# Towards Slicing-Enabled Multi-Access Edge Computing in 5G

Adlen Ksentini and Pantelis A. Frangoudis
EURECOM, Sophia Antipolis, France
Email: name.surname@eurecom.fr

**Abstract**

Multi-access Edge Computing (MEC) and Network Slicing are two key enablers for 5G, particularly to empower low-latency services, known as Ultra-Reliable Low Latency Communications (URLLC). However, MEC and Network Slicing are evolving in parallel, and are being defined by two different standardization bodies, ETSI and 3GPP, which limits their integration and their benefits as complementary solutions. In this paper, we fill this gap by providing a novel scheme, compliant with both ETSI and 3GPP, that integrates these two key technologies and brings enhanced slicing capabilities to the edge of the 5G network. In particular, we devise a novel management and orchestration architecture, based on the latest 3GPP specifications, which integrates MEC as a 5G sub-slice. Furthermore, we highlight several issues that emerge when extending network slicing to the edge, security and isolation included, providing a solution for each issue.

## I. Introduction

Network Slicing (NS)[1] is envisioned as one of the key enablers of the 5G system; it allows sharing a common physical infrastructure to provide virtual networks tailored to services' (or applications') needs. NS relies on network softwarization, i.e., Software Defined Networking (SDN) and Network Functions Virtualization (NFV), to provide flexible and dynamic virtual networks. A network slice, in the context of 5G, is composed of sub-slices covering the Radio Access Network (RAN), Core Network (CN) and the transport network. Each sub-slice is composed of a set of VNFs chained together (e.g., parts of the RAN or CN elements), or a mix of Virtual Network Functions (VNFs) and Physical Network Functions (PNFs); the latter typically are RAN components.

[1]For the sake of readability, abbreviations used in this paper are listed in Table I.

Edge computing is a complementary solution to sustain low latency for time-critical services, known in 5G as URLLC services. In this context, ETSI is leading standardization activities around Multi-access Edge Computing (MEC). Several 5G use-cases are expected to rely on MEC to deliver added value services to the end users. In addition to providing an execution environment for running applications at the edge, MEC provides services that supply information on end user and base station (eNB) context, such as the radio channel quality of users and their location in the network, allowing to build context-aware applications.

Meanwhile, 3GPP has put efforts [1], [2] into integrating NS in the future specification of both the RAN and CN. Importantly, 3GPP has created three Service and System Aspects (SA) groups, SA1, SA2 and SA5, which aim to, respectively, update the RAN, the CN, and describe a management framework for Network Slicing in 5G. First results of these groups are: (i) the usage of a slice identifier (S-NSSAI: Single-Network Slice Selection Assistance Information), when the User Equipment (UE) first connects to the RAN; (ii) the introduction of a new CN architecture, which is virtualization-ready, and integrates a Network Slice Selection Function (NSSF) that aims to help the RAN select the CN functions corresponding to a UE's S-NSSAI; (iii) a new framework that manages the life cycle of 5G network slices. However, the support of NS in MEC is in its infancy. A recent group [3] has been constituted to evaluate to what extent MEC can support NS. Its first document describes only some use-cases and requirements to support NS at the MEC level and many points are kept open. First, a new MEC architecture should be devised and aligned with (i) the current 3GPP specifications to fit with the 5G architecture at both the RAN and CN, and (ii) the integration of MEC in NFV, while considering the new Network Slicing management framework as introduced by 3GPP. Second, the MEC service model should be revised in order to guarantee security and isolation for network slices. Finally, the registration and discovery of MEC services, provided by third-party MEC applications, need to be adapted to the context of *sliced* MEC.

In this article, we address the aforementioned gaps by proposing a novel orchestration/management architecture that allows to deploy a MEC platform as well as MEC applications in a 5G environment that supports NS, while being aligned with the new MEC-in-NFV ETSI recommendations [4]. Furthermore, we discuss some issues regarding NS security and isolation, as well as the registration of MEC services by third-party application providers when slicing MEC; for each concern, we provide solutions. Finally, we report on our experiences implementing a fully-fledged, standards-compliant MEC system, providing some technical solutions and extensions for the support of NS in MEC-in-NFV, as well as presenting some early performance results from our testbed. To the best of our knowledge, this article is the first to introduce solutions for sliced MEC, and the integration of the latter in the 5G Network Slicing model.

