ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Supporting vPLC Networking over TSN with Kubernetes in Industry 4.0

Lorenzo Rosa , Andrea Garbugli, Lorenzo Patera, Luca Foschini

Department of Computer Science and Engineering
University of Bologna, Italy

1st Workshop on Enhanced Network Techniques and Technologies for the Industrial IoT to Cloud Continuum (IIoT-Nets)

September 10th, 2023

New York, NY, USA

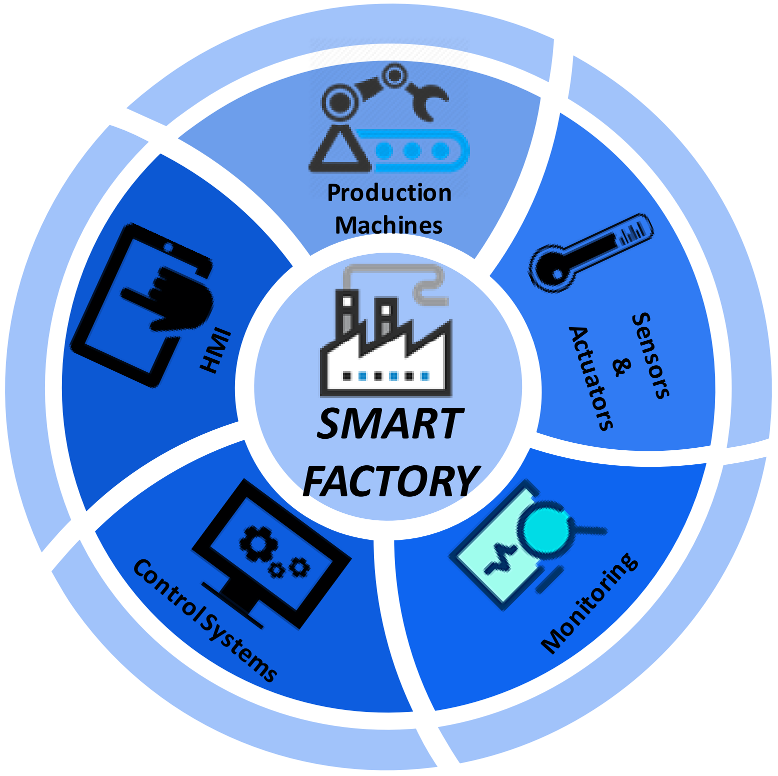
The promise of Industry 4.0

Industry 4.0: the huge amount of machine data can be analyzed to extract business insights, and to support and automate decision-making.

Industrial Internet of Things (IIoT) provides smaller, smarter, and pervasive connected devices that allow both real-time data ingestion and immediate actions on the surrounding environment.

Smart factories automate and improve industrial processes through Information Technologies (IT) for advanced integration between machinery and equipment.

Cloud-native communication enables innovative IT and networking technologies (e.g., Machine Learning) directly at the shop floor.



The IT/OT convergence: a challenging goal

Operational Technologies (OT) manage and control physical industrial processes with strong performance constrains. Traditionally closed and rigid, with *ad hoc* design, proprietary protocols, and special (expensive) hardware.

Information Technologies (IT) run software-based services, on commodity hardware, but with no performance guarantees. Traditionally open and flexible, general-purpose, standard protocols, integrated with cloud environments.

IT/OT convergence reduces operational costs, enhances flexibility, portability, maintenance and testing, improves reliability, and puts companies back in control of data ownership.

However, the demanding performance requirements of OT are difficult to satisfy with best-effort general-purpose IT.

→ **Example: vPLCs**

Virtual Programmable Logic Controllers (vPLC)

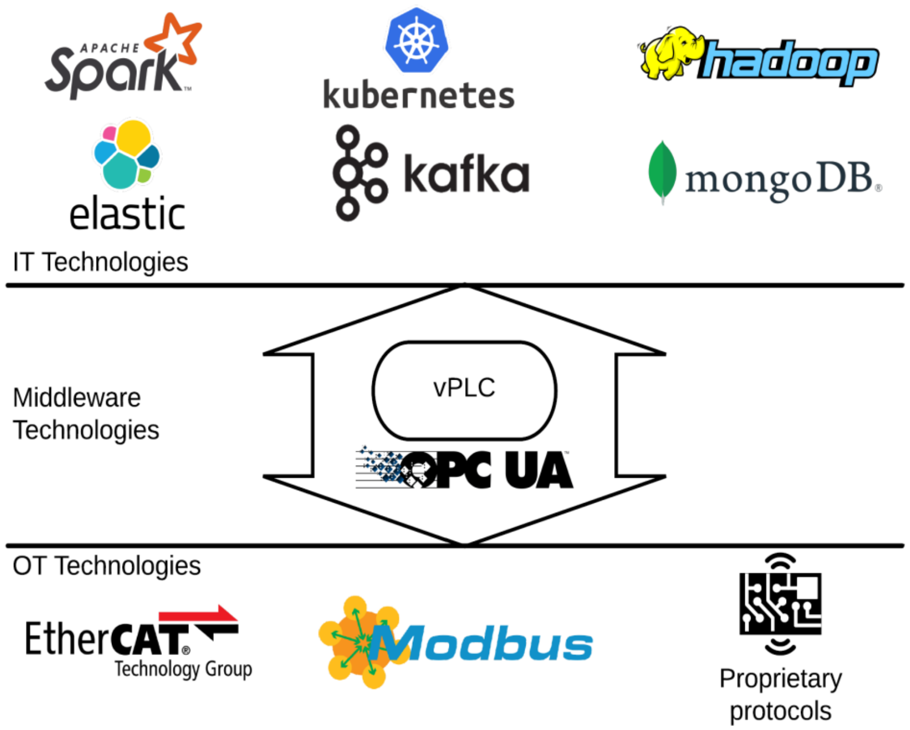
PLCs command the feedback control loops of sensors and actuators under **demanding performance constraints**.

