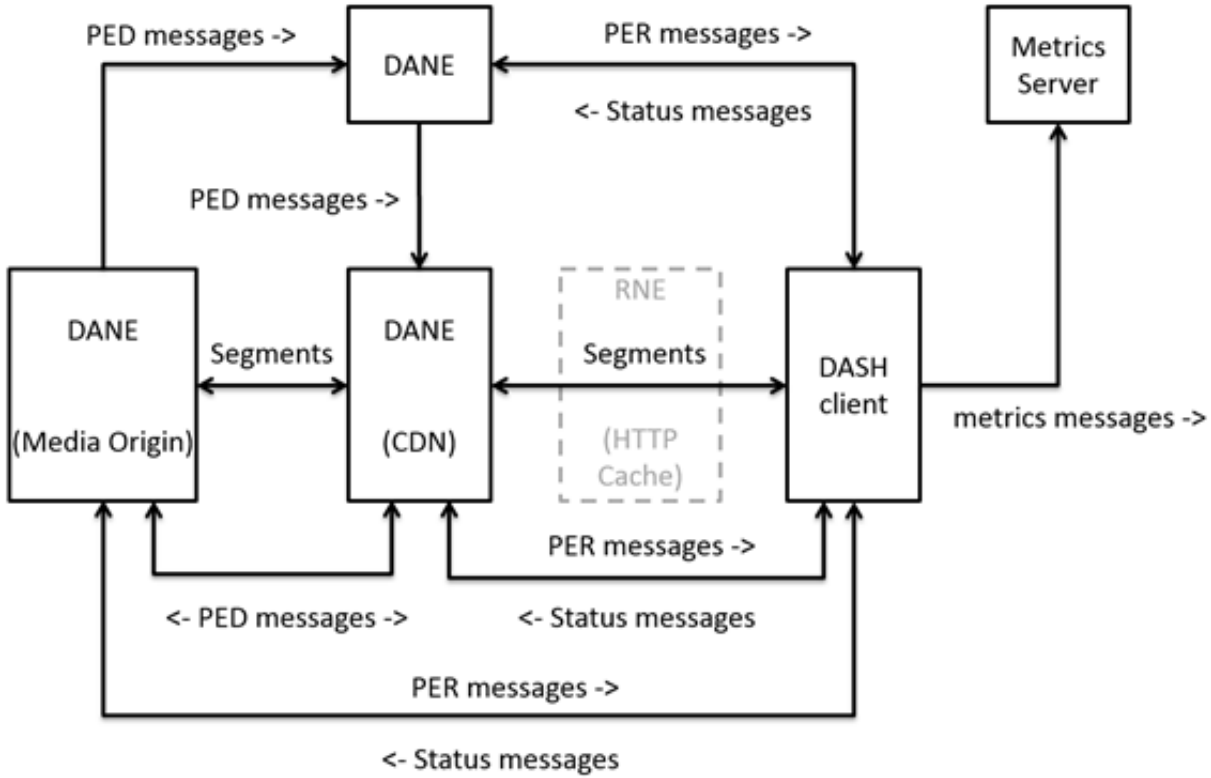
- A robust and well understood data transfer system
- Two headers are defined, in order to limit HPACK/QPACK table entries
  - **CMSD-Static** – data about the request that does not change over each link
  - **CMSD-Dynamic** - data about the request that changes with each link

```
CMSD-Static:ot=v,sf=h,st=v,d=5000,br=2000,n="OriginProviderA"
CMSD-Dynamic: "CDNB-3ak1";etp=96;rtt=8
CMSD-Dynamic: "CDNB-w35k";etp=76;rtt=32
CMSD-Dynamic: "CDNA-987.343";etp=48;rtt=30
CMSD-Dynamic: "CDNA-312.663";etp=115;rtt=16;mb=5000
```

