Low Latency MEC Framework for SDN-based LTE/LTE-A Networks

Anta Huang, Navid Nikaein, Tore Stenbock, Adlen Ksentini, and Christian Bonnet
Communication Systems Department, EURECOM, Biot, France 06410
firstname.lastname@eurecom.fr

***Abstract*—Mobile Edge Computing (MEC) consists of deploying computing resources (CPU, storage) at the edge of mobile networks; typically near or with eNodeBs. Besides easing the deployment of applications and services requiring low access to the remote server, such as Virtual Reality and Vehicular IoT, MEC will enable the development of context-aware and context-optimized applications, thanks to the Radio API (e.g. information on user channel quality) exposed by eNodeBs. Although ETSI is defining the architecture specifications, solutions to integrate MEC to the current 3GPP architecture are still open. In this paper, we fill this gap by proposing and implementing a Software Defined Networking (SDN)-based MEC framework, compliant with both ETSI and 3GPP architectures. It provides the required data-plane flexibility and programmability, which can on-the-fly improve the latency as a function of the network deployment and conditions. To illustrate the benefit of using SDN concept for the MEC framework, we present the details of software architecture as well as performance evaluations.**

***Index Terms*—MEC, SDN, LTE, OpenFlow, Programmability.**

I. Introduction

Having smart and diversified applications toward 5G mobile network requires pushing the boundaries of existing network and service infrastructure. Services desiring lower-latency are emerging; therefore pushing network services to the edge has the potential to enhance user latency and experience, as well as to offload Internet traffic. Due to the inevitable huge flood of traffic, one of the key emerging challenges is a need for faster throughput and lower latency to handle the ever increasing demand of content-rich services such as high-resolution video streaming and low-latency augmented reality.

Introduced and specified by ETSI (ETSI MEC 001-005), Mobile Edge Computing (MEC), concurrently evolving to Multi-access Edge Computing towards 5G, is a platform enabling applications to run at the mobile edge and in close proximity to end-users. From a high point-of-view, MEC itself is a cloud computing entity with access and capabilities to perform tasks at the edge of mobile networks allowing enhancements and innovation through network applications. For example, UE requests a video streaming service which matches one of the applications located in MEC platform. MEC has the ability to apply traffic rules to program the data path and redirect the traffic to the corresponding service provider, whether it's local or remote to lower the latency. In this case, the service can be served by the corresponding server to improve the perceived user experience in a totally transparent manner. Not only does MEC provide technical benefits, but it also creates a new market and value chain not seen before in mobile networks by opening the network to authorized third-parties, who can develop and rapidly install innovative network applications, benefiting both the third-parties and the network owners.

Software-Defined Networking (SDN) is a promising solution and already exploited extensively in non-mobile networks. It provides a network architecture where the control plane has been migrated from physical network devices to off-device remote controllers. The underlying infrastructure can therefore be abstracted creating unparalleled opportunity for innovation and customization for network applications and services. The noticeable success in non-mobile networks made by SDN gives the initiatives to apply it to the core network of LTE. With the separation of control and data plane, SDN provides the possibility to program and virtualize the LTE/LTE-A network components, such as MME, control plane of Serving-Gateway (SGW-C), and control plane of Packet-Gateway (PGW-C) as MEC applications. The programmability of the core network provided by SDN is exactly where MEC can cooperate and facilitate its programmability in Radio Access Network (RAN) by delegating control decisions. SDN also has the same objectives as MEC in the way of applying specific rules to data plane. SDN and MEC are complementary concepts and the interplay between them is studied and attempted in this work. Given the open specifications of MEC for vendor implementation, the SDN concept is applied in the proposed framework with OpenFlow-enabled switch.

Specifically, the contributions of this work cane be summarized as follows:

- (Section III and IV) An ETSI and 3GPP compliant low-latency MEC framework for SDN-based LTE/LTE-A network is proposed and implemented. With the low-latency feature, our framework enables massive number of connected devices into the mobile networks today and tomorrow. Besides the execution environment for mobile edge applications, OpenFlow protocol negotiation and switching integration for GTP are described systematically.
- (Section V) Both system and performance evaluation are measured in terms of latency and CPU loading to present the significant enhancements resulted from the interplay between SDN and MEC.

