Analyzing the impact of VIM systems over the MEC management and orchestration in vehicular communications

Nina Slamnik-Kriještorac*, Michael Peeters†, Steven Latré*, and Johann M. Marquez-Barja*

* University of Antwerp - imec, IDLab research group, Sint-Pietersvliet, 2000 Antwerp, Belgium.
E-mail: {name.surname}@uantwerpen.be
† imec, Antwerp, Belgium.
E-mail: michael.peeters@imec.be

***Abstract*—The combination of 5G and Multi-access Edge Computing (MEC) technologies can bring significant benefits to vehicular networks, providing means for achieving enhanced Quality of Service (QoS), and Quality of Experience (QoE) of wide variety of vehicular applications. Although beneficial in terms of latency reduction, the edge of the architecture for communication networks produces enormous heterogeneity of network services and resources. This challenge becomes even more severe when different administration domains are taken into consideration. Thus, efficient network Management and Orchestration (MANO) of network resources and services are inevitable. As ETSI provided guidelines and standardization for NFV MANO components, the MEC platform can be used to host network services, while MANO systems are in charge of network service management and orchestration. In this paper, we focus on the specific impact that the Virtualized Infrastructure Manager (VIM) has on the performance of the whole MANO system, used for management and orchestration of MEC services and resources in vehicular networks by enabling the on-demand service instantiation, and termination teardown. In our testbed-based evaluation, we measured the network service instantiation and termination delays when evaluating: a) OpenStack and Amazon Web Services (AWS) as VIMs for Open Source MANO (OSM), and b) OpenStack and Docker in case of Open Baton. Such performance analysis with a strong experimental component can serve as a baseline for researchers and industry towards exploiting the opportunities that existing MANO solutions provide.**

***Index Terms*—MEC, NFV, VNF, MANO, edge and cloud computing, orchestration, vehicular networks**

I. Introduction and Motivation

Nowadays, network operators, automotive industry, and service providers work closely together in order to provide a fruitful variety of vehicular services to their users, promising high levels of Quality of Service (QoS), which reflects on the Quality of Experience (QoE). To cope with the heterogeneous nature of network resources, technologies, vendors, as well as high dynamicity in network traffic, the efforts are focused on exploiting the synergy between technologies such as Software Defined Networking (SDN), Network Function Virtualization (NFV), and 5th generation of mobile communications (5G). Such collaboration is essential to address latency requirements in various vehicular applications, e.g, emergency electronic brake warning, lane change warning, traffic information systems, video streaming etc. [1–5].

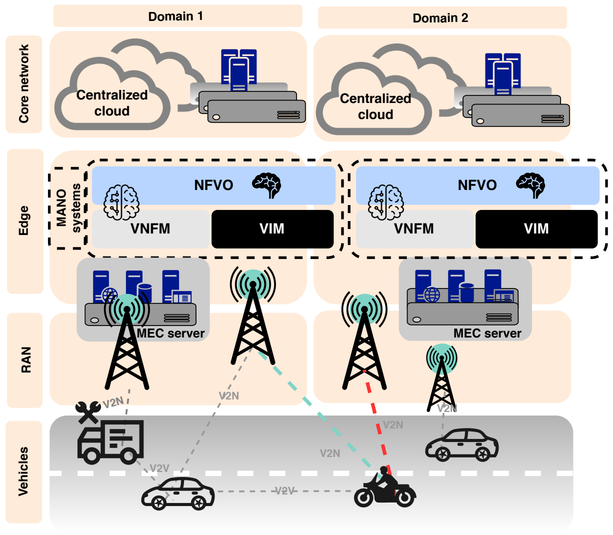

Fig. 1: Management and Orchestration of MEC resources and services in vehicular communications: 5G and MEC-enhanced vehicular network

