Received March 25, 2021, accepted April 10, 2021, date of publication April 21, 2021, date of current version April 30, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3074802

Network Slicing for TSN-Based Transport Networks

SUSHMIT BHATTACHARJEE[1], KOSTAS KATSALIS[1], OSAMA AROUK[2], ROBERT SCHMIDT[2], (Graduate Student Member, IEEE), TONGTONG WANG[1], XUELI AN[1], THOMAS BAUSCHERT[3], (Member, IEEE), AND NAVID NIKAEIN[2]

[1]Huawei Technologies Duesseldorf Gmbh, 80992 Munich, Germany
[2]Eurecom, 06410 Sophia Antipolis, France
[3]Chair of Communication Networks, Technische Universität Chemnitz, 09126 Chemnitz, Germany

Corresponding author: Sushmit Bhattacharjee (sushmit.bhattacharjee@huawei.com)

This work was supported in part by the Eurecom through the EU H2020 Affordable5G Project under Grant 957317 and the EU H2020 5G-Victori Project under Grant 857201.

ABSTRACT In this work, we present and analyze methods and mechanisms for interconnecting a network slice control and management system of the mobile network, with an IEEE Time-Sensitive Network (TSN) control plane. IEEE TSN is gaining momentum as a key technology that is able to provide network service guarantees for Ethernet-based communications. Although Ultra-Reliable Low-Latency Communications (URLLC) have been thoroughly investigated in 5G, incorporating TSN technologies in the Transport Network is expected to unleash the potential of end-to-end deterministic communications, especially in industrial environments and time-critical applications like factory automation.
We elaborate on the concepts of a TSN-aware Xhaul network, present a novel architecture, and describe a set of amendments required in order to enable network slicing. With the devised approach, a slice-aware TSN-enabled transport network can be controlled and managed in an end-to-end orchestrated way. Implementation experience and evaluation results are reported using TSN-enabled prototype devices, OpenAirInterface (OAI), and JOX slice orchestrator.

INDEX TERMS DetNet, Ethernet, 5G, IEEE TSN, mobile network, network slicing, orchestration, transport network, URLLC, Xhaul.

I. INTRODUCTION

New requirements set by fascinating use cases are driving the evolution of mobile networks. Traditional siloed application environments like in the Industrial or Automotive space, are now investigating the integration of potential 5G technologies as part of their network solution (see [1] and 5G-ACIA[1]). Techniques like Network Slicing and Service Based Architecture for the core network introduced in 3rd Generation Partnership Project (3GPP) rel-15, based on cloudification, micro-services, Software-Defined Networking (SDN), and Network functions virtualization (NFV), are now offering the ability of 5G networks to self-adjust and support a number of diverse use cases with extremely different requirements. One category of use cases is focusing on Ultra-Reliable Low-Latency Communications (URLLC) services. A comprehensive survey covering both fixed and wireless ultra-low latency communication is presented in [2].

Smart manufacturing is embracing cloudification, virtualization, and programmability as the main building blocks for realizing Industry 4.0 concepts [3]. However, on the network side, vendor lock-in and fragmentation still cause high development and maintenance costs and a lack of flexibility on the way new cloudified services can be integrated. IEEE TSN is an excellent candidate technology for building Ethernet-based industrial networks, able to provide extreme latency guarantees. As future industrial networks will be based on multiple technologies like IEEE TSN, IETF Deterministic Networking (DetNet), 5G networks, and IEEE 802.11be (Wi-Fi 7), having a single open, standardized and widely adopted Ethernet-based transport network technology will unleash the potential of the new generation of Smart Manufacturing.

The associate editor coordinating the review of this manuscript and approving it for publication was Sabu M. Thampi.

[1]https://www.5g-acia.org/

As time-critical flows require end-to-end handling inside the 5G System (5GS), the concept of Time-Sensitive Communications (TSC) has emerged as part of rel-17. This is relevant not only to identification mechanisms like TSC Assistance Information (TSCAI) and Quality of Service (QoS) profiling, but also to the actual resource allocation mechanisms. In order to realize the TSC concept, strong dependency also exists on the way in which deterministic low latency communications are enabled on the transport network side, which is used to interconnect the different mobile services.

In 3GPP rel-16 and rel-17, TSC exploits the IEEE TSN family of protocols as a means to realize deterministic communications in the transport network. In order to connect the IEEE TSN control plane with the 5GS, a set of new functionalities related to QoS mapping and profiling, stream identification, and PDU session mapping have been incorporated in the 5GS architecture and the relevant processes. These are the Network-side TSN translator (NW-TT) in the User Plane Function (UPF), the Device-Side TSN translator (DS-TT) in the User Equipment (UE), and the Application Function TSN translator (AF-TT) which enables communication of the CNC (TSN controller) with the 5G Application Function (AF). AF is the entity responsible for performing application-driven decision-making on for example, traffic routing, interacting with services like Network Exposure Function (NEF), and Policy Charging Function (PCF).

Fig. 1 illustrates the different approaches of coupling TSN with the 5GS. According to current 3GPP activities, the entire 5GS is exposed as a single TSN bridge entity to the TSN network. However, TSN can be also used to support internal 5GS transport network operations for the fronthaul, but also for the entire converged Xhaul network, while at the same time serving non-5G related time-critical traffic. Both scenarios will be analyzed in detail in the following sections.

Although the interaction between 5GS and TSN is already part of the 3GPP specifications, very little research work exists on how Network Slicing can be enabled over TSN. The relevant requirements and use cases have been analyzed by studies like [1], [4]. However, no technical solution or standardized interfaces currently exist to realize the concept. In this study, we address the problem of connecting the 5G Network Slice management systems with the TSN control plane. We propose a new network architecture that is able to support network slice life-cycle management in TSN-enabled transport networks. We elaborate on the TSN for the Xhaul concept, we introduce the notion of slice-aware TSN orchestration, while we still preserve compatibility with the 5GS as a "black box" (TSN bridge) approach, from both the IEEE TSN control and data plane perspective.

Many research and standardization activities have started to deal with TSN-5G integration. From the 5G pre-standardization and standardization perspective, 5G-ACIA and 3GPP TS 23.501, TS 23.502, TS 23.503 are describing the relevant architecture aspects and also the relevant mappings and processes. Lower layer integration with TSN, for example, TSN over Passive Optical Network (PON)

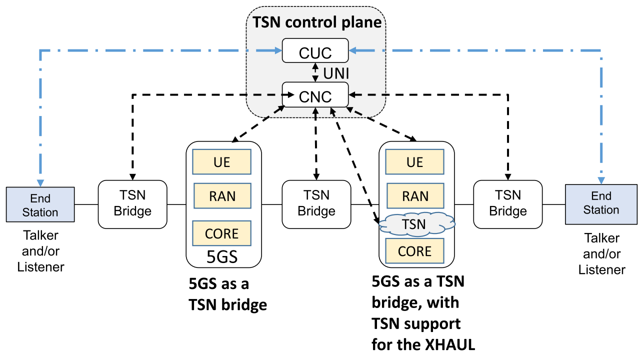


FIGURE 1. 5GS - TSN Integration: 5GS as a TSN bridge.

for the fronthaul link, is addressed in [5]. In [6], the authors propose to interconnect synchronized TSN domains through an optical backbone. In our previous work [7], we investigated the fronthaul performance when IEEE 802.1Qbv and IEEE 802.1Qbu are applied in the TSN data plane. We provide a detailed analysis of related work and current research activities in the following sections.

The contributions of this work are summarized as follows:

- We provide a detailed analysis of the current technology landscape for TSN - mobile network integration.
- We develop a novel control plane architecture for TSN networks that is able to support Network Slicing.
- We introduce the concept of a TSN slice-aware orchestration.
- We investigate how to expose capabilities of the TSN transport network and requirements to the mobile network through a multi-layer transport network controller.
- We handle the TSN-related network slice instance state at the transport network level.
- We show how to preserve slice isolation over a TSN-based Xhaul data plane and how to support slice-aware TSN network life-cycle management.
- We show how to perform coordinated actions between the TSN control plane and a multi-layer transport network controller.
- We present implementation experience using the JOX open source orchestration software [8], Open Air Interface (OAI) [9], and TSN-enabled prototype devices.

In contrast with related work, we describe in detail the way TSN technology can serve as an enabler for realizing the concept of network slicing under deterministic performance constraints. We also report implementation experience over a real testbed using a prototype solution with TSN-enabled hardware, OpenAirInterface [9], and JOX slice orchestrator [8]. An OAI-based disaggregated Radio Access Network (RAN) testbed was built, where TSN-enabled devices were used to interconnect the different components. We evaluate both the data plane and the control plane aspects under network slicing. However, because OAI does not support all the 5G functionalities, no TSN translation services (AF-TT, DS-TT) were implemented. This is planned for future work. On the data plane, our evaluation focuses on investigating the ability of TSN to differentiate real mobile traffic on a per slice basis and preserve isolation. By means of the control plane, we investigate the interaction of the TSN control plane with the orchestrator and measure both TSN network and mobile network provisioning time. More in-depth investigation of the relevant control plane aspects is planned for future work.

In section II, we provide background information and related work. In section III, we describe the key principles for incorporating TSN technology in the transport network. Subsection III-A describes both the TSN control and data plane; Subsection III-B describes TSN for the Fronthaul 802.1CM profile; Subsection III-C elaborates on how a 5GS can operate as a TSN bridge and Subsection III-D describes the case of using TSN in the Xhaul. In section IV, we describe our architectural proposal and develop methods to interconnect the TSN control plane with the network slice management system of the mobile network. In section V, we report implementation experience and performance evaluation results. In section VI, we summarize our findings and outline future research directions.

II. TECHNOLOGY LANDSCAPE

In this section, we provide some background information on the key building blocks related to network slicing for transport networks in 5G systems. The inter-working with IEEE TSN is described in the following sections.

A. NETWORK SLICING IN THE 5GS: CONCEPTS AND TECHNOLOGIES

1) 5G ARCHITECTURE

The core components of the 5G system (5GS) are the 5G Core Network (5G-CN) and the Radio Access Network (5G-RAN).

5G-CN: The 5G-CN is based on a Service-Based Architecture (SBA) comprising a set of interconnected Network Functions (NFs) like Session Management Function (SMF), Access and Mobility Management Function (AMF), Application Function (AF), and User Plane Function (UPF). Key functionalities are summarized in Table 1. The 5G architecture is defined in 3GPP TS 23.501, TS 23.502 contains the relevant procedures and TS 23.503 describes the relevant Policy, Control, and Charging architecture. Numerous studies like [10]–[14] and research projects like METIS, METIS-II, and 5G-Monarch analyzed the 5G architecture and also investigate its evolution beyond 5G.

5G-RAN & Disaggregated RAN: In 5G-RAN, the base station called gNodeB handles RAN-related functionalities similar to the eNodeB in 4G. In a 5G-RAN, the next generation eNodeB is usually denoted as gNB. The term ng-eNodeB is used to denote an LTE eNobeB that can also be connected to a 5G-Core. A 5G-RAN can comprise of gNBs or/and ng-eNodeBs. The option to connect a 5G gNB to a 4G core network also exists.

Cloud-RAN (C-RAN) was a technique introduced to decouple the radio part and the base-band processing part [15]. In C-RAN, the Remote Radio Heads (RRHs) are responsible for the lower layer PHY functions (Radio Frequency (RF), signal amplification, D/A, and A/D conversion), while the base-band processing and the higher layer protocols are performed in a centralized pool of Base Band Units (BBUs). The link between an RRH and a BBU is denoted as fronthaul.

TABLE 1. Main CN functions in 5G.

• **SMF:** responsible for controlling the life-cycle of Protocol Data Unit (PDU) sessions, according to the respective network policy.
• **AMF:** responsible for connection and mobility management.
• **AF:** is the entry point used to allow the exposure of and interaction with 5G Network resources. It is used for traffic description and traffic steering and interacts with the Policy and Control Framework (PCF).
• **UPF:** responsible for the forwarding of user traffic and connectivity to the Data Network (DN). Also responsible for per-flow QoS handling, including transport level packet marking for uplink (UL)/downlink (DL) traffic and rate limitation.

Despite the advantages introduced by C-RAN in terms of cloudification and SDN/NFV-awareness, its fundamental need for extreme bandwidth on the *Fronthaul* link makes its practical deployment difficult. The *functional split* concept was introduced to relocate RAN functions (like modulation/demodulation) from the centralized BBU pool to the RRHs [16]. Furthermore, a single BBU can be further subdivided into a centralized entity where for example, the PDCP layer is processed, and an entity where for example, the RLC/MAC functions are executed. The PHY layer is handled in the Radio Unit (RU), while a BBU can be decomposed into Centralized and Distributed Units (CUs and DUs respectively) to flexibly deploy RAN functions on different locations (cloud sites).

In [16], [17], a comprehensive analysis is presented describing the concept and the terminology mapping between different organizations and standardization bodies working on the functional split concept like 3rd Generation Partnership Project (3GPP), enhanced Common Public Radio Interface (eCPRI), extensible Radio Access Network (xRAN), Telecom Infra Project (TIP), Telecommunications Technology Association (TTA), Small Cell Forum (SCF), IEEE 1914 Next Generation Fronthaul Interfaces (NGFI). In Table 2, we summarize the terminologies adopted by 3GPP. Since our work focuses on the deterministic transmission aspects of the Ethernet-based 5G transport network, we suggest interested readers to refer to [16], [17] for an extensive discussion and analysis of all functional splits and related terminologies. Note, that several requirements (e.g., throughput, delay, and jitter) on the fronthaul link strongly depend on the chosen functional split. Although the functional split affects the actual physical network topology (where CU/DU entities are deployed), in our devised approach from the TSN slicing management and control perspective, this is transparent and any possible functional split can be supported.

TABLE 2. Disaggregated RAN: 3GPP terminology.

- Low Layer Split (**LLS**): split between RU and remaining RAN protocols.
- High Layer Split (**HLS**): split between DU and CU protocols.
- Radio Unit (**RU**): processes all protocol layers below LLS.
- Distributed Unit (**DU**): processes protocol layers between LLS and HLS.
- Centralized Unit (**CU**): processes all protocol layers above HLS and terminates inter-RAN interfaces. Aggregates several DUs.
- The CU can be separated into one Control Plane (CU-CP) and one or more User Planes (CU-UP). *E1* is the interface between CU-UP and CU-CP. Together, they form the gNB-CU.
- The RU plus DU represents the gNB-DU. It can be regarded as evolution of the eNodeB.

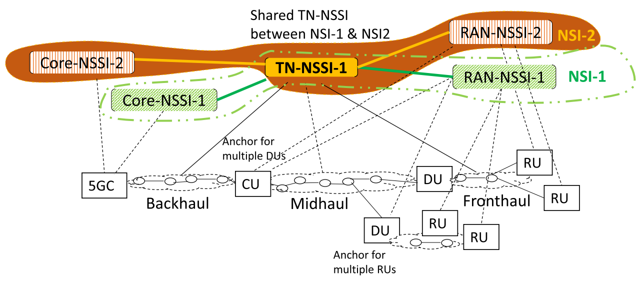


FIGURE 2. 5G Network Slicing under different functional splits. Two end-to-end slices sharing a transport network slice instance.

TABLE 3. Network slicing terminology (TS 23.501, TS 23.502, TR 28.801, TR 23.799, TS 28.530, TS 28.531, and TS 28.533).