II. Related Work

With the openness of the ETSI specifications, MEC working group has produced plenty of discussions and thus various architectures have emerged. ETSI presents the ecosystem of MEC and the main six use cases in [1]. In addition to providing the MEC blueprint, the applications for deploying a varieties of services at the mobile edge is categorized in [2]. For instance, a service is proposed to offload encoding tasks from mobile devices to MEC, allowing to reduce the power consumption of mobile devices [3]. A hierarchical architecture for multi-access edge computing is proposed by leveraging cloud computing and migrating mobile workloads for remote execution at the cloud [4]. Notice that MEC is a complementary approach to the current and future cellular architecture and an explanation of how a real-time context-aware application can be built by collaborating MEC and 5G RAN is in [5]. A comparison between MEC, fog computing and cloudlet is in [6] and a complete conceptual MEC architecture considering full functionalities, interfaces, and applications are also provided in [7].

Using SDN for the core network of LTE is intuitively the first step and there have been several research papers around this concept [8]–[11]. These papers are studying SDN in mobile networks from different aspects such as scalability, 3GPP interoperability, and performance evaluation, highlighted the needs and benefits of adopting SDN concept onto core networks. In addition, SoftRAN [12], a centralized control plane for radio access networks, which abstracts all base stations into a virtual big-base station comprised of a central controller. FlexRAN [13] provides a flexible control plane to build real-time RAN control applications, remains flexible to realize different degrees of coordination among RAN infrastructure entities and programmability for adapting control over time.

Most of aforementioned studies focus on a top-down view and examine the MEC concept from the application perspective; however, the underlying framework inside the *ME Host* is not fully specified. The ETSI MEC ISG initiates the *MEC* framework standardization; however, only the data-plane part is considered. A MEC framework adopting SDN concepts for both radio and core network is not currently in place to the best of our knowledge. This work, compared to the aforementioned studies, proposes and implements a MEC framework with ETSI and 3GPP compliance and focuses on how LTE/LTE-A network can fully cooperate with MEC and SDN.

III. Low-latency MEC Framework Overview

This section gives an overview of the proposed framework and describes how SDN-based LTE/LTE-A networks can operate over the MEC architecture. In this paper, the control plane and data plane of S-GW and P-GW are annotated as *X*-GW-C and *X*-GW-U. Fig. 1 provides a high-level view of the proposed MEC framework, mainly composed of three layers design: *MEC Application*, *MEC Platform*, and *Abstraction*. This framework fully separates the data plane from core nodes and functions upon multiple LTE eNodeBs and OpenFlow-enabled switches, whether it is physical or software, to provide RAN functionalities and comprise the data plane of a

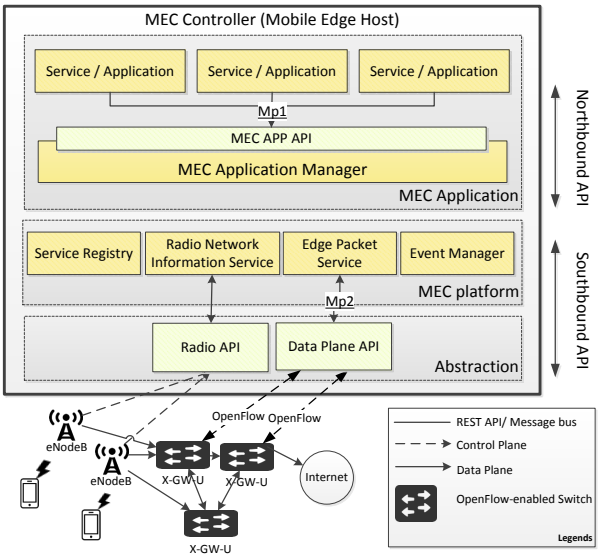


Fig. 1: High-level schematic of the proposed framework

LTE network. The entities and interfaces we realize in this framework follow the ETSI MEC specifications to remain full functionalities support while retaining the 3GPP compatibility. The Mp1 reference point is the interface between mobile edge platform and applications, and the Mp2 is the interface between mobile edge platform and the abstracted data plane as specified by ETSI (refer to section IV-B).

A. Abstraction

With the SDN concept introduced, *Abstraction* mainly includes two entities: *Radio API* and *Data Plane API*, and acts as an abstraction layer of control plane and data plane respectively by providing only the necessary information to the development of *MEC Applications*.