One of the strategies to cope with latency constraints is shifting cloud computing capabilities to the network edge (Fig. 1), thereby placing vehicular applications closer to the end-users. Hence, Multi-access Edge Computing (MEC), supported by SDN, NFV, and 5G, is recognized in research community [1–6] for significant reductions of latency for vehicular services. However, MEC platforms are usually resource constrained, and distinctive for wide resource and service heterogeneity. To address these challenges, efficient *network management and orchestration* that provide a fast reaction to network changes are a must. In Fig. 1, the position of NFV Management and Orchestration (MANO) systems in 5G-enhanced vehicular networks illustrates their main goal, which is to provide means for orchestration of resources and services that are placed on top of the MEC platforms, in one or multiple technological and administrative domains. Thus, in this paper we study the performance of existing solutions for MANO in MEC, inspecting the impact of the Virtualized Infrastructure Manager (VIM), which is the management system for NFV Infrastructure (NFVI) that is used for instantiation, operation, and termination of network services. As ETSI NFV MANO components, shown on the right side of the Fig. 2, manage and orchestrate life-cycle of a particular network service, in this paper we focus on VIM systems (black rectangle in Fig. 2), and their impact on the performance of the whole MANO system. Such an approach is significant for vehicular communications since one of the first steps towards addressing management and orchestration in realistic scenarios within vehicular context is to assess the opportunities that existing MANO platforms bring.

Our experimental setup includes two MANO platforms under development, i.e., Open Source MANO (OSM) and Open Baton, and different VIM environments that each of these MANO platforms supports (Table I). Both OSM and Open Baton are MANO solutions suitable for deployment at the network edge, as a result of their full compliance to the ETSI standardized framework [7], and their lightweight installation [8]. To evaluate the impact of VIM on the individual performance of both tools, we conduct two separate experiments (one per each MANO tool). These two tools do not support the same set of VIM drivers, and a fair environment for experimentation must be ensured.

The performance is measured in terms of the time needed for a service to be instantiated on top of the MEC platforms, i.e., Overall Instantiation Delay (OID), and time needed to release resources when terminating the service, i.e., Overall Termination Delay (OTD). Due to the programmable nature of vehicular applications nowadays, consisting of multiple Virtual Network Functions (VNFs) that can be realized as Virtual Machines (VMs) or containers [9], linked into Service Function Chains (SFCs), MANO systems can proactively make decisions to instantiate additional VNFs, or the whole VNF chains (i.e., application instance), in order to meet QoS and QoE requirements. Therefore, if the MANO system decides that an immediate application instantiation will be more efficient than scaling up or down existing application instances in order to cope with network fluctuation, or increased service demand, benchmarking instantiation and termination delays plays a significant role. Therefore, the experiment consists of two separate parts, each proceeding with the evaluation of time taken for a network service to be instantiated or terminated, executed by MANO platforms. We focus on the performance of MANO systems (being bounded to their limitations in operation), excluding the impact of network, and relying on the benefits of application placement within MEC [10]. Thus, measuring the delay at the user equipment is out of scope of this paper.

The first experiment measures how fast OSM can instantiate and terminate a network service, when OpenStack, and Amazon Web Services (AWS) are VIM systems. Similarly, in the second experiment Open Baton is used as a MANO platform, because it allows us to use Docker as a VIM, and to evaluate the performance of MANO in case it instantiates services as containers (Docker VIM), and VMs (OpenStack VIM). In order to mimic the realistic features of edge computing, we utilized the *testbed environment* for the case of OpenStack VIM, and Docker VIM, while for AWS VIM we used a *public cloud*. The aforementioned testbed environment is the high-performance Virtual Wall[1] testbed, located in Gent, Belgium. Consisted of more than 400 bare metal and GPU servers, Virtual Wall is a large-scale testbed for advanced networking, distributed software, cloud, big data, and scalability research and testing.

Going beyond a theoretical evaluation of both approaches, we wish to provide both research and industry communities with a realistic performance evaluation of such tools, and to help them compare different platforms as well as impact of VIM, as an unavoidable manager entity for the underlying NFVI. Extending our research presented in [6], which reviews the existing orchestration solutions based on their compliance to ETSI, support for different VIMs and descriptor standards, as well as their multi-domain capabilities, in this paper we show the performance analysis for Open Baton, and OSM.