TABLE I

GLOSSARY

| Abbreviation | Name |
|---|---|
| 3GPP | 3rd Generation Partnership Project |
| AF | Application Function |
| AMF | Access and Mobility management Function |
| API | Application Programming Interface |
| AppD | Application Descriptor |
| BSS | Business Support System |
| CN | Core Network |
| CSMF | Communication Service Management Function |
| DN | Data Network |
| DNS | Domain Name System |
| eNB | evolved Node B |
| ETSI | European Telecommunications Standards Institute |
| ISG | Industry Specification Group |
| MANO | Management and Orchestration |
| MEAO | MEC Application Orchestrator |
| MEC | Multi-access Edge Computing |
| MEO | MEC Orchestrator |
| MEP | MEC Platform |
| MEPM | MEP Manager |
| NEF | Network capability Exposure Function |
| NFV | Network Functions Virtualization |
| NFVI | NFV Infrastructure |
| NFVO | NFV Orchestrator |
| NS | Network Slicing |
| NSD | Network Service Descriptor (NSD) |
| NSI | Network Slice Instance |
| NSMF | Network Slice Management Function |
| NSSI | Network Slice Subnet Instance |
| NSSMF | Network Slice Subnet Management Function |
| NST | Network Slice Template |
| OAI | OpenAirInterface |
| OSS | Operations Support System |
| PCF | Policy Control Function |
| PKI | Public Key Infrastructure |
| PNF | Physical Network Function |
| RAN | Radio Access Network |
| RAM | Random Access Memory |
| RNIS | Radio Network Information Service |
| SA | System Aspects |
| SDN | Software Defined Networking |
| S-NSSAI | Single-Network Slice Selection Assistance Information |
| UDM | User Data Management |
| UE | User Equipment |
| URLLC | Ultra Reliable and Low Latency Communication |
| UPF | User Plane Function |
| VIM | Virtual Infrastructure Manager |
| VNF | Virtual Network Function |
| VNFD | VNF Descriptor |

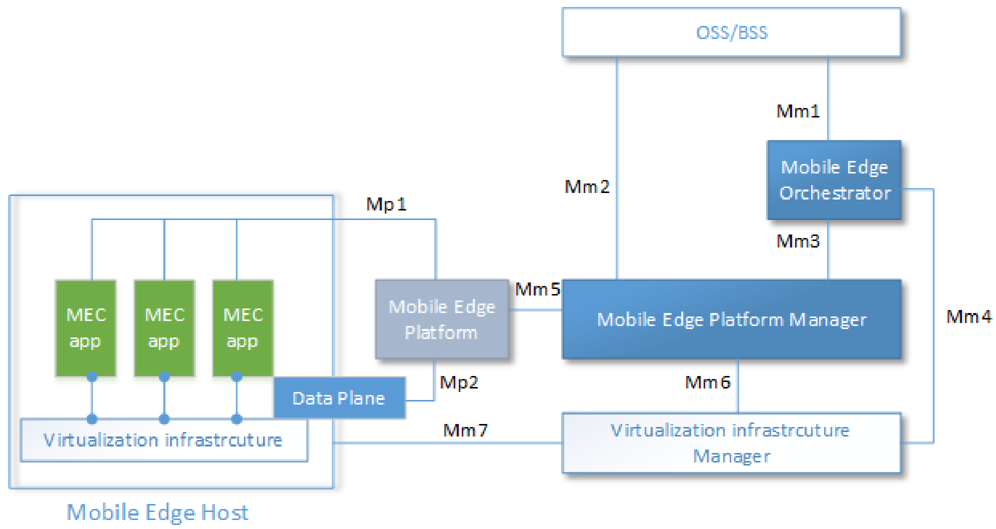


Fig. 1. High-level representation of the MEC architecture (based on [5]).

## II. Related work

### A. ETSI MEC

The ETSI MEC ISG has been working on the development of standardization activities around MEC since 2013. Its first released document covers the reference architecture [5]. A high-level representation of this architecture is shown in Fig. 1. It introduces three main entities:

- The MEC host, which provides the virtualization environment to run MEC applications, while interacting with mobile network entities via the MEC platform (MEP) to provide MEC services and data offload to MEC applications. Two MEC hosts can communicate via the Mp3 interface aiming at managing user mobility via the migration of MEC applications among MEC hosts.
- The MEC platform (MEP), which acts as an interface between the mobile network and the MEC applications. It has an interface (Mp1) for MEC applications to expose and consume MEC services, and another (Mp2) to interact with the mobile network. The latter is used to obtain statistics from the RAN on UEs and eNBs, e.g., in order to provide the Radio Network Information Service (RNIS) and the Location Service, and to appropriately steer user-plane traffic to MEC applications.
- MEC applications that run on top of a virtualized platform.

Another concept introduced by ETSI MEC is the MEC service, which is either a service provided natively by the MEP, such as the RNIS and traffic rules control, or a service provided by a MEC application, e.g., video transcoding. MEC services provided by third-party MEC applications should be registered with the MEP and made available over the Mp1 reference point. Once registered, a service may be

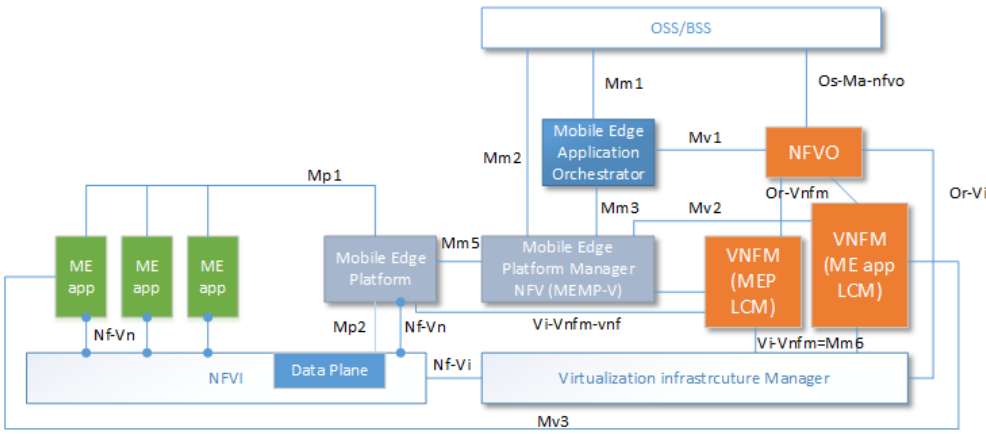


Fig. 2. Updated version of the MEC architecture featuring MEC in NFV (based on [4] and [5]).