Source: Will Law – CTA WAVE

SAND

- Published as ISO/IEC 23009-5:2017 Server and network assisted DASH (SAND)
- Introduces bi-directional messages between DASH clients and network elements or between various network elements for the purpose to improve efficiency of streaming sessions
- Standardized metric categories (23009-1 Annex D):
  - TCPList, HTTPList, RepSwitchList, BufferLevel, PlayList, DeviceInformationList
- Example Use Cases:
  - Metric reporting
  - Shared Resource Allocation
  - Real-time Multi-CDN switching
  - And many more …see DASH-IF whitepaper: https://dashif.org/docs/SAND-Whitepaper-Dec13-final.pdf



SAND Shared Resource Allocation: Use Cases

Use cases:

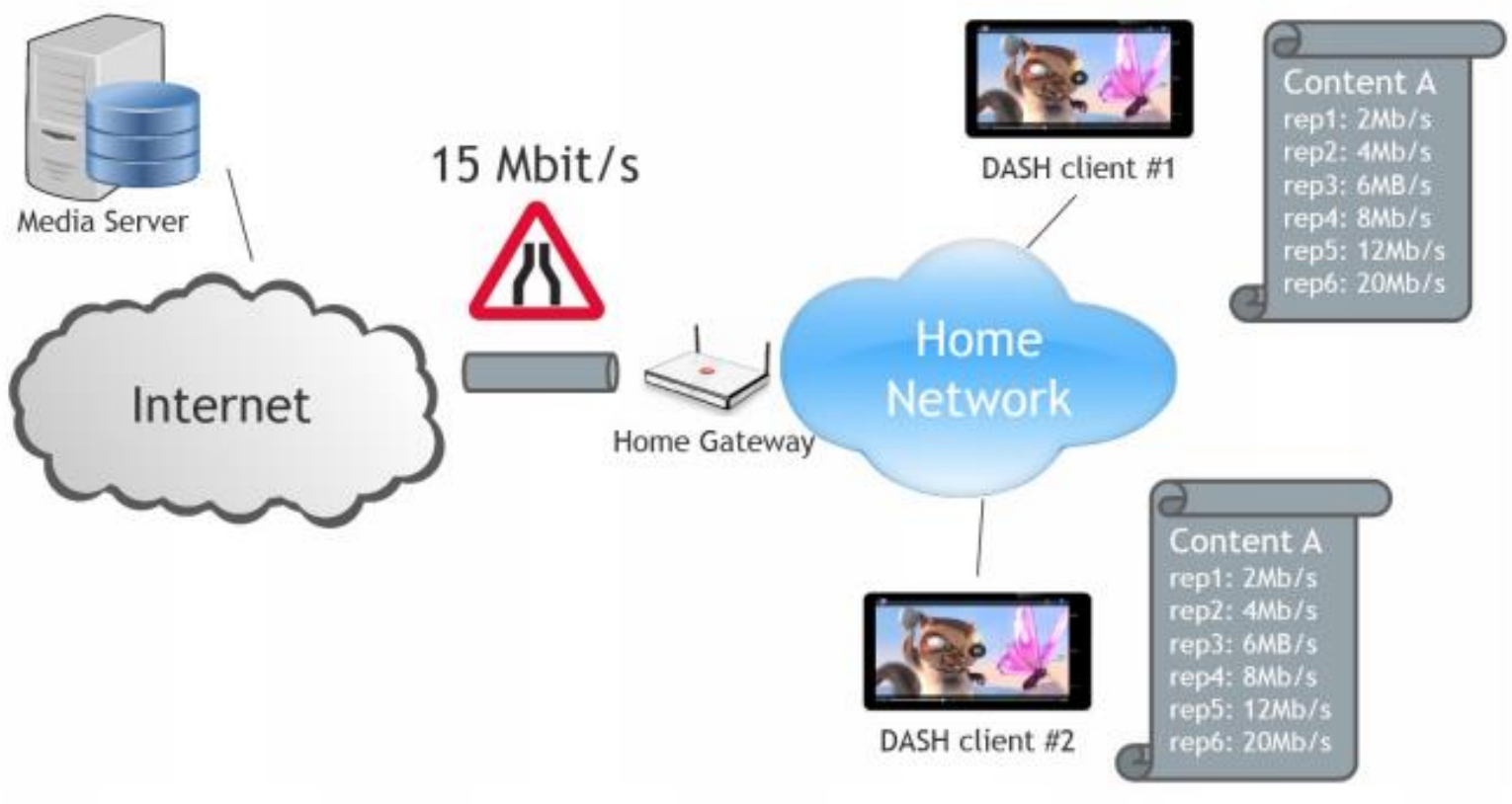
- Sports events (stadium)
- Airplanes, trains
- Home networks

