Kyber

Interactive Streaming

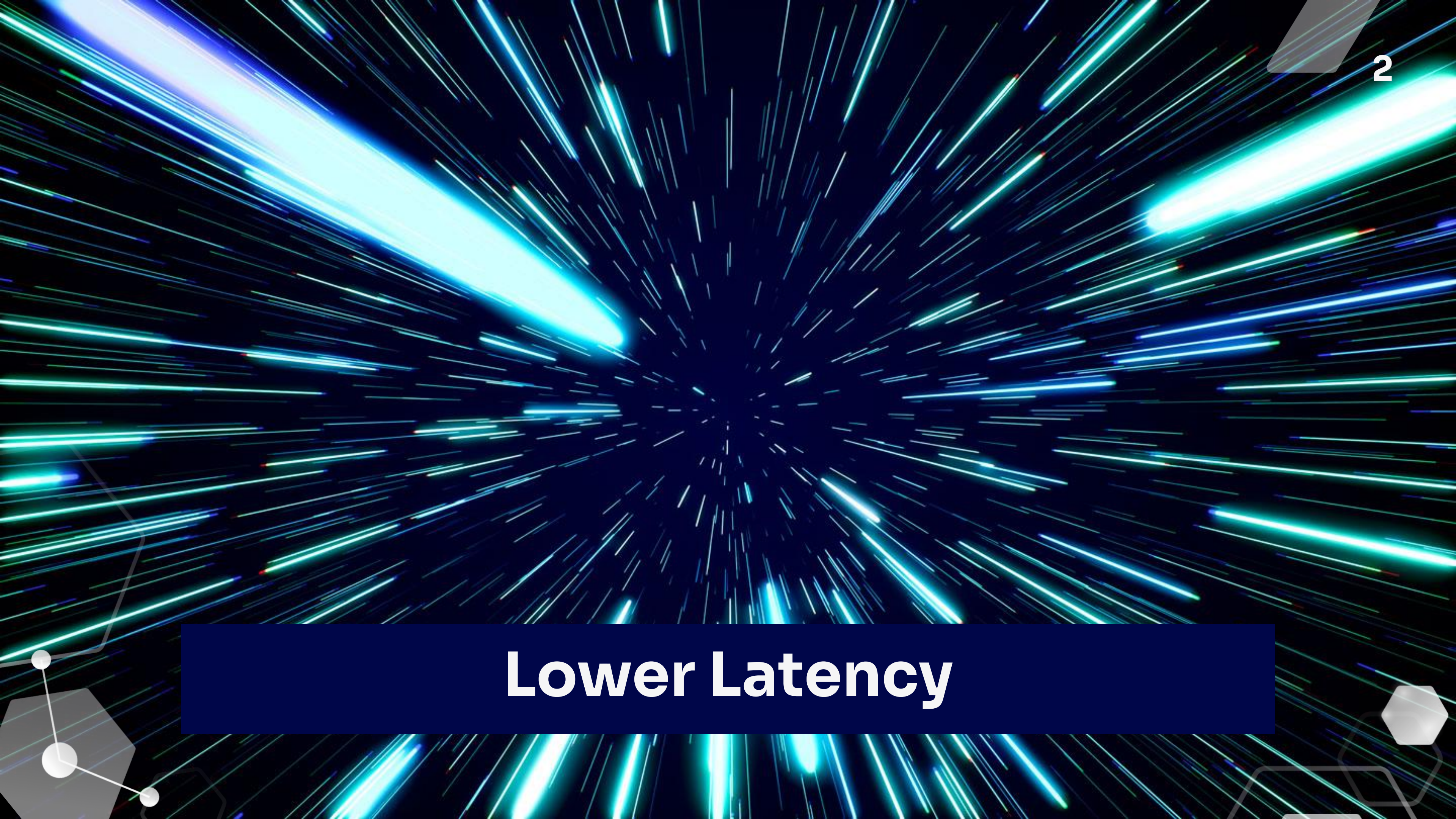
(near) Real Time Video and Controls streaming