discovered and consumed by other MEC applications. Regarding the management plane, ETSI MEC introduced the Mobile Edge Orchestrator (MEO), which is in charge of the life-cycle of MEC applications (instantiation, orchestration and management), and acts as the interface between the MEC host and the Operations/Business Support System (OSS/BSS).

Considering the advantages brought by NFV, and aiming to integrate and run all MEC entities in a common NFV environment, the ETSI MEC 017 working group drafted a document [4] to update the reference architecture as shown in Fig. 2. These updates have been included as an NFV-oriented variant in the most recent version of the MEC framework and reference architecture specification [5]. In this variant, the MEP and MEPM are run as VNFs. The MEO is renamed to MEAO (Mobile Edge Application Orchestrator), maintaining the same functionality, but using the NFVO to instantiate MEC applications as well as the MEP and MEPM. Consequently, all the processes of instantiation and management of resources follow the well-defined NFV interfaces. By doing so, edge resources can be seen as classical computation and storage ones, and can be managed by the same Virtual Infrastructure Manager (VIM) software.

### *B. Network slicing support in 5G networks*

Release 15 of the 3GPP standard includes NS specifications for 5G. Remarkably, the CN has been decomposed into fine-granular Network Functions (NFs), moving from a monolithic core network into a more modular one. Fig. 3 illustrates the service-based 5G reference architecture. The most prominent NFs are Access and Mobility Management Function (AMF), Session Management Function (SMF), User Plane Function (UPF), User Data Management (UDM), Network Slice Selection Function (NSSF), Network

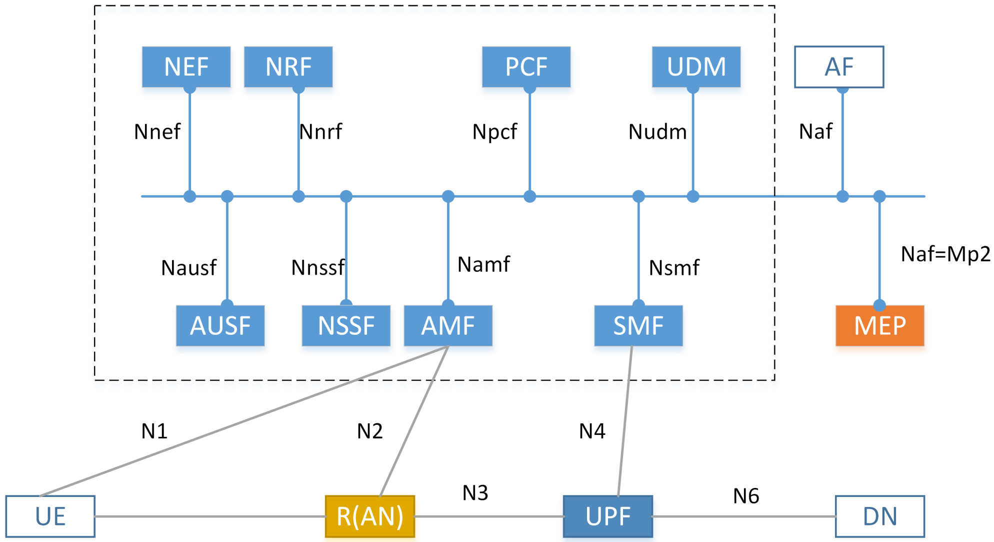


Fig. 3. 5G Core Network service-oriented architecture.

capability Exposure Function (NEF), Policy Control Function (PCF), and Application Function (AF). All the NFs expose APIs to provide one or more services to other NFs, following the producer-consumer concept.

In this article, we focus on user-plane functions (SMF, PCF and UPF), as MEC requires the definition of traffic policies to redirect traffic to the appropriate MEC applications. More details on the other 5G functions can be found in [2]. The UPF is the function in charge of routing user plane traffic to the appropriate Data Network (DN). It gets its configuration from the SMF, which is one of the key elements for user-plane traffic management. Among the various functions of the SMF, such as IP address allocation and management, and session management, is the control of the UPF by configuring traffic rules. The SMF exposes service operations to allow another function or 5G AF to use policy and traffic rules to reconfigure the UPF, via (i) the PCF, if the 5G AF is a trusted application, or (ii) the NEF, for untrusted AFs.

In the 5G architecture, the MEP will be integrated as a *5G AF* [6], trusted or not, depending on the use-case. It may request traffic redirection for a MEC application as per the request of the MEAO via the MEPM. Therefore, if MEP is a trusted 5G AF, it can use directly the PCF to generate a policy to offload traffic towards the MEC application. If it is not considered as a trusted 5G AF, it uses the NEF to access the SMF, via its traffic filter policy API, and requests the traffic redirection.

