RICA: Remote Instrument Camera Access

A system to enhance remote user experience using secure real-time video-streaming over the web

Outline

- Introduction
- Application requirements
- Streaming protocols
- Architecture
- Authentication and authorization
- Future work

Introduction

RICA: Remote Instrument Camera Access

-> A tool that provides a real time visualization of the instrument

Addresses scientists and support needs:

- See the specific areas of the instruments - hard to access.
- Standardise the actual camera systems at the ILL.
- Provide secure external access.

Project developed at Institut Laue Langevin by the SCI in collaboration with the SI.

Service de Controle des Instruments (SCI)

Service Informatique (SI)

Requirements

- Only real time display - no recording - no user on demand reposition.
  - Allow digital zoom.
- Controlled access.

  Authorization required: user needs to be known by the ILL and the access to video stream depends on rights given.

- We provide a web application.
  - Use of standard IP cameras.
  - In house development.

Streaming protocols

A streaming protocol is the method (set of rules) for delivering media, audio and video, over the Internet.

The most of the IP cameras use Real Time Streaming Protocol (RTSP) to deliver the video and audio data.

Problem: RTSP cannot be used directly on a HTML5 element.

so... **We need to convert the streaming protocol.**

Streaming protocols: our requirements

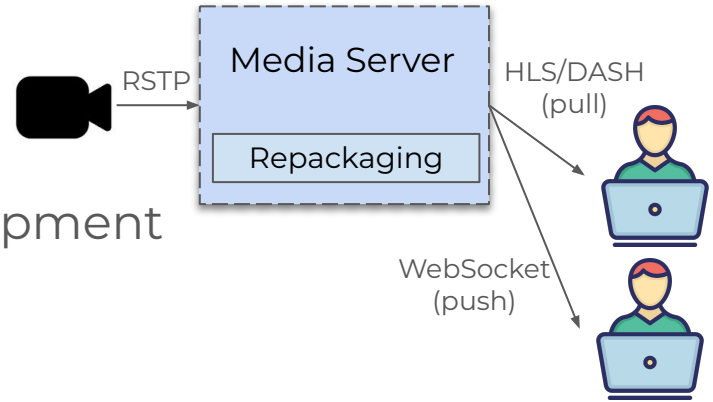
We were looking for:

- Low latency.
- Limit disk I/O.
- Little CPU consumption:
    - Avoid transcoding (decode, alter and encode the data) the stream.
- Avoid commercial software:
    - Open source solution.

Streaming protocols: options to deliver to a browser

Adaptive HTTP based protocols: HLS, DASH.

- Adaptive bitrate streaming. Client pull.
- Widely supported and scalable.
- Low Latency (LL) variants. Only in development when project was deployed.



Push based strategy: Websocket.

- Low latency: data actively sent to the clients.
- Simple to set up.
- No disk access required.

HTTP Live Streaming (HLS)

Dynamic Adaptive Streaming over HTTP (DASH)

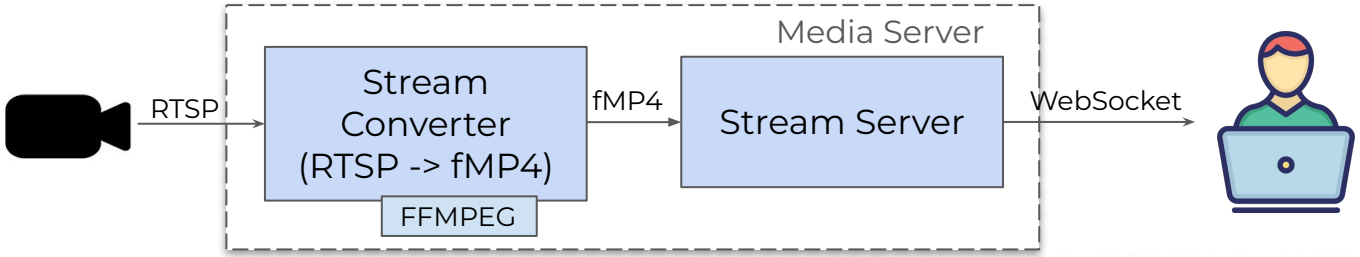
Streaming protocols: solution

Solution taken: Websocket based with fragmented MP4 (fMP4) on Media Source Extensions (MSE)

The server is divided in two applications:

- The main server which manages the communication between cameras and clients.
- A streamer which does the video conversion using FFMPEG.
  - No transcoding (same codec H.264) just repackaging (RTSP -> fMP4).