FRnOG 39

Jean-Baptiste Kempf

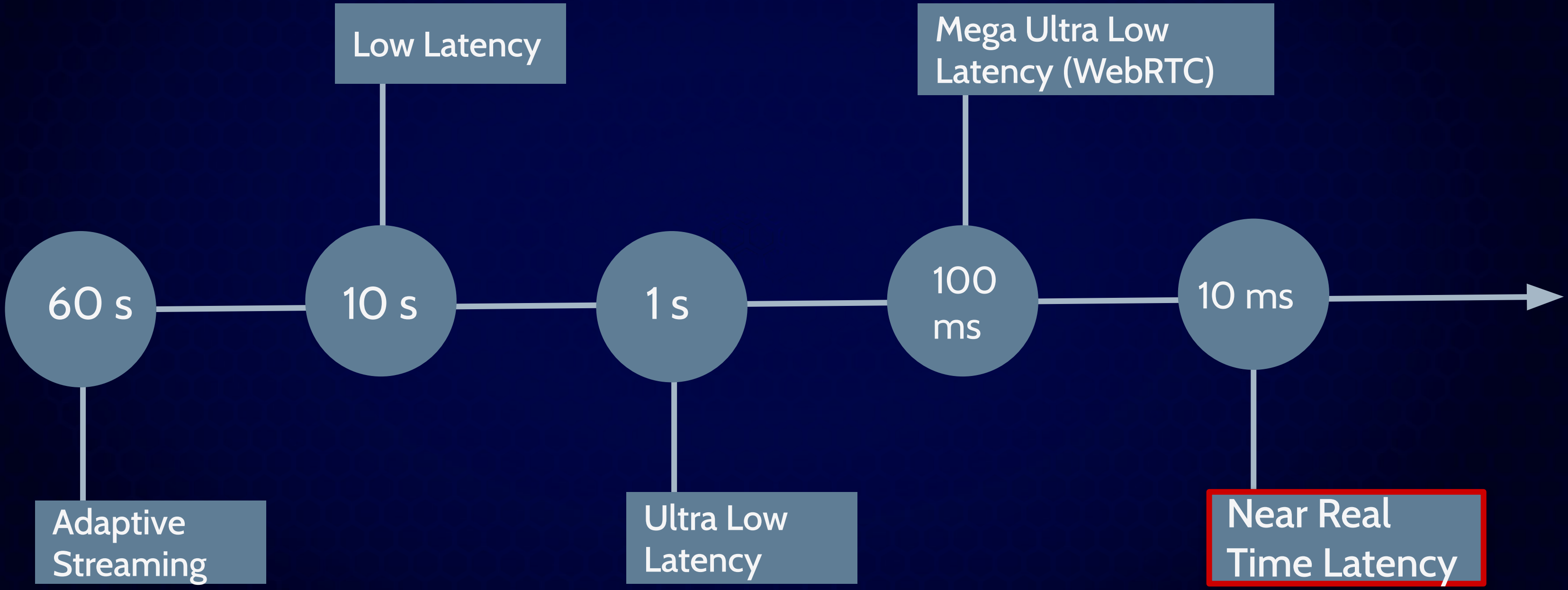
Resume of My Video Life

- **VideoLAN:** being working on the VideoLAN project for 17 years, president of the VideoLAN NPO since its creation *(2008)*
  Doing most of the non-developer tasks of VideoLAN
- **VLC:** Active developer since 2006, notably on GUI, Windows, Android ports, codecs and demuxers, packaging and releases
- **FFmpeg:** Active community member and peacekeeper of FFmpeg. De facto involved in releases and roadmaps.
- **Shadow:** ex-CTO of Cloud Gaming/Desktop company
- **Technical Consultant**: Video startups, scaleups and e-Commerce business



Lower Latency

Lower Latency Streaming



We're talking about encoded latency, else we would talk in lines :)

Why do we need lower latency?

Interactivity

Remote Desktop

VM / VDI / DaaS
Cloud Gaming
Cloud Desktop

Robots & Drones

Robots, Drones
AR Supervision
Cars?

Safety & Critical

Remote Monitor

Virtual Monitor
SlingBox
Industrial Supervision

App Streaming

Remote Video Production
Trial / Demo of Apps
Visual Cloud Apps

Kyber

Watizit?