The 3GPP, via the SA5 group, has also defined a framework for the orchestration and management of the Network Slice life-cycle. The 3GPP approach is based on two basic concepts: Network Slice Instance

(NSI) and Network Slice Subnet Instance (NSSI). The NSI, at the fundamental level, is composed of NFs (both AN and CN ones), realized with corresponding physical and logical resources, and its composition is described by a NS Template (NST) that can be individually enriched with some instance-specific information (parameters, policies).

The 3GPP approach defines the following management functions related to NSSI, listed below in the order corresponding to their hierarchy:

- Communication Service Management Function (CSMF). The CSMF manages Communication Services provided by the Network Operator according to the requirements of the Communication Service Customer, converts these requirements to NS requirements (e.g., network type/capacity, QoS requirements, etc.), and delegates the management of NSIs to NSMFs.
- Network Slice Management Function (NSMF). The NSMF manages NSIs, according to the requirements from the CSMF, and further converts/splits them to NSS requirements and delegates management of NSSIs to NSSMFs.
- Network Slice Subnet Management Function (NSSMF). NSSMF manages NSSIs based on the requirements received from the NSMF.

It should be noted that the NST describing a Network Slice is composed when the vertical (i.e., slice owner) requests the creation and deployment of a NSI.

## III. Network Slicing including MEC

Stemming from the facts that (i) 3GPP has released a new architecture model to integrate NS in 5G, and a new framework to manage NS, and (ii) the ETSI MEC group has proposed a solution to integrate MEC in NFV, there is a need to update the current MEC architecture to comply with these developments, aiming at supporting NS at the MEC level. We distinguish two models for the support of Network Slicing in MEC. The first model assumes that the MEP is already deployed at the edge NFVI and is shared among the slices; we term it the *multi-tenancy* model. In the second model, the MEP is deployed inside the slice. This is what we call *in-slice* deployment. For both models, we assume that the MEP is deployed as a VNF. The MEP and MEC applications are described using a VNF Descriptor (VNFD) and Application Descriptors (AppDs), respectively. VNFDs and AppDs describe the necessary information required by the NFV Orchestrator (NFVO) and VIM to deploy instances of virtual applications, either at centralized clouds or the edge. AppD is specific to MEC applications, containing special fields such as traffic steering rules and MEC services required by the application. Note that we consider the MEPM as the Element Manager (EM) of the MEP. Fig. 4 shows the global picture highlighting the envisioned network slicing orchestration/management architecture as proposed by 3GPP, and featuring MEC slicing.

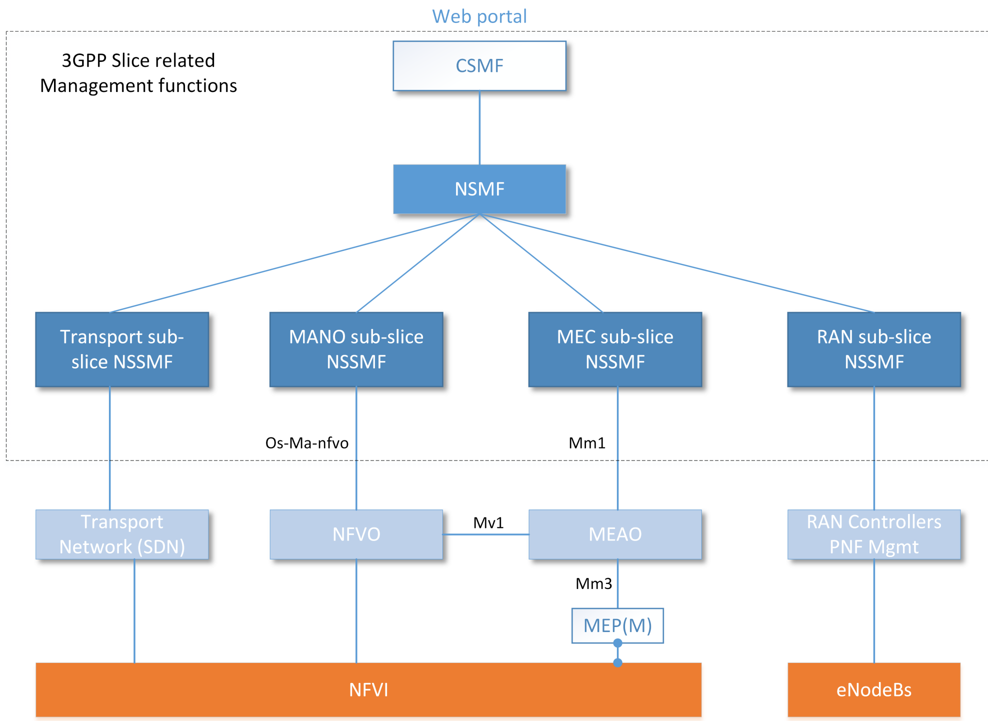


Fig. 4. The proposed network slicing orchestration/management architecture, including MEC, in a 5G environment.