Term	Message
Tenant	Represents an organization, agency, application, or business entity that is entitled to access the service and get guaranteed network resources through pre-defined Service Level Agreements and Policies with the network operator. In this paper, it is assumed, that a tenant can own multiple network slices.
Service Instance	An instance of an end-user service that is realized within or by a Network Slice.
Network Slice Instance (NSI)	Set of network functions and their corresponding resources which are arranged and configured to form a complete logical network to meet certain requirements of the Service Instance(s).
Sub-network Instance (NSSI)	Set of network functions and their corresponding resources which are arranged and configured to form a logical network. The Sub-network Instance is defined by a Sub-network Blueprint (TR 23.799). A Sub-network Instance is not required to form a complete logical network. A Sub-network Instance might be shared by two or more Network Slices.
Network Slice Blueprint	A complete description of the structure, configuration and the plans/work flows for instantiating and controlling the NSI during its life cycle. The Network Slice is described by a Network Slice Template (NST). The NSI is created by using the NST and instance-specific information.
Sub-Network Slice Blueprint	A Sub-network Blueprint refers to physical and logical resources and may refer to other Sub-network Blueprints. Network slice subnet template: description of the structure (and contained components) and configuration of the network slice subnet.

2) NETWORK SLICING

In principle, the evolution of the mobile network towards the 5GS was driven by the need to support a number of vertical industries and diverse use cases. These span from enhanced Mobile Broadband (eMBB), Massive Internet of Things (IoT), and URLLC to Vehicular to everything communication (V2X). The ability to concurrently support use cases with different requirements and KPIs is realized by the introduction of Network Slicing. Network Slicing enables multi-tenancy and allows the creation of end-to-end logical networks tailored to the needs of different use cases. These logical networks use shared physical resources by exploiting cloudification, Software-Defined Networking (SDN), and Network Function Virtualisation (NFV) technologies [18].

A realization of a Network Slice is called Network Slice Instance (NSI). An end-to-end NSI may be composed of Sub-network Instances (NSSIs), where a single NSSI may be shared by multiple NSIs. For example, an end-to-end NSI is composed of a RAN-NSSI connected to a 5G-Core NSSI through a Transport Network NSSI (TN-NSSI). In this paper, we investigate the way to create and manage TSN TN-NSSIs. As an example, in Fig. 2, a TN-NSSI is shared between two different NSIs. Note, that as NSSIs can be extremely complex, an important step for their design and creation is related to Network Slice Blueprints/templates. A summary of the key terms used in network slicing is provided in Table 3. In section IV, we describe our technical approach for the integration of the network slice management system with the TSN control plane. We also investigate how network slice templates can be used to facilitate the creation of TN-NSSIs by a TSN-aware transport network.

We highlight that 5G Network Slicing considers two concrete operations. The first one is the control of the life-cycle of network slice instances. As defined in TR 28.801 and TS 28530, the NSI life-cycle is related to the control and management of Provisioning, Instantiation/Configuration, Activation, De-activation, Termination, and Modification actions. The second one is the association of a UE to a specific slice. In this paper, we are focusing on the first operation, namely how to create and control Network Slice Instances over a TSN network rather than the process of UE stream association and operation over a TSN network.

Regarding end-to-end Network Slice Management [19], 3GPP defines the following key management entities in TR 28.801 regarding the orchestration of NSIs and NSSIs:

- Communication Service Management Function (CSMF): Responsible for translating the communication service requirements to network slice requirements.
- Network Slice Management Function (NSMF): for the E2E management and orchestration of the NSI.
- Network Slice Subnet Management Function (NSSMF): Responsible for the management and orchestration of the sub-network slice instance in a specific domain, e.g., the RAN-NSSMF and the Core-NSSMF are responsible for the management of the RAN and the CN sub-network slice instances respectively and the TN-NSSMF is responsible for the orchestration and management of the Transport Network (TN) sub-network slice instance.

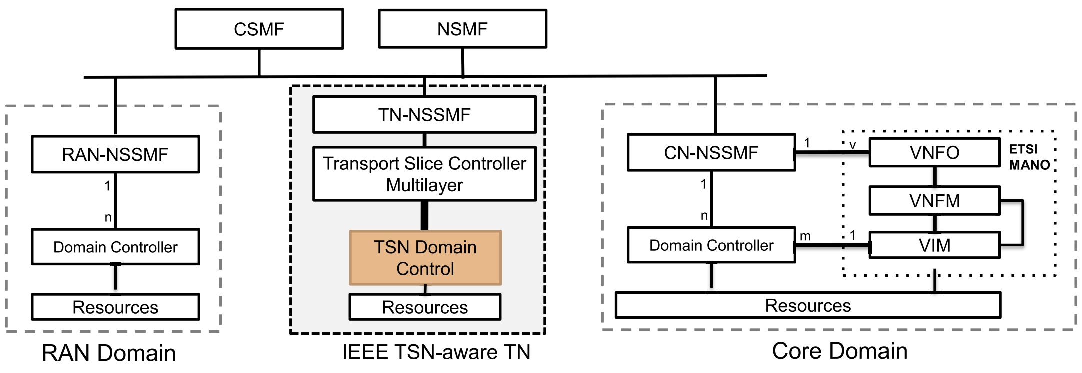


FIGURE 3. **5G Network Slice Management and Orchestration.**

Fig. 3 shows the interactions between the different management entities. For the 5G core, the ETSI MANO architecture [20] can also be incorporated. It is used for the management of both the physical and the virtual network resources and services (VNFs). How Network Function Virtualization (NFV) concepts can be incorporated also in the RAN is an open issue and currently being investigated by Open-RAN[2] and Telecom Infra Project (TIP).[3] For the transport network, industry consensus advocates that ETSI MANO is not necessary and the focus stays on SDN control. Resource allocation aspects on a per slice basis in the RAN and the Core are presented in [21], [22], while the relevant processes are described in detail in TS 23.501, TS 23.502, and TS 23.503.

Regarding the transport network, 3GPP provides the TN-NSSMF entity. However, it is not responsible for the actual operation of the TN. In section III, we present a technical approach for solving the problem of how a TSN-aware TN solution can be used by a TN-NSSMF entity. The Network Slice life-cycle management aspects described in TR 28.801 have been revised in Rel16 in different specifications, e.g., TS 28.530, TS 28.531, and TS 28.533. For example, in TS 28.530 the Network Slice as a Service (NSaaS) concept is described including the requirements for the transition to a service-based slice management architecture. In 3GPP TS 28.533, the network slice management entities are part of a service-based solution and are producing and consuming various management services. The main management entity functionalities provided, for example, from NSMF and NSSMF still remain the same.

B. TRANSPORT NETWORKS AND TRANSPORT NETWORK SLICING

The 5G Transport Network (TN) is used to interconnect the different Network Functions (NFs) deployed in the RAN (or disaggregated RAN) and the core network (and/or within the core network). These NFs are either deployed directly in hardware (like in legacy mobile networks) or as Virtual Network Functions (VNFs) running on cloud sites. In the case when functional splitting is applied, in order to realize a disaggregated RAN and Core, the RAN NFs reside in the Central Unit (CU), Distributed Unit (DU), and Remote Unit (RU) and the Core NFs reside in the Core Network (5GC). The different TN segments interconnecting these NFs are denoted as fronthaul, midhaul, and backhaul respectively (see [3GPP-TR 38.803] and [3GPP-TR 23.799] by 3GPP, [BBF TR-221], [MEF 22.2], and [ITU IMT2020 O-041]).

Depending on the deployment scenario, different technologies and different protocols operating in different layers (L1, L2, L3) have been exploited to realize the transport network segment. For example, in order to meet the high-bandwidth and low-latency requirements of the fronthaul link (L1), possible solutions are dark fiber, active Wavelength Division Multiplexing (WDM), passive WDM, and semi-active WDM solutions. The International Telecommunication Union Telecommunication Standardization Sector (ITU-T) Study Group 15 is standardizing 40G NG-PON2 and XGS-PON, as well as 10G symmetric PON systems. Dark fiber is suitable when there are a lot of spare fiber resources, while WDM-based PON is preferred when fiber resources are limited [23]. A solution for WDM-based optical networks is Optical Transport Network (OTN) - it is specified in ITU G.872. Regarding the protocols used at the fronthaul link, CPRI, eCPRI, and Radio over Ethernet (RoE) are the most important. For the sake of completeness, it should be mentioned that dependent on the deployment area and traffic demand, combinations of fixed and wireless (e.g., millimeter wave-based) technologies could also be used for realizing backhaul/fronthaul converged networks known as the 5G Xhaul network [24].

Considering L2 aspects, the use of Carrier Ethernet on top of OTN/WDM seems to be beneficial with the introduction of the *Xhaul* concept due to its flexibility in handling data traffic. An overview of Ethernet and its evolution in various fields of application is provided in [25]. We also highlight two technologies that will be increasingly important for TNs: Flexible Ethernet (Flex-E) and Flexible-OTN (FlexO). Flex-E is exploiting time multiplexing between client groups, performed in a layer between the MAC and the Physical Coding Sublayer (PCS) [26]. Flex-O is described in the ITU-T G.709.1/Y.1331.1

[2]https://www.o-ran.org/
[3]https://telecominfraproject.com/ran/

recommendation and provides OTN interfaces with comparable functionality to that of Flex-E-based Ethernet interfaces. In case Layer 3 connectivity is required, typically IP/Multi-Protocol Label Switching (MPLS) is used and techniques like Segment Routing, Ethernet Virtual Private Networks (VPNs) with Virtual Extensible LAN (VxLAN) encapsulation can be applied in IP/MPLS networks. In order to enable deterministic real-time performance in transport networks, novel full-stack approaches need to be adopted that also incorporate techniques like the ones proposed by IEEE TSN Task Group (TG) (layer 2 aspects) and IETF DetNet Working Group (WG) (layer 3 aspects). In this regard, ongoing research activities are investigating the use of IEEE TSN in an integrated MPLS-based network [27], [28]. These activities are also relevant for our work in which we consider a TSN-aware converged Xhaul. A number of research projects like EU H2020 5G-Xhaul, 5G-Crosshaul, 5G-Picture, 5G-Transformer, and some studies [24], [29]–[31] have already investigated Transport Network slicing in a multi-domain Xhaul and analyzed the data plane and control plane aspects as well as multi-domain orchestration and SDN control aspects.

Transport Network Slicing is mandatory in order to enable end-to-end slicing in the mobile network. Two categories of solutions denoted as *"Hard Slicing"* and *"Soft Slicing"* exist. In the first case, techniques like Flex-E [26] or TSON [32], [33] can be used to minimize multiplexing effects in the transport network nodes and to offer guaranteed service quality. WDM can also be used for the physical resource separation on wavelength level. For the realization of *"Soft Slicing"*, VPN techniques are exploited. Layer 2 VPN techniques are, for example, Virtual Private LAN Service (VPLS) (Ethernet-based communication over MPLS tunnels) or MAC-in-MAC encapsulation according to IEEE 802.1Qay (also known as Provider Backbone Bridge - Traffic Engineering, PBB-T). Layer 3 VPNs can be realized e.g., via Virtual Private Routed Network (VPRN) (MPLS-based VPN). Note, that although currently available *hard* and *soft Slicing* technologies are able to provide service differentiation and guarantees for throughput, they are not able to provide deterministic guarantees for delay and jitter without the exploitation of techniques mentioned before like IEEE TSN and IETF DetNet.

Note, that TN operation aspects are not covered by 3GPP activities. Contributions from standardization for TN Slicing stem mainly from ITU-T (ITU-T SG13, ITU-T SG11), Internet Engineering Task Force (IETF), Broadband Forum (BBF), and Metro Ethernet Forum (MEF). For example, ITU-T defines network slicing as Logically Isolated Network Partitions (LINP). In Table 4, references to the most relevant standards/recommendations for Transport Network Slicing are listed.

III. IEEE TSN FOR MOBILE NETWORKS

In this section, we outline the current activities related to TSN integration with the 5GS. The IETF DetNet working group is also investigating new mechanisms to provide deterministic QoS, spanning from explicit routes, packet replication and elimination, to congestion protection with end-to-end synchronization. At the moment of this writing (January 2021), the level of maturity of IETF DetNet developments is rather low.

TABLE 4. Transport network slicing overview.

Technology	Description
ITU-T Y.3102	Framework of the IMT-2020 network.
ITU-T Y.3011	Framework of network virtualization for future networks.
ITU-T Y.3012	Requirements of network virtualization for future networks.
ITU-T Y.3110	IMT-2020 network management and orchestration requirements.
ITU-T Y.3111	IMT-2020 network management and orchestration framework.
ITU-T Y.3112	Framework for the support of multiple network slices.
ITU-T Q.NS-LCMP	Protocol for network slice lifecycle management.
ITU-T G.7702	Architecture for SDN control of transport networks.
IETF	Network slice for 5G and its characteristics (*draft-rokui-5g-ietf-network-slice-00*) Framework for transport network slices (*draft-nsdt-teas-ns-framework-04*) Yang data model for transport network slices (*draft-wd-teas-transport-slice-yang-01*)
MEF	Slicing for a shared 5G Fronthaul and Backhaul.
ONF TR-526	Applying the SDN architecture to network slicing.
BBF SD-406	End-to-End network slicing.

A. IEEE TSN BASICS

IEEE TSN is a set of IEEE 802.1 amendments that enable deterministic QoS guarantees for delay and jitter of time-critical traffic flows even in cases where different traffic flows with different statistical characteristics are multiplexed. A categorization of relevant IEEE TSN standards is provided in Table 5. TSN synchronization is covered by IEEE 802.1AS and 802.1AS-Rev.

TSN Data Plane: The data plane delay guarantees can be provided through techniques like Scheduled Traffic (IEEE 802.1Qbv), Frame Preemption (IEEE 802.3br, IEEE 802.1Qbu), Asynchronous Traffic Shaping (ATS) (802.1Qcr), and Cyclic Queueing and forwarding (802.1Qch). These standards define how frames belonging to a particular traffic class and having a particular priority are handled by TSN-enabled bridges. Fig. 4 provides a high-level description of the TSN frame handling pipeline where traffic classification, 802.1Qbv, and preemption are applied.

Regarding algorithms for solving the TSN scheduling problem, we refer to the following studies: [34], for methods to compute static schedules via Satisfiability Modulo Theories (SMT); [35] for a joint scheduling and routing problem formulation; [36] for an investigation of the trade-off between the computation time and the maximum number of windows

TABLE 5. IEEE TSN standards overview.

Category	Standards
Time Synchronization Providing network wide precise synchronization of the clocks of all entities at Layer 2.	**IEEE 802.1AS & IEEE 802.1AS-Rev** (Network Timing & Synchronization)
Latency & Jitter Separating traffic into traffic classes and differentiated forwarding & queuing of the frames according to these traffic classes.	**IEEE 802.1Qav** (Credit Based Shaping) **IEEE 802.1Qbv** (Scheduled Traffic) **IEEE 802.3br & IEEE 802.1Qbu** (Frame Preemption) **IEEE 802.1Qch** (Cyclic Queuing) **IEEE 802.1Qcr** (Asynchronous Traffic Shaping)
Reliability & Redundancy Maintaining network wide traffic integrity by providing redundant paths and policing at the ingress queues.	**IEEE 802.1CB** (Frame Replication & Elimination) **IEEE 802.1Qca** (Path Control & Reservation) **IEEE 802.1Qci** (Per-Stream Filtering)
Resource Management Providing dynamic discovery, configuration and monitoring of the network as well as resource allocation & registration.	**IEEE 802.1Qat & IEEE 802.1Qcc** (Stream Reservation) **IEEE 802.1Qcp** (YANG Models) **IEEE 802.1CS** (Link-Local Reservation)

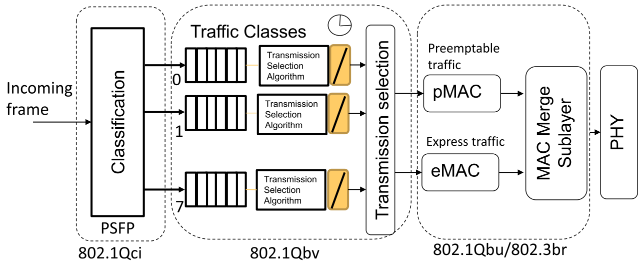


FIGURE 4. Frame handling in TSN considering 802.1Qci, 802.1Qbv and 802.1Qbu.

per queue; [37] for the calculation of worst-case latency bounds of high priority traffic in TSN networks by using network calculus.

