Kubernetes Orchestration in SDN-based Edge Network Infrastructure

Alessio Giorgetti[1,2], Davide Scano[1,3], Javad Chamanara[4], Mustafa Albado[5], Edgard Marx[6], Sean Ahearne[5], Andrea Sgambelluri[1,3], Francesco Paolucci[1], Filippo Cugini[1]

1: CNIT, Pisa, Italy; 2: IEIIT CNR, Pisa, Italy; 3: Scuola Superiore Sant'Anna, Pisa, Italy,
4: Leibniz University, Hannover, Germany; 5: DELL, Ireland 6: Eccenca, Germany
alessio.giorgetti@cnit.it, alessio.giorgetti@ieiit.cnr.it

Abstract: This demo presents a comprehensive framework providing effective cooperation among K8s scheduler, SDN controller, telemetry system, and SLA broker. The framework enables orchestrated provisioning and adaptation in distributed edge resources across a metro optical network. © 2022 The Author(s)

1. Overview

Kubernetes (K8s) has emerged as the most relevant open-source containers orchestration technology enabling automated computer application deployment and management. K8s is built on the concept of Pod (i.e., a unit of deployment) consisting of one or more containers that are co-located on a same host. The Kubernetes platform is based on a flat network structure that enables Pods to communicate with each other on their hosting K8s cluster [1]. Such flat K8s network, also called Pod network, does not account for network constraints in terms of limited bandwidth or bounded latency. For this reason, deploying K8s in edge computing environments over metro optical infrastructures to serve latency/QoS-critical applications (e.g., [2, 3]) requires a specifically designed and comprehensive framework. In particular, specific workflows are needed to efficiently interface K8s with various components such as Software Defined Networking (SDN) controller, Service Level Agreement (SLA) broker, and Telemetry Collector. So far, only few preliminary studies have addressed the K8s integration with SDN control [4-6], while the overall framework also encompassing the SLA broker and the Telemetry Collector is yet undiscussed.

In this work, we present a comprehensive framework enabling the K8s scheduler to interact with the SDN controller for deploying Pods and services in metro optical scenarios, taking into account network constraints. The framework also includes a telemetry system enabling effective SLA monitoring and enforcement. Finally, the framework encompasses an SLA broker interfaced with the telemetry system triggering the SDN controller to perform automated network adaptation upon detection of network performance degradation.

2. Components and workflow description

The demonstration will deploy an integrated environment including software and hardware components as illustrated in Fig. 1. In particular, the proposed framework is designed to operate over metro/edge data plane resources including (Fig.1, in blue):

- Disaggregated metro optical network leveraging on SDN control [7]
- Programmable P4 switches supporting in-band and/or post-card telemetry
- Edge computing nodes using a compact and modular design known as an Edge Micro Datacenter (EMDC). The EMDC enables a user-configurable hardware solution which can contain x86 or ARM CPU's, GPU's, or FPGA's according to the user's requirements at the edge.

The proposed framework for effective K8s orchestration at the edge is based on the following control plane components (Fig.1, in orange):

- **Kubernetes scheduler**: it receives a Pod specification and utilizes machine learning techniques to nominate the best possible node in the cluster that is able to accept and run the Pod. The Pod will be dispatched to the selected node and will be executed there. The node is chosen in accordance with the scheduler's objective e.g., minimizing total waiting time. To achieve its objective, the scheduler ingests the resource requirements of the Pod, resource availability of nodes of the cluster, maximum available resources, and updated network telemetry metrics such as latency and bandwidth.
- **SDN Controller**: it is based on ONOS that has been extended with a dedicated REST API to interact with other components. The developed REST interface includes methods to configure the connectivity on the network (i.e., connectivity creation/update/removal) and to dynamically activate P4 telemetry (i.e., in-band and postcard) along the traversed P4 switches.

- **Telemetry Collector**: it is devoted to collect the several telemetry flows, organize and store them into a dedicated database. Any element in the cluster and network may send its metric records. The database plays the source of truth for the telemetry data and is used by many other components such as the SLA broker, scheduler, and monitoring dashboard.
- **SLA Broker**: it is a distributed framework deployed across the architecture. Its role is to track the performance of certain system features listed in the SLA agreement and reacts based on pre-defined policies. It includes: a Rule Database, a Monitoring Engine, an Enforcer. SLA rules are described in the database representing the agreements to be held for a specific application. For example, a video processing application may ask for a specific latency threshold. The monitoring engine continuously interacts with the telemetry database to check the rules. If an SLA rule is violated, the Enforcer triggers a corrective action, e.g., notifying an administrator, requesting network rerouting, or relocating the application to another node.
- **Service Manager User Interface (SMUI)**: it provides an abstract way of managing services running at K8s edge clusters without required knowledge of the underlying data structures and APIs. The SMUI defragments the data across the distributed K8s clusters providing information such as the overall available computational resources (Pods, nodes, CPUs, storage and memory) as well as service metadata (states, logs, and failures) to the user. It also manages image and service deployment specifications. For example, it provides means for identifying the best node for running, training, or testing an AI workload while checking possible data accesses and running failures.