1) Radio API: Radio API basically provides an abstract view of the radio network status (e.g., topology, band, and signal strength) by extracting the parameters of interest from the RAN with the required level of granularity. Besides, it also gives the possibilities to modify the state of the underlying network and passes the control decisions based on the statistics and events gathered in the RAN. All the aforementioned features occur at the interaction between radio network and MEC platform through *Radio API*.

2) Data Plane API: Data Plane API plays the abstraction role for data plane. It provides the *Mp2* interfaces for *Edge Packet Services* (EPS) located in the MEC platform to control the data plane of the core network. EPS is able to pass the required rules to OpenFlow-enabled switches through *Data Plane API*. The rules can be default rules for initial establishment of data plane setup, specific rules for UE traffic data path establishment by X-GW-C, or MEC application specific rules setup by MEC applications.

B. MEC Platform

MEC Platform resides in a Mobile Edge Host as a middleware between the applications and the real RAN elements. It gives the possibilities for application developers to focus

on their specific application purpose rather on the detailed functionalities of underlying RAN. The detailed concept of each component is described as follows.

1) Radio Network Information Service: Specified by ETSI MEC, RNIS provides authorized applications with radio network related information. In our framework, RNIS exposes the information, such as real-time radio bearer statistics, measurements and statistics related to UE, state changes of UE, and power measurements to applications by interacting with *Radio API*.

2) Service Registry: The service registry identifies the available services, supported by the *MEC Platform* and gives the abilities for high-level applications to register interested services, such as radio network information and location information.

3) Edge Packet Service: Edge Packet Service (EPS) is the main component for managing data plane and also corresponds to traffic rules control entity specified by ETSI. EPS brings a native IP-service end-point to the MEC applications and acts as a local IP agent performing network functions, like IP forwarding, packet encapsulation/decapsulation and data transcoding. EPS also gives the abilities for applications to adapt the routing for their specific purpose. Traffic coming from UEs through OpenFlow-enabled switches will go along the routes based on the rules setup in the switches. In this case, applications can control the routing path through the `Mp1` interface towards EPS, and then the latter will set the OpenFlow rules in the corresponding switches to redirect the traffic.

4) Event Manager: An event dispatching system is utilized in this framework. Any raw data changes acquired from network abstraction such as variations in CQI (Channel Quality Indicator), newly-connected UE events, and packets mismatch can all be packaged as a simple event. Services and applications are therefore able to track interested simple events about things that happen in the underlying network. When simple events occur, those registered services and applications will be notified about the changes or more complex event like the level of network congestion which is composed of the information from simple events. In this framework, *Event Manager* is responsible for events publishing and subscribing. Based on the indicated reporting frequency, like periodical, one shot, or event-driven, event manager will send information to the registered applications and services accordingly.

C. MEC Applications

One of the main benefits coming with the separation of control and data plane is that the applications on top of platform has the limitless possibilities to be developed for any specific purpose without knowing the detailed knowledge for underlying network. In the proposed MEC framework, applications communicate with MEC platform through the northbound interfaces (MEC APP API), mapped as Mp1 reference point in ETSI MEC 003. The northbound interface allows the applications to access the information abstracted and organized from network or delegate the control decision toward network. Not only can the applications interact with MEC platform through provided API to consume and provide mobile edge services, but the applications also can provide services producing information and messages that other applications might be interested in (e.g., producing high-level organized and structured radio information as *eNodeB Producer*).

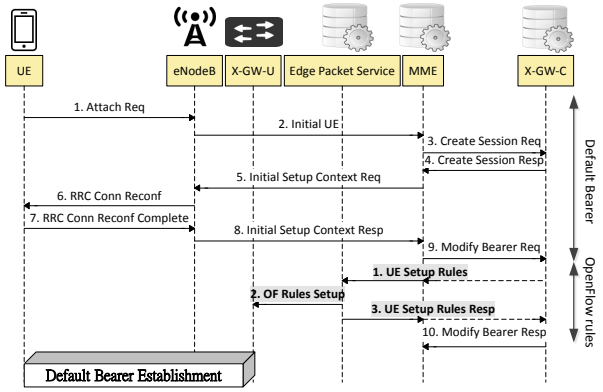


Fig. 2: Sequence diagram of default bearer establishment.