TSN Control Plane: Regarding the TSN control plane, resource management, configuration, resource allocation, and registration aspects are covered by a) 802.1Qcc, describing the different configuration models and focusing on the centralized case, b) 802.1Qdd, covering the fully distributed case and c) 802.1Qca, which enables explicit path selection and bandwidth reservation.

A deeper look on 802.1Qcc: TSN Control and resource management aspects are investigated by the IEEE 802.1Qcc amendment, which also describes enhancements of the Stream Reservation Protocol (SRP) to extend its capabilities to support complex traffic shaping mechanisms. SRP (IEEE 802.1Qat-2010) is a simple admission control protocol that uses decentralized registration and reservation procedure to achieve end-to-end resource management. However, it was designed primarily for networks using Credit Based Shaper (CBS, IEEE 802.1Qav-2009) which only defined two traffic classes. With the introduction of more complex traffic shaping mechanisms in TSN such as Scheduled Traffic (IEEE 802.1Qbv) and Frame preemption (IEEE 802.3br, IEEE 802.1Qbu), enhancements of SRP were required which are covered by IEEE 802.1Qcc.

The key elements in a TSN network in the context of configuration and management are talkers, listeners, Bridge, and the User-Network Interface (UNI). Talkers and listeners are the end-stations that produce and consume data-streams respectively. The user side of the UNI comprises the talkers and listeners, while the network side comprises the Bridge that transfers the data frames from talkers to one or more listeners. A stream in this context is a unidirectional flow of data. The main idea is that the users specify the requirements for the streams without any detailed knowledge of the network. The network obtains these requirements, analyzes the topology and capabilities of the Bridge, and then configures the Bridge accordingly. For this purpose, IEEE 802.1Qcc proposed three configuration models.

- **Fully Distributed Model**: In this model, the user requirements from the end-stations are propagated along the active topology by exploiting a distributed protocol. The UNI is located in between the end-stations and the Bridge over which they are connected in the topology. IEEE 802.1Qdd amendment is working on the Resource Allocation Protocol (RAP) that is exploiting a Link Registration Protocol (LRP) underlay transport to support the fully distributed case.
- **Centralized Network / Distributed User Model**: The key element in this model is the Centralized Network Configuration (CNC) entity. A CNC has the complete knowledge of the network topology and of all the streams in the network. It is responsible for configuring TSN features and performing complex operations required for Time-Aware Shaper (TAS), Frame Preemption, etc., at the Bridge using a remote network management protocol (like NETCONF [38], YANG [39]). The UNI is still located in between the end-stations and the Bridge. However, the Bridge at the edge of the network (connected to an end-station) communicates the user requirements to the CNC directly in this model.
- **Fully Centralized Model**: The fully centralized model considers another entity called Centralized User Configuration (CUC). The CUC is responsible for the discovery of end-stations, retrieval of end-station capabilities, and configuration of TSN features in the end-stations. The difference from the centralized Network/distributed user model is that in this model the communication and exchange of user requirements take place between the CNC and CUC i.e., the TSN UNI exists between the CNC and the CUC. The CUC retrieves the requirements from the end-stations and exchanges this information with the CNC through the UNI.

IEEE 802.1Qcw is an amendment that specifies YANG data models specifically for Scheduled Traffic, Frame Preemption, and Per-Stream Filtering and Policing and can be used to configure the TSN bridges.

For the integration of TSN with the 5GS, ongoing standardization activities focus on the fully centralized case. Centralized control and management of the transport network offers important vantage points against the distributed alternative. The reason is that a centralized SDN-based system for the transport network (either fronthaul, midhaul, or backhaul) can be easily incorporated into the existing 3GPP management and control systems. It can also be a part of an end-to-end orchestrated solution, while also expose TSN capabilities to the network slicing management systems [40]. SDN-based control plane solutions for the transport network are investigated in the context of many research activities like [31], [41]. However, the deliberation of the concerns regarding TSN control plane operations on an integrated orchestrated mobile network environment is still an open research issue.

Regarding existing research efforts for applying SDN methodologies in TSN, in [42], a SDN-based solution based on NEON software is proposed. The authors developed a proof-of-concept which allows automatic configuration of the IEEE 802.1AS (Network Timing and Synchronization) standard. Authors in [43] proposed a framework for self-configuration of real-time networks. The authors introduce a specific learning mechanism called Configuration Agent. This study combines OPC-UA and TSN networks with the fully centralized model of IEEE 802.1Qcc as the core of the framework. References [44], [45] discuss dynamic network reconfiguration for Cyber-Physical Systems using Time-Sensitive Software-Defined Network (TSSDN). TSSDN combines TSN and SDN to support real-time communication in SDN. Reference [44] focuses on optimizing transmission schedule calculations using Integer Linear Programming (ILP) for time-triggered traffic. Reference [45] focuses on the integration of SDN and TSN over a combined control-plane. While SDN inherently provides management flexibility in a re-configurable management system [46], the adaptation of SDN for TSN presents its own challenges. To this extent, [47] investigates the general suitability of SDN for real-time Ethernet. The authors model SDN-based network configuration protocols using a Compositional Performance Analysis (CPA) framework.

B. TSN FOR THE FRONTHAUL-802.1CM PROFILE

IEEE 802.1CM provides a TSN profile for the mobile fronthaul network. IEEE 802.1CM resulted from a collaborative effort between CPRI and IEEE 802.1 and describes how to meet the stringent fronthaul requirements in an Ethernet-based bridged network. In 802.1CM, both CPRI and eCPRI protocols are supported (Class 1 and Class 2 respectively). In both cases the following types of data are considered: a) User Data; b) Control and Management Data; and c) Synchronization Data. For example, for Class 2 (eCPRI), the maximum end-to-end one-way latency is 100us for high priority user plane data traffic between eREC and eRE. Moreover, 802.1CM mentions the components that contribute to the worst-case latency for a single hop from a bridge to a bridge. 802.1CM also discusses how the time synchronization requirements can be met for Precision Time Protocol (PTP) enabled devices, satisfying for example the ITU-T G.8275.1 telecom profile and ITU-T G.8272, ITU-T G.8273 depending on the deployment case. P801.CMde is now investigating several enhancements to the fronthaul profile.

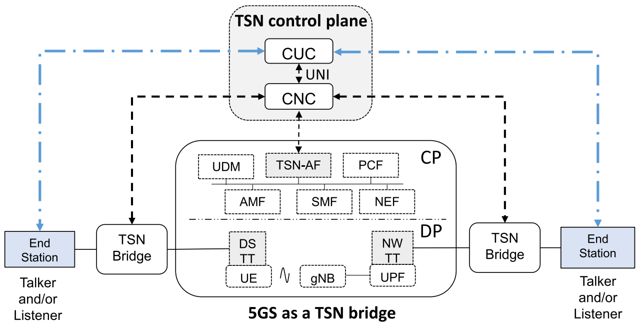


FIGURE 5. 5GS as a TSN Bridge, the "black box" approach.

In [48], FUSION platform demonstrated preemption for the fronthaul link over a 100G Ethernet-based transport. In [49], extreme packet delay percentiles contrary to maximum one-way end-to-end delays are considered. This comes at an expense of a high frame loss ratio (FLR). In [50], a time-aware shaper based on the IEEE 802.1Qbv standard is proposed for an Ethernet-based fronthaul network. By applying simulations it is demonstrated that contention of high priority traffic can be reduced and the frame delay jitter can be minimized. In [7], we evaluated the performance of Ethernet TSN networks based on IEEE 802.1Qbv and IEEE 802.1Qbu for carrying real fronthaul traffic and benchmarked it against Strict priority and Round Robin scheduling schemes.

C. THE BLACK-BOX APPROACH: 5GS AS A TSN BRIDGE

The technical approach followed in order to integrate TSN functionalities in the 5GS is to treat 5GS as a vanilla TSN bridge. This approach on one hand allows the independent development of TSN standards without having a strong coupling with 3GPP standardization efforts and timetables. On the other hand, it allows the use of IEEE TSN protocols/amendments as is by the 3GPP system, but also allows a smooth integration with other technology frameworks like TSN support for MPLS investigated by IETF. Note, that TSN features are part of Rel 16 and beyond. The description provided in the following, summarizes the key points of TS 23.501 and 23.502 regarding the 5GS as a TSN bridge concept. We refer to the standards specification for more details on the topic.

In principle, in order to expose the 5GS as a TSN bridge, TSN translation functionality is embedded for both user and control planes inside the 5GS (see Fig. 5 for a visual representation).

Control Plane: In the control plane, the 5G Application Function (AF) interacts with a fully centralized TSN control plane, namely with the CNC entity. AF is able to influence traffic routing and QoS handling for each PDU Session, through interaction with the SMF. In principle, CNC receives the stream specification from CUC and passes this information to AF. A feedback approach is used where AF gets configuration information coming from the CNC and also reports Flow QoS and characteristics to the CNC. Once the CNC retrieves the necessary information, it performs the appropriate scheduling and path calculation. Then CNC configures 5G TSN bridge (through AF) as a normal TSN bridge, using for example a southbound interface respecting the 802.1Qcw YANG model.

Data plane interaction: In the user plane, the translator services used inside the 5GS are Network-side TSN Translator (NW-TT) in the UPF and TSN translator (DS-TT) in the UE. In order to perform QoS mappings inside the 5GS and also apply the right resource allocation, the information needed is Per Stream Filtering and Policing (PSFP) and the schedule of transmission gates for every traffic class.

TSN Traffic class: AF retrieves the TSN QoS requirements and traffic class from the TSN stream specifications passed from CNC. Then, a QoS mapping table in the TSN AF is used as specified in TS 23.503. In the PCF, the 5G QoS Flow can be configured by selecting a 5QI as specified in TS 23.503.

PSFP: Two models are currently supported a) CNC exposes PSFP information to AF, and b) CNC does not expose PSFP information to AF. In the former case, PSFP functionality is executed by the DS-TT and the NW-TT according to the PSFP information received by TSN-AF. In the latter case, pre-configured QoS flows are used. DS-TT and NW-TT optionally support LLDP, hold and forward functionality for the purpose of de-jittering per-stream filtering and policing according to 802.1Qci.

Synchronization: In order to align forwarding plane operations (like 802.1Qbv aware scheduling), extreme end-to-end time synchronization is required. Note, however, that only NW-TT and DS-TT need to support the IEEE Std 802.1AS operations, while the rest of the 5GS components like UE, gNB, etc., are synchronized with the 5G grandmaster clock. The "clock bridging" between 5G grandmaster and TSN time-domain grandmaster is achieved by adding the measured residence time between the TTs into a Correction Field (CF) of the synchronization packets of the TSN working domain.

5GS internal interactions (see TS 23.501 for more details):

- The granularity of the 5GS TSN bridge is per UPF. There is only one PDU Session per DS-TT port for a given UPF.
- All PDU Sessions which connect to the same TSN network via a specific UPF are grouped into a single 5GS bridge.
- All PDU sessions which connect to the same TSN network via a specific UPF are handled by the same TSN AF.
- The SMF reports the MAC address of the DS-TT port of the related PDU Session to TSN AF via PCF.
- The association between the DS-TT MAC address, 5GS Bridge ID, and the port number on DS-TT is maintained at TSN AF and further used to assist the binding of the TSN traffic with the UE's PDU session.

D. TSN FOR THE XHAUL AND CONVERGENCE OVER ETHERNET

IEEE TSN convergence with the 5GS means binding the TSN traffic with the UE's PDU sessions and that the 5GS is making the necessary resource allocation to preserve the QoS level required by the TSN flows. It also means TSN-aware ports in the UE (DS-TT) and also in the devices hosting UPF and gNB (CU, DU, RU in the case of disaggregated RAN). It also means that TSN traffic is now crossing the network equipment supporting N3 (UPF to gNB), N6 (UPF to and from data network), and N9 (UPF to UPF) interfaces. However, the research question is how to configure the TSN bridges that now reside inside the 5GS black box TSN bridge.

Note, that under the umbrella of the Xhaul concept [24], [29]–[31], TSN is gaining momentum as an enabling technology. However, it cannot be used as a panacea to solve any transport network connectivity issue. For example, pure Layer 2 connectivity is impossible to scale. In this regard, it is expected that the outcome of the liaison activities between IEEE TSN and IETF DetNet will actually drive the TSN-aware Transport network, especially for large-scale deployments. In this context, relevant standardization and research activities are RFC 8655 (Deterministic Networking Architecture), RFC 8938 (DetNet Data Plane Framework), RFC 8939 (DetNet IP Data Plane), and [27], [28] where a TSN-aware MPLS data plane is specified for DetNet. Control plane signaling, for DetNet covering distributed, centralized, and hybrid signaling scenarios in the TSN and SDN domain are investigated in [51], IP inter-networking with TSN is investigated in [52], management operations for DetNet are investigated in [53]. Recent activities are also investigating lower layers' integration with TSN, for example, TSN operations over PON (for the fronthaul link) [5]. In [6], the authors are interconnecting synchronous TSN domains through an optical backbone.

Another dimension considered in our study is related to the ability of IEEE TSN to serve as the backbone Ethernet over which other protocols can be smoothly integrated. For example, in industrial environments, TSN is gaining momentum as the key technology over which protocols like PROFINET and Ethernet/IP will operate. In this regard, we are considering that the same TSN switch fabric supporting Xhaul is used to operate non-5G-related flows, which may or may not have deterministic requirements by means of delay. A visual representation of the concept is depicted in Fig. 6.

IV. TSN CONTROL PLANE INTEGRATION WITH NETWORK SLICING MANAGEMENT SYSTEMS

In this section, we describe a novel architectural design, analyzing the appropriate control and management entities and interfaces required to connect network slicing management systems with an integrated SDN-based TSN control plane. With our approach, by enabling slicing over TSN, we align network slicing requirements and map the network slice instance defined by 3GPP to the underlying TSN transport network, taking into account the desired transport

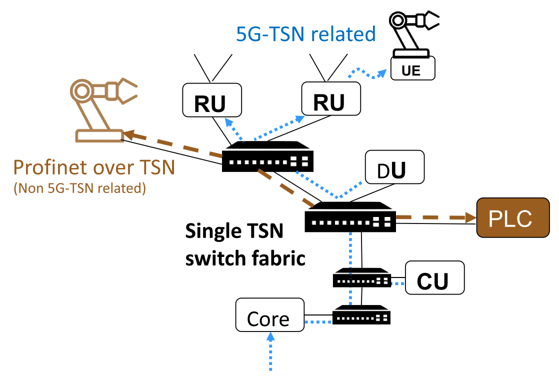


FIGURE 6. Convergence over TSN for industrial networks.

network performance attributes. Furthermore, we maintain network slice instance state at the transport network level, while preserving slice isolation over a TSN-based Xhaul data plane. For the overall transport network, we consider TSN-awareness. However, as TSN is a Layer 2 technology, we consider that it can harmonically operate with other technologies like MPLS, deterministic IP/DetNet, and segment routing, in order to deliver the integrated network service.