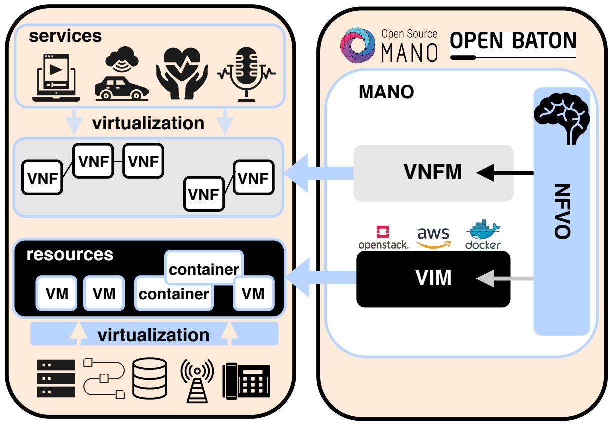

Fig. 2: ETSI NFV MANO components as a management and orchestration entity for a MEC platform.

	VIM environment	
Open Baton	OpenStack	Docker
Open Source MANO (OSM)	OpenStack	AWS

TABLE I: Supported VIM environments in Open Baton and OSM.

II. Related Work

In order to design and develop real-time network services capable to cope with stringent QoS and QoE requirements, containers are usually deployed as an alternative to VMs, as they usually demand low resource overhead, which makes them suitable for deployments on the resource-constrained network edge.

[1]Virtual Wall testbed: https://doc.ilabt.imec.be/ilabt/virtualwall/index.html

In their comparison between traditional VMs and containers, Doan *et. al.* [11] show that containers outperform VMs in terms of their suitability for MEC implementation, referring to specific service migration scenarios. However, the results provided by Salah *et. al.* [12] show that, although both deployed on top of the AWS Elastic Compute Cloud (AWS EC2), VM-based services outperform the container-based ones. This proves that impact of VIM environment is not negligible, and it has to be studied deeper. Therefore, we see the potential of inspecting the type of VIM when approaching MANO of network services in highly dynamic and resource-constrained platforms such as MEC.

Furthermore, there are some efforts to evaluate the overall performance of existing MANO systems, but without focusing on the impact of specific MANO element in the architecture (Fig. 2). For instance, the approaches presented in [8] provide a valuable but yet only theoretical overview of the orchestration solutions. One of the rare attempts to evaluate existing MANO platforms is introduced by Peuster *et. al.* [13], emulating the Points of Presence (PoPs) that need to be orchestrated. The authors used two different version of OSM, with no comparison of this tool to the other tools, and no discussion on how VIM influences the performance of OSM is provided.

Recently, there have been also some efforts to benchmark different VIM environments, based on the self-generated performance reports. In their approach to measure the performance and VM instantiation times of OpenStack and Nomad, Ventre *et. al.* present the performance measurements of both VIM solutions, focusing on the tunning of performance for each of the VIMs, and however, not focusing on the evaluating the impact each VIM has on the system, nor making the comparison between them. Moreover, Sechkova *et. al.* [14] focus on the VIM, measuring the time overhead that VMs provisioning brings to the system. For this evaluation, Sechkova *et. al.* [14] conducted a comparative analysis of two open-source VIMs, i.e., OpenStack and OpenVIM.

To the best of our knowledge, our approach is the first which presents the evaluation of different VIM environments used as a NFVI management system, thereby measuring the impact of VIM on the performance of particular MANO platform, within a real testbed environment.

III. Management and Orchestration Systems

The ETSI MEC architectural framework consists of management and orchestration elements, which are interconnected in order to enable hosting of applications and services on top of the MEC platform [7]. As the left side of Fig. 2 depicts, different MEC platforms include various virtualized and physical resources, plenty of services, and applications developed and maintained by different stakeholders. Such heterogeneity inevitably requires the standardization guidelines and recommendations, in order to increase the compatibility of such platform to various NFV environments.

The components of ETSI NFV MEC architecture which represent a so-called MANO, aiming to manage and orchestrate resources and services within MEC, are illustrated on the right side of Fig. 2. Thus, MANO incorporates NFV Orchestrator (NFVO), VNF Manager (VNFM), and VIM. In the specific context for heterogeneous MEC platforms, MANO is in charge of managing resources and all service instances which are running on top of these platforms. Based on the fluctuating nature of vehicular network traffic, and therefore high dynamicity in service requests, MANO takes care of allocating more resources for services, scaling VNFs, and releasing unnecessary resources by terminating ongoing services.

