Edge Virtualization in Multihomed Vehicular Networks

Miguel Silva*‡, Miguel Luís†, Susana Sargento*‡
*Instituto de Telecomunicações, 3810-193 Aveiro, Portugal
†ISEL - Instituto Superior de Engenharia de Lisboa, 1959-007 Lisboa, Portugal
‡University of Aveiro, 3810-193 Aveiro, Portugal

***Abstract*—Vehicular Networks (VANETs) are a critical component of a Smart City environment. They extended the connectivity plane with support for a wide range of applications, from safety to entertainment. Such services, when deployed outside the vehicular network, may imply an additional delay, which can be critical. In addition, these services become inaccessible whenever the vehicles lose contact with the infrastructure.**

This paper proposes a practical solution that aims to minimize the impact of the services' location and its inaccessibility in a VANET. The solution focuses on using Network Function Virtualization technologies to support the deployment of the services at the edge of a mobility-enabled multihomed VANET, thus allowing the services to be accessible in intermittent connectivity situations, as well as enabling lower delays for critical services. The results obtained show that the solution is capable of deploying services at the edge of the VANET with low delay and with a fast recovery when in handover and mobility scenarios.

***Index Terms*—Multihomed Vehicular Networks, NFV, MANO, Virtualization**

I. Introduction

Nowadays, the concept of Smart City considers the full connectivity of its elements, being them people, objects (mobile or static), or others. Vehicular networks (VANETs) are seen as one of the key enablers for the *always connected* paradigm, providing useful communications among vehicles, and between vehicles and the infrastructure. Besides providing access to the Internet, vehicular communications can be used for information sharing, therefore enabling efficient Intelligent Transportation Systems (ITS) [1].

With the 5^{th} generation of mobile networks gaining ground, which will go beyond the 5G New Radio and Cellular Vehicular to Everything (C-V2X) technology for vehicular communications, resource sharing and the use of softwarized networks, replacing hardware network functions through software functions, are becoming increasingly popular. This is where the concept of Network Function Virtualization (NFV) comes into play, a technology capable of decoupling software from hardware, enabling flexibility, programmability and extensibility to the network [2], [3]. Through NFV, the network functions are available in the cloud, and pushed into the edge of the network through the connection to the cloud. However, due to the intermittent connectivity of VANETs, the provisioning of softwarized network functions in the network nodes, such as the On-Board Units (OBUs) in vehicles, is not straightforward.

This work proposes a practical solution which uses NFV technologies to support the deployment of lightweight Virtual Functions (VxFs) at the edge of a vehicular network. This work focuses on allowing the deployment of VxFs as close as possible to the end-users. Several solutions have explored various types of integration of cloud computing into the scope of a VANET. Most of them are focused on the concept of Vehicular Clouds (VC), where the VANET infrastructure itself is part of the Cloud [4], or on the concept of Infrastructure-as-a-Service (IaaS) in a VANET, where the vehicles' resources can be used to provide different types of services [5].

This approach brings some of the Cloud characteristics to the edge of a multihomed VANET with mobility support, while it makes use of the IaaS concept. By extending the NFV Infrastructure up to the edge of a vehicular network with mobility and multihoming, it is possible to explore the use of additional computing, networking and storage resources closer to the end user. The results show the advantage of deploying VxFs at the edge of the VANET, as well as the its impact in the presence of mobility and connection disruption.

The remaining of this paper is organized as follows. Section II presents the related work on cloud computing and NFV solutions in dynamic environments. Section III presents the VANET platform, its main concepts and the main components. Section IV presents the proposed solution's design and implementation. Finally, Section V presents the evaluation results and Section VI presents the final remarks and future work.

II. Related Work

Previously, several authors have investigated ways to bring cloud concepts to the scope of VANETs. Hussain *et al.* [4] proposed potential framework architectures for different types of cloud scenarios in VANETs. The authors divide VANET clouds into three major architectures namely Vehicular Cloud (VC), Vehicles using Clouds (VuC), and Hybrid Vehicular Clouds (HVC). The VCs are sub-divided into two types, static and dynamic: stationary vehicles providing cloud services, and clouds that are formed on demand in an ad hoc manner, respectively. When talking about VuC, they connect VANETs to traditional clouds where VANET users can make use of the services hosted on the cloud while on the move. As for the HVC, they are a combination of both the VC and

978-1-7281-8086-1/20/$31.00 ©2020 IEEE

VuC where they serve as consumers, using services from the cloud, and as producers where they provide their resources. These architectural frameworks have been explored and used in various different solutions.