PLCs are **specialized embedded systems**, with dedicated hardware and communication stack, requires expert programmers.

Virtual PLCs fully embrace IT. It clearly separates the <u>software control logic</u> (programmable with general-purpose languages) from the machine-specific <u>physical interface</u>.

vPLCs dramatically improve flexibility, portability, maintainability, etc., allowing the **dynamic (re)scaling and (re)conguration** of the control infrastructure and the seamless integration with the cloud.

However, general-purpose IT might introduce unacceptable delays and unpredictability for industrial environments.



Virtual Programmable Logic Controllers (vPLC)

Two major sources of variability and performance overhead for vPLCs:

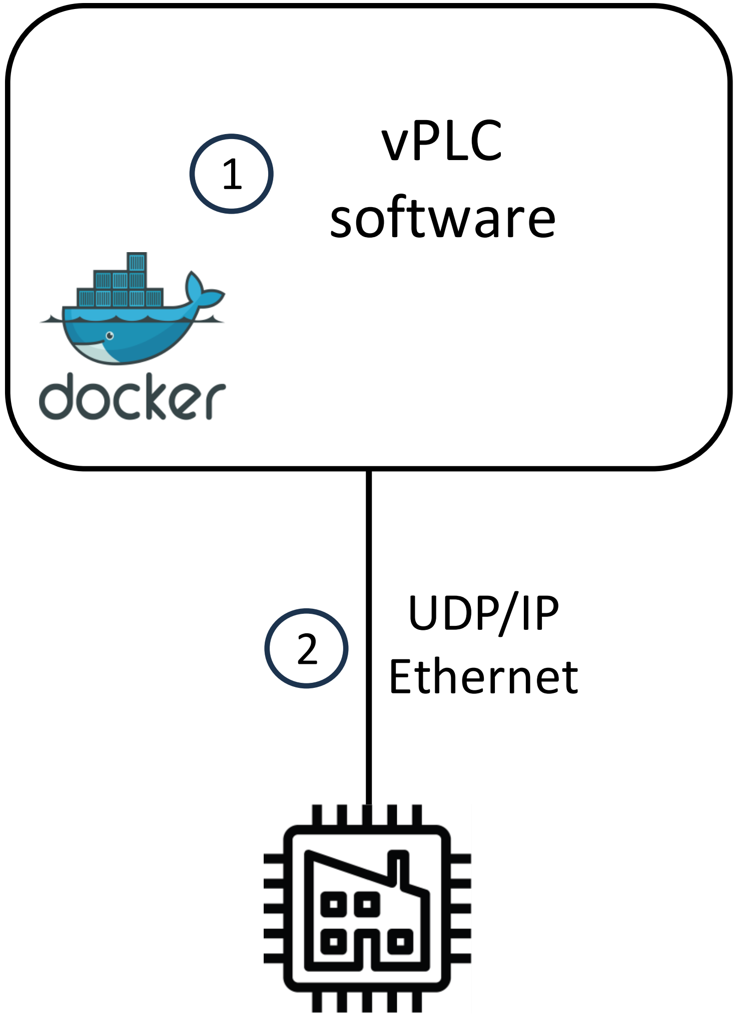
1. The use of lightweight **virtualization** mechanisms, such as Virtual Machines (VMs) or containers.

2. The adoption of **general-purpose communication** protocols and equipment.

To meet vPLC requirements, previous work on vPLCs:

- makes use of Real-Time Operating Systems (RTOS), reducing virtualization-induced variability but also hurting communication latency;
- adopts of specialized hypervisors which may reduce the virtualization overhead for critical components, but re-introduces vendor lock-in

No previous work specifically supports the communication requirements of vPLCs on general-purpose hardware and standard protocols.