D. Basic Workflow

Fig.2 shows the basic steps of how SDN-based LTE network operates on MEC and interacts with EPS to handle UE initial attach procedures for bearers establishment. The main point of the sequence diagram starts from the message calls initiated all the way from X-GW-C through EPS to X-GW-U. Notice that the entity to initiate the API call is not limited to be S-GW-C. Both MME and P-GW-C can start the procedure since they have the equal understanding about the bearer setup information. The OpenFlow rules setup procedure can be initiated at three (MME, S-GW-C, and P-GW-C) different timing for default bearer As soon as the UE is properly attached to the network with MME and X-GW-C knowing the GTP information, X-GW-C will initiate the procedure to transmit the UE information (*UE Setup Rules*) to EPS and then based on the rules, EPS is able to setup the OpenFlow rules (*OF Rules Setup*) in corresponding switches. By introducing the concept of SDN into LTE through integration of OpenFlow-enabled switches, the default bearer can be setup among UE, eNodeB, and X-GW-U as can be observed in the last step of sequence diagram. By configuring the OpenFlow rules as soon as UE completes the initial attach procedure, the UE can access the Internet as normal.

IV. Design and Implementation

The core of the proposed low-latency MEC framework is the EPS and the integration of RNIS. The layered structure of MEC framework can be seen in Fig. 3, showing the main software components. As illustrated in the figure, applications comprising the upper-most layer manage the network based on the information gathered through Mp1. The eNodeB and Open vSwitch form the data plane as the bottom-most layer. The core components, RNIS and EPS, located in the middle of them play the network abstraction role and provide the required interfaces for applications. The details of our design and implementation is provided as follows.

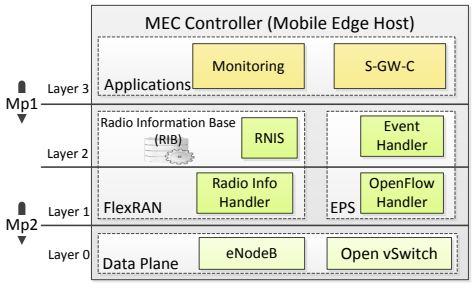


Fig. 3: Software components of MEC Controller (ME Host) and its interfaces

A. MEC Applications

There are many ways that MEC applications can acquire and pass network configuration of both the core and the radio side. A part of the specifications defines that MEC applications shall be provided with this information if requested. Basically, the structured and meaningful information can be provided from both radio and core networks. In this way, numerous applications based on radio information to adapt data plane decision could be easily achieved. Two applications, *monitoring* and *S-GW-C*, are implemented as examples in our proposed framework.

1) Monitoring: Thanks to the integration of RNIS and EPS, the monitoring task can have full coverage on RAN and core network including both control and data plane. A simple event notified form RAN such as CQI changes and resource block usage or from core network such as packets drop can be registered as interested events in an indicated reporting frequency. More complex events like the level of network congestion combined with simple events can also be registered and transformed from raw data to meaningful information.

2) S-GW-C: The control plane of S-GW (S-GW-C) is separated and extracted from core network as an application located in MEC platform. The S-GW-C acts as a gateway, as it is specified in 3GPP, controlling UE initial attach procedure, the mobility anchor point for inter-eNodeB handover, and so on. However, the data plane traffic does not go through control entities anymore. S-GW-C only notifies EPS about newly-connected UE through Mp1 and EPS handles the rest of procedures to establish the default bearers for each UE. Through EPS, S-GW-C can also gather data plane statistics without having the knowledge of OpenFlow.

B. Edge Packet Service

One of the advantages of SDN concept is the abstraction of the network layer allowing MEC applications to start interacting with the network through a set of interfaces. As can be seen in Fig. 3, EPS uses *Event Handler* to facilitate the event notifications for applications and it also covers the OpenFlow rules interpretation and configuration. The fundamental API calls that are needed for the system to work are defined and can be organized into two groups: *Mp1 API* and *Mp2 API*

1) Mp1 API: Mp1 is the interfaces for applications to establish the basic functionality in the core network and also fulfill the specific request from applications. The following features are implemented:

Default bearer: In the proposed framework, the OpenFlow-enabled switches in the core network acting as X-GW-U have dedicated APIs integrated into EPS for X-GW-C to notify that a new UE has just attached to the network through *UE Setup Rules* call, and then the OpenFlow rules to establish default bearer needs to be setup for the UE. When this *UE Setup Rules* API is called, the message must include the identities, e.g. UE IP, bearer ID, DL and UL tunnel ID, for bearer establishment. For each bearer, it has distinct bearer ID to differentiate between each others. All the information of each bearer needs to be sent through *UE Setup Rules*. Therefore, our design, following RESTful API for uniform interface, allows a bearer list consisting of one or more than one bearer information to be sent in one API call. The EPS will then store this UE information and update the network layer with default user-plane rules so that user traffic can traverse the network as normal.