In [6], Zhu *et al.* presented a solution which makes use of both cloud and NFV concepts. Currently many automotive companies release vehicles with an Intelligent Onboard System (IOS), which allows vehicles to have access to various types of services, such as updates on the traffic, services based on the vehicle's location (e.g. nearby gas stations), etc. However, IOSs have a closed architecture, meaning that, they cannot easily be upgraded to support new services.

Another NFV solution that was developed for a different type of Mobile *ad-hoc* Network is the one of Nogales *et al.* [7], where the authors designed an NFV system capable of deploying Network Services (NSs) over a cloud platform composed of an infrastructure of Small Unmanned Aerial Vehicles (SUAVs). The authors used Open Source MANO (OSM) with OpenStack (OS) as the Virtualized Infrastructure Manager (VIM), where the SUAVs made up the NFV Infrastructure (NFVI). The approach has been tested with a Voice over IP (VoIP) NS, and the results show that it is capable of deploying NSs over a SUAVs network, with good performance.

The solution presented in this work focuses on the concept of extending the cloud to the edge of a VANET. Much like in [6], this work aims to develop a solution which will enhance the VANETs, making it possible to easily deploy flexibly NSs in a vehicular network. But unlike [6], which focuses on using the hardware already present on the vehicles, this work focuses on developing a more open and flexible solution which is capable of using various types of hardware platforms.

III. Edge virtualization in a multihomed VANET

In a vehicular network scenario, most of the communications from the vehicles are performed with services or applications located outside the vehicular network (i.e. on the Internet). This happens mostly due to the applications and services used by the vehicles' occupants, but also due to vehicle hardware limitations and network configurations, for example in services such as safety applications.

Current solutions host virtualized versions of network functions, that make up the services which are used by the network and its end-users, in the Cloud. This way, the VANET's users can use the services whenever they need. Nevertheless, this solution is not the best when it comes to services with delay sensitive requirements and capabilities. In such a solution, it is also problematic that, given the fact that VANETs have a very dynamic network topology, handovers and loss of connection are to be expected, and access to the services would be dependent on the connectivity in the VANET.

In this work we provide a solution that aims to allow users to access specific services even when the vehicles are in intermittent connectivity situations with the VANET infrastructure, as well as allowing lower delays for critical services, such as in the case of road safety services, by bringing lightweight VxFs closer to the end-users, with the help of general purpose hardware platforms deployed at the edge of the network.

A. Multihomed VANET

The proposed solution is deployed over a VANET with mobility and multihoming support, i.e. with more than one wireless technology available for communication with the infrastructure. It is based on the NEMO-enabled Proxy Mobile IPv6 (N-PMIPv6) architecture, with additional mechanisms developed to support transparent handovers and simultaneous multihoming [8]. A simple representation of such network is shown in Figure 1, which main components are the following:

- *Local Mobility Anchor (LMA)* is the home agent for the mobile nodes. It is the topological anchor point for all mobile nodes and it manages their communications;
- *Road side units (RSUs)* are mobile access gateways (points of attachment to OBUs) to the LMA where clients, on OBUs, can connect to the Internet through them;
- *On-board units (OBUs)* act as mobile routers inside a vehicle and are responsible for creating an access link between the end-user and the mobility network.

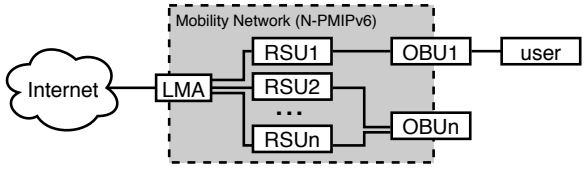

Fig. 1. Simple representation of a N-PMIPv6-based architecture.

The mobility protocol software executed in all the components of the VANET (i.e. the LMA, RSUs and OBUs) is responsible for guaranteeing all the mobility aspects (such as handovers), as well as the multihoming support. The OBUs also contain a connection manager which is capable of connecting each OBU to multiple RSUs using different access technologies (such as IEEE 802.11p/WAVE and Wi-Fi), and it is also responsible for selecting the best connections for the OBU, from the ones available at the time. Another main aspect of the VANET, given the mobility protocol it uses (i.e. N-PMIPv6), is the IPv6 support. In this protocol, IPv6 tunnels are created between the network nodes by the mobility protocol, which are used for protocol and data communications.