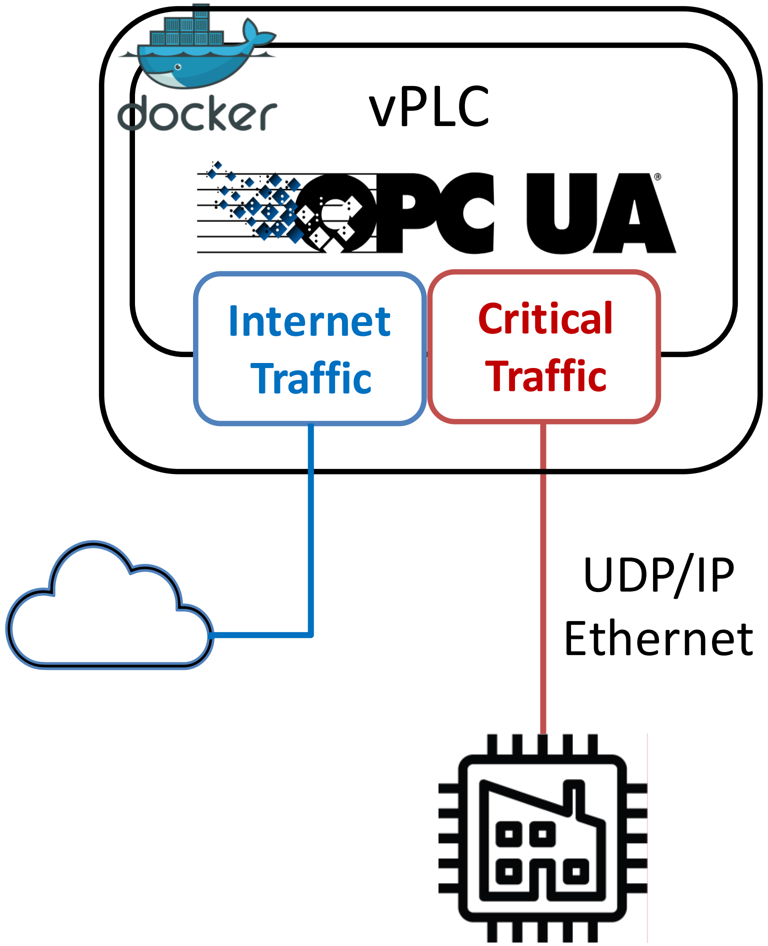


An open solution for vPLC Networking on Commodity Hardware

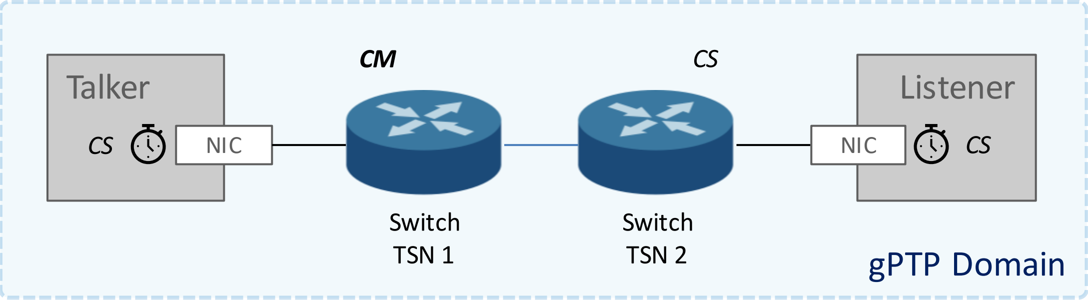
Our work aims to provide support **vPLC networking** through an **open solution** that **clearly separates** support for the <u>mixed-criticality</u> requirements of vPLCs:

- Traffic toward the **cloud**: best-effort communication through standard protocols and technologies.
- Traffic toward the **machines**: guaranteed communication in terms of (1) deterministic behavior and (2) minimal latency overhead.

To achieve that goal, we combine a set of **open source tools**: **<u>docker</u>** containers, standard protocols to ensure low network performance and variability (**<u>TSN</u>**), to ease portability (**<u>OPC-UA</u>**), and to enhance management and deployment (**<u>Kubernetes</u>**).



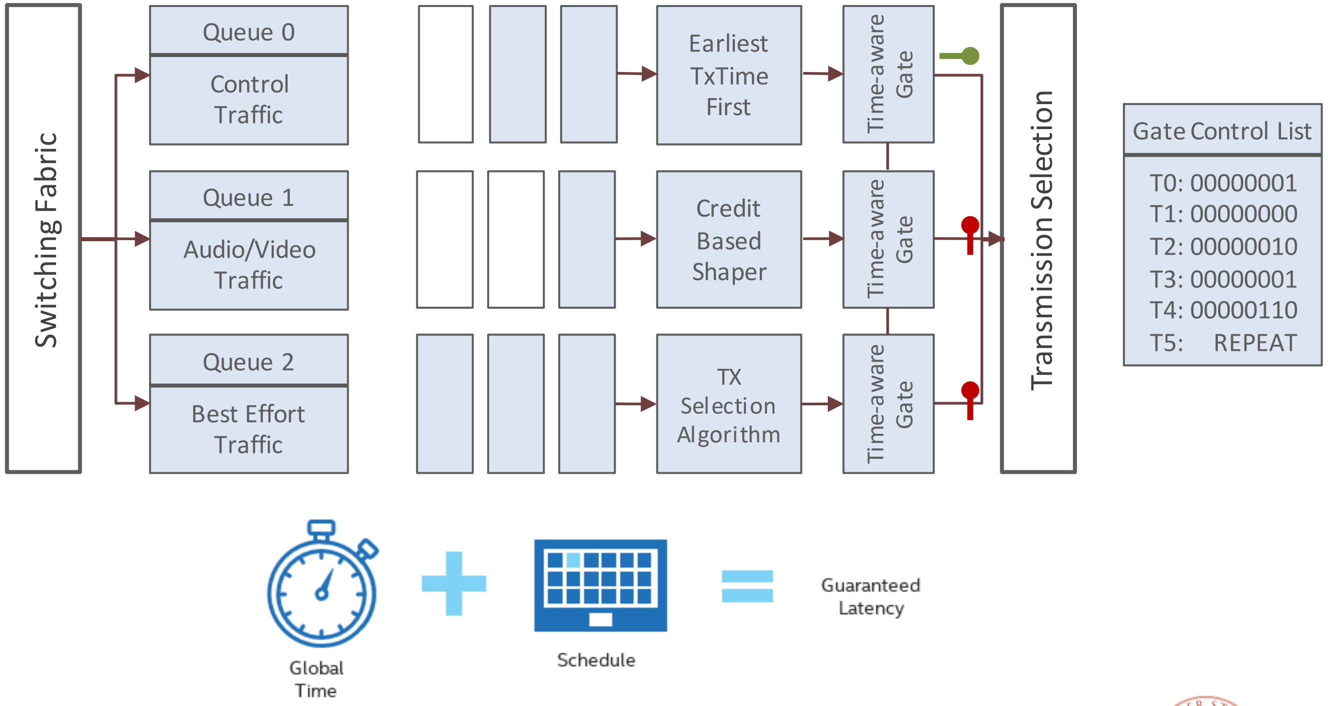
Time-Sensitive Networking (TSN)



A set of standards that make **Ethernet networks deterministic**, to support real-time industrial traffic.