A. PROPOSED ARCHITECTURE

Our architectural proposal is depicted in Fig. 7. Our architecture considers that a Multilayer Transport Network Slice Controller (MTNSC) exists, interacting a) with a transport slice management entity of the mobile network (TN-NSSMF) through interface *tn-ml-nsi*; and b) with the TSN control plane through a second interface *tn-tsn-nsi*. Our design is aligned with the approach described by [54]. Also, the interface *tn-ml-nsi* specification is according to the YANG definition in [54], [55]. However, in order to enable TSN, data plane functionality fields like (*"slice-template"*) need to be augmented with TSN features. The *tn-tsn-nsi* interface is used to interconnect MTNSC with a TSN slice-aware orchestrator.

A key novel component of our design is a TSN slice-aware orchestrator, responsible to orchestrate the creation of TSN network slices and support their entire life-cycle management. In more detail, through interaction with the MTNSC, the TSN orchestrator is responsible for:

- handling TSN-TN network slice requirements information.
- handling slice aware TSN TN-NSSI stream specification.
- TSN-TN slice instance creation.
- handling TSN-TN slice instance state information.
- handling TSN-TN slice instance policy information.
- handling TSN-TN slice instance configuration information.
- monitoring TSN-TN slice instance running state.
- TSN-TN slice instance decommissioning actions.
- exposing soft or hard TSN slice instance capabilities to MTNSC.
- driving the creation of soft or hard TSN slice instance based on the overall network state/slice requirements.
- receiving TN slice isolation requirements from MTNSC.

For example, the TSN control plane entity may support preserving slice isolation over a converged TSN-based data plane using specific schedulers and Gate Control Lists (GCLs) when 802.1Qbv is used. We envision a single TSN network, able to concurrently serve the 5G Xhaul but also support non 5G related connectivity. This is a very realistic case, especially for Industrial environments. Having two separate TSN networks, one with a slice-aware 5GS TSN bridge and one segment without it is just too expensive.

The need for a TSN orchestrator originates from the fact that a single TSN control plane (CNC/CUC as these are currently defined by 802.1Q) on one hand is not slice aware and cannot differentiate stream requirements on a per slice basis. On the other hand, TSN support on a per slice basis requires that the TN NSSI instance is created and activated before the streams (corresponding also to 5G sessions) really pass through the network. This means that the TSN control plane needs to perform decision-making with different time scales, one at NSSI provision time (without necessarily having full knowledge of all the flows passing through the TSN network) and one according to traditional CNC/CUC operations and workflows.

The TSN slice orchestrator interacts with the rest of the TSN control plane entities, namely CNC/CUC, in a network slice aware manner in order to realize the creation of TSN TN-Network Slice Subnetwork Instance (NSSI). To preserve backward compatibility, we consider that the CNC hosts the scheduler logic. However, it is further configured for controlling both TSN slice aware operation and/or a TSN non-slice aware operation, through interaction with the TSN orchestrator.

According to the current standardization landscape in 802.1Q, CUC/CNC and their interplay with "traditional" SDN control are not well defined. This further complicates the management of a convergent DetNet/TSN control plane and the way this can be realized. In our design, the TSN orchestrator is the control and management entity that interacts with both CNC/CUC (TSN aspects), but also SDN control (L2/L3/L4/topology, etc.), and the MTNSC in order to orchestrate TSN-NSSI and optimize the TSN slice-aware and slice-unaware operations.

Input to the TSN orchestration mechanism are as follows:

- TSN slice–aware information/requirements/policies for network slices.
- Per slice stream profiling – session dynamicity handling/filtering/aggregation.
- Input from other network controllers or engineering tools, as TSN can be used as a converged network over which other traffic can pass concurrently with 5G flows. For example, in industrial networks, engineering tools can describe the requirements of Profinet or Modbus traffic over TSN.

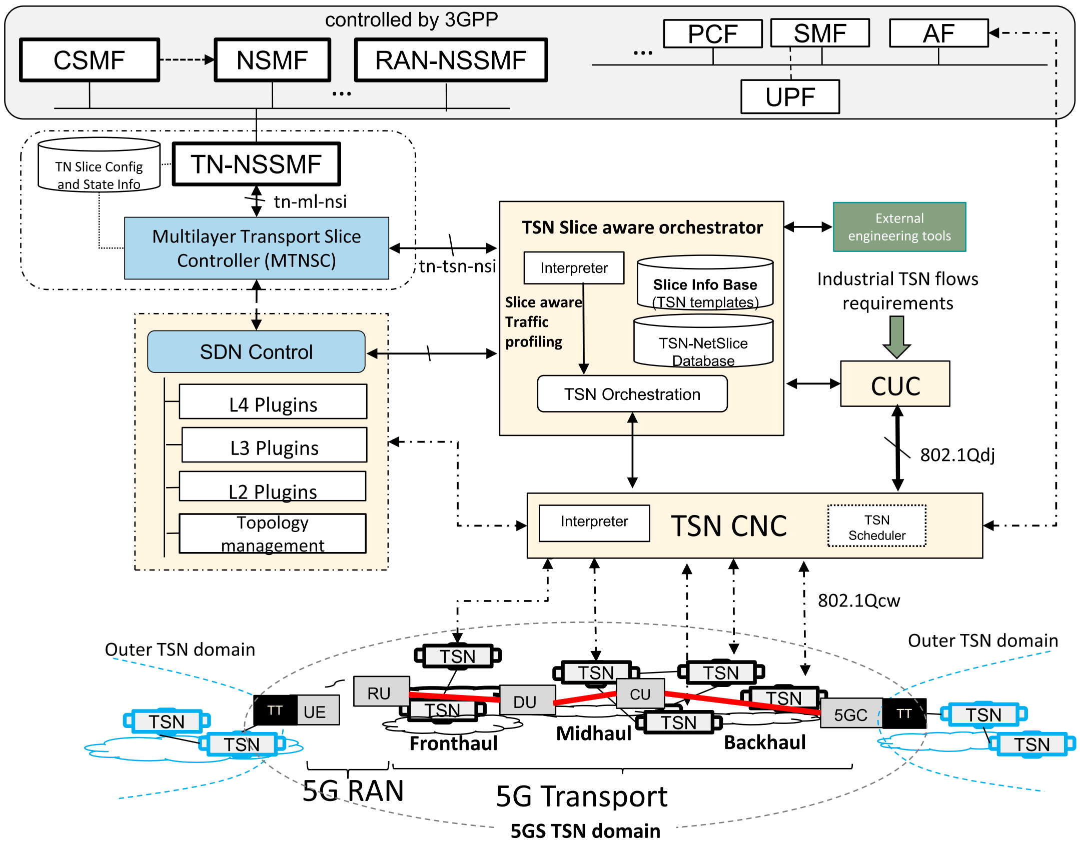


FIGURE 7. **Proposed architecture for enabling Network Slicing in a TSN-aware Xhaul.**

- Traffic prediction module: as in the case of applying TSN over 5G-Xhaul, flow dynamicity complicates the decision-making inside CNC. We consider a traffic profiling module that can operate in order to facilitate optimal decision making, instead of statically defining the traffic requirements. From one implementation perspective, this module can operate inside CNC, from another perspective this could be implemented as a part of the TSN orchestrator (or even inside CUC). However, it could also be independent and expose service to both CUC/CNC.

The different components of the architecture and their functionalities are detailed as follows:

- *TN-NSSMF:* TN-NSSMF is responsible for the orchestration and management of the TN-NSSI counterpart.
- *E2e Slice-DB: TN Slice Config and State info:* We assume that this is a database infrastructure with all the TN-NSSI information. This database is used to store all the information regarding NSI state, NSI templates, reserved resources, network functions, and configurations.
- *Multilayer Transport Network Slice Controller (MTNSC):* This entity is defined by IETF in [54], [55]. It is the entity that communicates with TN-NSSMF in order to control the different network control elements used to deliver the transport slice service. Note, that the control plane functionalities for the TN are provided by one or more domain controllers that are interacting with the TN-NSSFM through the MTNSC. For example, a different domain controller can be used to control the fronthaul network and a different one for the backhaul network. Different domain controllers can also be assigned to control different administration domains. One control entity for example, could be responsible for L2/L3 aspects and another for topology discovery or IP configuration. From an implementation perspective, a single software solution (like an SDN controller) could support all the necessary functionalities; a domain controller can be SDN-based.
- *TSN-NetSiceDB*: The database infrastructure with all the TSN TN NSSIs' state information. This database infrastructure is not controlled by 3GPP. It is used to store all the information regarding identification and mappings between NSI and NSSI, store the TSN TN NSSI state, templates, reserved resources, network functions, configurations, etc. It is also the entity where TSN TN NSSI operation and management information are stored. For every network element or network service, we consider that for each TSN TN NSSI, only specific Operations, Administration, and Maintenance (OAM) information is stored at the TN-NSDB that is relevant only to this TSN TN NSSI. This OAM filtering operation could be implemented by a domain controller, however, how this

operation is made is out of the scope of the interface specification.

- *Slice Aware/Slice Unaware CUC*: To enable backward compatibility with the 5G black-box approach, CUC initially parses slice unaware stream requirements from the different Talkers/Listeners. However, in case Network Slicing is enabled prior to sending to CNC the relevant stream TSpecs, stream requirements are passed through new *tn-tsn-nsi* interface to Transport Network Slice Controller to the management entities (like CSMF) responsible to describe the Slice requirements to NSMF. In case Network Slicing is not enabled, or in case all streams by default belong to a default network slice, the normal pipeline is followed and stream information is passed through 802.1Qdj to CNC and then to AF in order to make the resource reservation inside 5GS. This is a new design of the CUC entity in order to enable slice awareness. The new interface is used to update stream information with relevant slice identification and essentially enable communication between a Slice aware CUC/CNC with MTNSC.
- *TSN-NSI Templates:* A network slice template is used to describe the slice by means of resources, services, configurations, relationships, and service function chains required by the NSI. The network slice templates actually define all the details required by a network orchestrator to drive all the phases of the NSI life-cycle. For the TSN network, a new network slice template is required that is used to define the type of the NSSI like hard or soft slicing, shared or non-shared resources, traffic requirements, and QoS attributes. These templates can augment Generic Network Slice Template (GST) with TSN attributes [56], [57]. These are used to compile the relevant Network Slice Type (NEST) with TSN information. A Network Slice Type (NEST) is a GST with the values assigned. The invention considers that information passing between CSMF and NSMF should also consider the amendment of slice NEST with the relevant TSN parameterization.
- *Slice Info Base:* The definition of such templates can be found in the Slice Info Base.
- *Slice aware/Tenant aware CNC:* In principle, CNC receives input from CUC regarding configuration requests and network services like LLDP (or other tools) for topology information. Based on all this input scheduling, decision-making is performed for the whole network. However, according to current developments, there is no notion of tenant or slice to group different stream requests, in order to optimize the scheduling/forwarding decision. We consider that, for all the stream requests made by the CUC, an additional tenant/slice identifier is also used. After the CNC compiles the forwarding strategy (e.g., scheduling), this is applied to TSN-bridge devices through a management protocol (like NETCONF, RESTCONF, etc.). We consider that CNC has direct access to the Slice Info Base and the TSN-aware TN-NEST. All the interfacing between the TSN control plane and the MTNSC is to be handled by a TSN orchestrator, used for message interpretation, while also for interfacing with other TN control systems to cope with complexity minimization and relevant optimization decision-making.

The identification of NSIs, TN-NSSIs, TN-resources, TN-NFs, TN-interfaces, etc., is an important topic towards NSMF and TN-NSSMF integration, in order to provide end-to-end NSIs. In 3GPP, different identifiers are used to realize the concept of network slicing such as the NeS-ID, S-NSSAI, Tenant-ID, Temporary-ID, Token, and Tracking Area Identity (TAI) defined in 3GPP TS 23.501, TS 23.502, TS 38.300. For the work delivered so far, it may also be considered that existing identifiers like PLMN-IDs, logical channel identifiers, session identifiers, and so on, can all be exploited by the slice-aware orchestration and management system. For the TSN network part regarding the TN-NSSIs that are related to the assembly of one or multiple NSIs, it may be assumed that a similar identification mechanism exists that may assign for example, TNS-NSSI-ID and may further map it to the NSI-ID (provided by the NSMF) (see also [54], [55]). This information may be stored in the TSN-NetSliceDB.

While for our analysis, we consider a rather static network design, the proposed architecture also works for dynamic environments. Under 5G network dynamicity, incorporating new devices, new VNFs, adding and removing services, and so on, on the one hand, affects the physical connectivity and the TSN network supporting the Xhaul. On the other hand, it also affects the TSN streams actually being transported. However, from a design perspective, the control and management plane components of our design such as the CNC, CUC, SDN controller, and also the TSN Orchestrator remain the same also under dynamic network setups.

B. ROLE OF ANALYTICS

Indeed the key idea in our approach is that a TSN-based transport will serve as the underlay to support on one hand predefined slice-specific TSN traffic that is "crossing" the 5G system and on the other hand support non-5G traffic (as presented in subsection III-D) at the same time. The flow specifications in both these cases are provided by the CUC and the orchestrator to the CNC in order to optimize the scheduling decision. We also assume that the stochastics inside the 5GS, are handled by the appropriate resource allocation made by the 5GS.

However, in the case where the same TSN network serving the Xhaul needs to support other flows which are not externally predefined by the CUC, the TSN network optimization becomes extremely complex. As an example, this is the case where a UE associated with a URLLC slice starts transmissions that are defined inside the 5GS. The traffic profiling for these flows and the possible traffic fluctuations for all the flows need to pass the relevant CNC entity, in order to optimize the TSN network operation. This profiling needs to be on a per slice basis. As described in [58], 5G traf-

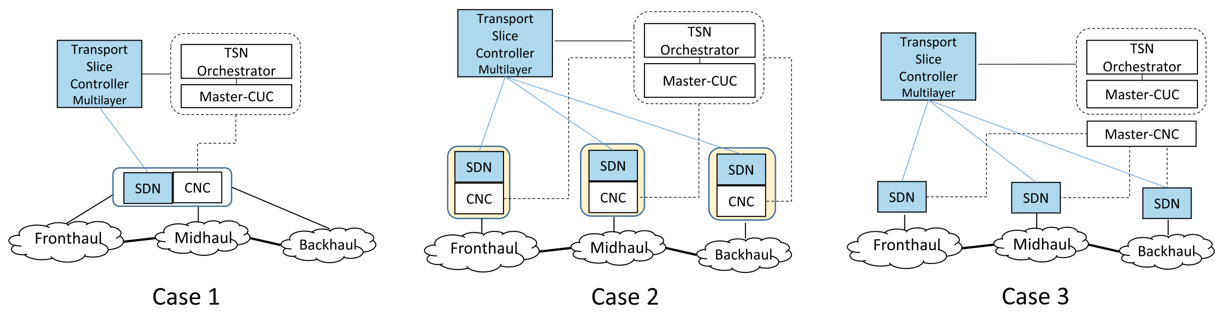


FIGURE 8. Hierarchical vs Centralized SDN/CNC control for the mobile network.

fic profiling and prediction models have been used for UE and session-related parameters like the UE context, mobility, QoS, and traffic load predictions. Furthermore, prediction models have been used for network-related parameters like average channel quality, interference, and user density in addition to being used for service-related parameters. The type of analytics that could be exploited by the TSN orchestrator can be real-time, near real-time, and non-real-time. In our design, we are interested in descriptive analytics, diagnostic, and predictive analytics. Regarding the mathematical models that could be used and a more detailed discussion on traffic analytics, we refer the interested readers to [58], [59].

