05/05/2025

# Asynchronous Internode Media Transfer with Libfabric

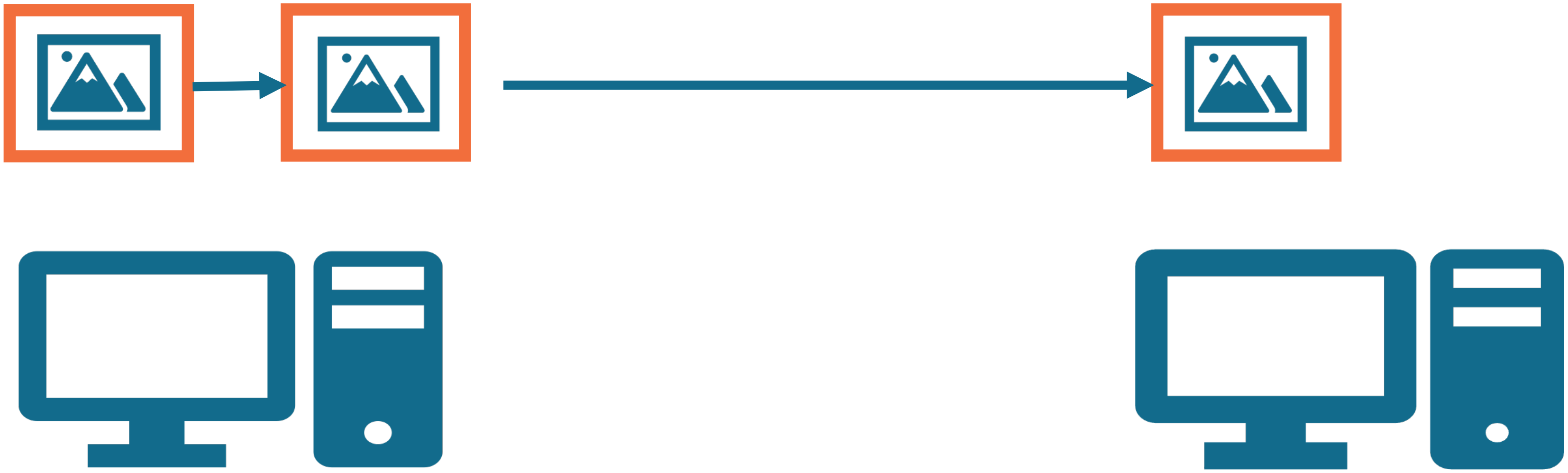
Michael Lefebvre
Sithideth Viengkhou

2025-02

RIEDEL

# The need: Exchanging data between hosts

- The Broadcast industry is moving to Software based computing,
- Data exchange is not that different from HPC
  - But we need it to be real time



# What is Libfabric and how could it address a need?

- What if we had a framework that lets the user share memory between compute nodes without the user having to do much of the heaving lifting?

OPENFABRICS ALLIANCE

- **Libfabric**: A Framework for exporting High Performance Networking Service to application.
  - API is application driven
  - Low level
  - Abstract diverse networking technologies
  - Supported by most OS (Linux, Window, MacOS...)
  - Can be used on premise and on most Cloud Vendors
  - **Designed to be out of the way and let users focus on the application data, wherever that is.**

VIDTRANS25

# What is Libfabric, and how could it address a need?

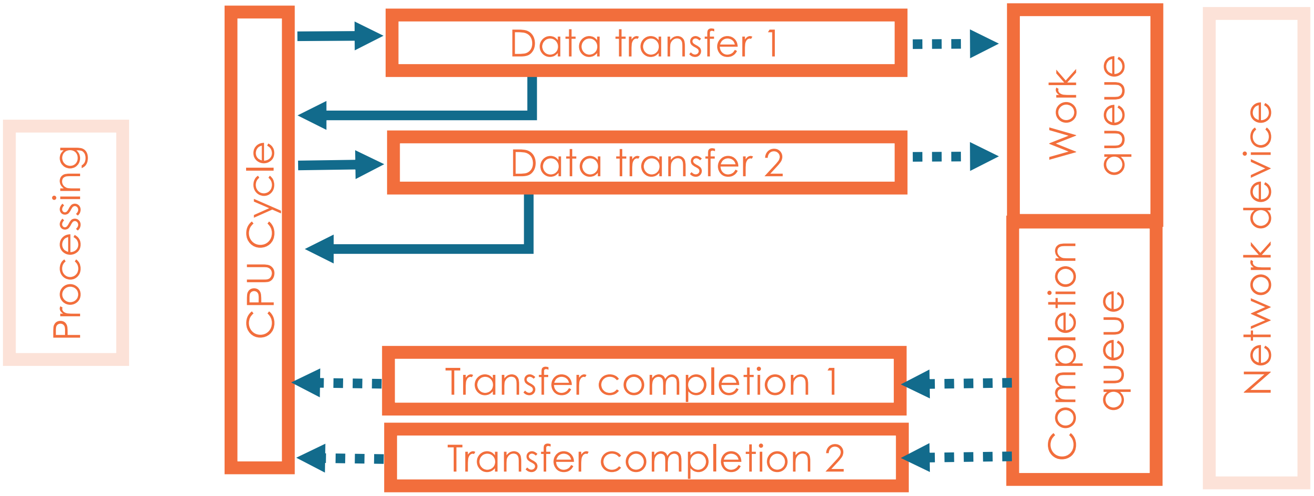
- A brief history
  - It has been developed by the Open Fabric Alliance
  - Launched in 2015
  - Active Open-source project
  - +192 Contributors
  - Used within HPC ~ 2018