MEC APP specific: In the proposed implementation, general MEC APP API end-points like monitoring the radio information or modification of an individual user traffic handling are provided for MEC applications. Regarding the data plane APIs, it currently allows to apply only two rules on per user traffic flow basis. The difference is that one rule allows a MEC application to request that all traffic for a certain UE has to be *copied* and sent to a receiver inside the calling MEC application. The other API is similar but instead of copying, all traffic is *redirected*. In other words, a traffic is either copied for analysis without delaying latency or hindering normal functions, while other traffics are directly processed at MEC. Both with their own benefits.

2) Mp2 API: Mp2 is, from EPS point-of-view, the OpenFlow connections to OpenFlow-enabled switch. First, when the application is running and a switch with OpenFlow support is independently turned on, an event will occur that triggers execution of the logic handling *switch-joined*. This consists of storing the provided information from the new switch connection and sending of default OpenFlow rules. Second, a *packet-in* handler is provided when any switch has detected a rule match that commands a packet to be sent to the EPS. In most cases, this means that a MEC application has requested packets from certain network devices for analysis or other services and the packet will be sent up through *Mp1* appropriately. Finally, a *switch-left* event occurs when the OpenFlow connection is closed. Subsequently the switch information is removed from the storage and the switch will no longer be considered for any future events until the next connection.

The types of OpenFlow rules that EPS creates and maintains can be categorized into three groups as follows:

Default rules: There are a set of rules that are pushed to a newly connected OpenFlow-enabled switch. The general idea is that this set of rules are pushed only once and hence are called the default rules. These rules consist of how to handle packets that have little or no changes in packet fields such as

Address Resolution Protocol (ARP), Domain Name System (DNS) queries, no match actions, and the control-plane rules according to configured network settings.

UE specific rules: UE specific rules are mainly to establish default bearers for UE in SDN-based LTE network. After the default rules are setup, the control plane (S1-MME, S11, and S5/S8) of SDN-based LTE network works as normal and UE can attach successfully without data plane functionality. At this stage, UE specific rules are required to be present in the flow table of switch in order to have default bearer established. With S-GW-C as an example knowing the UE information (TEID), rules with TEID is setup in OpenFlow-enabled switch through EPS and then default bearer is able to be established.

MEC application rules: When a user connects to a network, the EPS needs to be notified about this event and provided with the specific tunneling information for that UE from MME, which is accomplished through the Mp1 API. Then, appropriate OpenFlow rules are sent to the corresponding switch.

C. Radio Network Information Service

In order to have a clean separation of control and data plane for radio access network, *FlexRAN* is integrated into our framework [13]. *FlexRAN* mainly written in C++ follows two-layer architectures, controller managing agents and agent running above eNodeB, and integrated as RNIS and Radio Info Handler respectively.

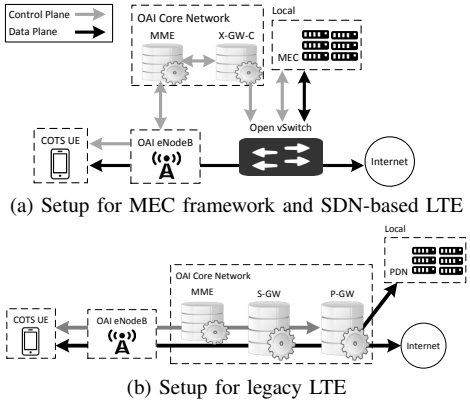
1) RNIS: One of the key features *FlexRAN* controller has is the RAN Information Base (RIB) to perform the main task of RNIS. All the statistics and configurations related information about the RAN, i.e. UEs and eNodeB, are all maintained in RIB. Only the RIB Updater component of the controller can update the RIB with the information received from the agents and real-time applications can be supported in this way instead of allowing write conflicts from multiple writers.

2) Radio Info Handler: To control and manage data plane of RAN, *FlexRAN* Agent is integrated as Radio Info Handler which defines a set of functions that basically act as an abstraction layer allowing the management of the higher-level control operations in a technology agnostic wayThis API defines a set of functions that can be used by the data plane to notify the control plane about events such as the initiation of a new Transmission Time Interval (TTI), the state change of a UE that has been turned off and the attempt for random access by a UE [13].