Goal:

- Controlling the available bandwidth
- Use bandwidth optimally

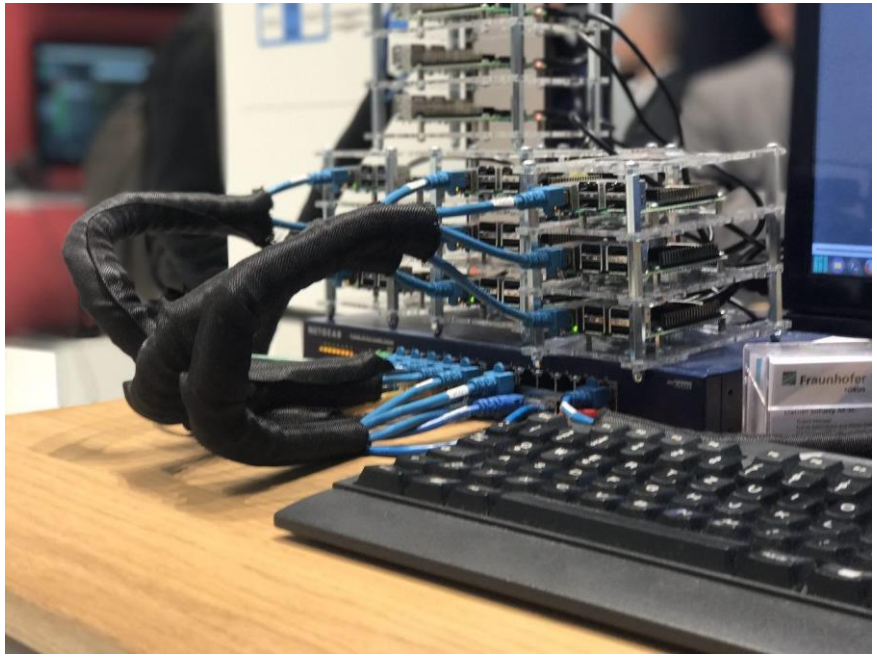
Different Strategies

- „Everybody served“
- “Premium privileged“
- Etc.