Kyber

Open Source

Real Time

Control of Machines

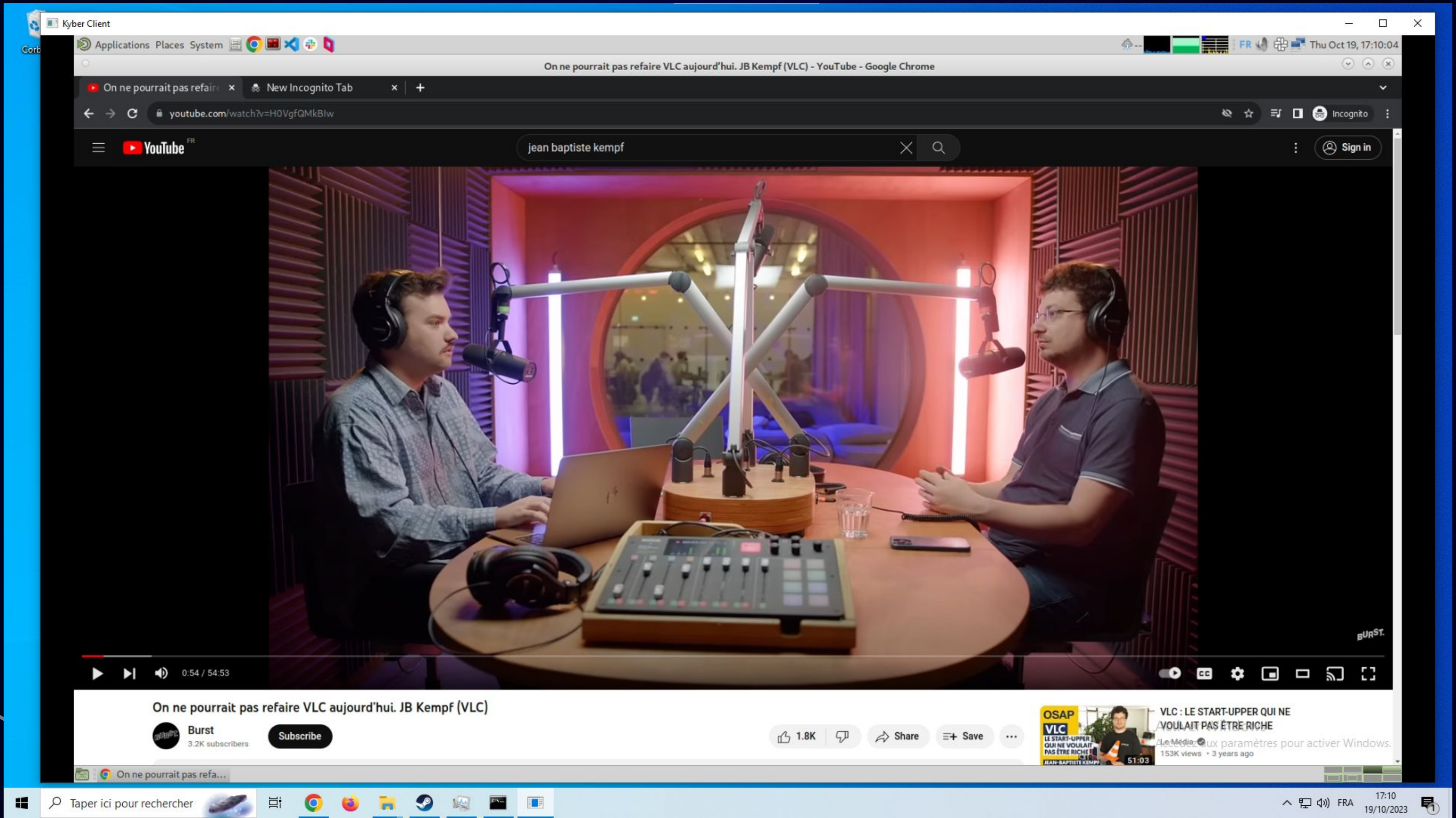
Solution SDK

Client, Server and Networking stack
Streaming video, audio, *subtitles* unidir
Streaming inputs bidirectionally
Modular SDK and application
Quic protocol & ~~WebRTC~~