C. A NOTE ON MULTILAYER CONTROL

In principle, orchestration of multiple domains requires the design of complex integrated solutions with multiple SDN controllers and network slice-aware orchestrators. See our previous work on this topic [29]. When incorporating the TSN technology in the transport network, this integration is even more complex if coupled with the multi-domain TSN concept as defined in IEEE 60802 profile. Hierarchical CNC/CUC approaches for TSN networks are discussed in [60]. From the practical perspective, for rather small network topologies, we can safely consider that a single TSN CNC entity is responsible for the fine-tuning of the TSN aspects through interaction with a simple SDN instance to cover full stack aspects. However, for large network topologies, careful control plane planning is required following a hierarchical SDN structure.

For example, we identify the following three cases as depicted in Fig. 8. In the first case, which is also the one we used to analyze the concept, all the TSN control and orchestration is performed centrally for all the different domains. In the second case, SDN and CNC control can be applied per domain and can be centrally orchestrated. We consider also a third case, where SDN control is applied on a per-domain basis, but the CNC control is performed in a centralized way. A more in-depth investigation of the topic will be part of our future work.

V. IMPLEMENTATION AND EVALUATION

In this section, we describe a primitive end-to-end implementation of the approach. We exploit a TSN hardware prototype solution, OAI testbed implementing the disaggregated RAN, and JOX open-source event-based orchestrator [8]. JOX is part of the MOSAIC-5G ecosystem[4] and is used as an integrated orchestrator for both the mobile network and the TSN network. However, we highlight that currently not all the 5G functionalities are supported by OAI, and in this regard, no TSN translation services were implemented (AF-TT, UPF-TT, DS-TT). This means for example, that the actual resource allocation in the radio part is not based on mapping with the TSN priorities. Furthermore, the proof of concept experimentation and results presented are sub-optimal by means of data plane performance, as all the software used is based on open source and all the TSN hardware on a prototype solution. For example, we are using 1 Gbps links for all TN segments, which is not realistic for real fronthaul networks. The purpose of this proof of concept demonstrator is to showcase the performance of TSN as an enabling technology, able to provide service differentiation on a per slice basis, investigate TSN service provisioning time, and demonstrate the interaction between the network slice management plane with the TSN control plane. Implementation of the translation services is left for future work.

A. IMPLEMENTATION

Fig. 9 depicts the testbed solution we built for our experimentation. JOX orchestrator controls the life-cycle of all the slices. A new TSN plugin was developed to interact with the TSN network through an SDN controller and a TSN agent was developed to interact with the JOX core. We used the OpenDaylight (ODL) controller as our SDN controller.

1) TSN PROTOTYPES

We use hardware prototype TSN switches equipped with 1 Gigabit ports. These prototypes support the following IEEE standards used in our experiments: Scheduled Traffic (IEEE 802.1Qbv), Frame Preemption (IEEE 802.3br, IEEE 802.1Qbu), and Network Timing and Synchronization (IEEE 802.1AS). The prototype switches also expose a NETCONF interface for the configuration of TSN, VLAN, and 802.1AS.

2) DISAGGREGATED RAN USING OpenAirInterface

For both the core and the RAN, we use OpenAirInterface (OAI) [9]. For the fronthaul, we use NGFI split 4.5, similar

[4]http://mosaic-5g.io/apidocs/JoX/

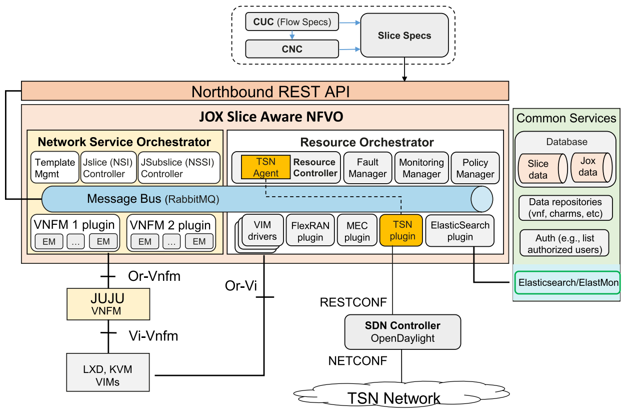


FIGURE 9. TSN agent-plugin implementation in JOX.

to 3GPP option 7-1 [7]. For this split option, OAI defines a custom format for packetization [61]. The split transports I/Q samples in the frequency domain, i.e., after removal of the cyclic prefix and Fast Fourier Transform (FFT), and before resource element (de-) mapping. Thus, it is a part of the cell-related fronthaul processing (as opposed to user-specific processing), exhibiting a constant bandwidth requirement since the I/Q samples related to the full cell capacity need to be transmitted, irrespective of the cell load. To reduce the fronthaul bandwidth, the samples are compressed using 8-bit A-law compression. For the midhaul, we consider the F1 functional split (TS 38.470) between the Radio Link Control (RLC) and Packet Data Convergence Protocol (PDCP) sub-layers. Traffic is user-specific and is separated into control and user plane parts. User plane traffic (F1-U) is encapsulated into (non-standard-conform) Google Protocol Buffers[5] and transported via User Datagram Protocol (UDP). The control plane traffic (F1-C), transported via Stream Control Transmission Protocol (SCTP), is negligible in the considered scenario. For the backhaul, we consider the S1 protocol (TS 36.410) between the base station and the core network. Again, traffic is user-specific and separated into control and user plane parts. The user plane traffic (S1-U) is encapsulated into GPRS Tunnelling Protocol (GTP-U) and transported via UDP. The control plane traffic (S1-C), transported via SCTP, is negligible in the considered scenario. In the testbed built, all the RAN and Core resources per slice are dedicated, while the TSN network is shared. In order to mark the corresponding slices, the packets are transported over UDP/IP and encapsulated in 802.1Q VLAN frame tags. This is done by routing all traffic through the VLAN interfaces in the corresponding dedicated machines hosting the RAN/Core resources. We use the VLAN ID to differentiate the traffic between slices.

3) JOX ORCHESTRATOR: KEY ELEMENTS

The key elements of JOX are summarized as follows:

- REST NorthBound Interface (NBI): used to manage and orchestrate (e.g., on-board slice template, deploy and monitor slice, delete slice, etc.) slices and sub-slices. The templates of slices and sub-slices are described according to OASIS TOSCA [62].
- Package Manager: It is responsible for creating and packaging the template of slices and sub-slices, which can be onboarded later via the JOX NBI.
- NFV Orchestrator (NFVO): responsible for Network service instantiating and life-cycle management.
- Network Functions Virtualization Infrastructure (NFVI): resource management through the interaction with the Virtualized Infrastructure Manager (VIM) (e.g., resource reservation and allocation to the network instances).
- Monitoring manager: Collect usage information of NFVI resources by VNF instances.

[5]https://developers.google.com/protocol-buffers, retrieved 08/12/2020.

JOX is operating on top of Canonical's JUJU, which serves as a Virtualized Network Function Manager (VNFM). Regarding the VIMs responsible for managing the infrastructure, we experimented with LXC containers, Kernel-based Virtual Machines (KVMs), and physical machines. JOX also supports a plugin architecture based on the microservice paradigm. By design, the interaction between plugins and the JOX core is made through a message bus (RabbitMQ), while each plugin serves a different purpose, like store periodic statistics into NoSQL elastic-search database, or interacting directly with the RAN using FlexRAN plugin.

In JOX, slices and subslices are created as follows. We first onboard the slice template for a slice and the associated subslices using JOX NBI. These are stored in *the JOX store*. Network Service Orchestrator (NSO) starts parsing the templates for the slice and the associated subslices to understand their requirements. After that, for both slice and subslices, NSO instructs the resource orchestrator (RO) to perform the resource allocation, through interaction with the VIMs. The intended services are deployed on the allocated resources via Network Service Orchestrator/VNFM (JUJU framework), while the necessary plugin(s) are interacting with the infrastructure when necessary.

4) TSN PLUGIN IMPLEMENTATION

A new JOX plugin was implemented in order to interact with a fully functional TSN-enabled prototype device. Through JOX, the new plugin exposes control functionalities for the following three categories: i) Time-Aware Shaper (IEEE 802.1Qbv); ii) VLAN/private VLAN configuration; and iii) Network Timing and Synchronization (IEEE 802.1AS). The specific methods exposed by the TSN plugin are listed in Table 6.

The TSN plugin interacts with OpenDaylight (ODL) controller through RESTCONF, while ODL interacts with the TSN switch fabric over NETCONF (Fig. 9). The communication between the TSN plugin with JOX is made through a message broker service, implemented using the open-source RabbitMQ framework. A TSN Agent resides on the JOX core side and acts as a consumer/producer of messaging, targeting

TABLE 6. TSN plugin functionality.

Category	Message
VLAN	vlan_create, vlan_add, vlan_remove, vlan_delete, Pvlan_set (used to set default vlan per port)
802.1AS	display interface, enable, LogSyncInterval, LogAnnounceInterval, LogMinDelayReqInterval, SyncReceiptTimeout, delayAsymmetry, minNeighborPropDelayThreshold, maxNeighborPropDelayThreshold, display clock-parent, display clock, clock domain, clock priority1, clock priority2, clock clockClass, clock profile, clock gmCapable, clock slaveOnly, display program, program timestamping, program delayMechanism, program transport, program timeaware-bridge, program phydelay_compensate, program phydelay_compensate_out, program servo_locked_threshold, program logging-level, program save-config, display time-properties
TAS	tsntas-cycle-time, tsntas-schedule-entry, tsntas-enable-schedule, tsntas-ptp-mode, tsntas-gate-ctrl-period, tsntas-profile, tsntas-status

the TSN plugin. With this modular architecture, the TSN plugin can run in an independent process even in a different machine from the one in which JOX is deployed. Furthermore, the same message brokering service is also exploited by the JOX REST interface, however, for every function, all the necessary wiring inside JOX is updating the right data structures responsible to keep the operational state of every NSI.

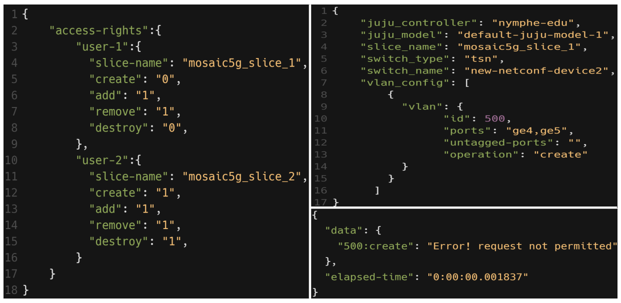
In our model as seen in Fig. 9, the flow specifications are retrieved by the CNC from the CUC, which invokes the TSN scheduler to generate the appropriate GateControlLists (GCLs) based on the flow specifications. These GCLs along with the flow specifications from the CUC are used to construct the Slice Specifications which are passed to the JOX orchestrator. Through JOX, the GCLs are passed from the TSN plugin to ODL through RESTCONF. The ODL in turn configures TSN switches over NETCONF.

Listing 1 illustrates an example of 802.1Qbv configuration for port 0 of TSN switch "new-netconf-device2", when constructing a specific TN-NSSI. As we can see, GCL with one entry (*tsntas-schedule-entries*) is configured while the Qbv feature is enabled on the interface (*iface*). Note, that since at each TSN switch the ports can be used by more than one slice, JOX controls the actions that can be performed in order to preserve isolation between slices.

The functionalities exposed in the message brokering service are also exposed through the NBI REST interface from JOX. As an example, two endpoints in the REST API are used to a) get the list of all supported switches, along with their configurations; and b) Create, add, remove, and destroy VLAN and their associated ports. Moreover, to preserve vertical slice isolation, we also implemented an access rights mechanism, where we defined the permitted actions for every slice owner. As illustrated in Fig. 10, the owner of slice "mosaic5g_slice_1" has the rights to add/remove ports, but does not have the rights to create or destroy a VLAN.

```
{
  "tsn-switch-name":"new-netconf-device2",
  "interface":[
    {
      "iface": 0,
      "tsnptp-interface": {
        "enable": "enable",
        "sync-receipt-timeout": 0,
        "delay-asymmetry": 0,
        "priority1": 100
      },
      "tsntas-interface-enable":"enable",
      "tsntas-cycle-time":{
        "base-time-sec": 3,
        "base-time-ns": 0,
        "cycle-time": 800,
        "extension-ns": 60
      },
      "tsntas-schedule-entries":[
        {
          "entry": 0,
          "hold-preempt":0,
          "time-interval": 5000,
          "gate-state": 80
        }
      ]
    }
  ]
}
```

Listing. 1. REST API example: Configuring Qbv.



```
{
    "access-rights":{
        "user-1":{
          "slice-name": "mosaic5g_slice_1",
          "create": "0",
          "add": "1",
          "remove": "1",
          "destroy": "0",
        },
        "user-2":{
          "slice-name": "mosaic5g_slice_2",
          "create": "1",
          "add": "1",
          "remove": "1",
          "destroy": "1",
        }
    }
}
```

```
{
    "juju_controller": "nymphe-edu",
    "juju_model": "default-juju-model-1",
    "slice_name": "mosaic5g_slice_1",
    "switch_type": "tsn",
    "switch_name": "new-netconf-device2",
    "vlan_config": [
        {
            "vlan": {
                "id": 500,
                "ports": "ge4,ge5",
                "untagged-ports": "",
                "operation": "create"
            }
        }
    ]
}
```

```
{
  "data": {
    "500:create": "Error! request not permitted"
  },
  "elapsed-time": "0:00:00.001837"
}
```

FIGURE 10. (left) Slice access rights, (top right) VLAN creation, and (bottom right) JOX NBI output for access violation.

B. EVALUATION

Our performance evaluation analysis is divided into two parts. In the first part, presented in subsection V-B1, we investigate the effects of statistical multiplexing in data plane performance, if network slice isolation is preserved, how well the TSN network scales, and TSN performance when carrying eCPRI traffic. In the second part presented in subsection V-B2, we investigate end-to-end provisioning time for both the mobile and TSN networks using the JOX orchestrator.