- https://github.com/ofiwg/libfabric

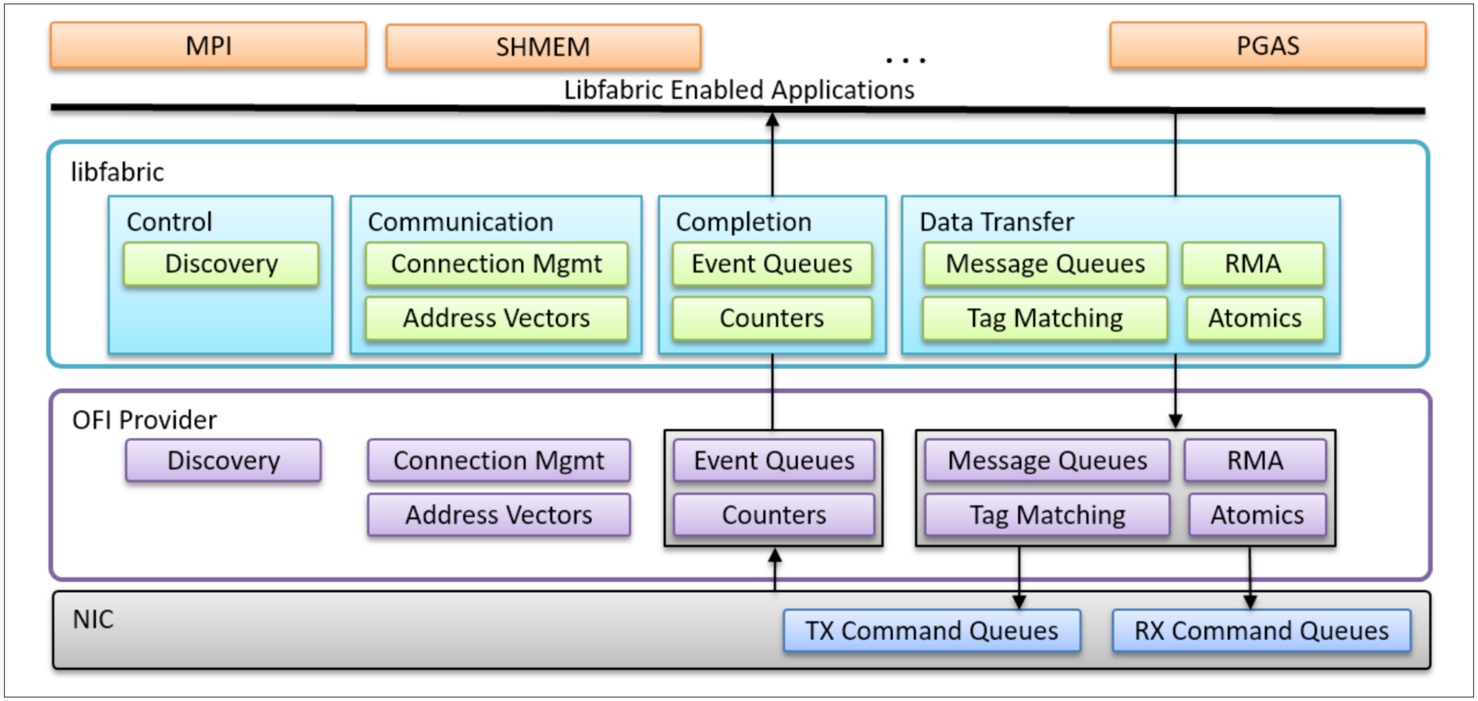
- Technical introduction
- Use case with video and audio content within multiple environments
- The cost of using Libfabric

# Asynchronous programming model

- Non-blocking submission of data transfer to queues
- Asynchronous feedback of data transfer completion
- Other threads/processes can accomplish useful work while waiting for completions