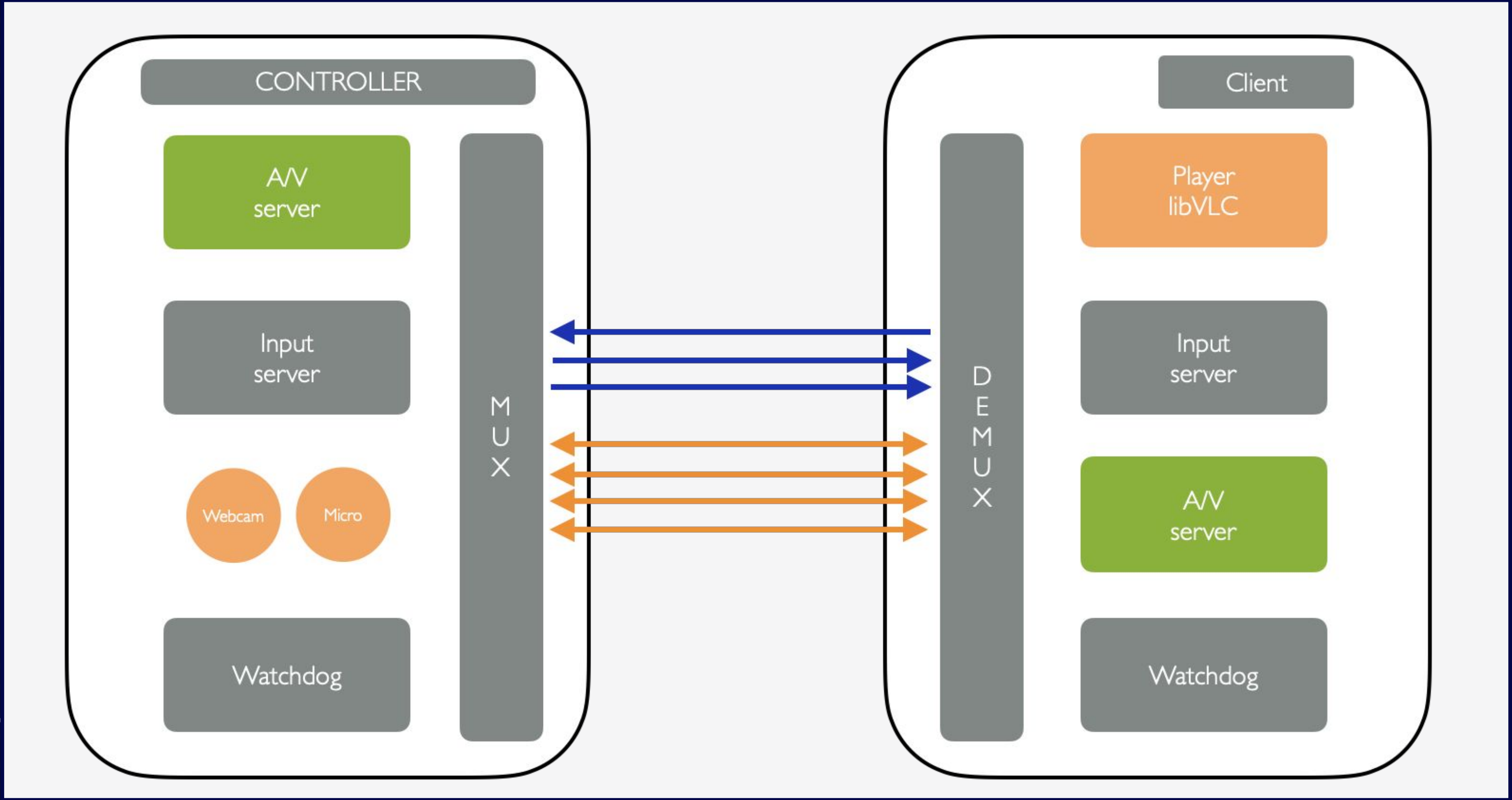
Multi-platform Client *(+ Web)*
Multiple platforms for the Servers
All Codecs (H.264, HEVC, VP9, AV1)
Multiple Hardware & Software Encoders
Based on VLC and FFmpeg