Following the orchestration decisions and instructions provided by NFVO, VNFM manages all network service instances (i.e., VNFs) running in MEC, while VIM represents the management system for NFVI that is used for instantiation and operation of network service. To be more specific, the roles of VIM are: a) performing allocation, management, and releasing of virtualized resources, b) preparing the underlying NFVI to run software images as a base for the required VNFs, and c) collecting fault reports and performance measurements about virtualized resources.

a) Open Baton: OpenBaton is a research MANO platform, developed by Fraunhofer FOKUS and Technical University of Berlin, which is aligned to the ETSI standardization. Its primary focus is on improving the performance and security of the overall infrastructure by merging the underlying infrastructures, software architectures, networking, management, and orchestration [8].

b) Open Source MANO: OSM is an ETSI-hosted MANO platform for automating end-to-end service deployment and orchestration [8]. It integrates several open source software initiatives to deliver ETSI NFV MANO functionalities: 1. Riftware is used as a network service orchestrator, 2. OpenMANO as resource orchestrator, and 3. Juju Server as VNFM [8]. The release five, which we used for experimentation, provides support to multiple VIM environments. Those tested in our experiment are shown in Table I.

IV. Experimentation setup

A. Virtualized Infrastructure Managers

Here we briefly provide an overview of OpenStack[2], AWS[3], and Docker[4], and particular settings that allow them to generate a corresponding VIM environment for MANO systems presented in the previous section. To run a service on top of OpenStack, the required image should be uploaded via Glance service. The name of the image is then used in VNF Descriptor (VNFD), and Network Service Descriptor (NSD), so when a request for instantiation comes from MANO to Openstack, image service retrieves the necessary image for VM instantiation. Furthermore, Nova and Neutron are services that provide compute and network resources based on the flavors and network specifications, that are also stated within VNFD and NSD. In order to register AWS as a VIM

[2]OpenStack documentation: https://www.openstack.org/

[3]AWS documentation: https://docs.aws.amazon.com/

[4]Docker documentation: https://docs.docker.com/

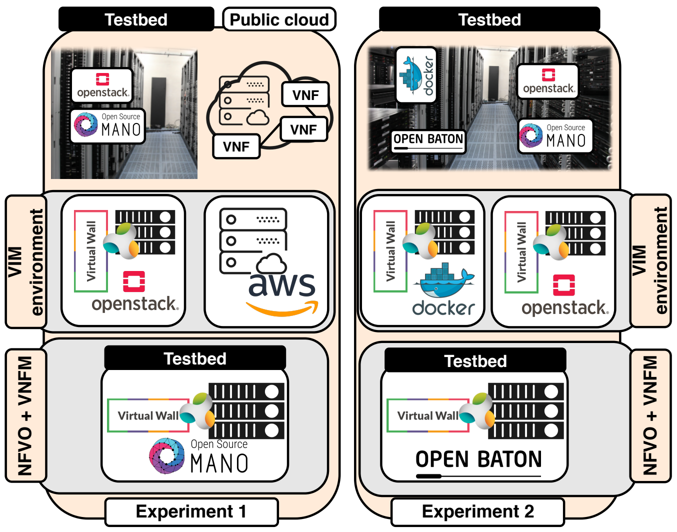

Fig. 3: Experimentation setup on the Virtual Wall testbed, and the public cloud.

for MANO, we needed *access* and *security* keys for our account, flavor of instances that will be instantiated, as well as a corresponding availability zone. Furthermore, to run an instance from MANO, it was necessary to specify a *key pair*, a security group, a subnet, and a location of the image needed for service instantiation. Finally, in our experimentation with Docker as a VIM, for the purpose of running specific network services, we created Docker images instead of creating VNFD, and then used these custom images to generate NSD, and to launch a network service. Therefore, network service is instantiated as a container on top of the Docker machine in a testbed environment.