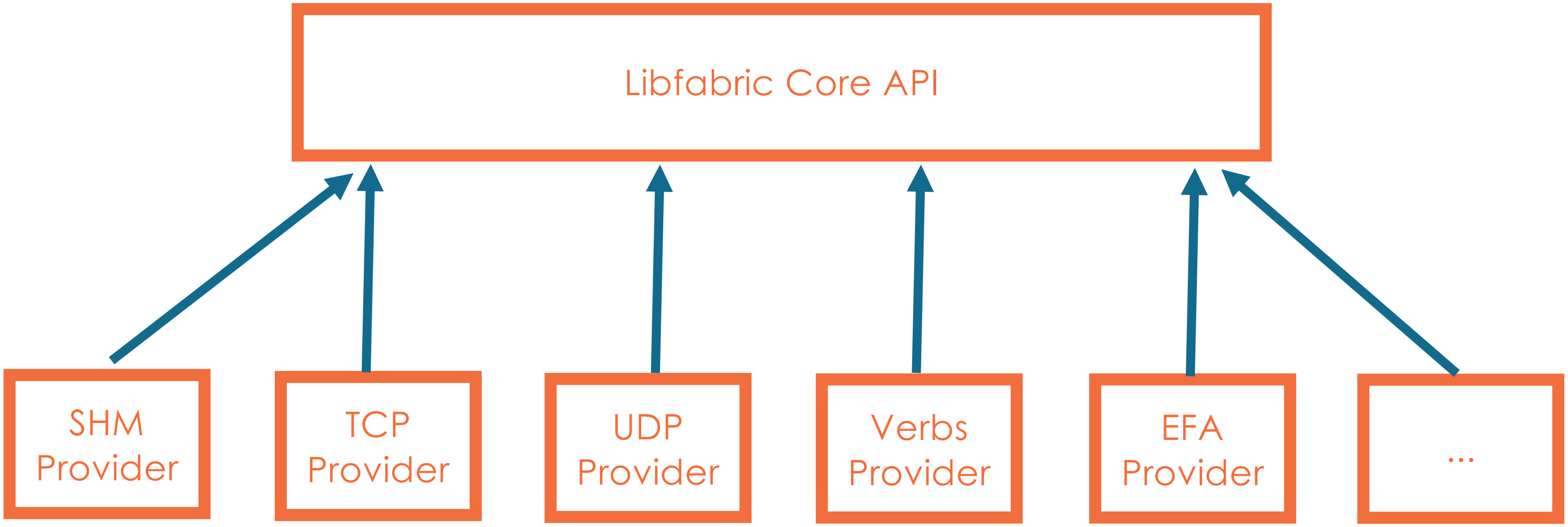
# Introduction to Libfabric



© Copyright VSF 2024
Confidential

# Libfabric providers

- Implemented by manufacturers
- Implementer can implement only a subset of Libfabric core API

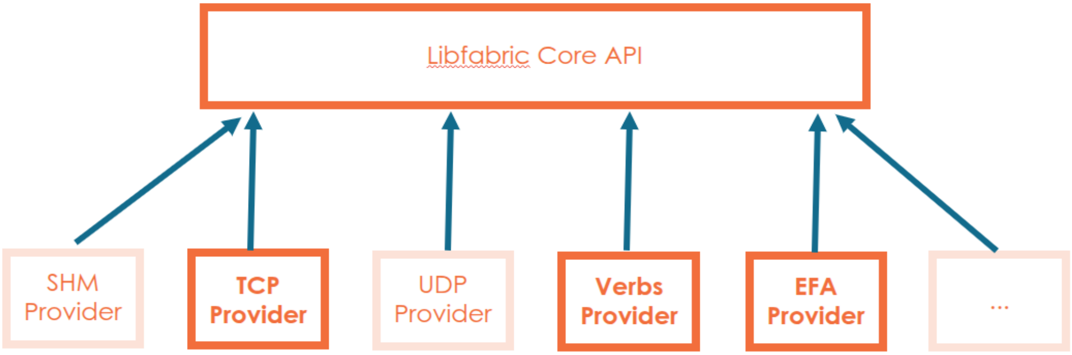


# Selecting a provider

hints->fabric_attr->prov_name = "tcp";

hints->fabric_attr->prov_name = "verbs"

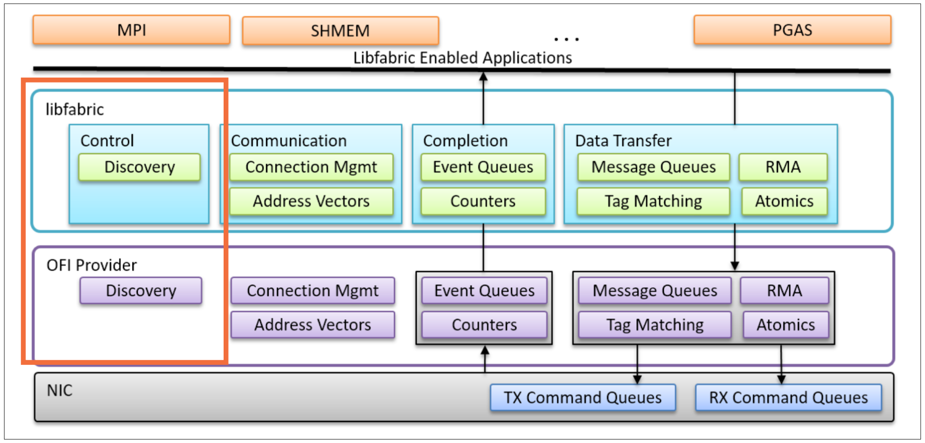
hints->fabric_attr->prov_name = "efa"



# Provider discovery API

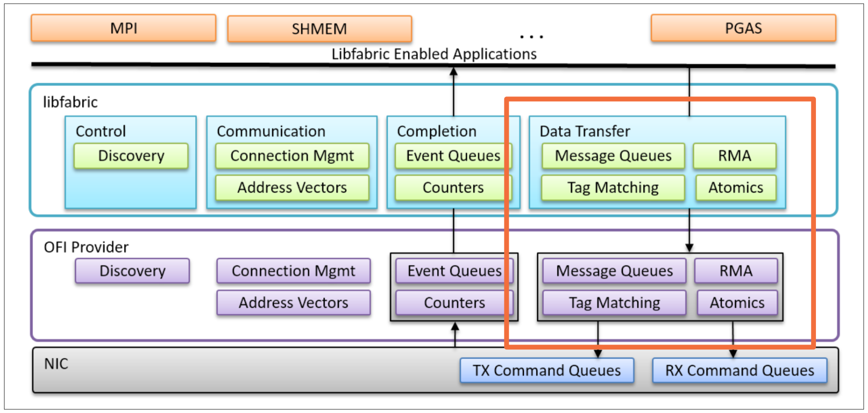
- Identifying available Libfabric devices and their capabilities
- **fi_info** binary let you inspect available providers on your node
- **fi_getinfo()** allows you to inspect and select providers

```
mlefebvre@jarvis ~> fi_info -l
opx:
   version: 122.0
ofi_rxm:
   version: 122.0
ofi_rxd:
   version: 122.0
shm:
   version: 122.0
udp:
   version: 122.0
tcp:
   version: 122.0
```