Demo

Linux Server


Kyber Client
Applications Places System
FR
Thu Oct 19, 17:10:04
On ne pourrait pas refaire VLC aujourd'hui. JB Kempf (VLC) - YouTube - Google Chrome
On ne pourrait pas refaire
New Incognito Tab
youtube.com/watch?v=H0VgfQMkBIw
Incognito
YouTube FR
jean baptiste kempf
Sign in
BURST
0:54 / 54:53
On ne pourrait pas refaire VLC aujourd'hui. JB Kempf (VLC)
Burst
3.2K subscribers
Subscribe
1.8K
Share
Save
OSAP
VLC
LE START-UPPER QUI NE VOULAIT PAS ÊTRE RICHE
JEAN-BAPTISTE KEMPF
51:03
VLC : LE START-UPPER QUI NE VOULAIT PAS ÊTRE RICHE
Le Média
153K views • 3 years ago
Accédez aux paramètres pour activer Windows.
On ne pourrait pas refa...
Corb
Taper ici pour rechercher
FRA
17:10
19/10/2023

How does it work?



Features

Video

Audio-Video Server based on FFmpeg libraries, through txproto

Pushed-based Streaming server, graph-based and multi-threaded per node Video Server can composite GPU overlays

Player based on libVLC, tuned for 0-latency *(push-based approach)*

Video Codecs tested: *H.264, HEVC, AV1, VP9*

Audio Codecs tested: *PCM, Flac, Opus*

Hardware and Software encoders

4:2:0 -> 4:4:4 upsampling on client

Input

New Input Server written in Rust from scratch

Push-based Input streaming server, graph-based, able to filter and merge inputs

I/O Support:
Keyboard, Mouse *(+Cursor)*, Gamepads *(+Rumble)*, Copy-Paste, File-Transfer, USB/IP

Virtual Video, Mouse, Keyboard and Gamepad Drivers

Cross-Platform, Client = Server

Network

Multiplexer Server written in Rust from scratch

Multi Protocol: Quic and WebTransport

Opens only one port *(TCP+UDP)*

TLS and Security handled at connection

Multi-user support *(Main, Student)*

Selectable features (audio, video, inputs)

Input latency is independent from Video latency

Separate process

Measures

Desktop @ 60 Hz

libVLC H.264:	~16ms	- 1 frame

Desktop @ 120 Hz

libVLC H.264:	~12ms	- 1 ½ frames
libVLC HEVC:	~12ms	- 1 ½ frames

Desktop @ 240 Hz

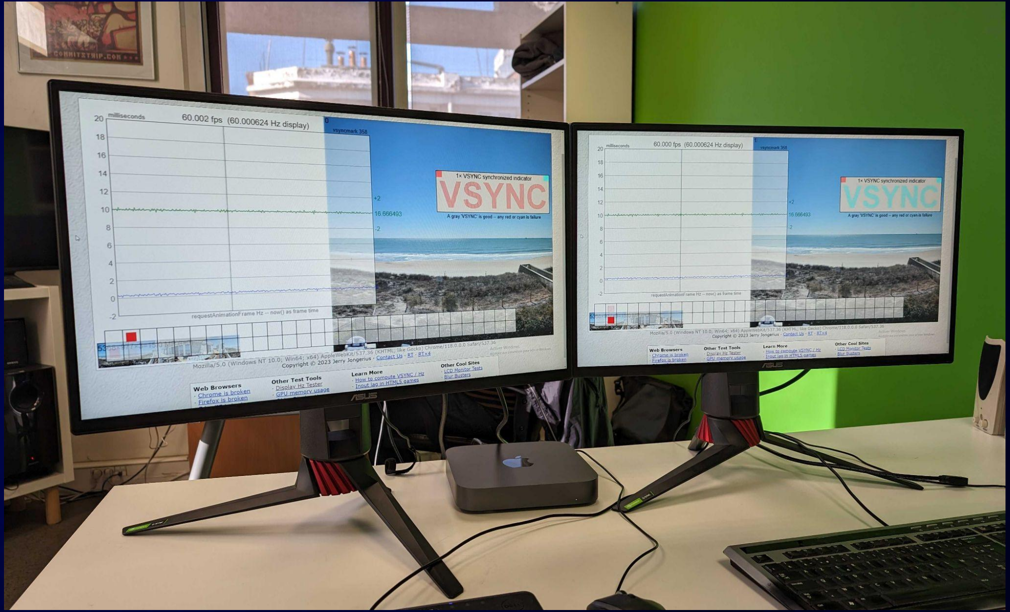
libVLC HEVC:	~10ms	- ~2 ½ frames
libVLC H.264:	~10ms	- ~2 ½ frames