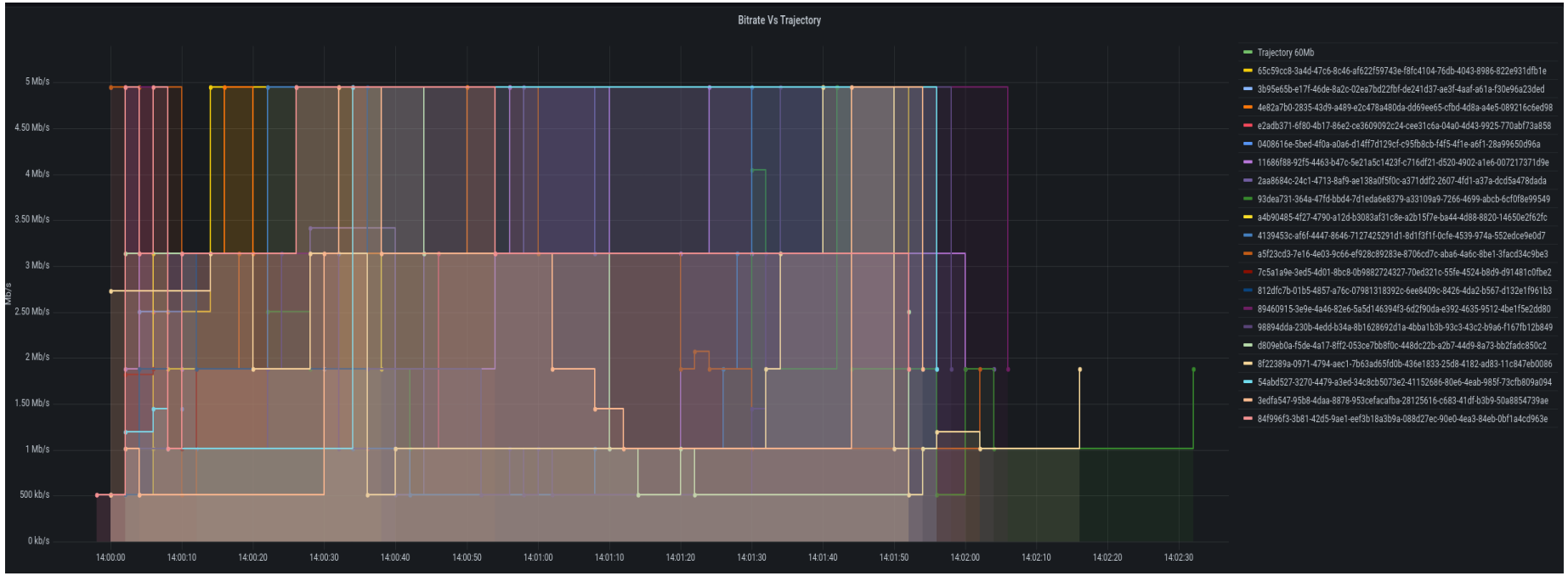
Evaluation Setup

- Raspberry Pis running latest dash.js in Chromium
  - 12 Pis with 2 browser instances each
- Metric reporting via CMCD/SAND
- ABR algorithm: BOLA
- Video Quality levels (kBit/s): 507, 1013, 1883, 3134, 4952
  - VBR encoding



Challenge: Multiple Clients – Shared Network Bottleneck

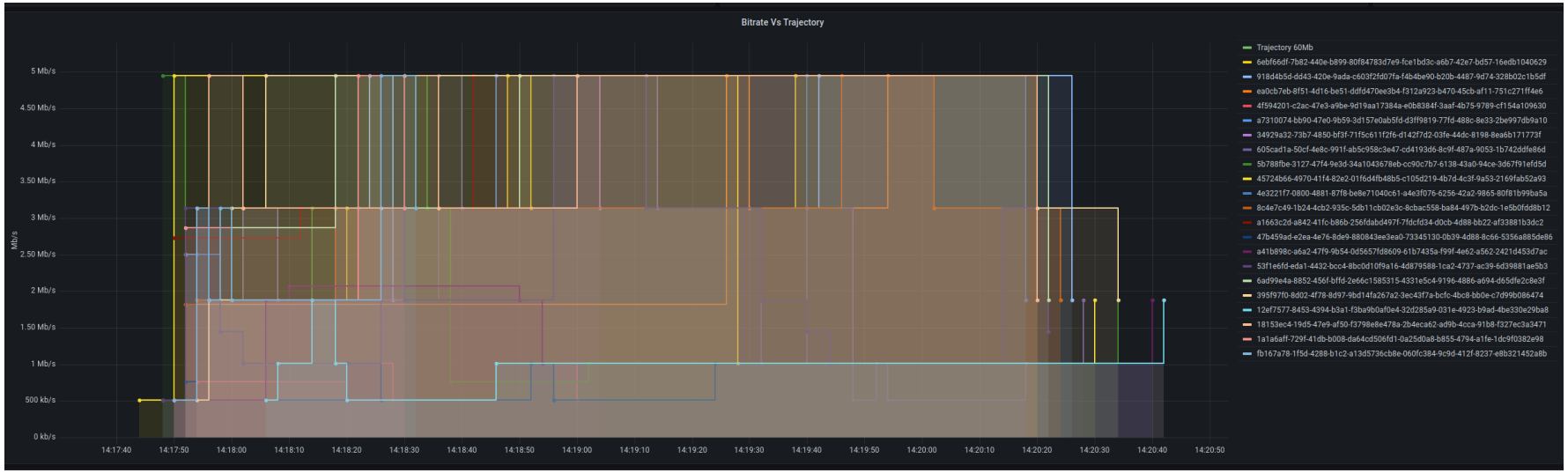
- ABR clients not aware of each other → aggressiveness
- Oscillating ABR behavior when connected to the same network bottleneck
- Increasing bandwidth requirements (UHD, HDR etc. etc.)
- Bandwidth limited and underutilized