# Libfabric data transfer API

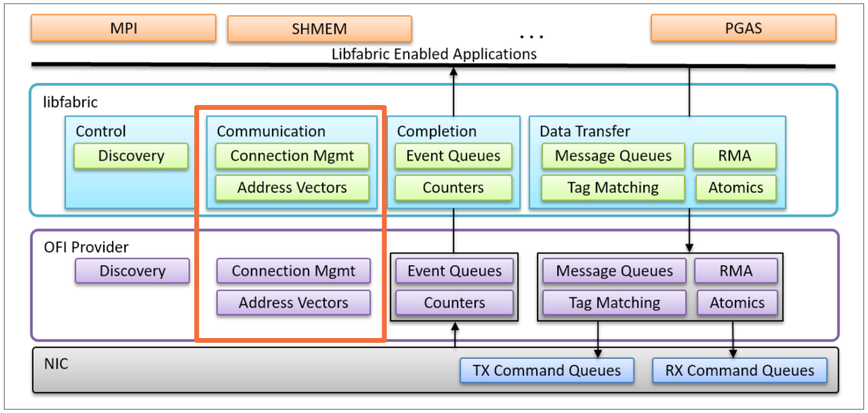
- Two-ways operations
  - Both the initiator and the target
  - Send/Recv
- One-way operations
  - Only Initiator node
  - Remote Write/Read
  - Requires memory registration
- Collective operations
  - Arbitrary number of peers
  - Complex operations



# Communication API

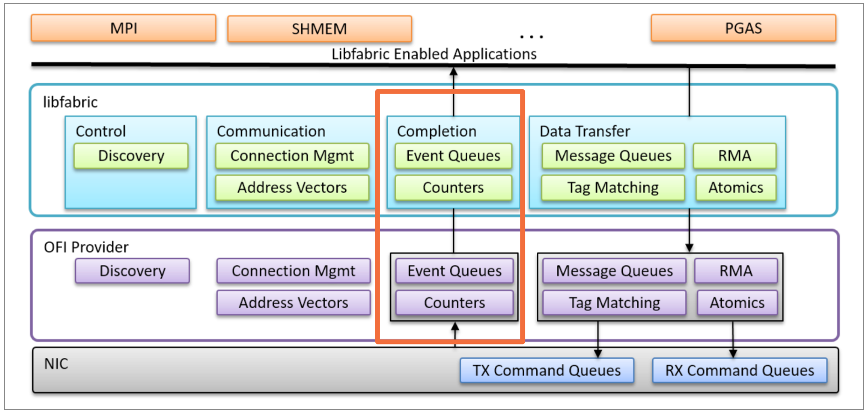
- Connected communications
  - Server/client architecture
  - Incomplete support

- Connection-less communications
  - A record of destination endpoints
  - Establish transfer paths at setup



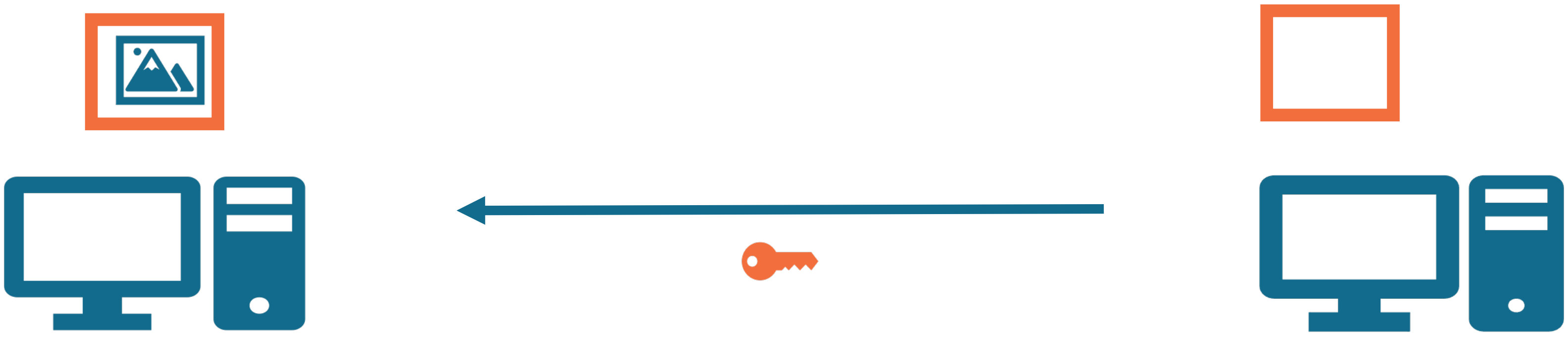
# Completion API

- Transfer completion feedbacks
- Blocking call
  - Not supported by all providers
  - Latency affected by the scheduler
- Non-blocking API
  - Choose a polling rate
  - Lowest latency possible
  - More wasted CPU cycles



# Libfabric memory registration

- Potentially giving up ownership of our application buffers
- Access permissions
- Memory descriptor for local buffer
- Remote protection key (rkey) for remote buffers
  - Grant access after key exchanges



# Step to perform transfers (tag matching)

1. Select a provider and create an endpoint
2. Create a CQ and attach to the endpoint
3. Create an AV and attach to the endpoint
4. Add peers to the AV
5. Register local buffers
6. Data transfer

   ```
   fi_tsendv(endpoint, iov, mem_descs, iov_len, dst_addr, tag, ctx);

   fi_trecvv(ep, iov, mem_descs, iov_len, FI_ADDR_UNSPEC, tag, mask,
   ctx);
   ```