B. Network Function Virtualization

To enable the deployment of VxFs on the edge of a vehicular network, this work explores the use of NFV technologies, following the conventions set by European Telecommunications Standards Institute (ETSI). The main components of the NFV framework are briefly explained as follows:

- *Virtualized Network Functions (VNFs)* are software implementations of functions previously carried out by dedicated hardware;
- *Network Functions Virtualization Infrastructure (NFVI)* describes the hardware and software resources that are required to deploy, execute and manage the VNFs;

- *Management & Orchestration (MANO)* is the collection of various functional blocks that, as the name suggests, enable the management and orchestration of NFVI and VNFs. The main blocks within MANO are: the Virtualized Infrastructure Manager (VIM) which is responsible for managing the NFVI resources; Virtual Network Function Manager (VNFM) which is responsible for managing the VNFs; and finally the NFV Orchestrator (NFVO) which coordinates NFVI resources from different VIMs.

IV. Proposed Solution

A. Design

This work studies the applicability of NFV technologies in a VANET scenario, whose main goal and motivation is to develop a solution capable of supporting the automated deployment and configuration of moderately complex services for provisioning of smart cities services and to enhance the VANET's end-users experience. These services are implemented by VxFs, which can be seen as the abstraction of VNFs. Unlike VNFs that specifically implement network functions, VxFs can implement any type of function, thus they can be used to provide more diverse services that can range from security to entertainment. With this objective in mind, the use of NFV technologies, along side general purpose hardware on-boarded in the vehicles, will allow the solution to use the hardware's resources and virtualization to provide varied services with the help of the VxFs.

An overview of the proposed solution is depicted in Figure 2, which relies mainly on the following two components:

- The MANO system, located outside the VANET's scope, is the component that supports both the management and orchestration of the NFVI and the NSs;
- The hardware on-boarded on the vehicles alongside the OBUs which supports the execution of the NSs and VxFs.

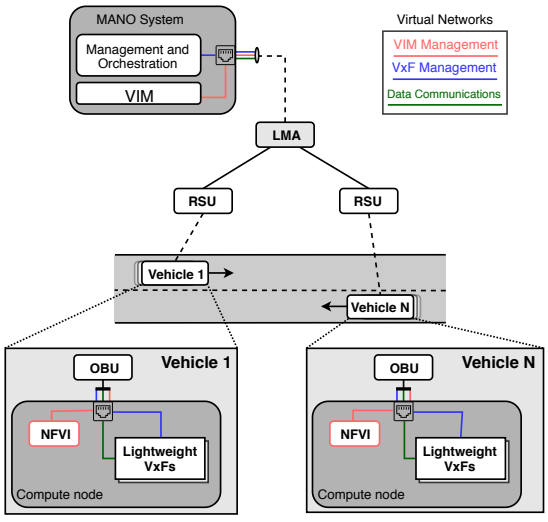

Fig. 2. Overview of the proposed architecture.

Another important component comprises the virtual networks that will sit on top of the pre-existing VANET's mobility network and will allow the communication between every element of the NFV solution. In what follows, we provide a detailed overview of each component.

1) MANO System: This is the component in charge of the MANO of the solution's NFVI, as well as the deployment of the NSs. It provides an orchestration service, through the NFVO and a VNFM, as well as a VIM. In the proposed solution, this component was implemented using open-source software: for the MANO we used the OpenSource MANO[1] stack to provide the management and orchestration, and for the VIM we used the cloud computing software OpenStack[2]. The MANO system is located outside of the VANET's scope.

The choice to use OSM as the MANO of the solution was based on a few specific reasons and those are: (1) it is a project hosted by ETSI, which is the entity that defined the NFV and MANO reference architectural frameworks [2], that provides a production-quality, extensive and working implementation of the MANO stack; (2) it is a project backed by a large group of network operators and other NFV related parties; (3) it supports fully automatic instantiation of NSs alongside their VxFs, as well as, their automatic configurations via the use of Juju charms present in its VNF Configuration and Abstraction component; (4) it is a fully open-source solution which provides extensive documentation and a straightforward installation process.