Bitrate Vs Trajectory
5.20 Mb/s
5 Mb/s
4.80 Mb/s
4.60 Mb/s
4.40 Mb/s
4.20 Mb/s
4 Mb/s
3.80 Mb/s
3.60 Mb/s
3.40 Mb/s
3.20 Mb/s
3 Mb/s
2.80 Mb/s
2.60 Mb/s
2.40 Mb/s
2.20 Mb/s
2 Mb/s
1.80 Mb/s
1.60 Mb/s
Mb/s
14:28:30
14:28:45
14:29:00
14:29:15
14:29:30
14:29:45
14:30:00
14:30:15
14:30:30
14:30:45
14:31:00
14:31:15
14:31:30
14:31:45
14:32:00
14:32:15
14:32:30
14:32:45
14:33:00
14:33:15
Trajectory 60Mb
ef179405-0547-4ad8-8be5-bc5128dd5414-ebc1d24f-1ba9-4b26-a329-25c1d6e249b1
d5503aa6-fdc5-4ef3-a139-6131a4597e4b-de202e0a-5404-4f4b-9268-59d7ed6d3110
dbe72f21-36a6-44d6-8481-5d2157b5db84-c95ce8c7-5f21-4a5e-97f0-21811aabb30b
d2ae60d4-2982-4d1f-b883-474a23cde963-b94b5bfd-9d4d-4f04-b378-608c484e3dc0
1edc2b6f-0143-4c4c-9f34-6182a70752a2-8c6a4726-3e1a-4295-a88c-13e58dedb4e8
cd0d6729-78b1-4b87-b167-a6758140fb80-6df69d91-483c-471a-9702-f6c540edad01
2319afe8-e2f5-4012-87d3-5afc7a82a717-63d536d9-1490-474b-92ab-65d1d645192d
cf1d496b-856b-4872-beeb-afcaecc08ec3-436ba1b0-045d-4e46-9f31-1816d8a4c675

Using the CMSD ‘mb’ attribute

- 24 sessions are started one after the other, reporting their bitrate
- The ‚mb‘ attribute is set dynamically (bandwidth / #clients)
- The CMSD implementation in dash.js parses the ‚mb‘ attribute and sets the maximum bitrate
- The player picks a representation from the MPD that is below the ‚mb‘ value
- In this test run 60MB/s is available for 24 sessions -> 2,5MB/s for each session -> 1,8MB/s is the available representation