7. Poll the completion queue

   ```
   fi_cq_read(cq, &entry, 1);
   ```

**Setup phase**

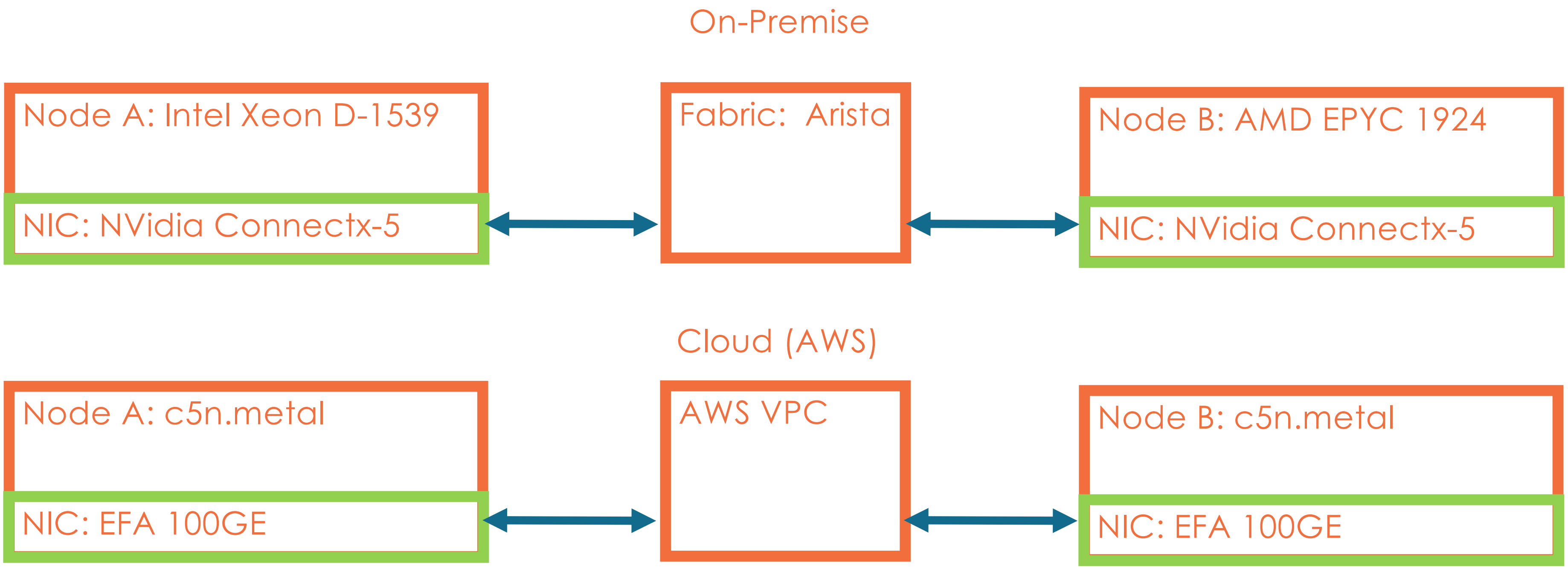
**Data Transfer phase**

- Technical introduction
- **Use case with video and audio content within multiple environments**
- The cost of using Libfabric

RIEDEL

VID 25 TRANS

# On-Prem and Cloud Setup



On-Premise

Node A: Intel Xeon D-1539

NIC: NVidia Connectx-5

Fabric:  Arista

Node B: AMD EPYC 1924

NIC: NVidia Connectx-5

Cloud (AWS)

Node A: c5n.metal

NIC: EFA 100GE

AWS VPC

Node B: c5n.metal

NIC: EFA 100GE

# Performance of different providers

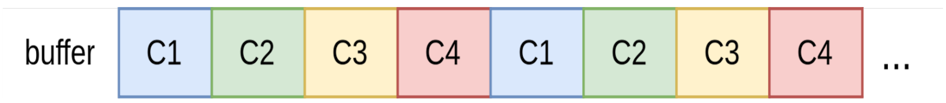
Uncompressed Video stream: 1920x1080p60 3 bytes per pixel

Media throughput: 373.248 MB/s
Max Latency: 16ms

| Provider | Throughput | Latency | CPU Usage (EPYC 1924) |
|---|---|---|---|
| TCP | 390.621 MB/s | 4.133 ms | 11% |
| Verbs | 395.1674 MB/s | 1.344 ms | 0.7% |
| EFA | 374.9351 MB/s | 2.235 ms | 3%* |

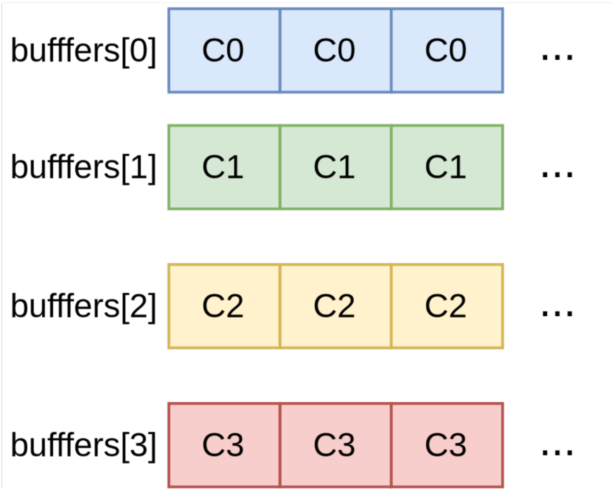
# Interleaved vs. Split audio channels

## Interleaved audio samples



| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| buffer | C1 | C2 | C3 | C4 | C1 | C2 | C3 | C4 | … |