As for the VIM, the choice came down to OpenStack (OS), since the other VIM options which are supported by OSM are either limited in terms of features or they are commercial solutions. The solution proposed works on the minimal installation of OS, given the fact that it provides all the requirements of a VIM for this work.

2) On-Boarded Hardware: The second main component that makes up the solution is the on-boarded hardware. Considering the nature of the VANET used in this work, the node that sits closer to the end-user is the OBU, therefore it would make sense to use it as the platform on which the VxFs could be deployed. However, considering that OBUs were conceived with the intent to solely run the mobility-related software, they have limited capabilities, thus it was better use an extra hardware element alongside them in the vehicle, one capable of supporting the execution of the VxFs. This hardware has to be relatively small in size, which means that it is going to be limited in terms of both processing and storage capabilities.

With that in mind, we decided to use the Raspberry Pi as the on-boarded hardware platform. The RPi does not support the typical hypervisor alternatives, but it does support container virtualization via the use of Linux containers (LXC), as was presented in [9]. This works very well for the solution, since OS offers support for LXC too, making it possible to use the RPis as the NFVI of the MANO system. The RPis will then be the compute nodes of the OS VIM.

3) Overlay Networks: Another important component of the proposed solution concerns the virtual overlay networks that sit

[1]http://osm.etsi.org
[2]http://www.openstack.org

on top of the pre-existing VANET's mobility network. These networks will allow the communication between every element by interconnecting the components of the NFV solution. The choice to use overlay networks was based on a few specific reasons, mainly related with constraints imposed by the software that makes up the MANO system. For example, OSM requires that itself and all the compute nodes have a network interface present on the same sub-network, so that OSM can configure the VxFs once they are running. By using virtual overlay networks, the proposed solution is able to fulfill the requirements while not requiring major changes to the VANET or its mobility protocol.

The two main virtual overlay networks are the VIM management network and the VxF management network (depicted in Figure 2). The former is the network responsible for allowing all communications between the VIM and its compute nodes; the latter is the network responsible for allowing all communications between OSM and the VxFs. There is still another type of virtual networks that needs to be mentioned: these are the virtual internal networks required for data communication between the VxFs that make up a NS (also depicted in Figure 2). In order for the solution to work, both the management networks have to be pre-created on the respective nodes of the solution, particularly on the MANO system and on the on-boarded hardware. As for the virtual internal networks, these are not pre-created; they are instead created by the MANO system whenever a NS requires such a network.

B. Deployment

After a brief explanation about the solution's main components, this sub-section explains the methodology used to deploy them. It is important to recall that such virtualization solution will be deployed on top a vehicular network with mobility and multihoming support.

1) Connectivity between MANO System and On-boarded Hardware: The first and most critical step is to establish connectivity between the host computer, that runs the MANO system, and the RPis, that sit inside the mobility network's scope. Given the fact that the LMA already has an interface outside the mobility network, allowing the communication with the Internet, the connection between the LMA and the host that runs the MANO system was straightforward, requiring the addition of simple routes. To connect the RPis to the mobility network, we made use of both the OBUs' and RPis' Ethernet ports. The next step was to provide an internal network and subsequently IP addresses for the RPis. This was accomplished by creating a network between the OBUs and the RPis where all the traffic originated from the RPis is routed through the OBUs and into the mobility network's scope: this allows the RPis to reach the outside of the mobility network, thus reaching the host running the MANO system, all while using the VANET's infrastructure.

2) Deployment of the Overlay Networks: Through the connectivity between the host and the RPis, it was possible to start deploying the support overlay networks depicted in Figure 2. For this process, we decided to use Virtual eXtensible Local Area Network (VXLAN) as the technology responsible for supporting these virtual networks. VXLANs are a good choice as they provide a way to create a logical network for machines across different networks, allowing the creation of a layer 2 network on top of an already existing layer 3 network through the help of encapsulation, thus making it possible to create virtual overlay networks on top of the VANET's mobility network. The creation of the required virtual overlay networks was done on each component of the proposed solution using VXLAN devices. Figure 3 shows the network configuration done on the RPis (the compute nodes).