The client uses native HTML elements (MSE) to display the video.



Architecture validation

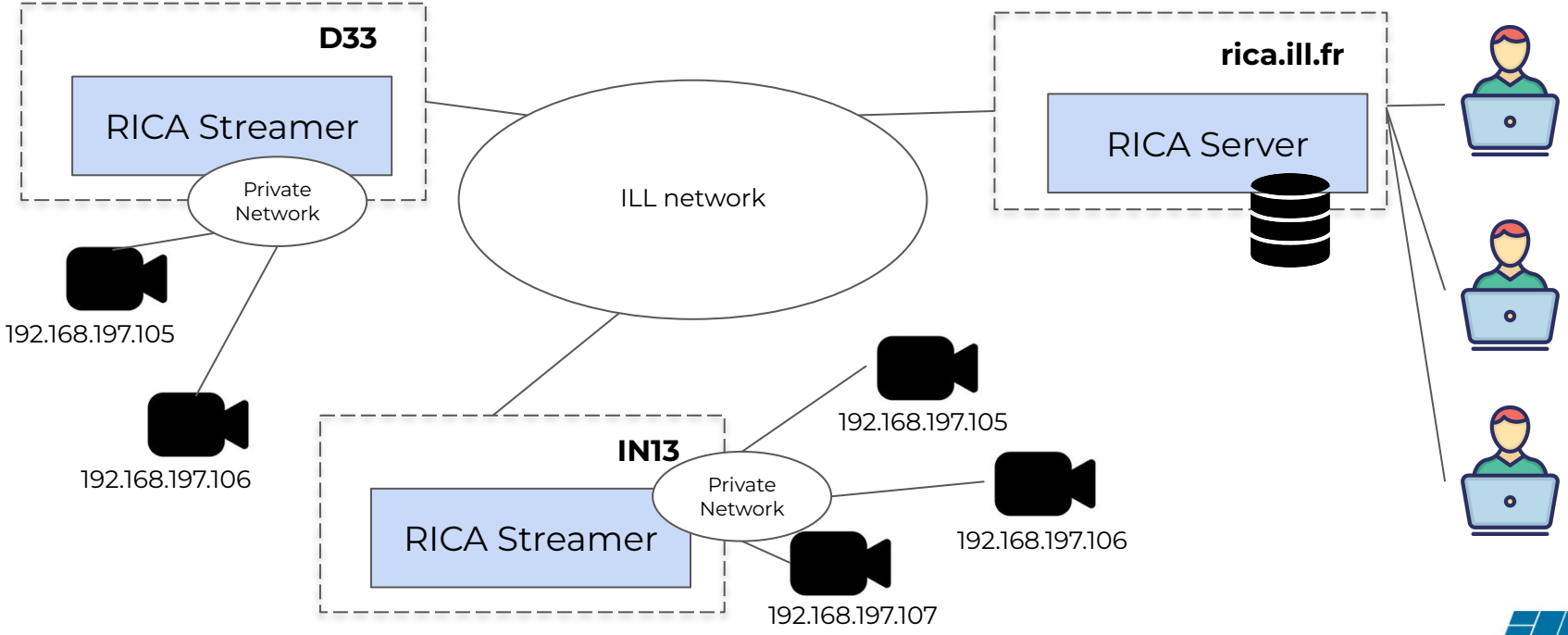
Benchmark was done to validate the prototype:

- Streamer load not influenced by the number of clients: only one stream per camera.
- Server load depends on number of clients.
  - Load balancing can be implemented if needed.
- Camera configuration is important.
  - Resolution from HD to 4k increases 50% the CPU of streamer.