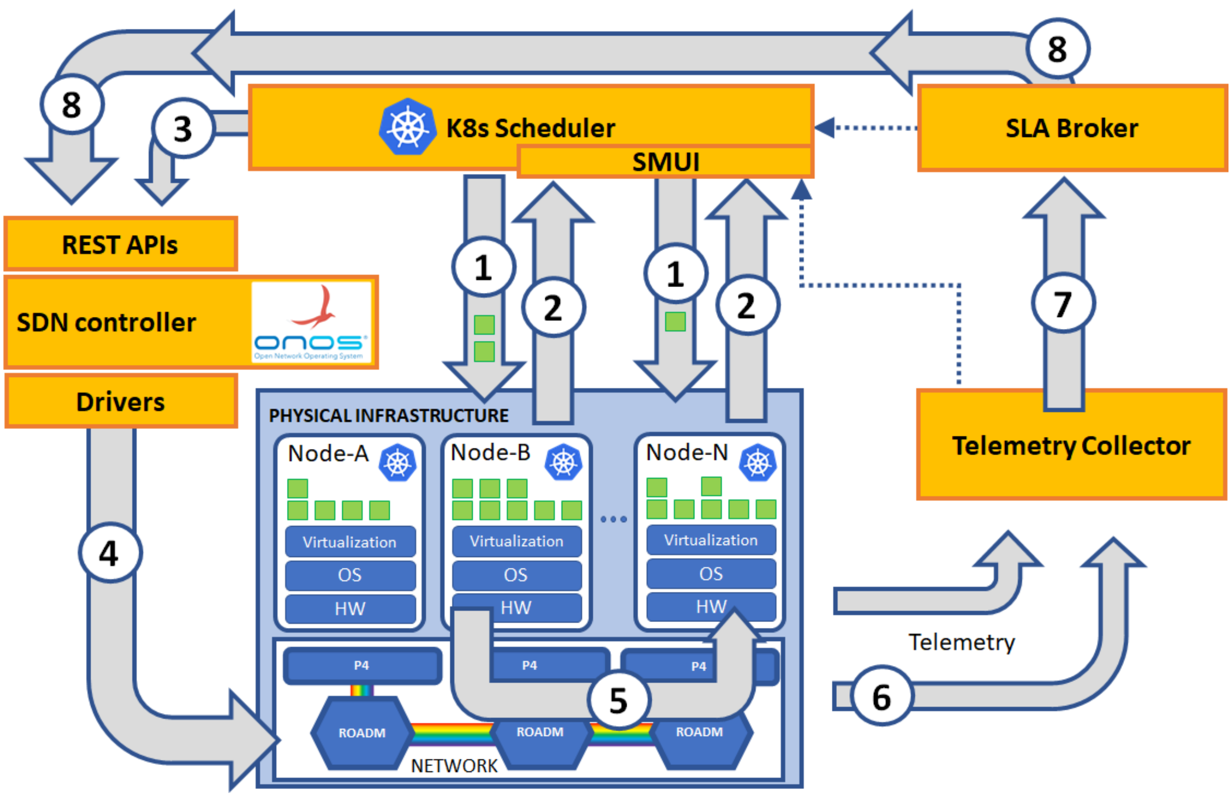

Fig. 1. Demonstration work-flow.

The demonstration work-flow realizes the following steps: **Step 1:** The scheduler places the incoming Pods with their own requirements on different nodes/locations on the metro area cluster. **Step 2:** The K8s scheduler retrieves the network parameters of the deployed Pods. **Step 3:** the K8s scheduler submits a connectivity request to the extended SDN controller feeding the specifically designed ONOS REST interface with the network parameters of the deployed Pods. **Step 4:** The SDN controller initiates the configuration of the connectivity including both the packet network based on P4 equipment (using P4-Runtime protocol) and the disaggregated metro optical network based on OpenConfig and OpenROADM yang models (using Netconf protocol), in the same step the SDN controller activates the post-card telemetry on the traversed P4 switches. The telemetry could be also started/stopped in a subsequent step. **Step 5:** Once the connectivity is configured the traffic starts to flow into the network. **Step 6:** The related postcard telemetry is generated toward the Telemetry Collector. **Step 7:** When the SLA broker, that is continuously monitoring the telemetry database, detects a service level degradation (e.g., increased packet loss or latency) it triggers a service upgrade request to the SDN controller using a dedicated method of the designed REST APIs. **Step 8:** In turn, the SDN controller modifies the network connectivity parameters in accordance with the received request (e.g., reserve more bandwidth, improve traffic priority).