Web @ 60 Hz

Kyber H.264 Soft:	~33ms	- 2 frames
Kyber HEVC Hard:	~33ms	- 2 frames

Web @ 120 Hz

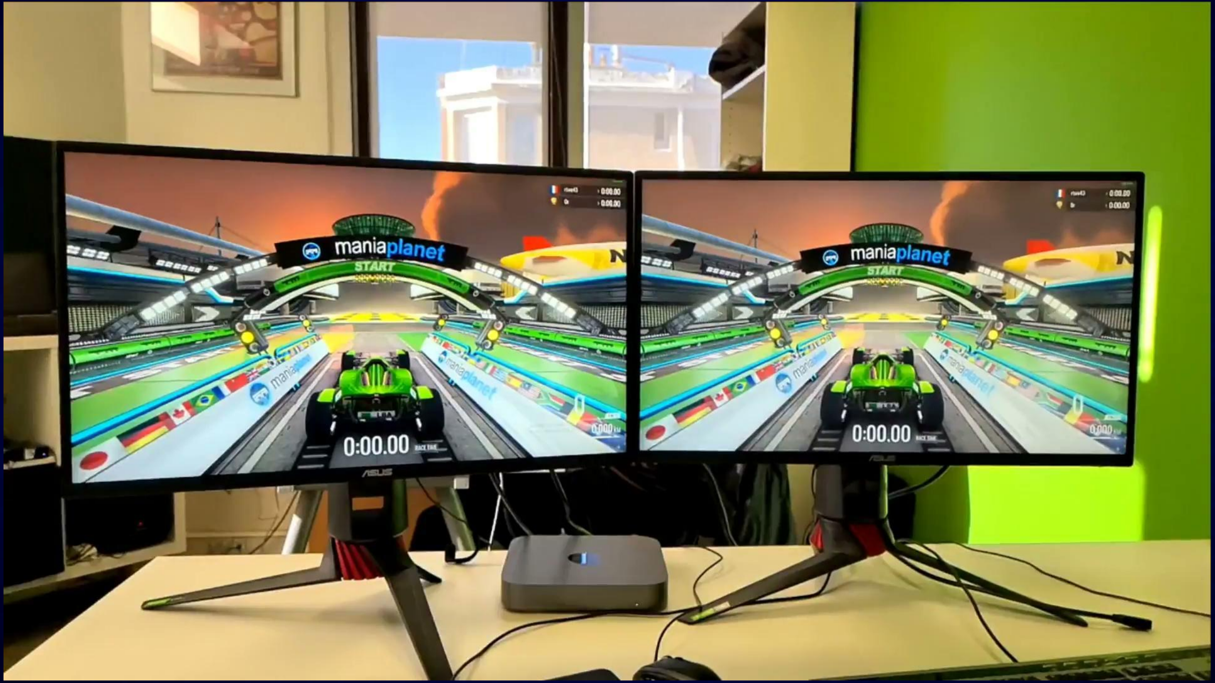
Kyber HEVC Hard:	~16ms	- ~2 frames
Kyber H.264 Soft:	~24ms	- ~3 frames



60.002 fps (60.000624 Hz display)
60.000 fps (60.000624 Hz display)
VSYNC
VSYNC
ASUS

We can do better!

Extra Low



We can do better!