B. Experimentation

For the purpose of inspecting the impact of VIM on the performance of MANO systems, we created an experimental setup that is illustrated in Fig. 3, including the Virtuall Wall testbed, and a public cloud. We made sure that machines selected for installation of Open Baton, OSM, OpenStack, and Docker, meet their resource requirements. Therefore, the capabilities of these machines are stated in Table II. The performance we measured is described as the time needed for a service to be instantiated, and terminated on top of the MEC platforms (i.e., OID, and OTD, respectively), as shown in Fig. 4, which also provides the definition of OID, and OTD. Due to the limitations imposed by release versions of MANO systems that we utilized for the experimentation, run-time operations such as on-demand service scaling are not supported, and yet not measured. The experimentation consists of two separate experiments (Fig. 3), both measuring the performance evaluation of network service instantiation/termination, as follows:

- Experiment 1: setup combining OSM for orchestration (MANO), and OpenStack and AWS for VIMs,
- Experiment 2: setup combining Open Baton for orchestration (MANO), and OpenStack and Docker for VIMs.

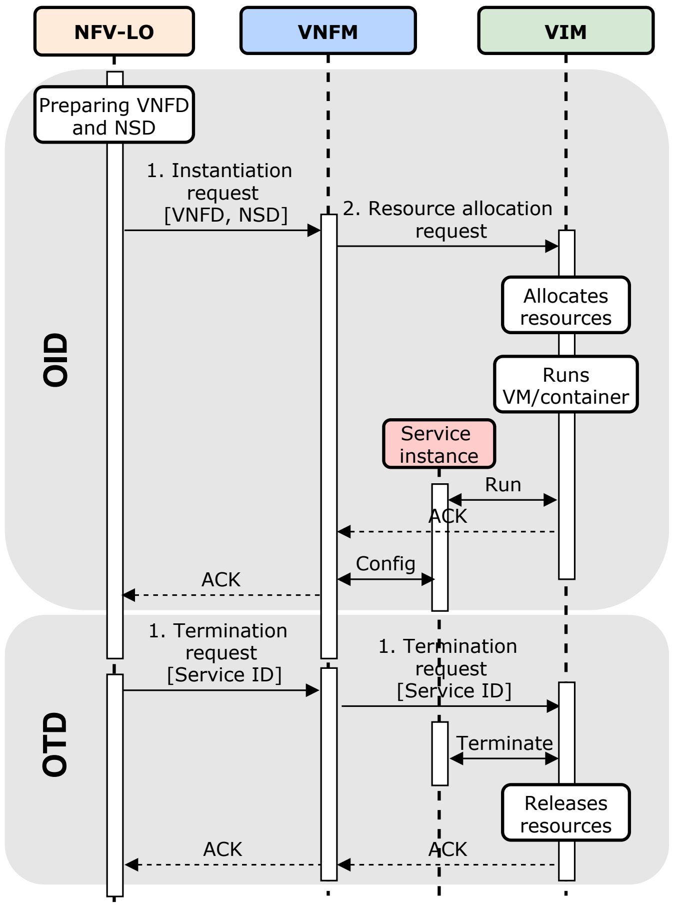

Fig. 4: Measuring OID and OTD.

For both experiments, we tested the performance for three chains of VNFs (SFCs), based on their complexity that is expressed as a number of VNFs contained in the chain. In Figures 5, 6, 7, and 8, SFCs are: 1) SFC_1 containing one VNF, 2) SFC_2 containing two VNFs, and 3) SFC_3 containing three VNFs. The testbed configuration of OpenStack mimics the realistic features of edge computing, while for AWS resources, we used the public cloud. Furthermore, in the Experiment 1, in order to create a fair environment for performance evaluation, we instantiated the same types of service (i.e., the same NSD) for both OpenStack and AWS. After instantiation, VMs with Ubuntu operating system (i.e., image uploaded to OpenStack, and present in us-east-1 zone in AWS EC2) were running on top of the OpenStack, and AWS cloud. For the Experiment 2, we measured the OID and OTD of Docker containers that are deployed using the testbed resources, and of VMs of the same functionality, instantiated on top of the OpenStack.