TSN requires a NIC that supports:

- *Hardware clock.* Host must **synchronize** via a specific profile of the Precision Time Protocol (PTP) called generic PTP (gPTP).
- *Multi-queues*. Traffic classes associated to NIC queues.



TSN defines also **algorithms** to select the packets to be sent, and a Gate Control List to create **cyclical time-aware windows**.

Kernel-bypassing networking with DPDK

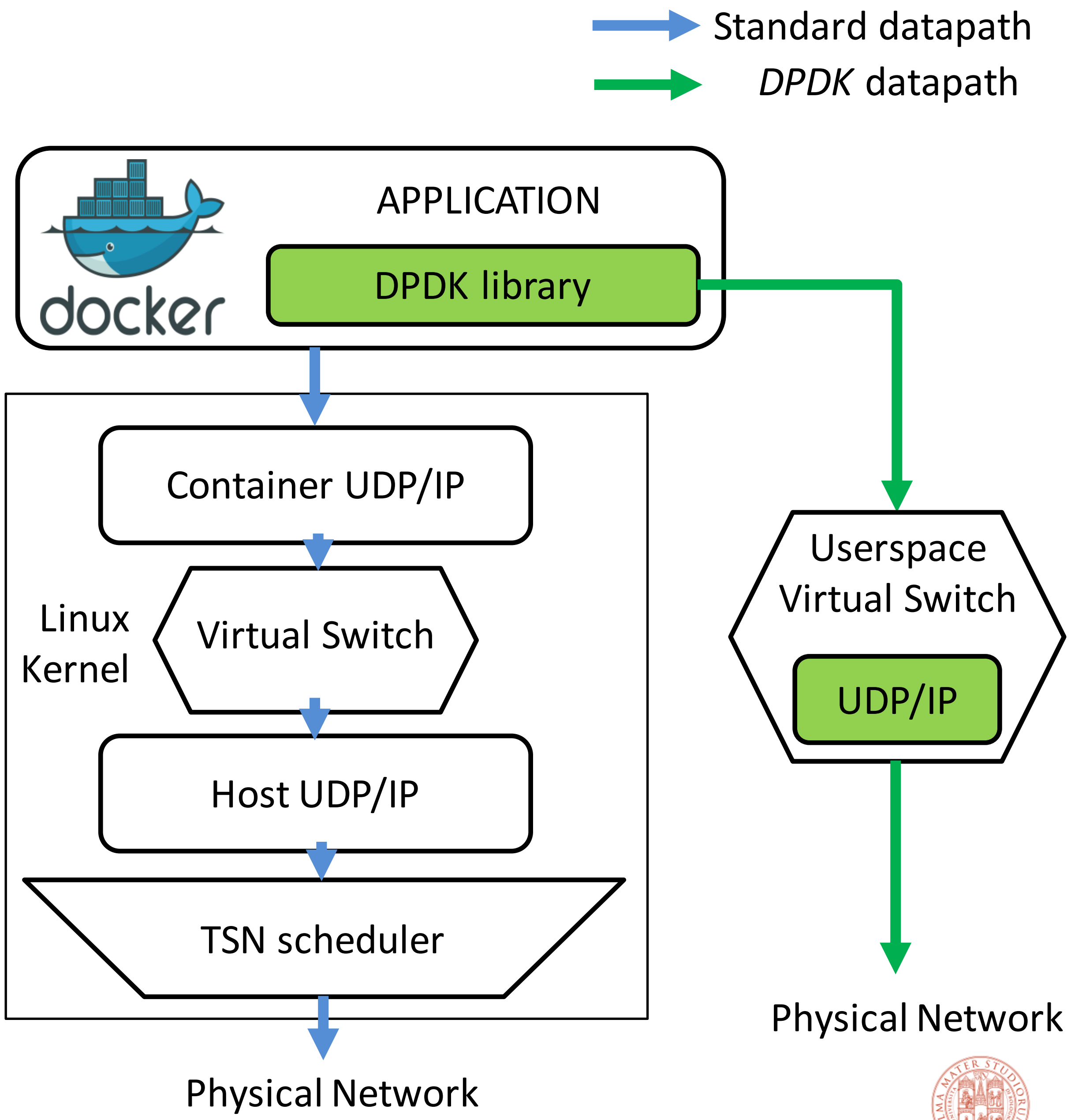
Standard UDP/IP networking from a containerized application let packets cross multiple software layers, adding significant overhead.