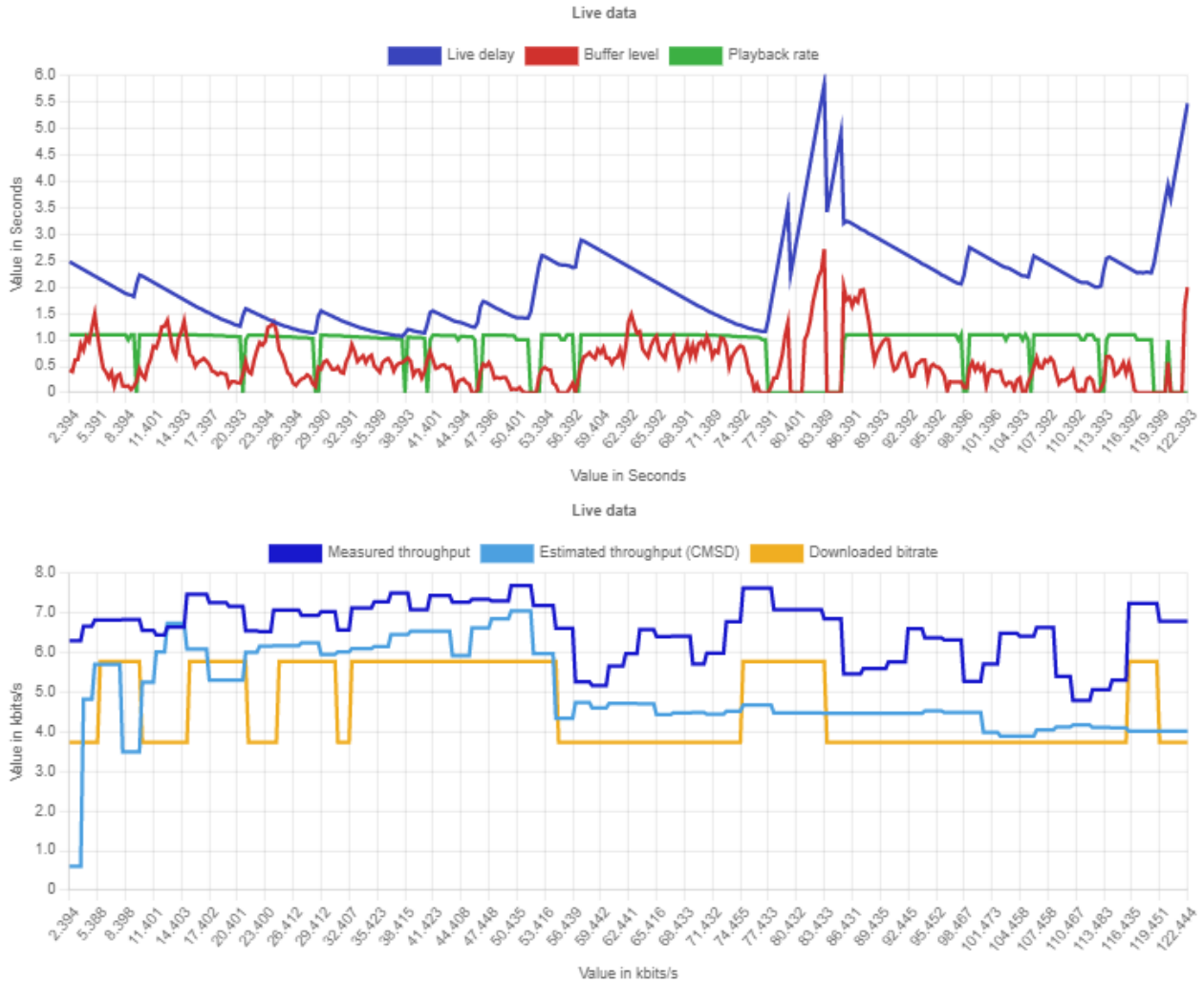


What can be done with ‘etp’?