For both the experimental parts, we exploited the same baseline setup, depicted in Fig. 11. We created URLLC and eMBB slices in an end-to-end deployment (indicated by blue and red respectively), where for each slice, Core, CU, DU, RU, and USRP are dedicated, while all slices share the fronthaul, midhaul, and backhaul segments that were built using the TSN-enabled prototype devices. Each slice is

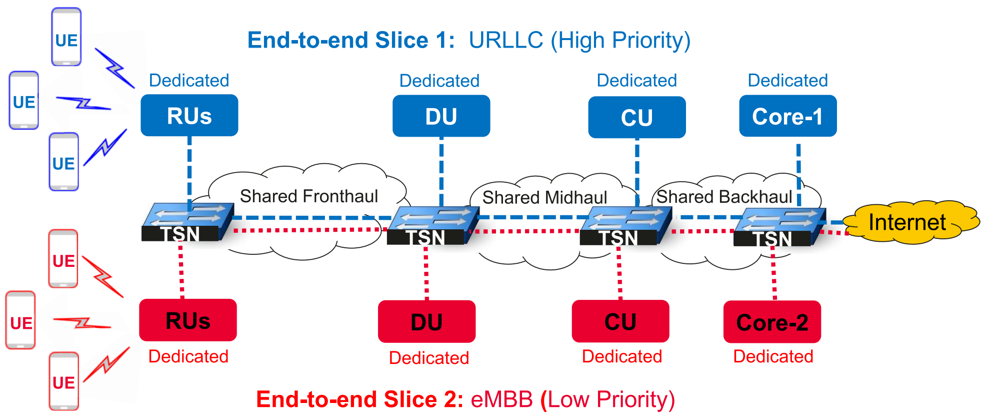


FIGURE 11. Experimental baseline network setup. One URLLC and one eMBB slice created with dedicated RAN and Core resources and shared TSN segments.

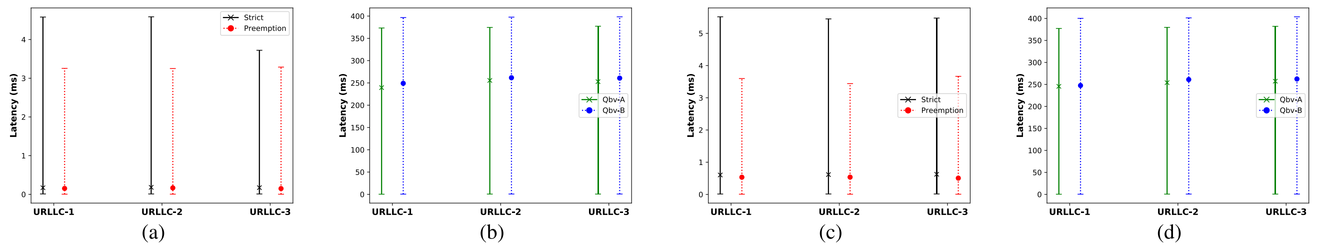


FIGURE 12. Evaluation Results for Traffic Multiplexing: Latency in underload conditions ((a) and (b)), Latency in overload conditions ((c) and (d)).

differentiated on the basis of the assigned VLAN identification in each experiment and is able to carry a number of flows. VLAN ID is used to differentiate the traffic between slices and QoS handling is performed based on the VLAN PCP field of each frame. For both the URLLC and eMBB slices, we carry real traffic as described in the previous section. We further used a traffic generator to add emulated background eMBB traffic and also investigated the performance of TSN when carrying eCPRI traffic. We used the IXIA traffic generator from Keysight Technologies to generate eCPRI messages. For our delay and jitter measurements, a hardware tap-based mechanism was used based on an Intel Ethernet Controller (I210) that supports hardware time-stamping.

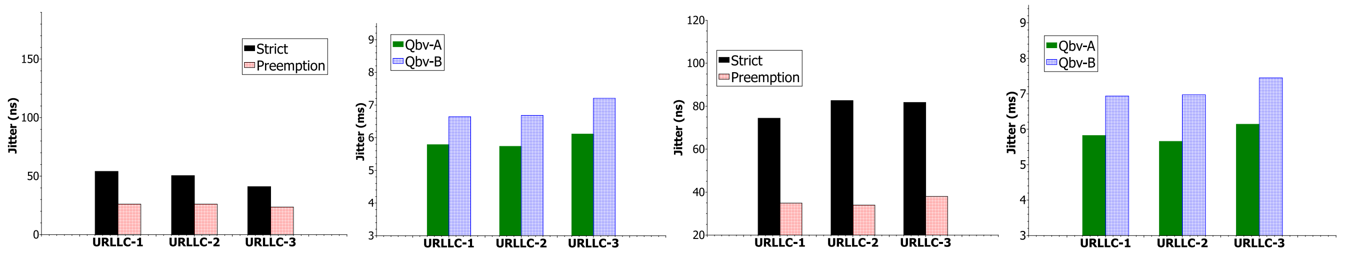
1) DATA PLANE PERFORMANCE

a: TRAFFIC MULTIPLEXING

In the first set of experiments, we evaluate the impact of multiplexing multiple traffic flows on latency and jitter. In this experiment, we created three URLLC NSIs and one eMBB NSI, each carrying a single flow. The traffic flows associated with the URLLC slices are assigned a high priority (VLAN priority 7), whereas the eMBB traffic flow is assigned a low priority (VLAN priority 0). Here, we compare the performance of TSN 802.1Qbv against the Strict Priority (SP) scheme and Frame Preemption. For TSN 802.1Qbv, we evaluate two different types of schedulers. "Qbv-A" represents the performance of using the scheduler mentioned in [63]. Under "Qbv-B" scheme, we give an equal scheduling opportunity to the three URLLC traffic flows i.e., equal time in the GCL for each of the flows to pass through. We consider two scenarios namely underload and overload. In the underload scenario, only the above-mentioned four real traffic flows traverse the network via the two network slices and in the overload scenario, we add background traffic (best effort) and visualize its impact.

Figures 12(a) & 12(b) show the latency of the URLLC traffic flows in the underload scenario, where the ticks at the top and bottom of each line indicate the maximum and minimum values. The average value is also shown in the plots. The higher values of latency for TSN-Qbv (both "Qbv-A" and "Qbv-B") compared to SP and Preemption is expected because the mobile traffic is not scheduled and the TSN switches transmit the packets only according to the schedule and not immediately upon arrival. Thus, TSN-Qbv is more effective when the traffic is scheduled. This has been experimentally validated in our previous work [7]. In the underload scenario, we notice the lowest values of latency in the case of Preemption, which always preempts the lower priority eMBB traffic and prioritizes the URLLC flows.

Figures 12(c) & 12(d) show the latency of the same flows in the overload scenario. The latency in the case of SP and Preemption increases slightly because of the presence of background traffic. However, in the case of TSN-Qbv, the latency remains the same. This is because the background traffic is completely blocked during the time in which the high priority traffic is allowed to pass by the switches thus, ensuring determinism. Figures 13(a) & 13(b) show the average jitter measured end-to-end of the URLLC flows in the underload scenario. As can be seen from these figures, TSN-Frame-Preemption achieves the lowest jitter. The scale



(a) (b) (c) (d)

FIGURE 13. Evaluation Results for Traffic Multiplexing: Jitter in underload conditions ((a) and (b)), Jitter in overload conditions ((c) and (d)).

of the jitter values for SP and Preemption is at a nanosecond level whereas that of TSN-Qbv is in milliseconds and this is because the mobile traffic is not scheduled.

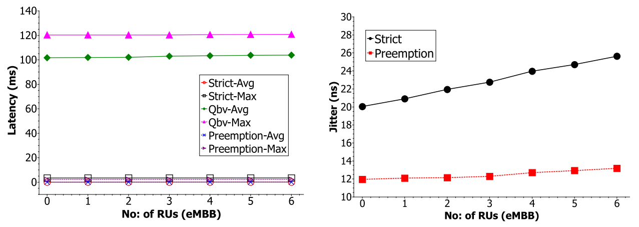
Figures 13(c) & 13(d) show the average jitter measured in the overload scenario. Once again it can be noticed that the jitter increases in the case of SP. In the case of Preemption, the increase in jitter is negligible (few nanoseconds). However, it remains constant in the case of TSN-Qbv since it manages to completely block the best-effort background traffic and ensures determinism. Note, the lowest values of jitter are noticed in the case of Preemption.

b: SLICE ISOLATION

We conducted three sets of experiments to validate slice isolation over a TSN-enabled network. In the first experiment, we created two network slices, one URLLC and one eMBB. Initially, we only have one URRLC traffic flow (VLAN priority 7) traversing through the network. We then add the eMBB traffic flow to the network. At this point, each of these slices has only one traffic flow, each associated with one RU. We then progressively increase eMBB traffic by increasing the number of RUs and the associated eMBB flows and evaluate its impact on the other slice (URLLC).

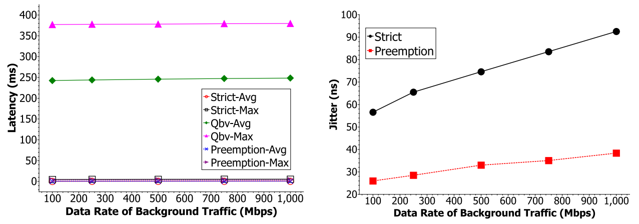
Figures 14(a) & 14(b) show the latency (maximum and average) and jitter (average) of the URLLC flow under SP, TSN-Qbv, and Preemption schemes. The TSN-Qbv in this experiment and in all the experiments from here on uses the same scheduler as that of "Qbv-A" mentioned in the previous experiment. We notice that as we increase the number of RUs and the load in the eMBB slice, the latency of the URLLC flow remains constant for all the schemes which are able to preserve slice isolation, under the assumption that priorities are preserved and there is no traffic multiplexing on the same traffic class. Fig. 14(b) shows that the average jitter does increase for SP and Preemption, however, this is in the scale of 1-6ns which is negligible. We do not plot the jitter values for TSN-Qbv as they are at a different scale. However, we do notice that the jitter for TSN-Qbv remains constant at 0.25us proving that the addition of traffic flows to the eMBB slice has no impact on the URLLC slice. We again notice that Preemption provides the lowest values for latency and jitter.

In the second experiment, we again create two slices, one URLLC and one eMBB. In this experiment, we



(a) (b)

FIGURE 14. (a) Latency and (b) Jitter of URLLC traffic on increasing the number of RUs (eMBB).



(a) (b)

FIGURE 15. Impact of increasing the data rate of background traffic on (a) Latency and (b) Jitter of the URLLLC traffic.

artificially add background traffic (best-effort) to the eMBB slice and evaluate its impact on the URLLC traffic flow. We progressively increase the data rate of the background traffic from 100 Mbps to 1Gbps (the maximum line rate). Figures 15(a) & 15(b) show the latency (maximum and average) and jitter (average) of the URLLC flow respectively. We again notice that the background traffic in the eMBB slice has no impact on the latency of the URLLC flow thus proving slice isolation. The jitter however does increases slightly in the case of SP and Preemption but this increase is in the scale of few nanoseconds.

The third experiment is the same as the previous experiment, but in this case, we have three URLLC traffic flows instead of one and just the best-effort traffic (VLAN priority 0) in the eMBB slice. In this experiment, we evaluate only the fronthaul traffic. We evaluate slice isolation in terms of throughput and packet loss. The data rate of the traffic flow in the eMBB slice is progressively increased. We compare the results for SP and three different schemes of TSN-Qbv namely: "Qbv-A", "Qbv-50/50", and "Qbv-70/30". In the case of "Qbv-50/50", we fix the schedule such that both the high-priority and low-priority traffic have equal opportunities i.e., the high-priority traffic is allowed to pass for 50% of the cycle-time and the low-priority traffic passes for the remaining 50% of the cycle-time. For "Qbv-70/30", we fix the schedule such that the high-priority traffic is allowed to pass for 70% of the cycle-time and the low-priority traffic passes for the remaining 30% of the cycle-time. Fig. 16(a) shows the throughput achieved for both the URLLC and the eMBB traffic flows. We notice that as we increase the data rate of the eMBB traffic flow, the throughput of the URLLC flow remains unaffected by the eMBB traffic for all the schemes. The received throughput is in accordance with the scheduling scheme. For instance, in the case of SP which always prioritizes the URLLC flow, we notice the received throughput to be the same as the combined sending rate of the three URLLC flows which is 231Mbps over the Fronthaul. In the case of "Qbv-50/50" and "Qbv-70/30", since the high-priority traffic is allowed to pass for 50% and 70% of the cycle-time respectively, the received throughput is the same as the sending throughput as well. However, in the case of "Qbv-A", the scheduler allows the high-priority traffic to pass for a short period of time, and hence the received throughput is only 22.7Mbps and this remains constant irrespective of the data rate of the eMBB traffic flow. In this experiment, we also notice the impact of the TSN-Qbv on the eMBB traffic flow. In the case of "Qbv-50/50", we notice that the received throughput reaches a maximum of only 490Mbps after which it remains constant irrespective of increasing the sending rate. This is because the low-priority traffic is scheduled to pass for only 50% of the cycle time and hence the maximum throughput received is 490Mbps which is half the line rate (1Gbps). The same can be noticed for "Qbv-70/30" wherein the received throughput saturates after reaching 300Mbps (30% of the line rate). Fig. 16(b) shows the packet loss of the traffic flows in terms of percentage. This remains constant for the URLLC flow and is unaffected by the increasing eMBB traffic. However, for the eMBB traffic, the packet loss percentage increases with the increasing rate of eMBB traffic. In the case of SP, since the URLLC traffic is always given priority, we notice no packet loss for the URLLC traffic, however, we notice some packet loss for the eMBB traffic. In the case of TSN-Qbv, the packet loss percentage is observed to be in accordance with the scheduling percentage i.e., in the case of "Qbv-50/50", a maximum of 50% packet loss is observed and in the case of "Qbv-70/30", a maximum of 70% packet loss is observed for eMBB traffic.

The evaluation of network slice isolation was conducted in three sets of experiments. The impact on the URLLC slice was noticed upon increasing the traffic in the eMBB slice by i) increasing the number of RUs and the associated eMBB traffic flows; and ii) adding best-effort background traffic to the eMBB slice. The latency, throughput, and packet-loss of the URLLC traffic remain unaffected thus demonstrating slice isolation. The jitter of the URLLC traffic in the case of TSN-Qbv also remains unaffected. The jitter in the case of SP and Preemption does increase, but this is negligible (few nanoseconds).