Kernel-bypassing solutions are **faster** because:

- Remove data copies (zero-copy transfers)
- Remove user/kernel context switches
- Uses a more efficient network stack

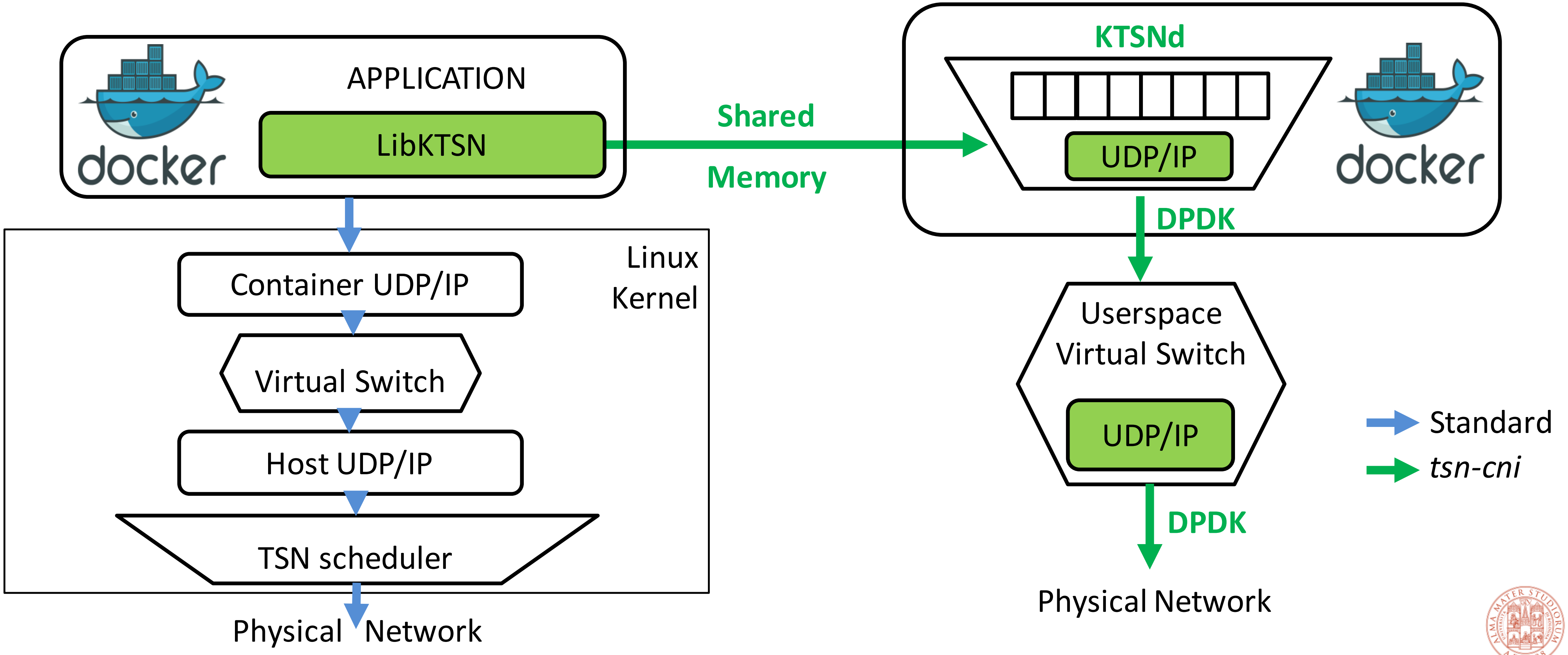
The **Data Plane Development Kit** (DPDK) is a kernel-bypassing software library that:

- Requires userspace vSwitch (e.g., OVS-DPDK)
- Removes the TSN scheduler from the datapath



KuberneTSN: A new userspace TSN scheduler for deterministic overlay networks

KuberneTSN defines a **new Kubernetes networking plugin** called ***tsn-cni*** that builds a **<u>userspace packet scheduler</u>** to configure TSN from the application container. Implemented as a daemon (*KTSNd*), it lives in a container and *shares memory* with applications.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

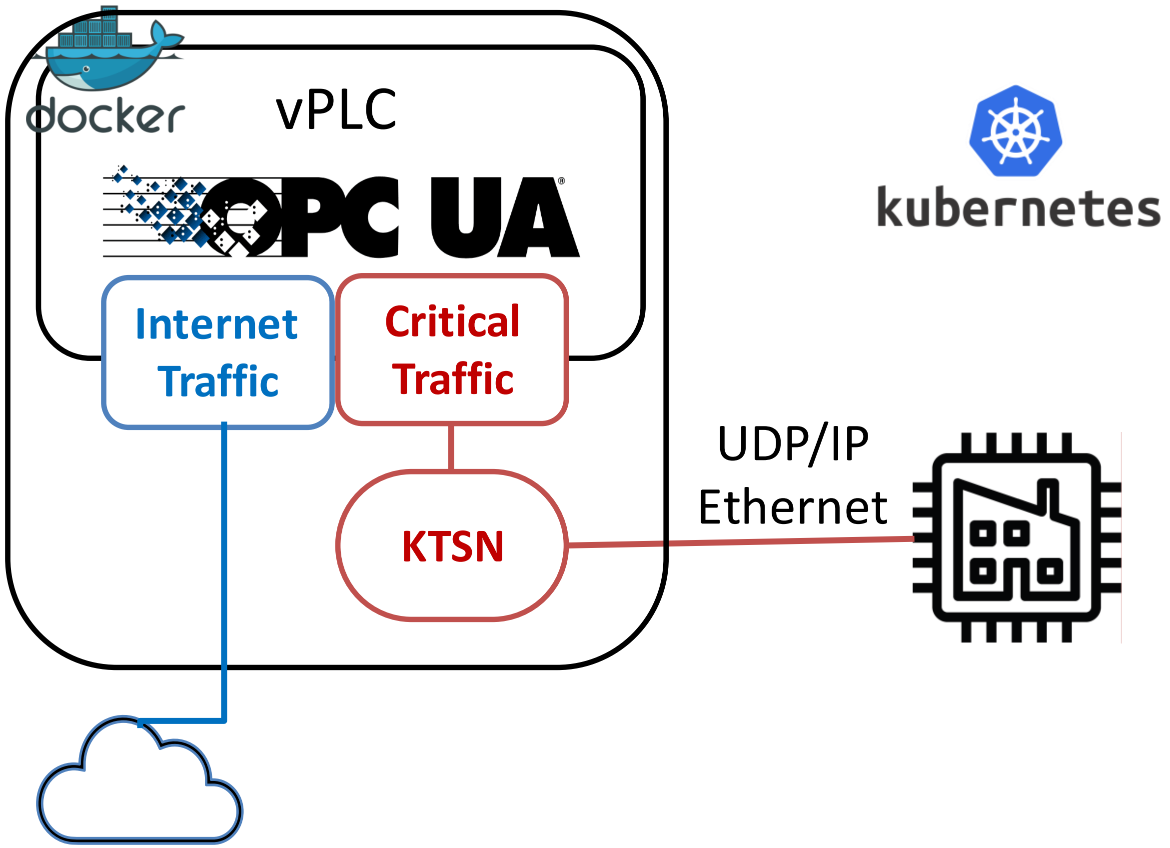
An open solution for vPLC Networking on Commodity Hardware