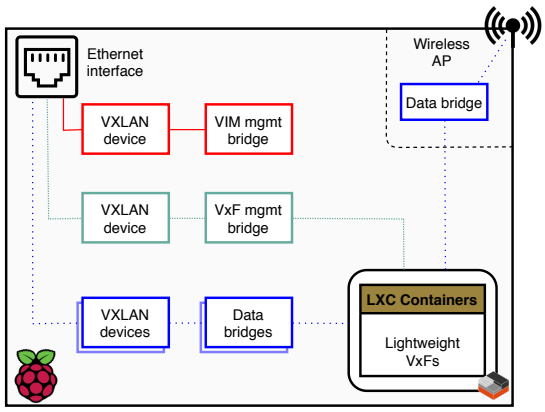

Fig. 3. Virtual Networks setup on the RPis (compute nodes).

The process of creating both the VIM and VxF management virtual networks is the same. First, the VXLAN devices are created over the RPi's Ethernet interface. Then, each VXLAN device gets attached to a Linux Bridge to enable multiple VxFs to share a single interface. The same process is done on the MANO system for both virtual networks, thus finalizing the overlay networks. As for the data communications networks, the OS configurations performed on the MANO system and the compute nodes enables the dynamic creation of such networks between VxFs, as requested by the OS VIM and according to instructions provided by the MANO system. Another aspect worth mentioning is the wireless interface of the RPi. This interface is configured to be managed by the OS in order to allow specific VxFs, like an Access Point (AP) VNF, to provide wireless connectivity to end-users.

V. Evaluation

This section presents the evaluation of the proposed solution. It focuses on the performance of two Intelligent Transportation System applications with two distinct configurations. The first use case assumes a NS with a single compute node (i.e. with a single vehicle), while the second setup explores the cooperation of multiple computes nodes (multiple vehicles).

A. Equipment

The system components are showcased in Figure 2. In the case of the VANET's components, the LMA is running on a dedicated computer, whereas RSUs and OBUs are implemented in single-board computers (NetRiders) with IEEE

802.11p, 802.11b/g/n and cellular interfaces. The MANO system is running on a dedicated computer, and the RPis are RPi 3 Model B. Table I shows their specifications.

TABLE I
MAIN SPECIFICATIONS AND CHARACTERISTICS OF THE EQUIPMENT.

Equipment	CPU [MHz]	Memory [MB]	Linux Kernel	Operating System
LMA	3600 (2 cores)	4096	4.14.3	Ubuntu 16.04.3 LTS
NetRider V3	680	64	3.7.4	VeniamOS 19.2
RPi 3 Model B	1200 (4 cores)	1024	4.4.38-v7+	16.04.6 LTS
MANO System	3600 (4 cores)	8192	4.4.0-154-generic	16.04.6 LTS

B. Results

Two uses cases have been developed to showcase the advantages of the proposed approach. The first main advantage is the ability to deploy services which encompass one vehicle, where even when the vehicle loses connection to the VANET's infrastructure, the service still works without any issues. If we compare this with the typical case where the services are hosted on the Cloud, whenever a vehicle gets disconnected from the infrastructure it cannot use the services until it gets reconnected. Building upon that, the second main advantage is the lower latency that the solution can provide by allowing the deployment of the services as close as possible to the end-users, as opposed to having the services on the Cloud.

1) Use Case 1: Safety service in a single compute node: It consists of a NS that aims to show how a safety service could be created and provided using the proposed solution. For this use case, the NS shown in Figure 4 was developed. This NS provides a simple safety service that can be deployed on a single compute node, where a specific VxF processes a video feed in order to detect distinct objects. The results obtained can then be accessed or viewed by the end-users. This NS is made up of two different VxFs, an *Object Detector* VxF and an AP VNF. The AP VNF is used to provide a way to support the data communications between the *Object Detector* VxF and the video feed, and between the VxF and the users.

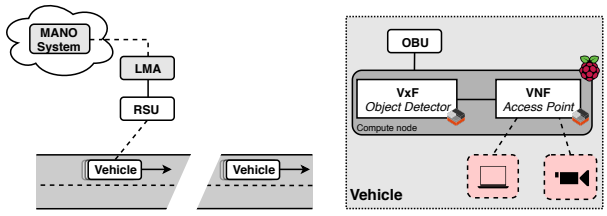

Fig. 4. Overview of the safety service used in Use Case 1.

Once this NS is deployed, it is possible to test and showcase the main advantages mentioned earlier. In order to do that, the OBU and thus the compute node which hosts the NS, is disconnected from the infrastructure after the deployment and configuration of the NS, considering that these processes require communication with the MANO system. The NS is compared in terms of communication delay with a different version of the same NS where the *Object Detector* VxF was deployed outside the scope of the VANET, on the cloud.