In terms of interfaces, we mainly highlight those needed to orchestrate and manage core and edge virtual applications. The RAN controller is the element that provides a northbound control interface to manage eNBs, while using a southbound protocol, such as FlexRAN [7], in order to remotely configure eNBs (e.g., to associate to a new AMF of a slice) or to obtain RAN-level information, such as UE statistics, which can be used by the operator or exposed to interested applications over the RNIS MEC API.

We assume that a vertical first accesses a front end (such as a web portal) to request the creation of a network slice, using the NST made available by the CSMF. The NST can be extended according to the vertical needs, and by integrating network functions displayed by the CSMF through its network functions store or catalogue (i.e., add more MEC applications). The CSMF forwards the NST to request the creation of an end-to-end network slice composed by several sub-slices that span the RAN, CN, MEC and transport network. The NSMF organizes the NST into sections corresponding to each sub-slice. The Management and Orchestration (MANO) NSSMF component covers the CN functions and VNFs that need to be deployed over the cloud. All the network functions that need to be deployed over MEC should be managed by the MEC NSSMF. The NSSMF accepts as input a Network Service Descriptor (NSD) [8] that contains VNFDs as well as AppDs. The NSMF requests the creation of each sub-slice to the corresponding NSSMF, as illustrated in Fig. 4. The RAN NSSMF is in charge of updating the configuration of the RAN, via a RAN controller that interacts with the involved eNBs (PNFs) indicated in the NST. The NSSMF in charge of CN and VNF instantiation, requests the instantiation of the NSD to the NFVO using the Os-Ma-NFVO interface [9]. The MEC NSSMF interacts with the MEAO by providing the AppDs of the applications that need to be deployed at the edge NFVI. The MEAO will use the same NFVO (as specified in [4]) to request the creation of the AppD instance at the selected edge NFVI. Among the available edge NFVIs, the MEAO can pick the appropriate by executing its internal placement algorithm, considering different criteria such as latency and service availability [10]. Once the application is instantiated, the MEAO is informed of the MEC application's IP address, which it communicates to the MEC platform along with parameters such as specific traffic filters to enforce traffic steering. The last sub-slice is about the transport part, where we assume that the NSSMF managing it interacts with Software Defined Networking (SDN) controllers to isolate and forward NS traffic to the Internet.

Once each sub-slice is created, the NSMF is in charge of stitching them together to build the end-to-end slice. Stitching consists in interconnecting the different sub-slices using a sub-slice border API, as described in [11].

### A. *Multi-tenancy model*

In the case of MEP multi-tenancy, the MEP and UPF are already deployed. The MEP is aware of the IP addresses and interface endpoints of the NEF or PCF for traffic redirection, as well as those of the RAN controller, from which it can gather the necessary RAN-level data to provide MEC services such as the RNIS. Once the MEC application is deployed by the NFVO, the latter informs the MEAO about the successful instantiation of the MEC application, along with its IP address. The MEAO then, via Mm3, requests the MEP to enforce traffic redirection rules as indicated in the AppD. Based on the description presented in Section II-B, the MEP, via the PCF's API, requests the redirection of specific traffic (via a traffic policy) toward the newly created MEC application. Here, the MEP uses the PCF, as it is considered a *trusted* 5G AF: the MEP has been deployed by the network operator as a common 5G AF for all slices.

### B. *In-slice deployment model*

In this case, the MEP has to be deployed along with the MEC application at the edge NFVI. Unlike the multi-tenancy model, here the MEAO requests the instantiation of both the MEP and MEC application at the same time. The NFVO deploys both, and ensures that there is a virtual link between them. As in the previous case, the NFVO acknowledges the creation of the MEP and MEC application instances and indicates their IP addresses.

We differentiate between two cases: (i) all the CN elements (including the UPF) are deployed inside the slice; (ii) the UPF is already deployed. In the first situation, the UPF is deployed also at the edge (for the sake of performance), and the MEP can implement traffic redirection using the internal PCF of the network slice. For the second scenario, the MEP has to discover the NEF of the operator, as the MEP is not considered as a trusted 5G AF. To solve this, we propose that the DNS service running at the edge NFVI is used: Once instantiated, the MEP sends a DNS request to discover the NEF's IP address, and communicates with the latter to apply traffic redirection rules.

Regarding the needed access to the eNBs in order to provide MEC services (e.g., RNIS, Location Service), we propose to use the concept of *zones*, as introduced in [12]. A zone indicates an area covered by a group of eNBs associated with a MEC host. These eNBs are assumed to be managed by a single RAN controller. For both scenarios, the MEP can use DNS to discover the RAN controller that corresponds to the zone where it is instantiated, which in turn allows the MEP to retrieve RAN-level information from all eNBs of the zone.

## IV. Security and Isolation

One of the major requirements in Network Slicing is traffic isolation and security enforcement. Each NS should share the common infrastructure in a secure way. It shall not "see" the traffic of other slices, nor have access to information on other running slices. Two challenges thus arise with respect to MEC slicing: (i) The traffic redirection mechanism should ensure that a NS (i.e., the MEC application instances it includes) cannot specify a traffic redirection policy for traffic it does not "own"; (ii) a Network Slice should not be able to use MEC services (e.g., native ones, but also MEC services exposed by other slices) in a way that it gets unauthorized access to information on other running Network Slices, or consume MEC services not available for it. In the following, we propose solutions to overcome the aforementioned issues.