We propose a framework that combines a set of open-source, general-purpose technologies:

- Docker container
- Kubernetes orchestrator
- OPC-UA (TSN profile)
- **KuberneTSN** = TSN + DPDK

to support the mixed-criticality **networking requirements** of vPLCs.

Our approach significantly reduces the development and operationalization cost of traditional PLCs, guaranteeing both flexibility **and also** predictable performance, with no risk of vendor lock-in.



Performance evaluation

Goals

1. Assess the network overhead introduced by containerized vPLCs
2. Assess the network behavior and performance of our solution

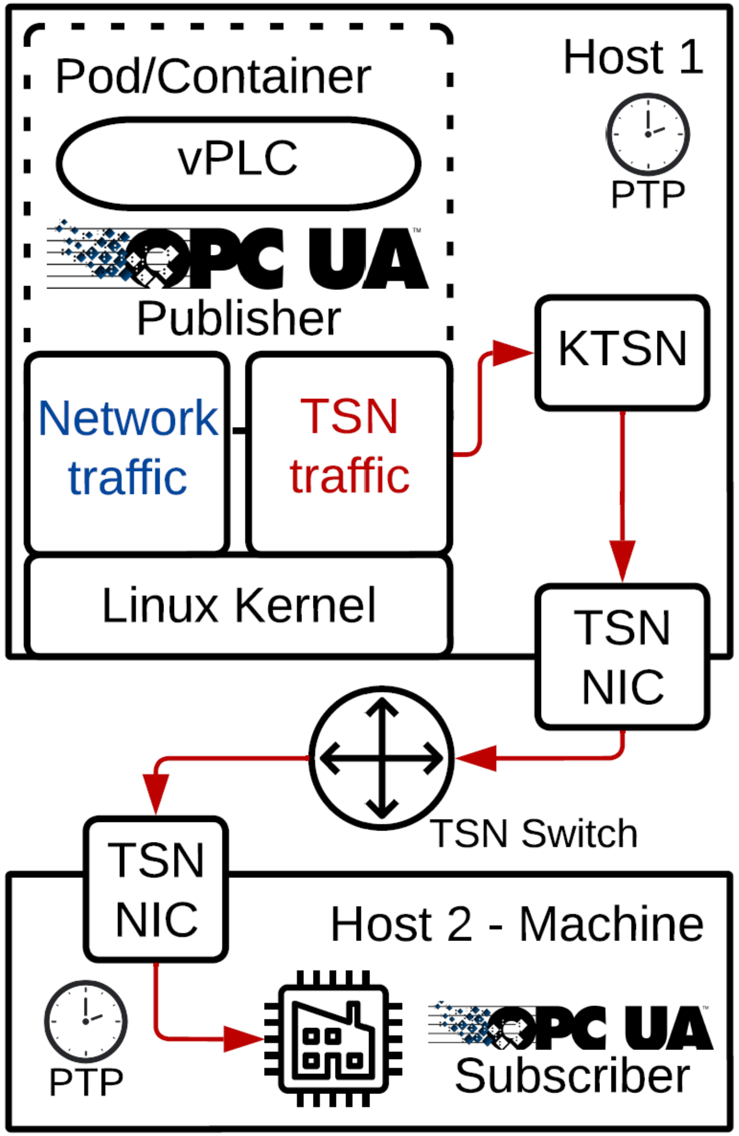
Testbed. 2 machines equipped with an Intel I225 NIC, an Intel i9-10980XE 18/36 CPU, and 64GB RAM, connected through a TSN-compliant switch.

Test App. 1 pub, 1 sub, running in containers on separate hosts, exchange UDP packets with **25 µs** publishing cycle, typical of hard real-time industrial applications.