V. Results and Discussion

Here we provide a thorough discussion on results shown in Figures 5, 6, 7, and 8:

1) As shown in Fig. 5, AWS requires more time (s) to instantiate a service. This is reasonable since it is a public cloud, and all the internal procedures prior to

Component	Machine type	Capabilities		
		RAM (GB)	CPU	storage (GB)
OpenStack	pcgen4	48	2 x 8 core Intel E5 (2.6GHz)	250
OSM	pcgen5	16	1 x 4 core Intel E3 (3.1GHz)	250
Open Baton and Docker	pcgen5	16	1 x 4 core Intel E3 (3.1GHz)	250

TABLE II: Characteristics of machines running on top of the Virtual Wall testbed.

instantiation are hidden from the user. At the same time, OpenStack provides a dedicated platform (i.e., a private cloud) for user's needs, and it is located at a geographically suitable place for a MANO to orchestrate it.

2) Although OpenStack outperforms AWS in terms of OID (Fig. 5), there are configurations and custom installations that need to be done prior to using OpenStack as a VIM, and of course, custom machines are needed (Table II).
3) The more complex the service is, the higher OID and OTD are for all VIMs (Figures 5, 6, 7, and 8). This is somewhat expected, because each VNF, either it is a container or VM-based, takes time for instantiation and termination.
4) Interestingly, AWS needed less time to terminate more complex network services (Fig. 6), i.e., services with two and three VNFs. Thus, once instantiated and went through security procedure, the resources needed for network services can be released in a faster way.
5) Regarding configuration complexity, setting up AWS as a VIM for OSM is not well documented, since additional configurations have to be set-up on AWS as well (security groups, virtual private cloud, and subnets, have to be public in order to communicate with OSM). Such public configuration is not necessary in OpenStack, i.e., networks can be private.
6) Although more variety in flavors and images is present in AWS, there is a certain limitation in creating custom images and flavors based on the users' needs, while in OpenStack this task is straightforward.

For the edge network implementation in MEC, OSM performs better with OpenStack than AWS, due to the reasons presented above. However, the installation, configuration, and maintaining of OpenStack are unavoidable, and must be done by e.g., network administrators.

As it can be seen in Fig. 7 and 8, Docker outperforms OpenStack in terms of both OID and OTD. As containers are a lightweight solution comparing to VMs that we instantiated on top of the OpenStack, based on this result they prove to be more suitable for implementation on the resource-constrained network edge.

As it was already stated in the paper, in case MANO systems decide to instantiate additional application instances to meet QoS and QoE requirements, it is important to obtain the values of OID and OTD. Concerning values of these two metrics, expressed in the order of tens of seconds, we see that neither OSM Release 6 nor Open Baton Release 6 are ready to be used in a real deployment for vehicular networks, performing MANO of resources and services in MEC platforms. Potentially, in order to decrease the impact of such high delays on QoS, some predictions for service instantiation can be done in order to preempt the users' service requests. The results show that the impact of VIM is essential for the operation of MANO systems, since the same network services operating on top of NFVI managed by different VIMs take significantly more/less time to be instantiated/terminated.

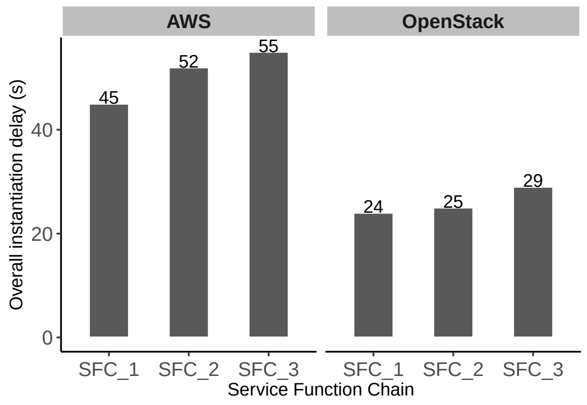

Fig. 5: Experiment 1: OID.

As a part of our future work, we plan to extend the experiment, creating more realistic network services which are suitable for hosting at the network edge of vehicular network. Although still unstable, the recently released version of OSM provides support to container instantiation by providing support to Kubernetes as a VIM. Since Kubernetes is a container orhcestration system that automates the deployment of microservice applications, OSM can be used as an orchestrator of life-cycle management operations for microservices deployed on top of the MEC platforms. Thus, interesting experimentation can be conducted in a setup that is similar to one presented in our paper. Also, once when AWS is supported and documented in Open Baton, we tend to compare different MANO tools from the perspective of different VIM environments.