### *A. Traffic redirection*

Each MEC application is described using an AppD. The AppD integrates information specific to a MEC application, among which the `appTrafficRule` field. This specifies the characteristics of the traffic requested by the application. It includes a traffic filter, which allows to specify the traffic to offload, such as flows identified by combinations of IP source address, IP destination address, source port, destination port, protocol, and others. Also, the MEC application provider is allowed to add a list of `appDNSRule` elements, which, combined with `appTrafficRule` ones, allow traffic offloading using DNS domains. Therefore, a potential issue is that a slice owner may encode in the AppD DNS rules for domains it does not own, or use a traffic filter that corresponds to a traffic flow of another running network slice. Such situations may introduce significant security threats. A malicious application instance can (i) intercept traffic flows it is not supposed to have access to, causing confidentiality breaches, and (ii) perform "black hole" or other denial of service attacks by diverting and dropping UE traffic destined for victim MEC instances. Therefore, we argue that the MEC NSSMF is augmented with security and access control functionality so that it can check that each MEC application has the necessary permissions to request traffic redirection as indicated in its AppD. To mitigate these threats and offer sufficient protection to MEC slice instances, Public Key Infrastructure (PKI) technologies can be used. In particular, we propose the use of a trusted third party, which may coincide with the network operator and which can guarantee that the slice owner has the appropriate permissions to indicate specific DNS entries and traffic filters in the AppD. This necessitates extending the AppD with a specific field where a signature of the trusted third party over the set of `appDNSRule` and `appTrafficRule` entries will be placed.

### *B. MEC services: RNIS and Location Service*

Another important issue for MEC slicing is related to MEC services that expose privacy-sensitive information about the UEs of a slice, such as their (coarse) location or channel quality. Depending on the considered use case, access to this type of information should be restricted only to the slice's MEC applications. To this end, we propose two solutions, which depend on the considered deployment scenario (multi-tenancy vs. in-slice).

In the case of multi-tenancy, the MEP should check the identifier of the MEC application, and whether the latter can have access to the specific MEC service. Furthermore, it should check which are the users that the MEC application can request information about. Thus, we propose that along with any MEC service request, the S-NSSAI identifier of the slice where the requesting MEC application belongs is included. The RNIS and location APIs should be modified to integrate the S-NSSAI of the UE in addition to the UE identifier. Thus, the MEC application can access only the UE information that belongs to its slice. In this first use case, the proposed solutions improve the MEP, by allowing it to obtain more information on the network slices along with their associated users and authorizations. The MEP will be S-NSSAI-aware, in order to know to which network slice an application or set of UEs belong to. We assume that the MEP includes an authorization database, which allows it to map the MEC services that a network slice has the right to access.

Regarding the second scenario, i.e., in-slice deployment, the solution is slightly different. In this case, the MEP is not implementing the access control mechanism, as this belongs to the slice itself. We propose in this case to rely on the RAN controller to filter the information to provide to the MEP. That is, when the MEP discovers the RAN controller in charge of the zone, it includes its S-NSSAI along with the request. The RAN controller can be considered as a 5G AF, which can access the NSSF via the NEF to check which are the users that belong to the S-NSSAI, and filter accordingly the information provided to the MEP.

## V. Service registration and discovery

A MEC application can expose a service to other MEC applications. For example, a video transcoding function can be provided by a third-party MEC application to other MEC application instances as a MEC service. To ensure this, the MEP provides an API to a MEC application to register itself as a MEC service, which the MEP can then expose to other MEC applications. This takes place over the Mp1 reference point. However, in case of a sliced MEC, we can identify the following issues: (i) For a MEP deployed in-slice, MEC applications can provide services only to the other MEC applications inside the same slice. Thus, a vertical (tenant) cannot directly expose a service to another vertical. (ii) In the multi-tenant MEP case, the problem stems from the fact that the new MEC service should be advertised to the MEAO, as well as to the CSMF, which need to include it in their available-function catalogues.

For the first case, if a vertical wants to provide a MEC service, it should indicate it to the CSMF, which updates accordingly its catalogue by advertising the availability of the service to other verticals. When a vertical requests a MEC service provided by another MEC application, it should indicate it via the NST. The MEAO should then keep track of the location of the MEC application providing the service and place the new MEC application in the same edge NFVI, ensuring that there is a link between the two applications. The creation of this (virtual) link should be requested by the MEAO to the NFVO. Regarding the second case, our approach is that the MEP, upon the registration of a new MEC service, provides this information to the MEAO via the Mm3 reference point. The latter updates a registry database, which indicates the MEP hosting a MEC service provided by another MEC application. The MEAO informs the MEC NSSMF about the new MEC service. The information is forwarded towards the CSMF, which updates its function catalogue. The MEAO places the MEC application at the edge host where a MEP is providing the MEC service. In this case, the discovery process is done at the CSMF level, while the registration process is kept at the MEP level, whereas in the first scenario (i.e., MEP inside the NS) both registration and discovery happen at the CSMF level.