For security reasons the cameras are in the private network of the instrument so the streamer has to run in the instrument computer.

Final architecture of the project

One central server and a streamer on each instrument where cameras were installed.



Authentication and authorization

Authentication

- Use OpenID Connect protocol.

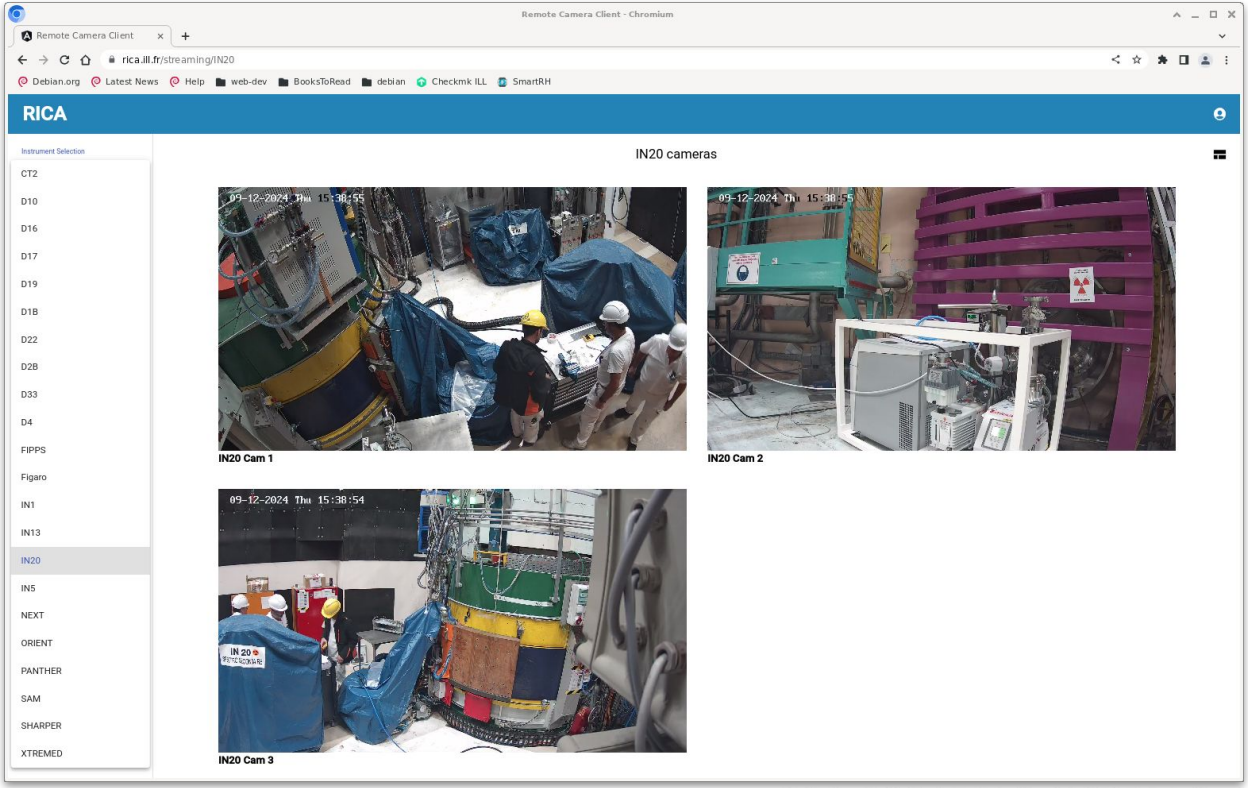
Authorization

Rules depend on who is accessing:

- **ILL staff:** access to all of the instrument cameras all the time.
- **External researchers:** access only to the instrument cameras of their active experiments.
  - Streaming stops when experiment is finished.

RICA nowadays

RICA is running since end of 2022 in 20 instruments so far.



Future work

Add more cameras!

Improve admin tools:

- Configure the camera.
- Add system metrics.

Evaluate using Low-Latency DASH:

- Check disk usage.

Improve the client providing more functionalities.