V. System Evaluation

Throughout the whole evaluation, we use Open vSwitch v2.5.1 with OpenFlow v1.0 Protocol to handle data plane traffic. Besides, in order to have GTP tunneling function, a OVS patch[1] is applied on the Open vSwitch 2.5.1 implementation. It is however important to note that a physical switch with OpenFlow and GTP supports can be used with the proposed framework. All the components, such as eNodeB, core network, Open vSwitch, and *ME Host* are running on

[1] "Basic GTP-U tunnel implementation in OVS" developed by Niti Rohilla.



(a) Setup for MEC framework and SDN-based LTE

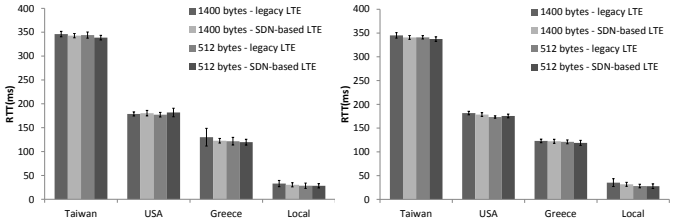
(b) Setup for legacy LTE

Fig. 4: The evaluation setup conducted

commodity Linux-based PC. Two types of evaluations are conducted to evaluate our proposed framework: *Compatibility*, where the measurements are taken in a real LTE network with COTS UEs to evaluate the latency performance; and *Scalability*, where measurements are taken in an emulated LTE network to evaluate the benefit of core network offloading when redirecting the increased user traffic.

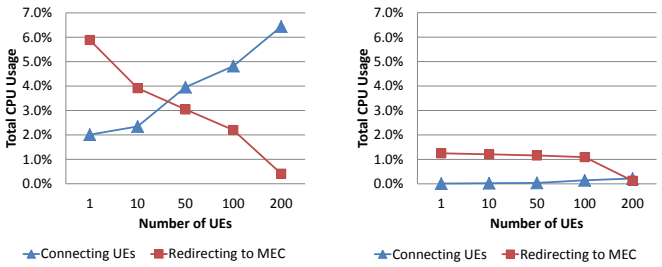
A. Compatibility

OpenAirInterface (OAI), a software LTE platform, is used as a real-time LTE environment. OAI permits to verify that the proposed MEC framework featuring SDN can operate with full LTE functionalities, and therefore the related latency measurements can be also provided. The setups for legacy LTE and SDN-based LTE are shown in Fig. 4. All the entities except COTS UE are connected with real-time LTE network via Gigabit Ethernet connections. COTS UE (HUAWEI E392 4G LTE dongle) is connected through a real RF front-end (Ettus B210 USRP). All the conducted measurements use the same eNodeB configuration, namely FDD with transmission mode 1 (SISO) and 5MHz bandwidth in band 7. In addition to compatibility experiment, the measured latency corresponds to Round Trip Time (RTT) of packets sent from one UE to different servers located in: Taiwan, USA, Greece, PDN, and local (that includes the MEC platform). For each server location, the packets are sent with different sizes of packets (512 bytes, 1400 bytes) using different interval of times between transmissions (1.0s, 0.2s). The measurement results for each server location are shown as a test group in Fig. 5, and the result for server located at PDN is combined with the server located at MEC as one test group, denoted as local, in the last test group of each figure. It can be seen that for each test group from left to right, the overall latency is significantly decreased using the proposed MEC framework by a factor of 10 when comparing the server located at Taiwan and local, and by a factor of 5 when comparing the server located at USA/Greece and local. We can also notice that in each test group, no or little improvement in latency can be obtained between legacy and SDN-based LTE setting. This is because the data plane for both x-GW and OVS is performed in the kernel space, which has a comparable GTP tunneling performance. The little improvement is achieved by offloading