Protocol

Quic / WT

Multiple protocols supported by the muxer

Big focus on Quic, because TLS, Uni-Socket, Multi-Stream, Bi-Directional and Datagrams

Audio / Video data can use Datagrams
Inputs are reliable and Bi-Directional

Use of WebTransport in very similar way than Quic, including DataGrams inside WebBrowser

Multiple Modes

Reliable: sends each channel on one QUIC/WT stream

GOPstream: 1 QUIC/WT stream per GoP

Unreliable:
Video Packets are sent in DataGrams mode
Config and Control packets are always sent in reliable streams

Unreliable_fec:
Use of FEC (RaptorQ) to recover info without needing retransmissions when loss of packets

Unreliable Protocol

Channels

One connection with multiple channels, similar to MoQ tracks

Request to Server to subscribe to the right channels *(channel_id)*

Stream vs Datagram

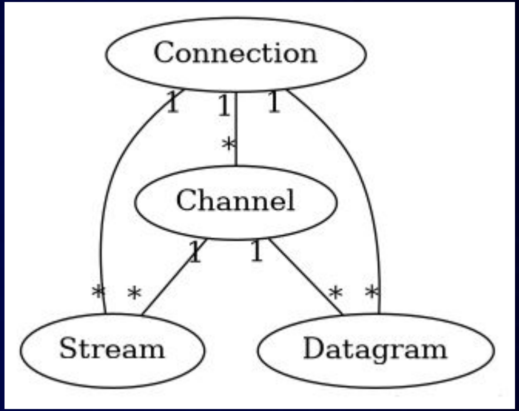
Some channels are either on DataGram or Stream mode

One channel is composed of multiple Streams or multiple Datagrams

Video can be datagrams and Inputs are always BiDi streams

Groups

Packets are grouped in groups for Media, to keep config (SPS/PPS) + data together



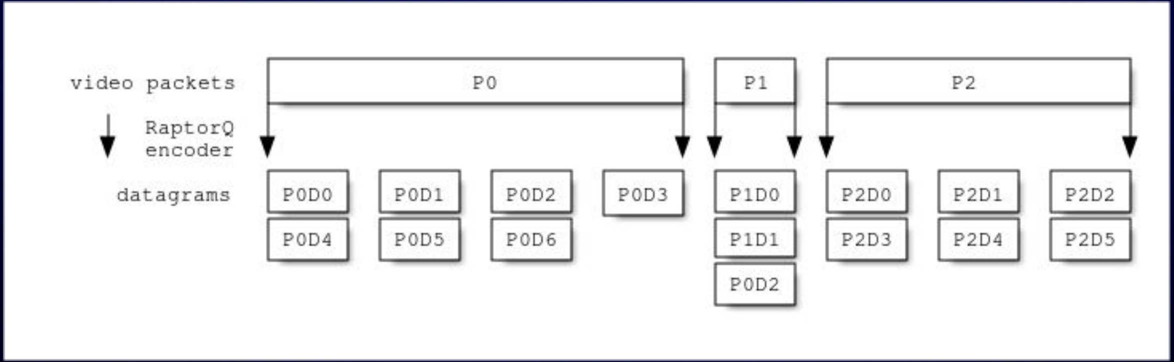
Unreliable Protocol + FEC

FEC

Use of RaptorQ *(for now)* but other schemes are possible

RaptorQ source symbols of size configured to maximize the max Quic Datagram size.

Hopefully, the RaptorQ encoded symbols are going on their own UDP datagrams



Web Version

Wasm

All the desktop Rust code is running in the web browser using Webassembly with same codebase