## Split audio samples

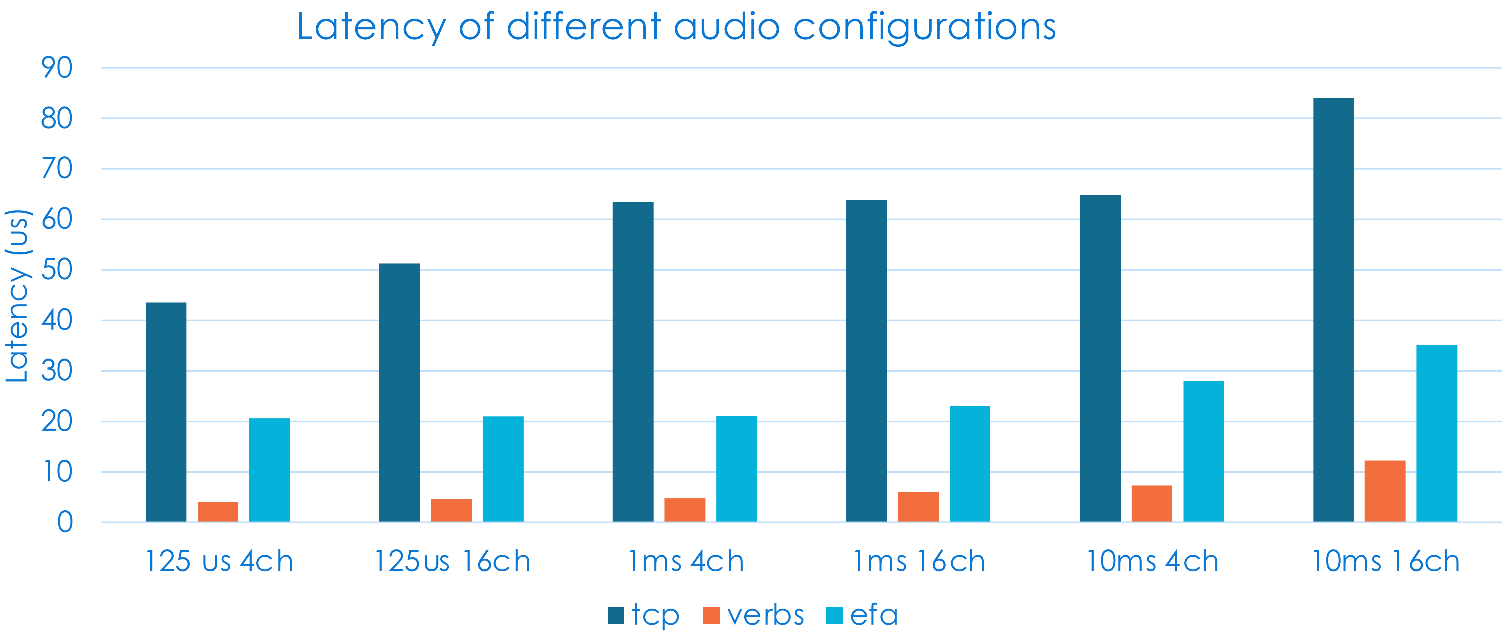


| | | | | |
|---|---|---|---|---|
| bufffers[0] | C0 | C0 | C0 | … |
| bufffers[1] | C1 | C1 | C1 | … |
| bufffers[2] | C2 | C2 | C2 | … |
| bufffers[3] | C3 | C3 | C3 | … |

# Scatter-gather for split audio channels

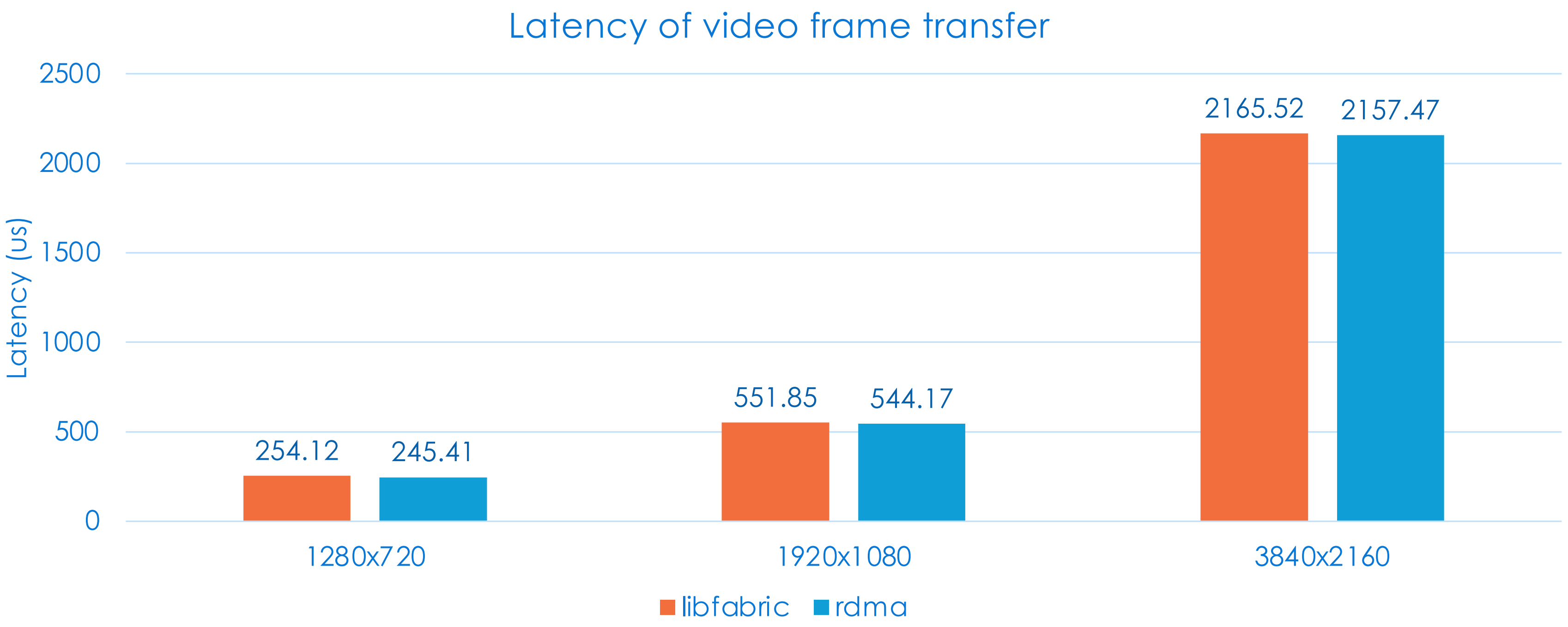
- Leverage hardware DMA scatter-gather capabilities