## VI. Implementation experiences

We have implemented a ME(A)O with an Mm1 interface that fully complies with ETSI MEC 010-2 [13], and a fully-fledged MEP, featuring traffic rule management, DNS, RNIS and location services with standard interfaces, as well as service registration and discovery APIs. Our system is tailored to OpenAirInterface (OAI)[2]: We have extended the OAI core network with the appropriate functionality for control-user plane separation [14] and for communicating with our MEP over the Mp2 reference point (REST API, in our case). At the same time, we are using the OAI RAN, after extending it for RAN slicing support, and with some additional features for the FlexRAN-based Mp2 interface. At the MEC host level, we have also implemented a lightweight VIM appropriate for resource-constrained edge deployments, building on lxd[3] and allowing the execution of MEC application instances as containers. Our edge VIM is written in python, interacts with lxd using the latter's REST API, and provides interfaces to the MEO for onboarding and deleting MEC application container images, as well as instantiating, querying, and terminating MEC application containers, also managing their network configuration. Notably, it has been tested with edge compute nodes of different footprints, from Raspberry PIs to powerful workstations. Fig. 5 presents our implementation and testbed.

While the related standard interfaces are evolving, in our MEAO API implementation we take advantage of placeholder fields in the information model specified in ETSI MEC 010-2 to support slicing and to give hints on how to handle special types of MEC application instances (e.g., a virtualized MEP and MEPM). In particular, during application package onboarding, we exploit the `userDefinedData` field of the standard package onboarding request message to signal that the package is a virtualized MEP component (MEP-V). Then, at application instantiation time, we use the `selectedMEHostInfo` element to add slice identification information (S-NSSAI); the MEAO can then use the slice identifier to select the appropriate virtualized MEP(M) instance and communicate with it in order to, e.g., configure traffic steering rules for the MEC application or discover the API endpoints for other services provided by this MEP instance, and configure the MEC application appropriately, so that the latter is able to consume them or expose its own.

In order to experiment with the two MEP deployment scenarios, we prepared our MEP/MEPM software for deployment on our edge VIM. As an example of a MEC application, we used a robot control tool that we have implemented for demonstration purposes. The size of the lxd application images was 514 and 285 MB, respectively. During package onboarding, the MEAO downloads the image from the

[2]https://www.openairinterface.org

[3]https://linuxcontainers.org/lxd/

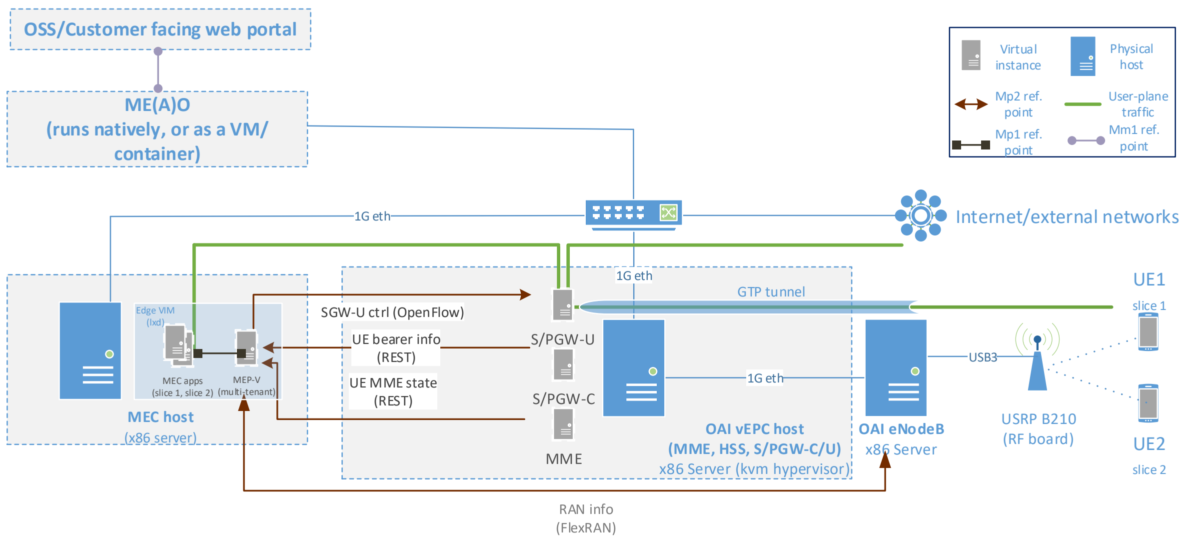


Fig. 5. MEC system implementation and testbed. In this figure, the MEP is deployed in its multi-tenant version and two MEC application instances, corresponding to the two different deployed slices, are sharing it. Traffic from two UEs (each belonging to a different slice) connected to a slicing-capable OAI eNB is appropriately redirected to the MEC applications after the MEP has installed traffic rules at the Serving GW-U (equivalent to UPF in 4G), which in our case is a modified version of OpenVSwitch.