The results, show that, when the vehicle lost connection to the RSU, and thus to the infrastructure, the NS deployed on the vehicle had no problems in terms of its communication. This made sense given the fact that, for this NS, since all of its VxFs are deployed on a single compute node (one vehicle), communication between them is always possible even when the vehicle loses connection to the infrastructure. This means that the vehicle does not need to be connected to the infrastructure for the NS to be operational and usable.

As for the latency, the results, which can be seen in Figure 5, show lower values for the communication between both the VxF and the video feed, as well as between the user and the VxF for the case where the NS is deployed at the edge. The differences in terms of delay are not large, but nevertheless they are significant. It is worth noting that, for the comparison, even though the VxF was outside the scope of the VANET, it was still close to the LMA. In a more typical scenario, the Cloud VxF could be hosted on a machine behind several networks and, in that case, the delay would be more noticeable.

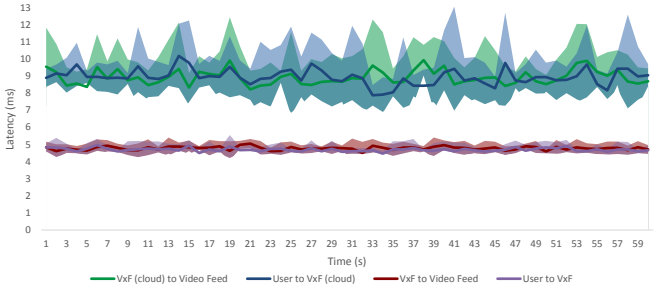

Fig. 5. Use case 1: Latency results for local vs. cloud VxF.

2) Use Case 2: Safety service in multiple compute nodes: The second use case consists of a NS involving more than one compute node. For that, the NS shown in Figure 6 is developed. This NS provides a safety service where a video feed from one vehicle is transcoded and the output can be viewed by other vehicles. This can be useful in situations where smaller vehicles are trying to overtake a bigger vehicle like a truck: the video feed from the truck is transcoded and viewed by the overtaking vehicles to check if the road is clear and the overtake maneuver can be done safely. This NS is made up of three VxFs: two AP VNFs and a *Transcoding* VxF. The AP VNF, like in the previous use case, provides a way to support the data communications between the end-user equipment and the *Transcoding* VxF.

Like in the previous use case, once the NS is deployed, it is possible to showcase how a multi-vehicle NS works in the proposed solution. In order to do that, similar tests to the ones presented in the previous use case were carried out, where vehicle B's OBU was disconnected from the infrastructure for a specific period of time. This NS was also tested in terms of communication delay by comparing it with a different version of the same NS where the *Transcoding* VxF was deployed outside the scope of the VANET, on the cloud.

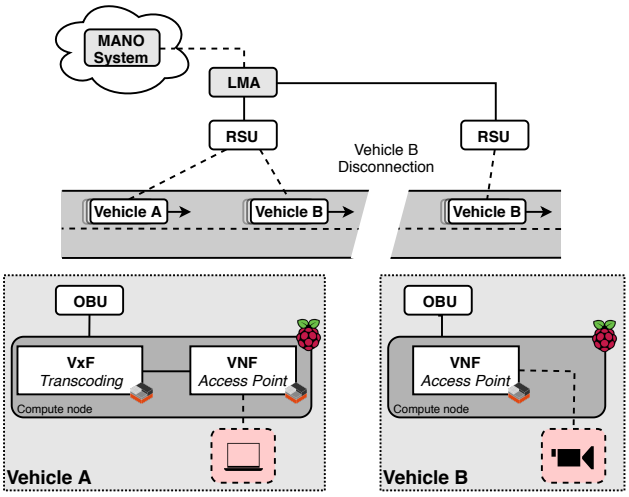

Fig. 6. Overview of the safety service used in Use Case 2 (multiple compute nodes).

Unlike in the previous use case, the results show that, for this NS to work properly, connection with the VANET's infrastructure is required at all times. Once deployed, the NS requires connection to the infrastructure for both of the vehicles, given that the communication between them is done using the underlay network (i.e. the mobility network through the infrastructure). When one or both of the vehicles lose connection to the RSU, the communication between the vehicles is not possible, meaning that the service is unusable until the connection is re-established for both of the vehicles.