Metric. *E2E Latency* Reception time – Scheduled Time

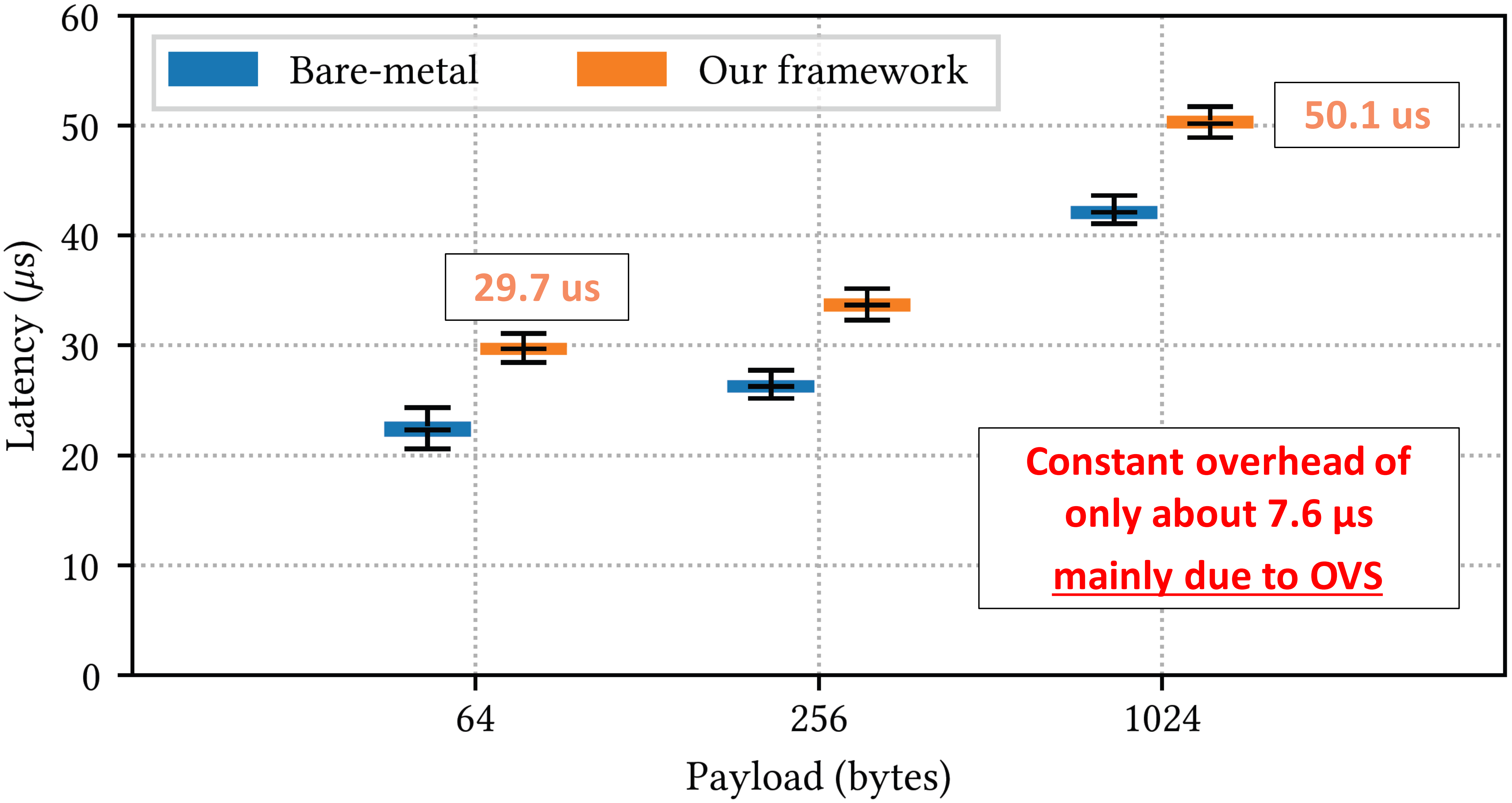
Jitter Skew from expected reception time

We **compare** the behavior of the same vPLC test application running in two configurations, within our framework and bare-metal, for different payload sizes, in a real industrial testbed.