➤ TCP provider handles packing unpacking buffers for you

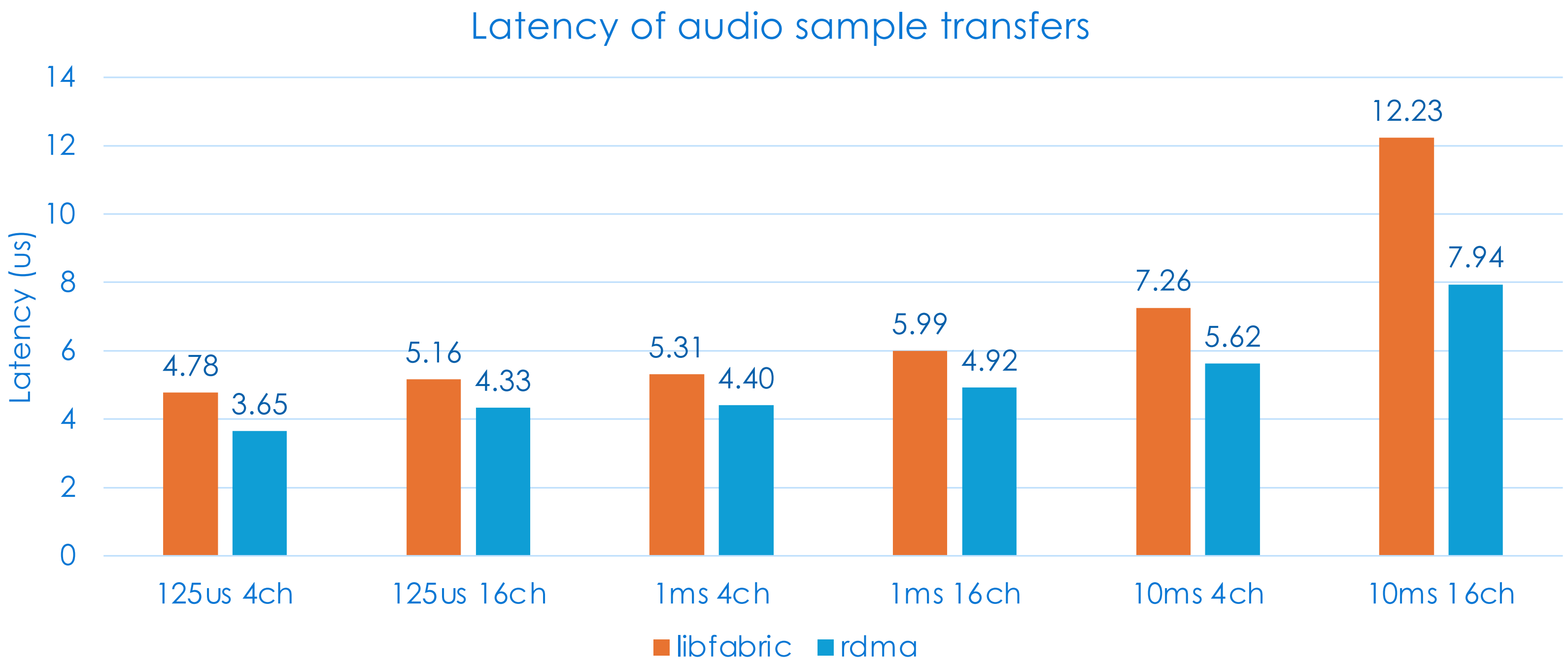


- Technical introduction
- Use case with video and audio content within multiple environments
- **The cost of using Libfabric**