'etp’: Estimated throughput

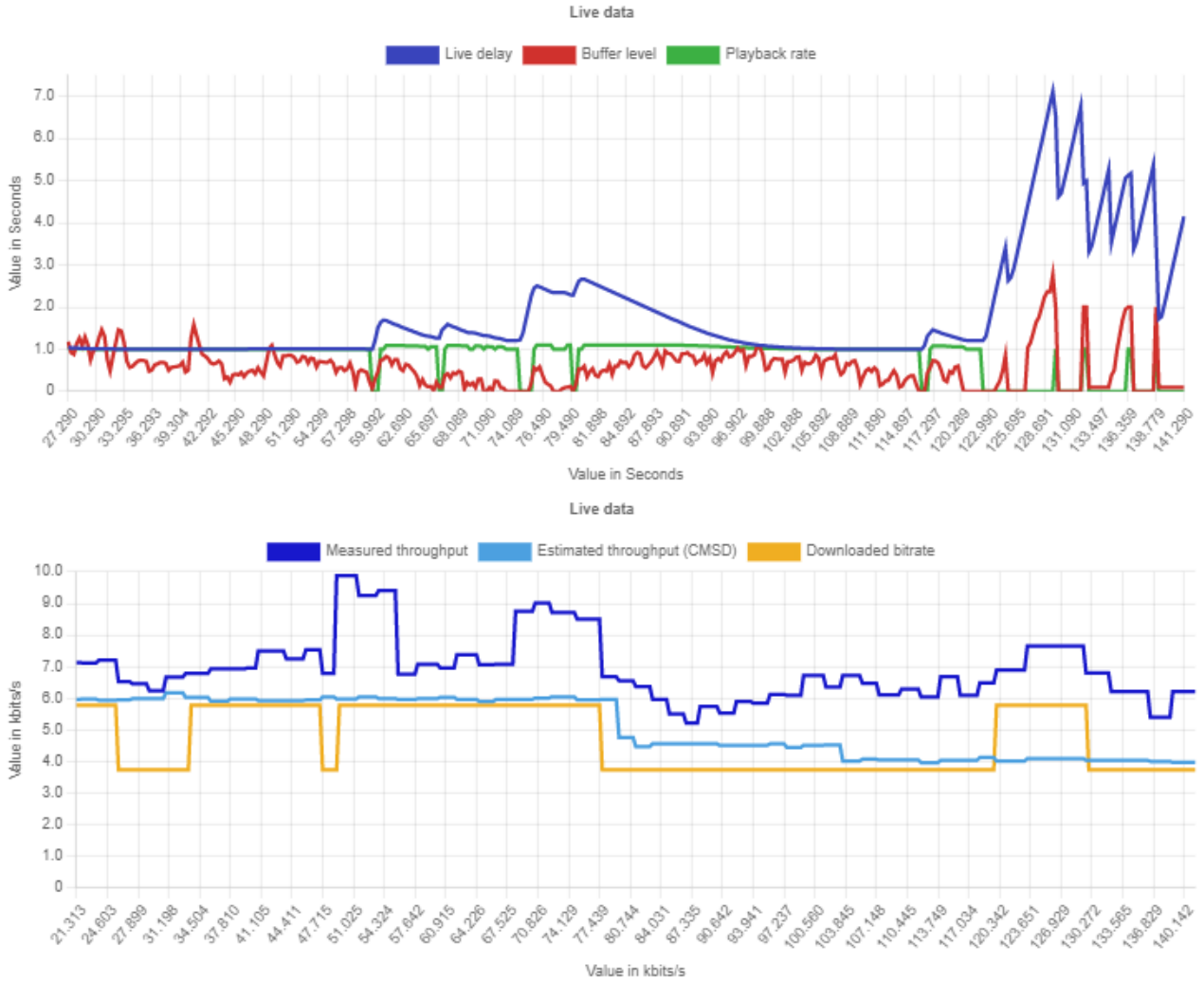
- Improve the selection of the initial bitrate
  - ABR algorithms conservatively select lower bitrates at the startup
  - An estimated throughput value helps the player to select a more appropriate bitrate
  - After startup, server-side throughput estimates can be combined with client-side estimations for higher accurary
- QoE improvements for low-latency streaming using chunked-transfer encoding
  - Client-side throughput estimation for chunked-transfer encoding is challenging, e.g. in Web browsers
  - Server-side throughput estimation using the transport layer’s congestion control layer leads to more precise estimates