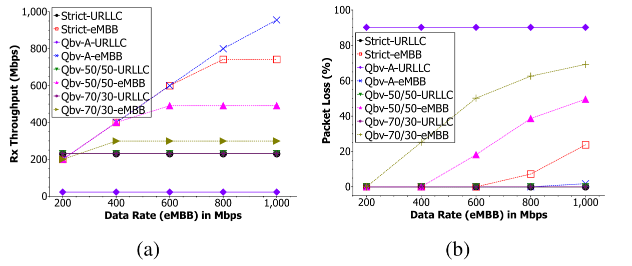


FIGURE 16. **Impact of increasing the data rate of eMBB traffic on (a) Throughput and (b) Packet loss.**

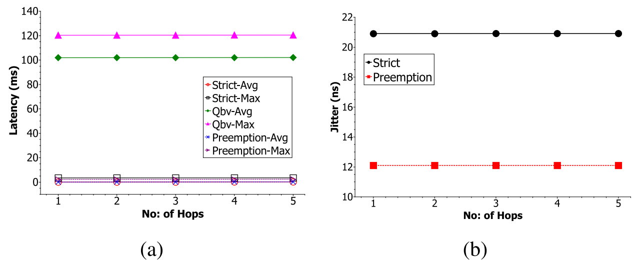


FIGURE 17. **(a) Latency and (b) Jitter of URLLC flow for increasing number of network hops.**

c: SCALABILITY

In this experiment, we evaluate the impact of the network diameter on network performance. For this, we progressively increase the number of hops in the TSN Fronthaul from two to six. We allocate two slices, one for URLLC traffic and one for eMBB with one traffic flow per slice. Fig. 17(a) shows the measured average and maximum latency of the high-priority URLLC traffic flow. We notice that there is a very slight increase in the average latency values as the number of hops increase. The average latency increases by 0.04 milliseconds per hop which is negligible and hence cannot be viewed in the plot. This small increase in the delay can be attributed to the switching latency and hence as the hops increase, so does the end-to-end switching latency. However, the increasing number of hops has no impact on the average end-to-end jitter as seen in Fig. 17(b). Note, that the lowest values of end-to-end latency and jitter are noticed in the case of Preemption. Note, that in the case of 802.1Qbv scheduling, while increasing the number of hops, the clock drift also needs to be taken into account.

d: eCPRI TRAFFIC VALIDATION

For the next set of experiments, we investigate the performance when introducing eCPRI traffic. Since eCPRI implementations are still not available, we use a traffic generator IXIA from Keysight Technologies to generate eCPRI traffic. In this experiment, we allocate two network slices over the Fronthaul, one for URLLC (high-priority) and one for eMBB (low-priority). Each of these slices allows eCPRI user plane IQ traffic. In addition, we also have control data which is designated to the eMBB slice with a lower priority. We vary the

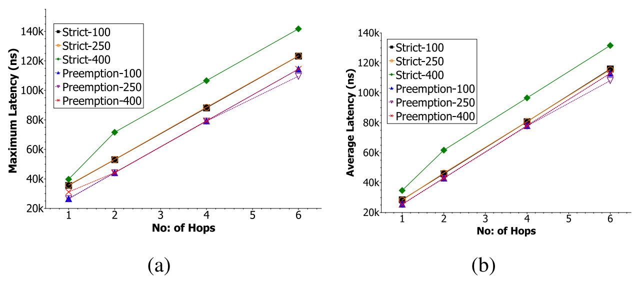


FIGURE 18. Results for eCPRI user plane traffic: (a) Maximum Latency and (b) Average Latency of URLLC flow.

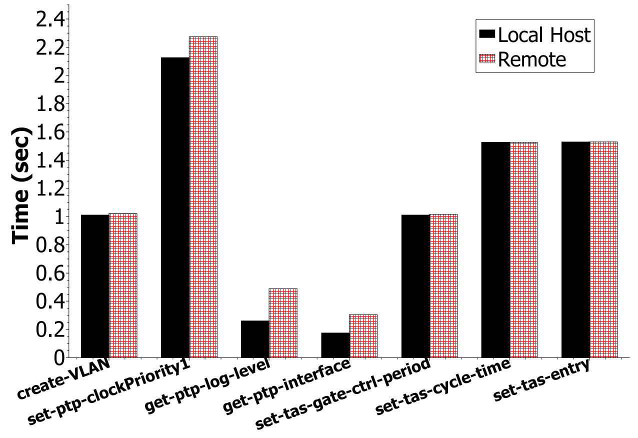


FIGURE 19. TSN network configuration time.

data rate of the traffic in the eMBB slice (100Mbps, 250Mbps, and 400Mbps) and notice its impact on the URLLC slice. We also increase the number of network hops (intermediate switches) to notice its impact. Figures 18(a) & 18(b) show the measured maximum and average latency of the URLLC traffic respectively. Here, we only compare SP and Preemption. We notice that Preemption fares better than SP. The impact of increasing the data rate of the eMBB traffic on Preemption is negligible, however, in the case of SP the latency increases as can be seen from the curve Strict-400 which represents the latency of SP scheme when the eMBB data rate is 400 Mbps. It can also be noticed that the latency increases as the number of network hops increases and this increase is due to the switching latency. As the number of intermediate switches increases, the switching latency increases, and hence the end-to-end latency increases as well. Note, the observed values of latency are only for the fronthaul segment in this experiment.

2) CONTROL PLANE PERFORMANCE

a: TSN CONFIGURATION TIME

In order to evaluate the control plane, we measure the time taken to configure the various TSN attributes via the TSN Plugin for the JOX orchestrator mentioned in the implementation section. We measure the configuration time for creating VLAN, setting PTP clock priorities, and setting Qbv cycle time and schedules. We measure the configuration time for two scenarios. In the first scenario, the TSN plugin as seen in Fig. 9 is hosted in the same machine as the one in which JOX is deployed (Local Host) and in the second scenario, the TSN plugin is running on a different machine (Remote). The results of the TSN configuration time for both these scenarios are depicted in Fig. 19. In principle, the overhead introduced by the message broker which is at the level of few microseconds is slightly lower when hosted in the same machine as the JOX installation. Calls from the JOX agent to the TSN plugin through the message broker and then to OpenDayLight and through NETCONF to the TSN switches are in the magnitude of few seconds (0.2 to 2 seconds) for the TSN, PTP, and VLAN operations. However, we noticed that the PTP convergence time especially in the case of PTP errors was in the level of minutes (not shown in this figure due to scaling). PTP clock recovery in case of failures is an open issue.

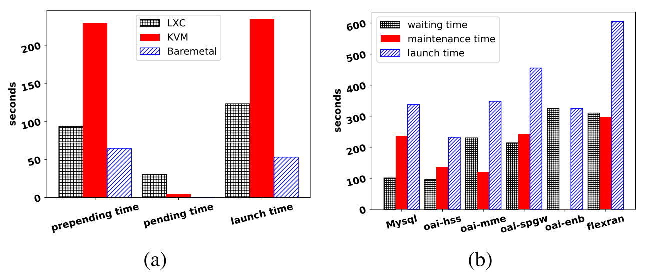


FIGURE 20. Network provision time: (a) Resources provision (b) Services provision.

b: MOBILE NETWORK PROVISIONING TIME

Figures 20(a) & 20(b) illustrate the provisioning time in JOX orchestrator for both the services and the resources on which the concerned services will be deployed. Three different types of time are illustrated: i) average pre-pending time; ii) average pending time; and iii) average launch time. The pre-pending time is the average time from the moment when the Network Service Orchestrator (NSO) of JOX, more specifically template manager, receives the request of allocating the resources (e.g., creating Linux Containers (LXC)/Kernel-based Virtual Machine (KVM)), until the moment when the Resource Orchestrator (RO) creates the resources (e.g., creating LXC/KVM machine). On the other hand, the pending time is the average time from the moment when the NSO receives the ID of the allocated resources from RO until the moment when the NSO is ready to deploy the concerned application on the resources. This time comprises of the time to update the system of the allocated machine or the time of waiting for another application to be ready before deploying the current application. Launch time is the average time from the moment of receiving the request of allocating the resources until the moment the resources are ready to deploy the service. It is thus equal to the sum of the pre-pending time and the pending time. In Fig. 20(a), we see that the majority of the time pertains to the pre-pending stage of the actual creation of the resources. Therefore, it is generally preferable to use LXD/LXC containers since they have a shorter pre-pending time. However, some applications require KVM or bare-metal resources, like SPGW.

Note, that the launch time for bare-metal is lower than that for LXD/LXC and KVM since the process of allocating resources does not include the actual creation of resources.

Another important provision time considered for JOX is the provision time for the services as illustrated in Fig. 20(b), which is composed of: i) waiting time; ii) maintenance time; and iii) launch time. Note, that since OAI does not support all the 5G functionalities, the relevant services used to build the mobile network are annotated using the LTE terminology. Waiting time is the average time from the moment when the request of deploying the service is received until the moment when the resources for the concerned service are ready to start deploying the service on the allocated resources. Maintenance time is the average time from the moment when the service is deployed on the allocated resources until the service becomes ready to start servicing (ready to provide the service). Thus, this is the time to do everything needed to get the service up such as installation, configuration, waiting for the other services to become ready, etc. We can define two main parts for the maintenance time: i) self-maintenance time which is related to the installation, configuration, and update; and ii) waiting time for other services to be ready (e.g., Mobility Management Entity (MME) is waiting for Home Subscriber Server (HSS) to be ready). Active time is the average time from the moment of receiving the request of provisioning a service until the moment when the service is ready to serve the users or service consumer. The important point to notice from the service provision time is that some services require more maintenance time. As mentioned previously, this is due to either the service itself requiring more time to be ready or it needs other services to be ready.

VI. CONCLUSION AND FUTURE WORK

In this work, we presented and analyzed methods and mechanisms required for interconnecting a network slice control and management system of the mobile network with an integrated SDN-based IEEE Time-Sensitive Network (TSN) control plane. We elaborated on the concepts of a TSN-aware Xhaul network, while also describing a new architecture and amendments that can be used to enable network slicing over IEEE TSN. Implementation experience was reported using TSN-enabled prototype devices, OpenAirInterface, and JOX slice orchestrator.

We evaluated the performance of TSN as an enabling technology to provide service differentiation on a per slice basis and also investigated the service provisioning time in addition to demonstrating the interaction between the network slice management plane and the TSN control plane. The performance evaluation conducted revealed the following:

1) Both the IEEE TSN 802.1Qbv and 802.1Qbu techniques can be used to protect high-priority critical traffic flows (URLLC slice) even in overload scenarios.
2) Addition of traffic flows to one slice has no impact on the delay, jitter, throughput, and packet-loss of the other slice thus, preserving network slice isolation.
3) For 802.1Qbv, the network diameter in terms of the number of network hops has a minor impact on delay (attributed to the switching latency) and no impact on jitter as long as synchronization is preserved.
4) The time required to configure the TSN-network can be in the level of few seconds.
5) The bottleneck towards fast network deployment and orchestration is the resource provision time required for the mobile network services.
6) In the case of 802.1Qbv, the scheduling solution calculation time can also be a bottleneck depending on the network size and number of flows.
7) The benefits of 802.1Qbv are maximized when all the sending entities are synchronizing their sending process with the network cycle which is extremely challenging to be achieved and is an open research issue. For non-synchronized sending (all nodes' clocks may be synchronized, but sending time is not aligned with the network cycle), strict priority and 802.1Qbu outperform 802.1Qbv.

Our future work plans include the implementation of NW-TT, DS-TT, AF-TT functionalities, and the implementation of all functionalities provided by the TSN orchestrator. We also plan to investigate hierarchical SDN/CNC solutions, incorporate in our solution a solid analysis and mechanism for the coupling between TSN and DetNet, and migration of the implementation solution to a Kubernetes cloud environment.

REFERENCES

[1] *5G Non-Public Networks for Industrial Scenarios*, 5G-ACIA, Frankfurt, Germany, Jul. 2019. [Online]. Available: https://www.5g-acia.org/publications/5g-non-public-networks-for-industrial-scenarios-white-paper/

[2] A. Nasrallah, A. S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. ElBakoury, "Ultra-low latency (ULL) networks: The IEEE TSN and IETF DetNet standards and related 5G ULL research," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 88–145, 1st Quart., 2019.

[3] A. G. Frank, L. S. Dalenogare, and N. F. Ayala, "Industry 4.0 technologies: Implementation patterns in manufacturing companies," *Int. J. Prod. Econ.*, vol. 210, pp. 15–26, Apr. 2019.

[4] *Integration of Industrial Ethernet Networks With 5G Networks*, 5G-ACIA, Frankfurt, Germany, Nov. 2019. [Online]. Available: https://www.5g-acia.org/publications/integration-of-industrial-ethernet-networks-with-5g-networks/

[5] Y. Pointurier, N. Benzaoui, W. Lautenschlaeger, and L. Dembeck, "End-to-end time-sensitive optical networking: Challenges and solutions," *J. Lightw. Technol.*, vol. 37, no. 7, pp. 1732–1741, Apr. 1, 2019.

[6] K. Christodoulopoulos, W. Lautenschlaeger, F. Frick, N. Benzaoui, T. Henke, U. Gebhard, L. Dembeck, A. Lechler, Y. Pointurier, and S. Bigo, "Enabling the scalability of industrial networks by independent scheduling domains," in *Proc. Opt. Fiber Commun. Conf. (OFC)*, 2020, p. 24.

[7] S. Bhattacharjee, R. Schmidt, K. Katsalis, C.-Y. Chang, T. Bauschert, and N. Nikaein, "Time-sensitive networking for 5G fronthaul networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–7.

[8] K. Katsalis, N. Nikaein, and A. Huang, "JOX: An event-driven orchestrator for 5G network slicing," in *Proc. NOMS - IEEE/IFIP Netw. Oper. Manage. Symp.*, Apr. 2018, pp. 1–9.

[9] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet, "OpenAirInterface: A flexible platform for 5G research," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 33–38, Oct. 2014.

[10] A. Zafeiropoulos, *5G PPP Architecture Working Group: View on 5G Architectur*. Belgium, Brussels: European Commission, 2019.