Figure 7 shows the results in terms of delay. It is clear to see that, for the NS presented in this use case, the delay value is significantly lower between the user and the VxF. This makes sense given the fact that the user and the VxF are located on the same vehicle, which means that the communication between them is local. As for the communication between the VxF and the video feed, the latency values are significantly greater. This makes sense given the fact that, between the VxF and the video feed, the traffic has to cross the infrastructure (mobility network) from vehicle A to vehicle B and back. When comparing with the VxF deployed on the cloud, the results show that the values are obviously greater for the case where the user communicated locally with the VxF, but still lower when compared with the values for the communication between the VxF and the video feed. Like in the previous use case, even though the VxF is outside the scope of the VANET, it is still close to the LMA, which is not introducing a significant delay.

VI. Conclusion

This paper proposed an approach for virtualization at the edge of a multihomed VANET using NFV technologies. The results obtained by developing and testing two use cases show that the proposed approach is capable of deploying the NSs at the edge of the network, while offering lower delay values for critical services. For future work, as a way to mitigate

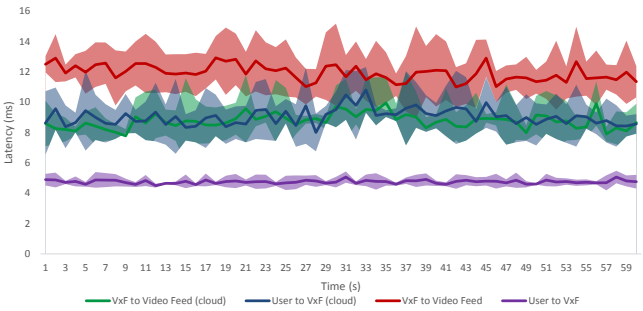

Fig. 7. Use case 2: Latency results for local vs. cloud VxF.

the connectivity problem shown in V-B2, where a NS which encompasses more than one vehicle is extremely dependent on the connection to the VANET, we will make use of the VANET's multi-hop capabilities. We will also add elements to the VANET, like SUAVs, to extend the effective range of the network and provide new functionalities.

Acknowledgments

This work is supported by the European Regional Development Fund (FEDER), through the Regional Operational Programme of Lisbon (POR LISBOA 2020) and the Competitiveness and Internationalization Operational Programme (COMPETE 2020) of the Portugal 2020 framework [Project 5G with Nr. 024539 (POCI-01-0247-FEDER-024539)] and National Financial Support National (FCT/OE), through project MobiWise (POCI-01-0145-FEDER-016426).

References

[1] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, "A comprehensive survey on vehicular ad hoc network," *J. Netw. Comput. Appl.*, vol. 37, pp. 380–392, 1 2014.

[2] "Network Functions Virtualisation (NFV) Architectural Framework," European Telecommunications Standards Institute, Sophia Antipolis, France, Tech. Rep. GS NFV 002, 1 2014.

[3] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "NFV and SDN—Key Technology Enablers for 5G Networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2468–2478, 11 2017.

[4] R. Hussain, J. Son, H. Eun, S. Kim, and H. Oh, "Rethinking Vehicular Communications: Merging VANET with cloud computing," in *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, 12 2012, pp. 606–609.

[5] M. Abuelela and S. Olariu, "Taking VANET to the Clouds," in *Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia*, ser. MoMM '10, New York, NY, USA, 2010, pp. 6–13.

[6] M. Zhu, J. Cao, Z. Cai, Z. He, and M. Xu, "Providing flexible services for heterogeneous vehicles: an nfv-based approach," *IEEE Network*, vol. 30, no. 3, pp. 64–71, 5 2016.

[7] B. Nogales, V. Sanchez-Aguero, I. Vidal, F. Valera, and J. Garcia-Reinoso, "A nfv system to support configurable and automated multi-uav service deployments," in *Proceedings of the 4th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications*, ser. DroNet'18, 2018, p. 39–44.

[8] N. Capela and S. Sargento, "An intelligent and optimized multihoming approach in real and heterogeneous environments," *Wireless Networks*, pp. 1935–1955, 2015.

[9] C. Pahl, S. Helmer, L. Miori, J. Sanin, and B. Lee, "A Container-Based Edge Cloud PaaS Architecture Based on Raspberry Pi Clusters," in *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, 8 2016, pp. 117–124.