ABR based on L2A-LL



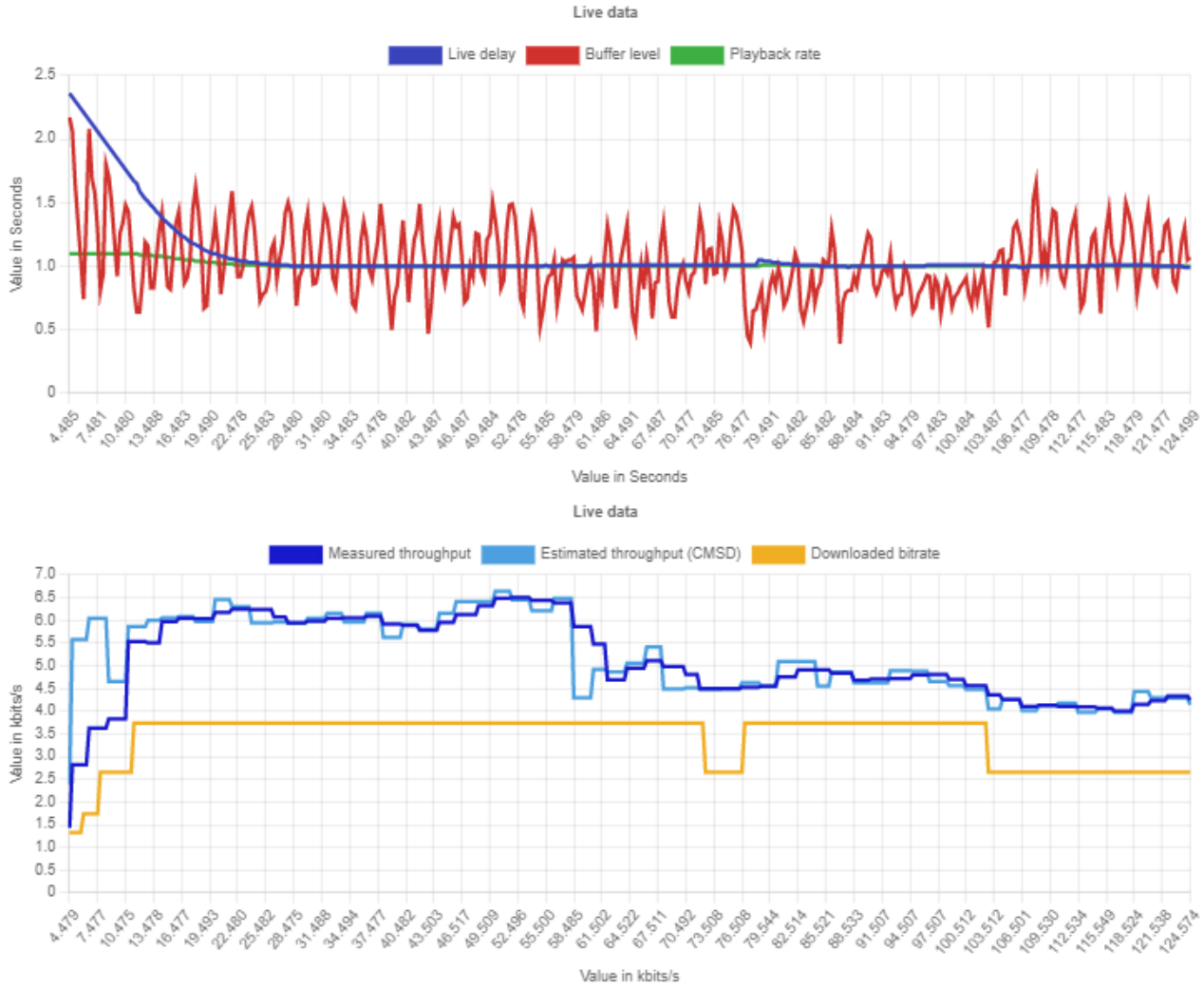
Source: Bertrand Berthelot - Broadpeak

ABR based on moof parsing throughput estimation



Source: Bertrand Berthelot - Broadpeak

ABR based on CMSD ‘etp’



Source: Bertrand Berthelot - Broadpeak

Contact

Stefan Pham
Phone +49 (30) 3463 - 7103
stefan.pham@fokus.fraunhofer.de

Fraunhofer FOKUS
Institute for Open Communication Systems
Kaiserin-Augusta-Allee 31
10589 Berlin, Germany
info@fokus.fraunhofer.de
www.fokus.fraunhofer.de