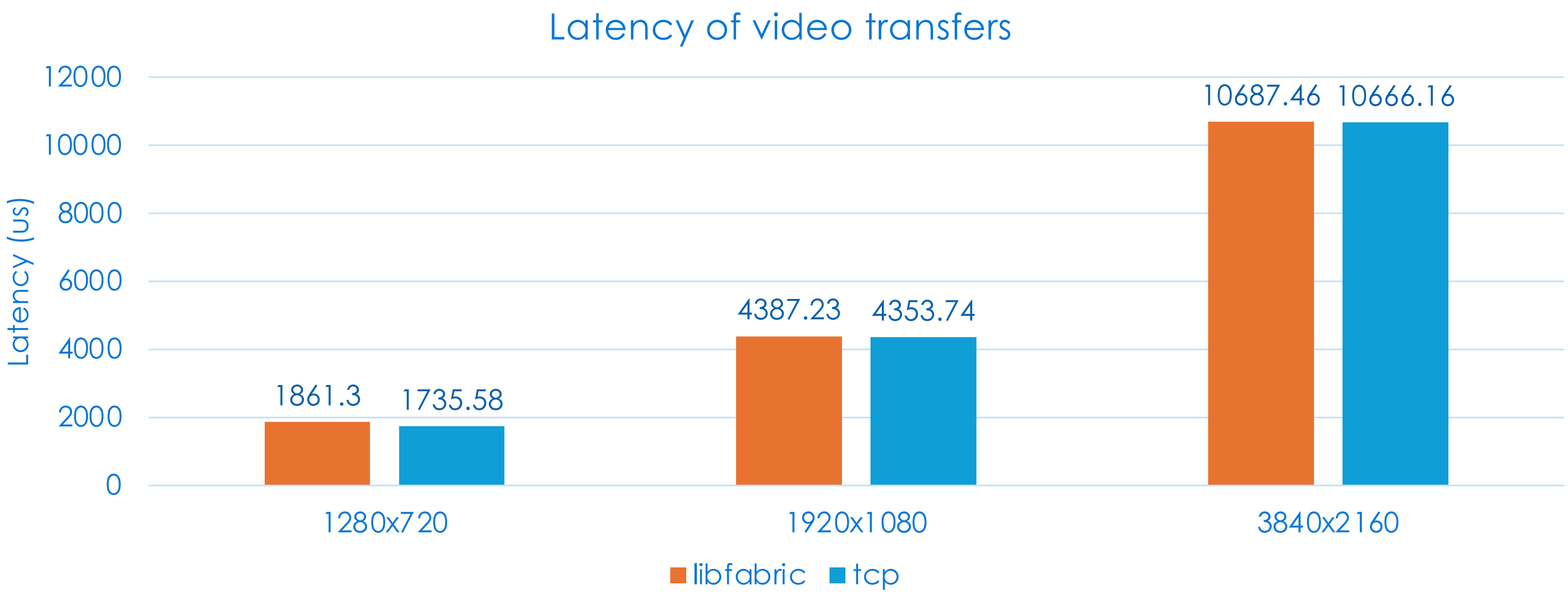
# Libfabric cost vs. Pure RDMA



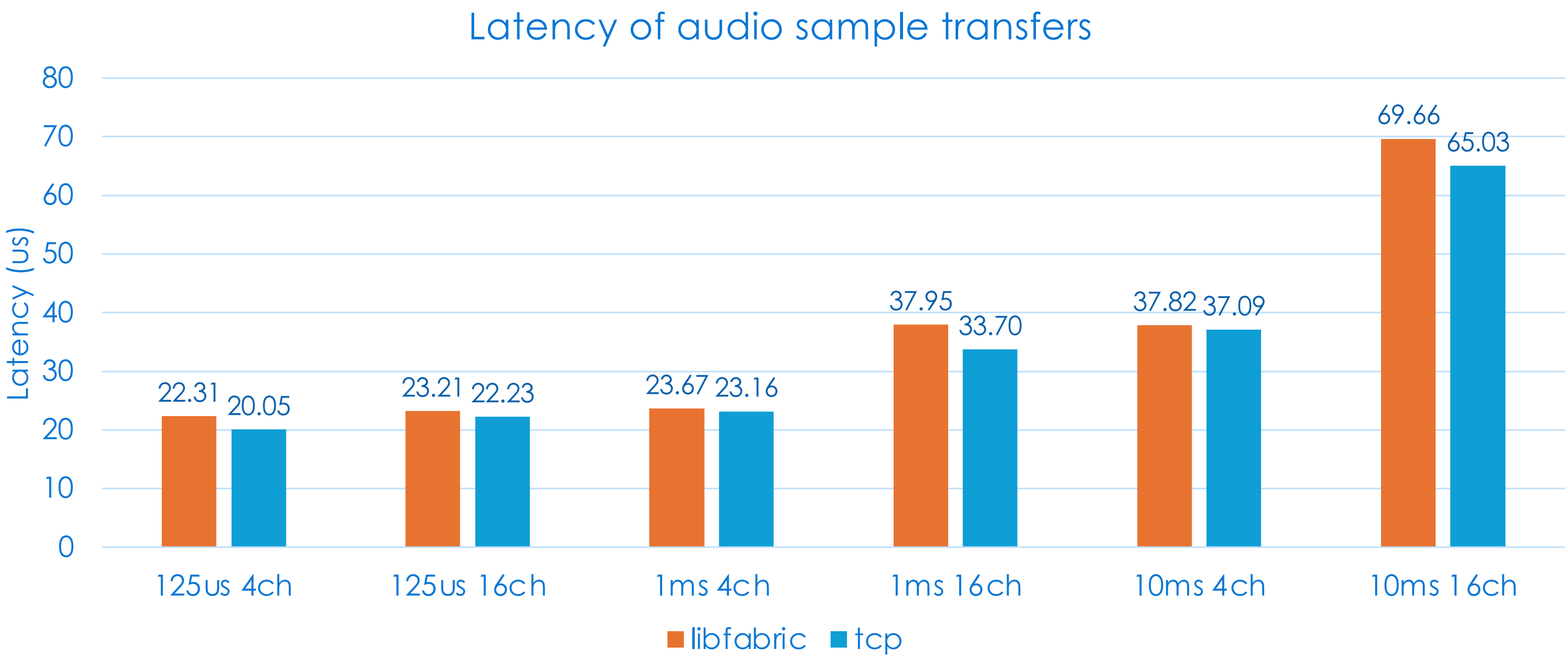
# Libfabric cost vs. Pure RDMA



# Libfabric cost vs. Pure TCP



# Libfabric cost vs. Pure TCP



# In conclusion with libfabric..

THE USER CAN FOCUS ON APPLICATION DATA

PROVIDES HIGHER LEVEL SEMANTICS

THERE'S NO REAL TRADE-OFF

VID 25
TRANS

## Thank you

vsf.tv

RIEDEL