VI. Conclusion

In this paper, we measure the impact of VIM environment on the performance of MANO systems used in MEC-based vehicular networks. To mimic the resource-constrained network edge, we utilized the high-performance Virtual Wall testbed, and a public cloud AWS, evaluating the OID and the OTD as indicators of the performance of OSM and Open Baton. Our results show the impact of OpenStack and AWS on the performance of OSM, as well as the superiority of container-based service deployment over VM-based in case of Open Baton. Although our results indicate that these MANO platforms have not reached a level of maturity for a deployment in real vehicular networks with such VIM environments,

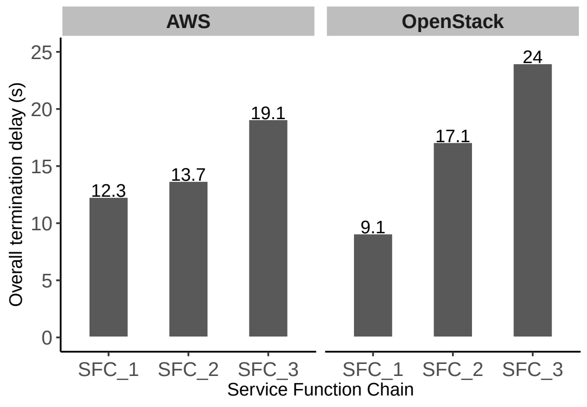

Fig. 6: Experiment 1: OTD.

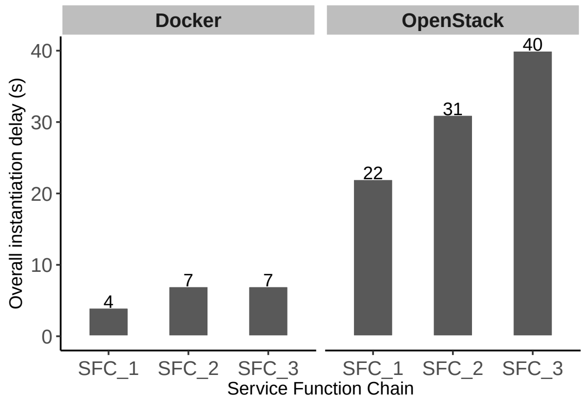

Fig. 7: Experiment 2: OID

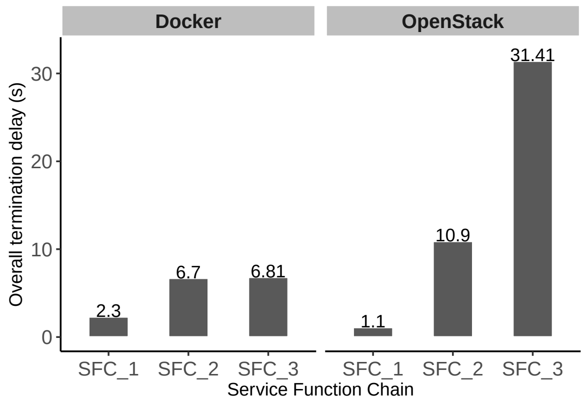

Fig. 8: Experiment 2: OTD

this performance analysis and its construction as a repeatable testbench will serve to benchmark existing and future MANO solutions for MEC.

VII. Acknowledgement

This work has been performed in the framework of the European Union's Horizon 2020 project 5G-CARMEN co-funded by the EU under grant agreement No. 825012, and the Horizon 2020 Fed4FIRE+ project, Grant Agreement No. 723638. The views expressed are those of the authors and do not necessarily represent the projects. The Commission is not liable for any use that may be made of any of the information contained therein.

References

[1] S. A. A. Shah, E. Ahmed, M. Imran, and S. Zeadally, "5G for Vehicular Communications," *IEEE Communications Magazine*, vol. 56, pp. 111–117, Jan 2018. doi: http://dx.doi.org/10.1109/MCOM.2018.1700467.