(a) IDT = 1.0s (b) IDT = 0.2s

Fig. 5: UE RTT measurements



(a) S-GW CPU% Load (b) P-GW CPU% Load

Fig. 6: Impact of offloading with increasing number of UEs

the data plane to the switch due to the separation of the control and data plane. This suggests that the main benefit that can be stemmed form a SDN-based LTE is the one-the-fly data plane programmability through the consolidation of the control plane within the SDN and MEC controller. Further, we can also observe from the figure that a small latency gain can be achieved by placing the server at MEC when comparing to the PDN (see the local test group in Fig. 5). This is due to the proximity of S-GW and P-GW in the considered small scale LTE deployment that can be found, for instance, in public safety use-cases. However, when considering a large scale deployment, where S-GW and P-GW are far away from each other geographically, the induced delay (i.e. transport and processing) can be avoided through MEC. The results not only validate the feasibility of the proposed low latency MEC framework, but also show that the proposed SDN-based MEC can provide the required flexibility in steering the user traffic on-the-fly to a local or remote servers so as to improve the latency as a function of the network deployment and conditions while retaining the compatibility with the legacy LTE architecture.

B. Scalability

Regarding the scalability performance, we use the same setup as in Fig 4(a), excepting the fact that the control plane traffic (S1-MME) is also handled by the switch. In order to determine the load of SDN-based core entities, a Python script is developed as an UE traffic generator to transmit a small collection of values every 10 seconds. After SDN-based OAI LTE platform is up and waiting for connections, the UE traffic generator will be added so as to increase the number of connected users from 1, 10, 50, 100, to 200. At the same time, CPU resource starts being monitored and recorded. As can be observed in the Fig.6, the CPU usage, when connecting UEs, is increasing in both figures and reach around 7% and 1% in S-GW and P-GW respectively. The next step is that the EPS starts increasing the number of UEs traffic being redirected as well as the CPU usage is kept recording. As can be seen in Fig. 6, the CPU utilization, at X-GW, scales with the number of connected UEs; validating not only the benefit of MEC in terms of dynamic traffic offloading but also in terms of applying UE specific rules when redirecting only a group of UEs.

VI. Conclusions and Future Work

In this paper, a low-latency MEC framework based on SDN is proposed and implemented; providing a flexible data plane programmability for both MEC and mobile core network, while retaining the compliance with 3GPP. The obtained results, through a real implementation of the framework, have shown a significant latency reduction and traffic offloading can be achieved using MEC; indicating that the benefit of SDN in MEC stems from the consolidation of the control plane of mobile network in conjunction with MEC. The low-latency feature of the proposed framework is a key in the support of emerging use-cases found in vehicular communications and content optimization. In the future, we plan to extend the framework to support dedicated bearers and perform large scale experiments consisting of thousands of UEs and measure traffic offloading gain under different conditions.

Acknowledgments

Research and development leading to these results have received funding from the European Framework Program under H2020 grant agreement No. 671639 for the COHERENT project and grant agreement No. 723172 for the 5G!Pagoda project.

References

[1] M. Patel *et al.*, "Mobile-edge computing - introductory technical white paper," *ETSI*, 2014.
[2] M. T. Beck *et al.*, "Mobile edge computing: A taxonomy," in *International Conference on Advances in Future Internet*, 2014.
[3] ——, "ME-VoLTE: Network functions for energy-efficient video transcoding at the mobile edge," in *ICIN*, 2015.
[4] L. Tong *et al.*, "A hierarchical edge cloud architecture for mobile computing," in *IEEE INFOCOM 2016*, 2016.
[5] S. e. a. Nunna, "Enabling real-time context-aware collaboration through 5G and mobile edge computing," in *ITNG*, 2015.
[6] R. Roman *et al.*, "Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges," *arXiv: 1602.00484*, 2016.
[7] C.-Y. Chang *et al.*, "Mec architectural implications for lte/lte-a networks," ser. MobiArch '16, 2016.
[8] K. Pentikousis *et al.*, "Mobileflow: Toward software-defined mobile networks," *IEEE Communications Magazine*, 2013.
[9] M. Martinello *et al.*, "Keyflow: a prototype for evolving sdn toward core network fabrics," *IEEE Network*, 2014.
[10] V. Nguyen and Y. Kim, "Proposal and evaluation of sdnbased mobile packet core networks," *EURASIP*, 2015.
[11] A. Ksentini *et al.*, "On using sdn in 5g: The controller placement problem," in *IEEE Globecom*, 2016.
[12] A. Gudipati *et al.*, "Softran: Software defined radio access network," ser. HotSDN '13, 2013.
[13] X. Foukas *et al.*, "Flexran: A flexible and programmable platform for software-defined radio access networks," in *CoNext*, 2016.