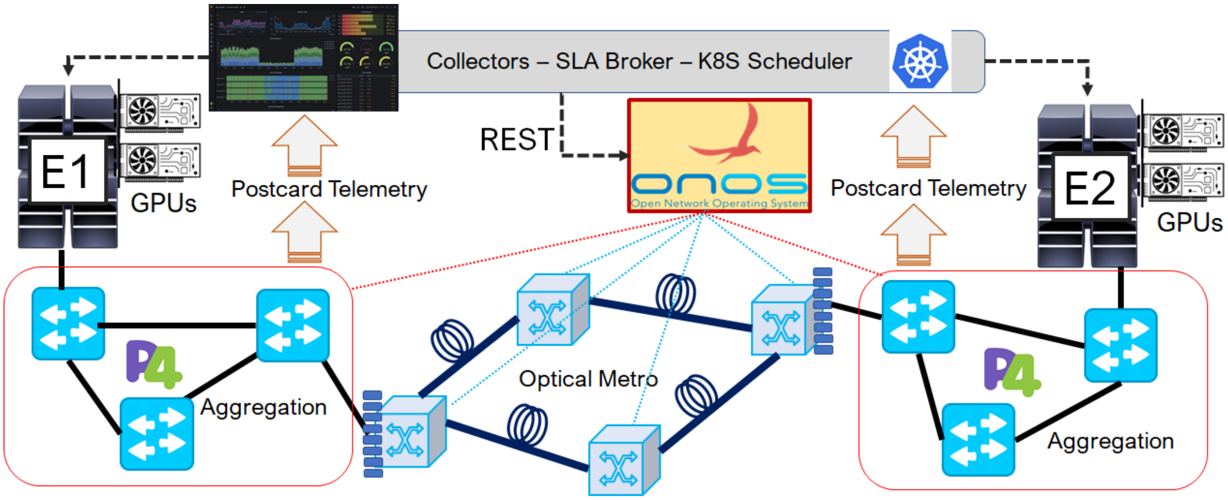

Fig. 2. Demo physical data plane setup: including computational and network resources.

3. Demo Implementation

The demo will be remotely executed on the CNIT testbed located in Pisa, Italy. The testbed encompasses: edge computing resources (three DELL nodes equipped with GPU acceleration), disaggregated metro optical network resources (ROADMs, 100G transponders, etc. [3]), and P4-based network switches (either physical and emulated switches). The demo will show dynamic K8s orchestration exploiting provisioning and adaptation of network resources efficiently coordinated by the proposed framework. On-site participation is expected with live interaction with the audience (e.g., the audience could interactively degrade the network performance triggering the network reconfiguration), providing feedback on both implementation details as well as architectural aspects. Discussions will also cover challenges and open issues for the efficient deployment of K8s orchestration in metro optical networks encompassing edge computing resources.

4. Innovation and OFC Relevance

This demo will show the first comprehensive framework enabling effective interaction among (i) the K8s orchestrator, (ii) the ONOS SDN Controller, (iii) the telemetry system, and (iv) the SLA broker. The innovative framework enables Pod orchestration in distributed edge computing resources leveraging on provisioning and adaptation of network services across a metro optical network. This demo is designed for the OFC audience, mainly telco and cloud/edge operators and vendors, interested in the potential innovation capabilities driven by a coordinated cooperation between the K8s Orchestrator and the SDN Controller of metro/edge resources.

Acknowledgment. This work has been partially supported by the BRAINE Project, funded by ECSEL Joint Undertaking under grant agreement No. 876967.

References

[1] Narūnas Kapočius et al, “Performance Studies of Kubernetes Network Solutions”, eStreams Conf., 2020
[2] Muqaddas, Abubakar, et al. "NFV orchestration over disaggregated metro optical networks with end-to-end multi-layer slicing enabling crowdsourced live video streaming", Journal of Optical Communications and Networking 13.8 (2021): D68-D79.
[3] I. Pelle et al, “Latency-sensitive Edge/Cloud Serverless Dynamic Deployment over Telemetry-based Packet-Optical Network”, IEEE Journal on Selected Areas in Communications, 2021
[4] Adeel Rafiq, “Intent-Based Slicing between Containers in SDN Overlay Network”, J. of Communications Vol. 15, No. 3, March 2020
[5] Robert Botez et al, “SDN-Based Network Slicing Mechanism for a Scalable 4G/5G Core Network: A Kubernetes Approach”, Sensors, 2021
[6] R. Figueiredo and K. Subratie, “Demo: EdgeVPN.io: Open-source Virtual Private Network for Seamless Edge Computing with Kubernetes”, IEEE/ACM Symposium on Edge Computing (SEC), 2020
[7] A. Giorgetti, A. Sgambelluri, R. Casellas, R. Morro, A. Campanella, P. Castoldi, “Control of open and disaggregated transport networks using the Open Network Operating System (ONOS)” Journal of Optical Communications and Networking,12,2, A171-A181,2020