[2] Z. Ning and X. Wang, "Mobile Edge Computing-Enabled 5G Vehicular Networks: Toward the Integration of Communication and Computing," *IEEE Vehicular Technology Magazine*, vol. 14, no. March, pp. 54–61, 2019. doi: http://dx.doi.org/10.1109/MVT.2018.2882873.

[3] J. Guo, B. Song, Y. He, F. R. Yu, and M. Sookhak, "A Survey on Compressed Sensing in Vehicular Infotainment Systems," *IEEE Communications Surveys and Tutorials*, vol. 19, pp. 2662–2680, Fourthquarter 2017. doi: http://dx.doi.org/10.1109/COMST.2017.2705027.

[4] R. C. Abeywardana, K. W. Sowerby, and S. M. Berber, "Empowering Infotainment Applications: A Multi-Channel Service Management Framework for Cognitive Radio Enabled Vehicular Ad Hoc Networks," in *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pp. 1–5, June 2018. doi: http://dx.doi.org/10.1109/VTCSpring.2018.8417749.

[5] Z. Laaroussi, R. Morabito, and T. Taleb, "Service Provisioning in Vehicular Networks Through Edge and Cloud: An Empirical Analysis," in *2018 IEEE Conference on Standards for Communications and Networking (CSCN)*, pp. 1–6, Oct 2018. doi: http://dx.doi.org/10.1109/CSCN.2018.8581855.

[6] N. Slamnik-Krijestorac, H. C. Carvalho de Resende, C. Donato, S. Latré, R. Riggio, and J. Marquez-Barja, "Leveraging mobile edge computing to improve vehicular communications," in *2020 IEEE 17th Annual Consumer Communications Networking Conference (CCNC)*, pp. 1–4, 2020. doi: http://dx.doi.org/10.1109/CCNC46108.2020.9045698.

[7] ETSI, "ETSI Multi-access Edge Computing (MEC): Framework and Reference Architecture," vol. 1, pp. 1–21, 2019. Online [Available]: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/02.01.01_60/gs_MEC003v020101p.pdf.

[8] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, pp. 1657–1681, third quarter 2017. doi: http://dx.doi.org/10.1109/COMST.2017.2705720.

[9] T. Soenen, W. Tavernier, M. Peuster, F. Vicens, G. Xilouris, S. Kolometsos, M. Kourtis, and D. Colle, "Empowering Network Service Developers: Enhanced NFV DevOps and Programmable MANO," *IEEE Communications Magazine*, vol. 57, pp. 89–95, May 2019. doi: http://dx.doi.org/110.1109/MCOM.2019.1800810.

[10] R. Ju, W. Wang, J. Li, F. Li, and L. Han, "On Building a Low Latency Network for Future Internet Services," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pp. 1–6, Dec 2017. doi: http://dx.doi.org/10.1109/GLOCOM.2017.8254436.

[11] T. V. Doan, G. T. Nguyen, H. Salah, S. Pandi, M. Jarschel, R. Pries, and F. H. P. Fitzek, "Containers vs Virtual Machines: Choosing the Right Virtualization Technology for Mobile Edge Cloud," in *2019 IEEE 2nd 5G World Forum (5GWF)*, pp. 46–52, Sep. 2019. doi: https://doi.org/10.1109/5GWF.2019.8911715.

[12] T. Salah, M. J. Zemerly, C. Y. Yeun, M. Al-Qutayri, and Y. Al-Hammadi, "Performance Comparison between Container-based and VM-based Services," in *2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN)*, pp. 185–190, March 2017. doi: https://doi.org/10.1109/ICIN.2017.7899408.

[13] M. Peuster, M. Marchetti, G. García de Blas, and H. Karl, "Automated testing of NFV orchestrators against carrier-grade multi-PoP scenarios using emulation-based smoke testing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, p. 172, Jun 2019. doi: http://dx.doi.org/10.1186/s13638-019-1493-2.

[14] T. Sechkova, M. Paolino, and D. Raho, "Virtualized Infrastructure Managers for Edge Computing: OpenVIM and OpenStack Comparison," in *2018 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, pp. 1–6, June 2018. doi: https://doi.org/10.1109/BMSB.2018.8436858.