[11] P. Agyapong, M. Iwamura, D. Staehle, W. Kiess, and A. Benjebbour, "Design considerations for a 5G network architecture," *IEEE Commun. Mag.*, vol. 52, no. 11, pp. 65–75, Nov. 2014.
[12] A. Gupta and R. K. Jha, "A survey of 5G network: Architecture and emerging technologies," *IEEE Access*, vol. 3, pp. 1206–1232, 2015.
[13] P. Marsch, O. Bulakci, O. Queseth, and M. Boldi, *5G System Design: Architectural and Functional Considerations and Long Term Research*. Hoboken, NJ, USA: Wiley, 2018.
[14] S. Husain, *End-to-End Mobile Communications: Evolution to 5G*. New York, NY, USA: McGraw-Hill, 2020.
[15] A. Checko, H. L. Christiansen, Y. Yan, L. Scolari, G. Kardaras, M. S. Berger, and L. Dittmann, "Cloud RAN for mobile networks—A technology overview," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 405–426, 2nd Quart., 2015.
[16] L. M. P. Larsen, A. Checko, and H. L. Christiansen, "A survey of the functional splits proposed for 5G mobile crosshaul networks," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 146–172, 1st Quart., 2019.
[17] N. Alliance, "NGMN overview on 5G RAN functional decomposition," *Next Gener. Mobile Netw.*, vol. 24, p. 15, Feb. 2018.
[18] A. S. D. Alfoudi, S. S. Newaz, A. Otebolaku, G. M. Lee, and R. Pereira, "An efficient resource management mechanism for network slicing in a LTE network," *IEEE Access*, vol. 7, pp. 89441–89457, 2019.
[19] B. Han, J. Lianghai, and H. D. Schotten, "Slice as an evolutionary service: Genetic optimization for inter-slice resource management in 5G networks," *IEEE Access*, vol. 6, pp. 33137–33147, 2018.
[20] *Network Functions Virtualisation (NFV) Release 2; Management and Orchestration; Architectural Framework Specification*, ETSI, Sophia Antipolis, France, Jan. 2021.
[21] R. Su, D. Zhang, R. Venkatesan, Z. Gong, C. Li, F. Ding, F. Jiang, and Z. Zhu, "Resource allocation for network slicing in 5G telecommunication networks: A survey of principles and models," *IEEE Netw.*, vol. 33, no. 6, pp. 172–179, Nov. 2019.
[22] P. L. Vo, M. N. H. Nguyen, T. A. Le, and N. H. Tran, "Slicing the edge: Resource allocation for RAN network slicing," *IEEE Wireless Commun. Lett.*, vol. 7, no. 6, pp. 970–973, Dec. 2018.
[23] M. Peng, Y. Sun, X. Li, Z. Mao, and C. Wang, "Recent advances in cloud radio access networks: System architectures, key techniques, and open issues," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 2282–2308, 3rd Quart., 2016.
[24] D. Camps-Mur, J. Gutierrez, E. Grass, A. Tzanakaki, P. Flegkas, K. Choumas, D. Giatsios, A. F. Beldachi, T. Diallo, J. Zou, P. Legg, J. Bartelt, J. K. Chaudhary, A. Betzler, J. J. Aleixendri, R. Gonzalez, and D. Simeonidou, "5G-XHaul: A novel wireless-optical SDN transport network to support joint 5G backhaul and fronthaul services," *IEEE Commun. Mag.*, vol. 57, no. 7, pp. 99–105, Jul. 2019.
[25] J. Sommer, S. Gunreben, F. Feller, M. Kohn, A. Mifdaoui, D. Sass, and J. Scharf, "Ethernet–a survey on its fields of application," *IEEE Commun. Surveys Tuts.*, vol. 12, no. 2, pp. 263–284, 2nd Quart., 2010.
[26] K. Katsalis, L. Gatzikis, and K. Samdanis, "Towards slicing for transport networks: The case of flex-Ethernet in 5G," in *Proc. IEEE Conf. Standards for Commun. Netw. (CSCN)*, Oct. 2018, pp. 1–7.
[27] E. Varga. (Jun. 2020). *DetNet Data Plane: IEEE 802.1 Time Sensitive Networking Over MPLS*. [Online]. Available: https://tools.ietf.org/id/draft-ietf-detnet-tsn-vpn-over-mpls-03.txt
[28] E. Varga. (Sep. 2020). *DetNet Data Plane: MPLS*. [Online]. Available: https://tools.ietf.org/id/draft-ietf-detnet-mpls-12.txt
[29] S. Draxler, H. Karl, H. R. Kouchaksaraei, A. Machwe, C. Dent-Young, K. Katsalis, and K. Samdanis, "5G OS: Control and orchestration of services on multi-domain heterogeneous 5G infrastructures," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, Jun. 2018, pp. 1–9.
[30] X. Li, R. Casellas, G. Landi, A. de la Oliva, X. Costa-Perez, A. Garcia-Saavedra, T. Deiss, L. Cominardi, and R. Vilalta, "5G-crosshaul network slicing: Enabling multi-tenancy in mobile transport networks," *IEEE Commun. Mag.*, vol. 55, no. 8, pp. 128–137, Aug. 2017.
[31] X. Costa-Perez, A. Garcia-Saavedra, X. Li, T. Deiss, A. de la Oliva, A. di Giglio, P. Iovanna, and A. Moored, "5G-crosshaul: An SDN/NFV integrated Fronthaul/Backhaul transport network architecture," *IEEE Wireless Commun.*, vol. 24, no. 1, pp. 38–45, Feb. 2017.
[32] A. Tzanakaki, M. P. Anastasopoulos, G. S. Zervas, B. R. Rofoee, R. Nejabati, and D. Simeonidou, "Virtualization of heterogeneous wireless-optical network and IT infrastructures in support of cloud and mobile cloud services," *IEEE Commun. Mag.*, vol. 51, no. 8, pp. 155–161, Aug. 2013.
[33] B. R. Rofoee, K. Katsalis, Y. Yan, Y. Shu, T. Korakis, L. Tassiulas, A. Tzanakaki, G. Zervas, and D. Simeonidou, "First demonstration of service-differentiated converged optical sub-wavelength and LTE/WiFi networks over GEANT," in *Proc. Opt. Fiber Commun. Conf.*, 2015, p. 15.
[34] S. S. Craciunas, R. S. Oliver, M. Chmelík, and W. Steiner, "Scheduling real-time communication in IEEE 802.1Qbv time sensitive networks," in *Proc. 24th Int. Conf. Real-Time Netw. Syst.*, 2016, pp. 183–192.
[35] P. Pop, M. L. Raagaard, S. S. Craciunas, and W. Steiner, "Design optimisation of cyber-physical distributed systems using IEEE time-sensitive networks," *IET Cyber-Phys. Syst., Theory Appl.*, vol. 1, no. 1, pp. 86–94, 2016.
[36] W. Steiner, S. S. Craciunas, and R. S. Oliver, "Traffic planning for time-sensitive communication," *IEEE Commun. Standards Mag.*, vol. 2, no. 2, pp. 42–47, Jun. 2018.
[37] L. Zhao, P. Pop, Z. Zheng, and Q. Li, "Timing analysis of AVB traffic in TSN networks using network calculus," in *Proc. IEEE Real-Time Embedded Technol. Appl. Symp. (RTAS)*, Apr. 2018, pp. 25–36.
[38] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman, "Network configuration protocol (NETCONF)," Internet Eng. Task Force, Tech. Rep. RFC 6241, 2011.
[39] M. Bjorklund, "YANG-a data modeling language for the network configuration protocol (NETCONF)," Internet Eng. Task Force, Tech. Rep. RFC 6022, 2010.
[40] K. Katsalis, N. Nikaein, E. Schiller, A. Ksentini, and T. Braun, "Network slices toward 5G communications: Slicing the LTE network," *IEEE Commun. Mag.*, vol. 55, no. 8, pp. 146–154, Aug. 2017.
[41] D. Camps-Mur, K. Katsalis, I. Freire, J. Gutierrez, N. Makris, S. Pontarelli, and R. Schmidt, "5G-PICTURE: A programmable multi-tenant 5G compute-RAN-transport infrastructure," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, Jun. 2019, pp. 469–474.
[42] S. B. H. Said, Q. H. Truong, and M. Boc, "SDN-based configuration solution for IEEE 802.1 time sensitive networking (TSN)," *ACM SIGBED Rev.*, vol. 16, no. 1, pp. 27–32, 2019.
[43] M. Gutierrez, A. Ademaj, W. Steiner, R. Dobrin, and S. Punnekkat, "Self-configuration of IEEE 802.1 TSN networks," in *Proc. 22nd IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2017, pp. 1–8.
[44] N. G. Nayak, F. Dárr, and K. Rothermel, "Time-sensitive software-defined network (TSSDN) for real-time applications," in *Proc. 24th Int. Conf. Real-Time Netw. Syst.*, 2016, pp. 193–202.
[45] M. Boehm, J. Ohms, M. Kumar, O. Gebauer, and D. Wermser, "Time-sensitive software-defined networking: A unified control-plane for TSN and SDN," in *Proc. Mobile Commun.-Technol. Appl. ITG-Symp.*, 2019, pp. 1–6.
[46] N. G. Nayak, F. Durr, and K. Rothermel, "Software-defined environment for reconfigurable manufacturing systems," in *Proc. 5th Int. Conf. Internet Things (IOT)*, Oct. 2015, pp. 122–129.
[47] D. Thiele and R. Ernst, "Formal analysis based evaluation of software defined networking for time-sensitive Ethernet," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, 2016, pp. 31–36.
[48] J. Zou, S. A. Sasu, J. Messenger, and J.-P. Elbers, "Options for time-sensitive networking for 5G fronthaul," in *Proc. 45th Eur. Conf. Opt. Commun. (ECOC)*, 2019, pp. 1–3.
[49] G. O. Pãrez, D. L. Lãpez, and J. A. Hernãindez, "5G new radio fronthaul network design for eCPRI-IEEE 802.1CM and extreme latency percentiles," *IEEE Access*, vol. 7, pp. 82218–82230, 2019.
[50] M. K. Al-Hares, P. Assimakopoulos, D. Muench, and N. J. Gomes, "Modeling time aware shaping in an Ethernet fronthaul," in *Proc. IEEE Global Commun. Conf.*, Dec. 2017, pp. 1–6.
[51] D. Trossen. (Oct. 2020). *DetNet Control Plane Signaling*. [Online]. Available: https://tools.ietf.org/id/draft-trossen-detnet-control-signaling-00.txt
[52] E. Varga. (Oct. 2020). *IP Over IEEE 802.1 Time Sensitive Networking (TSN)*. [Online]. Available: https://www.ietf.org/archive/id/draft-ietf-detnet-ip-over-tsn-04.txt
[53] G. Mirsky. (Aug. 2020). *Operations, Administration and Maintenance (OAM) for Deterministic Networks (DetNet) With IP Data Plane*. [Online]. Available: https://tools.ietf.org/id/draft-mirsky-detnet-ip-oam-03.html
[54] R. Rokui. (Nov. 2020). *IETF network slice for 5G and its characteristics*. [Online]. Available: https://tools.ietf.org/id/draft-rokui-5g-ietf-network-slice-00.txt
[55] B. Wu. (Nov. 2020). *A Yang Data Model for IETF Network Slice NBI*. [Online]. Available: https://www.ietf.org/archive/id/draft-wd-teas-ietf-network-slice-nbi-yang-01.txt
[56] GSMA. (2020). *Generic Network Slice Template Version 4.0*. [Online]. Available: https://www.gsma.com/newroom/wp-content/uploads/NG.116-v4.0-2.pdf

[57] (2020). *GSMA*. [Online]. Available: https://www.gsma.com/aboutus/workinggroups/wp-content/uploads/2019/09/NEST79_Doc_004_NEST-WP4-Global-slice-availability_v1.docx

[58] E. Pateromichelakis, F. Moggio, C. Mannweiler, P. Arnold, M. Shariat, M. Einhaus, Q. Wei, O. Bulakci, and A. De Domenico, "End-to-end data analytics framework for 5G architecture," *IEEE Access*, vol. 7, pp. 40295–40312, 2019.

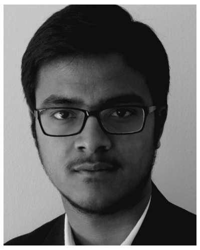
[59] J. Hagerty, R. L. Sallam, and J. Richardson, "Magic quadrant for business intelligence platforms," Gartner Group, Stamford, CT, USA, Gartner RAS Core Res. Note G00225500, Feb. 2012.

[60] A. Abdul, "Hierarchical CUC/CNC management model," in *Proc. IEEE Time-Sensitive Netw. (TSN) Task Group*, 2020, pp. 1–5. [Online]. Available: https://www.ieee802.org/1/files/public/docs2020/60802-abdul-hierarchical-CUC-CNC-management-model-0520-v02.pdf

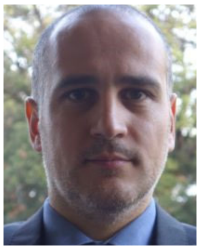
[61] C. Chia-Yu, S. Ruggero, N. Navid, S. Thrasyvoulos, and B. Christian, "Impact of packetization and functional split on C-RAN fronthaul performance," in *Proc. IEEE ICC*, 2016, pp. 5506–5513.

[62] OASIS. (2018). *TOSCA Simple Profile in YAML Version 1.2*. [Online]. Available: http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.2/TOSCA-Simple-Profile-YAML-v1.2.html

[63] M. Vlk, Z. Hanzalek, K. Brejchova, S. Tang, S. Bhattacharjee, and S. Fu, "Enhancing schedulability and throughput of time-triggered traffic in IEEE 802.1Qbv time-sensitive networks," *IEEE Trans. Commun.*, vol. 68, no. 11, pp. 7023–7038, Nov. 2020.



SUSHMIT BHATTACHARJEE received the M.Sc. degree in communications engineering from the Technical University of Munich (TUM), Munich, Germany, in 2018. He is currently pursuing the Ph.D. degree with the Technical University of Chemnitz (TUC). He is also working with the Applied Network Technology Laboratory, Huawei Munich Research Center. His research interests include deterministic networks, time-sensitive networking, and orchestration and management, with his current focus being on network control and management of time-sensitive networks.



KOSTAS KATSALIS is currently a Senior Research Scientist with Huawei, Munich, Germany. He is serving as the Technical Project Manager for a number of projects in the area of time-sensitive networking (TSN) and the TSN Laboratory Manager with Huawei's Munich Research Center. He has been participating in numerous EU FP7 and H2020 projects, such as CONTENT, COHERENT, Q4Health, and 5G-Picture. His research interests include new network designs and technologies, SDN/NFV, time-sensitive deterministic network communications, network programmability, network optimization, and orchestration and management. He serves as a Regular Reviewer for a number of conferences/journals, such as IEEE Transactions on Networking, IEEE Transactions on Network and Service Management (TNSM), IEEE Infocom, and IEEE Globecom.

OSAMA AROUK received the M.S. degree in signal processing and the Ph.D. degree in information sciences from the University of Rennes 1, France, in 2012 and 2016, respectively, with a dissertation on congestion control and power management for M2M communications in LTE and beyond networks. From 2016 to 2018, he was a Postdoctoral Fellow with INRIA, Sophia Antipolis. In 2018, he joined the Department of Communication Systems, Eurecom, as a Research Engineer. He is currently working on network slicing, 5G cloud-native, auto-management, and orchestration for 5G and beyond. His research interests include cloud-native, network slicing, AI/ML, NFV, M2M, and management and orchestration.

ROBERT SCHMIDT (Graduate Student Member, IEEE) received the Diploma degree (Hons.) in information systems engineering from the Dresden University of Technology, Germany, and the Diploma degree in engineering from the Ecole Centrale Paris/CentraleSupélec, France, in 2017. He is currently pursuing the Ph.D. degree in communications with the Department of Communication Systems, Eurecom, France. He is involved in collaborative research projects in the context of the EU H2020 framework program and is an Active Contributor to the OpenAirInterface and Mosaic5G projects. His main research interests include 4G and 5G wireless cellular networks, heterogeneous software-defined (radio access) networks, network slicing, and MAC layer scheduling.

TONGTONG WANG received the B.S. degree in computer science from the Beijing University of Posts and Telecommunications, and the M.S. degree in electrical engineering from Linköping University. She is currently an IP Network Expert with the Department of Wired Network Research, Huawei Technologies Company Ltd. After seven years in IEEE 802 Ethernet standard research and development, her current research interests include network latency guarantee and optimization. She is also an Editor of IEEE P802.1 DF TSN profiles for service provider networks.

XUELI AN received the master's and Ph.D. degrees in electrical engineering from the Delft University of Technology (TU Delft), The Netherlands. She is currently a Principal Researcher and an Industry Development Specialist with the Munich Research Center, Huawei Technologies, Germany. Within Huawei, she has the responsibility for mobile communication enabled vertical industry-related research, standardization, and industry development-related activities. She has a deep engagement with the 5G-enabled Industry 4.0 ecosystem. She has been serving for 5G Alliance for Connected Industries and Automation (5G-ACIA) Working Group "Use cases and Requirements" as the Co-Chair, since 2018. She is the Vice-Chair of Networld 2020 Enabling Technologies for Future Vertical Ecosystem Transformation Working Group. She has over 50 international journal/conference publications and over 20 patent applications in the field of wireless communication and networking.

THOMAS BAUSCHERT (Member, IEEE) received the Dipl.-Ing. and Dr.-Ing. degrees from the Technical University of Munich (TUM), in 1990 and 1997, respectively. From 1997 to 2007, he was with Siemens and Nokia Siemens Networks, Munich. Since 2007, he has been a Full Professor with the Technical University of Chemnitz (TUC), and heading the Chair of Communication Networks at the Faculty of Electrical Engineering and Information Technology. His research interests include methods and architectures for flexible and reliable communication in fixed, and wireless networks with a special focus on network design and automation and on network security.

NAVID NIKAEIN received the Ph.D. degree in communication systems from the Swiss Federal Institute of Technology, EPFL, in 2003. He is currently a Professor with the Department of Communication System, Eurecom. He is also leading a group on experimental 4G/5G system research related to radio access and core networks coupled with edge computing with a particular focus on industry-driven use-cases. He is leading the development of the radio access layer of OpenAirInterface and coordinating the Mosaic5G initiative whose goal is to provide software-based 4G/5G service delivery platforms.