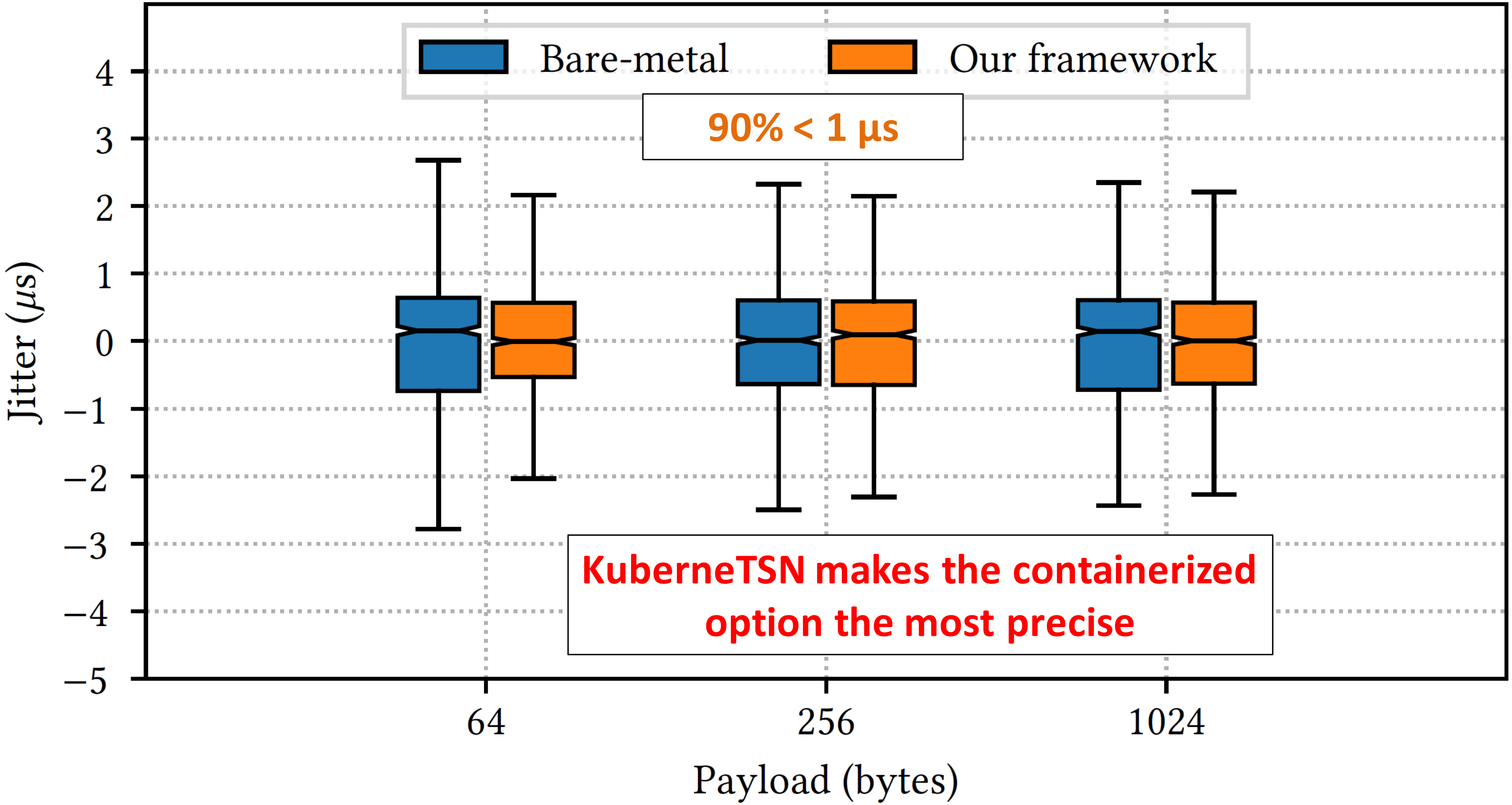


ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

Virtualization overhead: End-to-End Latency



Determinism: Jitter



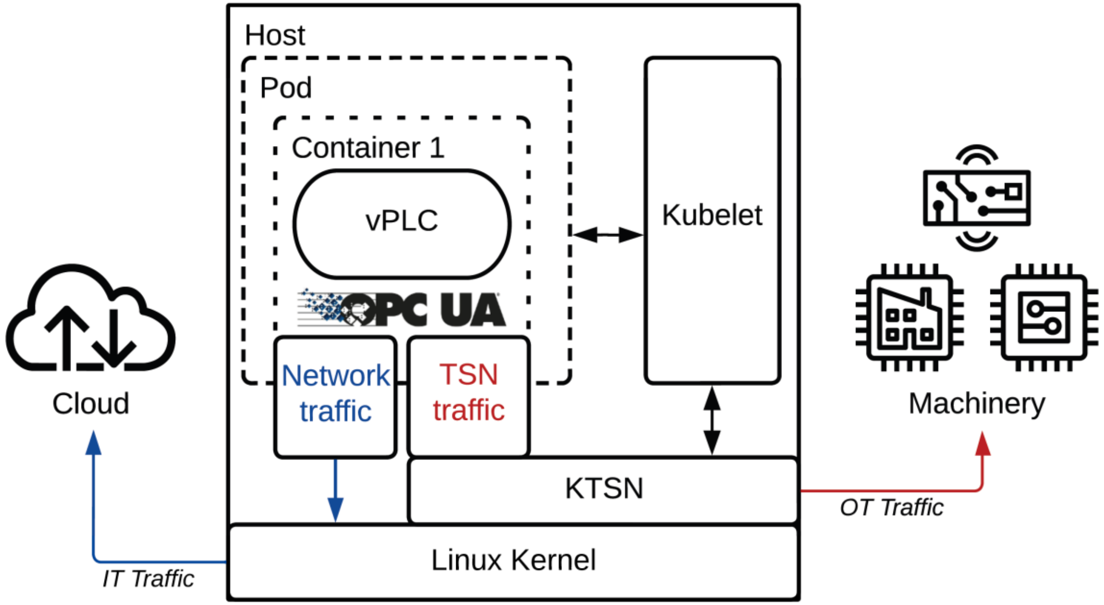
Conclusion and future work

We propose a **framework** based on **open-source tools** to support fully virtualized PLCs (vPLCs) while also satisfying the performance constraints of real-time industrial applications

The **KuberneTSN** (KTSN) solution we developed is an open-source Kubernetes networking plugin, freely available [1].

Future work

- Assess the <u>scalability</u> of our solutions in more complex settings involving multiple vPLCs and switches.
- Integrate our solution within existing initiatives for open PLC development (e.g. *Open-PLC*) and standards (e.g., *IEC 62443*).
- Investigate the adoption of AI logic in the control path



[1] https://github.com/MMw-Unibo/KuberneTSN

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Lorenzo Rosa, Andrea Garbugli, Lorenzo Patera, Luca Foschini

Department of Computer Science and Engineering
University of Bologna, Italy

lorenzo.rosa@unibo.it
andrea.garbugli@unibo.it
lorenzo.patera@unibo.it
luca.foschini@unibo.it