Notably Muxer and Input Server/Client

WebTransport

The Web version is using, in Rust/Wasm the same codebase and the same protocol

WebTransport instead of Quic

FEC running in Wasm

WebCodec

Video Decoding is done through WebCodec, in Rust/Wasm

Rendering is done through a Canvas
Compositing time is controlled by the app

Audio is done through WebCodec

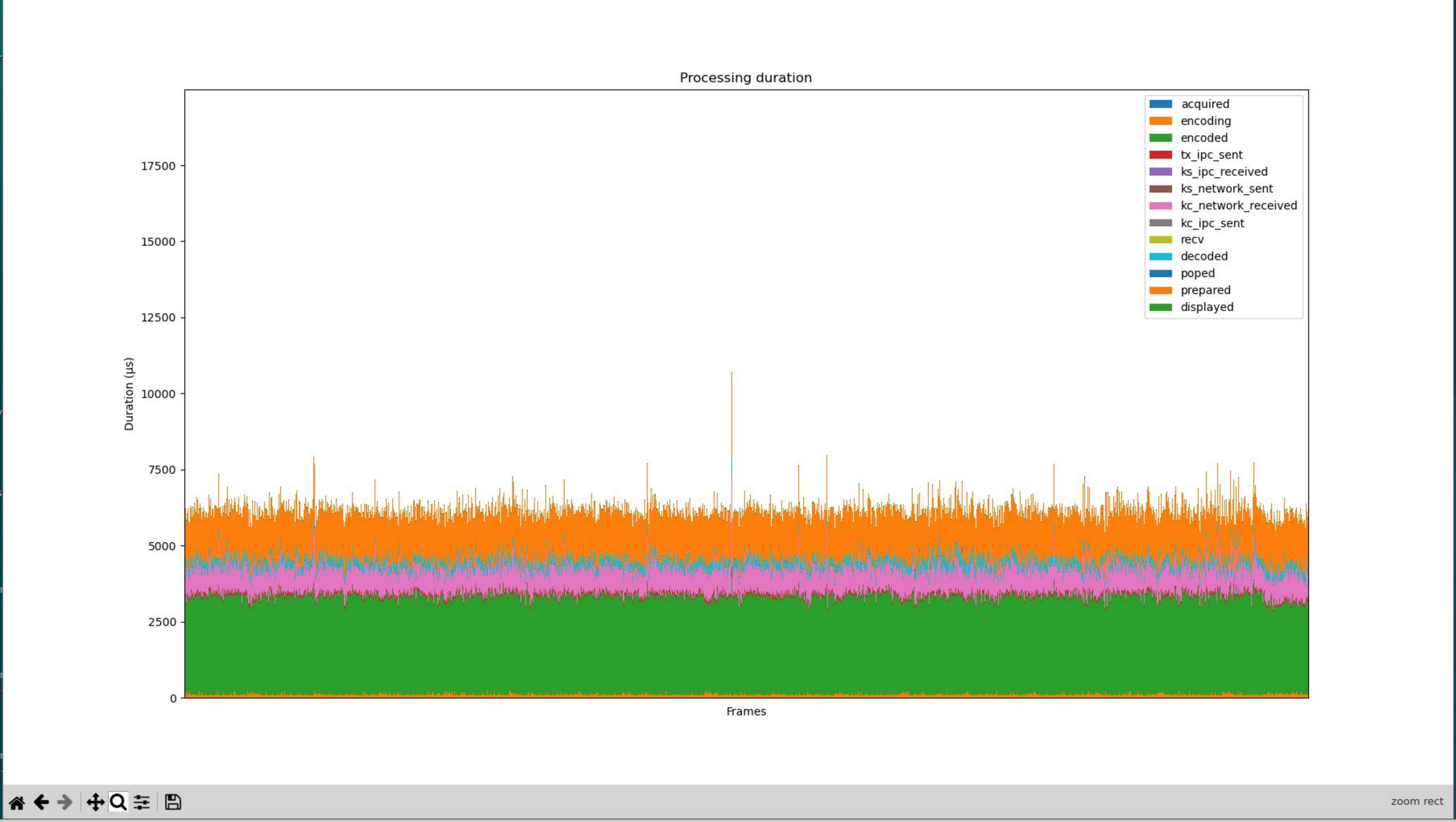
VLC.wasm

Video Decoding can also be done through VLC.wasm

No WebRTC :)

Thanks

Do you have any questions?
jb@videolan.org
kyber.media


Processing duration
Duration (µs)
17500
15000
12500
10000
7500
5000
2500
0
Frames
acquired
encoding
encoded
tx_ipc_sent
ks_ipc_received
ks_network_sent
kc_network_received
kc_ipc_sent
recv
decoded
poped
prepared
displayed
zoom rect