URL indicated in the AppD, which in our case pointed to a locally hosted HTTP server. The in-slice deployment scenario involves the onboarding and instantiation of one distinct (virtualized) MEP per application instance. In this case, the MEAO first deploys the MEP, and, as soon as it is up and running, it spins up the MEC application. In order to set up the necessary DNS and traffic rules, it discovers the appropriate MEP instance by the slice ID included with the instantiation request. On the contrary, in a multi-tenant MEP scenario, the default MEP instance is used. As a performance metric, we use the time it takes for a MEC slice to be operational from the moment this is requested to the MEAO over the Mm1 reference point, breaking it down to onboarding and instantiation time. For regular MEC applications, instantiation includes, among others, (i) the time it takes to launch a new instance on the edge VIM, (ii) the time to communicate with the MEPM to set up traffic offloading and DNS rules, and (iii) the time it takes for the MEP to apply these rules to the data plane over the Mp2 interface (in our implementation, to apply a set of rules remotely to the OpenVSwitch that is controlling user plane traffic).

In order to demonstrate the feasibility of our design and implementation and stress its lightweight characteristics, we chose to run our experiments on a low-end compute environment: Our MEC host is an AMD FX-7500 Radeon R7, with 4 CPU cores at 2.1GHz (maximum CPU frequency) and 8GB RAM, running Linux kernel 4.4.0-97. Table II summarizes the results of our tests.

TABLE II

MULTI-TENANT VS. IN-SLICE MEP DEPLOYMENT: COMPARISON OF THE TIME TO DEPLOY A MEC APPLICATION. THE REPORTED VALUES ARE IN SECONDS.

| Scenario | MEP onboarding | App. onboarding | MEP instantiation | App. instantiation | Total |
|---|---|---|---|---|---|
| In-slice | 215.4 | 118.19 | 46.99 | 58.5 | 439.09 |
| Multi-tenant | - | 116.4 | - | 56.6 | 173 |

Despite its qualitative advantages in terms of isolation, in-slice MEP deployment comes with the inherent cost of instantiating one MEP per slice. This can be critical in some deployments, as edge compute resources are typically scarce. Performance-wise, this significantly impacts slice deployment time, especially if the MEP is not already onboarded, since this is the operation that in our implementation dominates all overheads (it involves downloading a raw application image and computing its fingerprint).

## VII. CONCLUSION

This article introduced the latest developments on MEC and Network Slicing in 5G, addressing architectural, but also security and isolation-related challenges to be faced towards *sliced* MEC, where MEC resources are integrated in a 5G Network Slice. We proposed a novel design compliant with both ETSI and 3GPP specifications that enables the integration of MEC as a sub-slice, and presented solutions to the critical security and isolation issues raised. Finally, we presented preliminary experimental results from our MEC implementation to compare different deployment scenarios for MEC slicing.

## ACKNOWLEDGEMENT

This work was partially supported by the European Union's Horizon 2020 Research and Innovation Program under the 5G!Drones (Grant No. 857031) and MonB5G (Grant No. 871780)projects.

## REFERENCES

[1] *Study on management and orchestration of network slicing for next generation network*, 3GPP Technical Report TR 28.801, V15.0.1, Jan. 2018, Release 15.

[2] *System Architecture for the 5G System*, 3GPP Technical Specification TS 25.501, Mar. 2019, Release 15.

[3] *Multi-access Edge Computing (MEC); MEC support for network slicing*, ETSI Group Report MEC 024, V2.1.1, Nov. 2019.

[4] *Mobile Edge Computing (MEC); Deployment of Mobile Edge Computing in an NVF environment*, ETSI Group Report MEC 017, V1.1.1, Feb. 2018.

[5] *Mobile Edge Computing (MEC); Framework and Reference Architecture*, ETSI Group Specification MEC 003, V2.1.1, Jan. 2019.

[6] S. Kekki *et al.*, "MEC in 5G Networks," ETSI, White Paper 28, Jun. 2018.

[7] X. Foukas, N. Nikaein, M. M. Kassem, M. K. Marina, and K. P. Kontovasilis, "Flexran: A flexible and programmable platform for software-defined radio access networks," in *Proc. ACM CoNEXT*, 2016.

[8] *Network Functions Virtualisation (NFV); Management and Orchestration*, ETSI Group Specification NFV-MAN 001, Dec. 2014.

[9] *Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; Os-Ma-Nfvo reference point - Interface and Information Model Specification*, ETSI Group Specification NFV-IFA 013, Aug. 2018.

[10] L. Yala, P. A. Frangoudis, and A. Ksentini, "Latency and availability driven VNF placement in a MEC-NFV environment," in *Proc. IEEE Globecom*, 2018.

[11] S. Kuklinski *et al.*, "A reference architecture for network slicing," in *Proc. IEEE NetSoft*, 2018.

[12] *Mobile Edge Computing (MEC); Location API*, ETSI Group Specification MEC 013, V1.1.1, Jul. 2017.

[13] *Mobile Edge Computing (MEC); Mobile Edge Management; Part 2: Application lifecycle, rules and requirements management*, ETSI Group Specification MEC 010-2, V1.1.1, Jul. 2017.

[14] P. Schmitt, B. Landais, and F. Y. Yang, "Control and User Plane Separation of EPC nodes (CUPS)," 3GPP, Tech. Rep., Jul. 2018. [Online]. Available: http://www.3gpp.org/cups