

Journal of Visualized Experiments

Automated deployment of an IP telephony service on Unmanned Aerial Vehicles using Network Functions Virtualization --Manuscript Draft--

Article Type:	Invited Methods Article - JoVE Produced Video
Manuscript Number:	JoVE60425R1
Full Title:	Automated deployment of an IP telephony service on Unmanned Aerial Vehicles using Network Functions Virtualization
Section/Category:	JoVE Engineering
Keywords:	Unmanned Aerial Vehicles (UAVs), Network Functions Virtualization (NFV), Management and Orchestration (MANO), Cloud computing platform, Virtual Network Function (VNF), IP telephony service, open source, 5G.
Corresponding Author:	Ivan Vidal Universidad Carlos III de Madrid Leganés, Madrid SPAIN
Corresponding Author's Institution:	Universidad Carlos III de Madrid
Corresponding Author E-Mail:	ividal@it.uc3m.es
Order of Authors:	Borja Nogales Ivan Vidal Victor Sanchez-Aguero Francisco Valera Luis F. Gonzalez Arturo Azcorra
Additional Information:	
Question	Response
Please indicate whether this article will be Standard Access or Open Access.	Standard Access (US\$2,400)
Please indicate the city, state/province, and country where this article will be filmed . Please do not use abbreviations.	Leganés, Madrid, Spain

Automated Deployment of an IP Telephony Service on Unmanned Aerial Vehicles using Network Functions Virtualization

Borja Nogales¹, Ivan Vidal¹, Victor Sanchez-Aguero^{1,2}, Francisco Valera¹, Luis F. Gonzalez¹, Arturo Azcorra^{1,2}

¹ *University Carlos III of Madrid, Av. Universidad, 30, 28911, Madrid, Spain*

² *IMDEA Networks Institute. Avda. del Mar Mediterráneo 22, 28918, Madrid, Spain*

Abstract – The Network Function Virtualization (NFV) paradigm has currently consolidated as one of the key enabling technologies in the development of the 5th Generation of mobile networks. This technology aims at alleviating the hardware dependencies in the provision of network functions and services, using virtualization techniques that allow the softwarization of those functionalities over an abstraction layer. In this context, there is an increasing research interest in exploring the potential of unmanned aerial vehicles, or UAVs, to offer a flexible platform capable of enabling cost-effective NFV operations over delimited geographic areas.

To demonstrate the practical feasibility of utilizing NFV technologies in UAV platforms, we present a protocol to set up a functional NFV environment, based on open-source technologies, in which a set of small UAVs supply the computational resources that support the deployment of moderately complex network services. Then the protocol details the different steps needed to support the automated deployment of an IP telephony service over a network of interconnected UAVs, leveraging the capacities of the configured NFV environment. Our experimentation results demonstrate the proper operation of the service after its deployment. We want to highlight that, although the protocol focuses on a specific type of network service (i.e., IP telephony), the described steps may serve as a general guide to deploy other type of network services. On the other hand, the protocol description considers concrete equipment and software to set up the NFV environment (e.g., specific single board computers and open source software). The utilization of other hardware and software platforms may be feasible, although the specific configuration aspect of the NFV environment and the service deployment may present variations with respect to those described in the protocol.

Keywords: Unmanned Aerial Vehicles (UAVs), Network Functions Virtualization (NFV), Management and Orchestration (MANO), Cloud computing platform, Virtual Network Function (VNF), IP telephony service, open source, 5G.

The authors also want to state that:

1. The paper is novel and has not previously been published, completely or in part, in another journal.

2. The paper has not been submitted for publication anywhere else, and it will not be submitted to a different journal until a decision has been made by *Jove*.
3. The authors declare that there is no conflict of interest regarding the publication of this paper.

TITLE:

Automated Deployment of an Internet Protocol Telephony Service on Unmanned Aerial Vehicles Using Network Functions Virtualization

AUTHORS AND AFFILIATIONS:

Borja Nogales^{1,*}, Ivan Vidal^{1,*}, Victor Sanchez-Aguero^{1,2,*}, Francisco Valera^{1,*}, Luis F. Gonzalez^{1,*}, Arturo Azcorra^{1,2,*}

¹Department of Telematic Engineering, University Carlos III of Madrid, Madrid, Spain

²IMDEA Networks Institute, Madrid, Spain

*These authors contributed equally.

Corresponding Author:

Ivan Vidal (ividal@it.uc3m.es)

Email Addresses of Co-authors:

Borja Nogales (bdorado@pa.uc3m.es)

Victor Sanchez-Aguero (victor.sanchez@imdea.org)

Francisco Valera (fvalera@it.uc3m.es)

Luis F. Gonzalez (luisfgon@it.uc3m.es)

Arturo Azcorra (azcorra@it.uc3m.es)

KEYWORDS:

unmanned aerial vehicles (UAVs), network functions virtualization (NFV), management and orchestration (MANO), cloud computing platform, virtual network function (VNF), IP telephony service, open source, 5G

SUMMARY:

The objective of the described protocol is twofold: to configure a network functions virtualization environment using unmanned aerial vehicles as computational entities providing the underlying structure to execute virtualized network functions and to use this environment to support the automated deployment of a functional internet protocol telephony service over the aerial vehicles.

ABSTRACT:

The Network Function Virtualization (NFV) paradigm is one of the key enabling technologies in the development of the 5th generation of mobile networks. This technology aims to lessen the dependence on hardware in the provision of network functions and services by using virtualization techniques that allow the softwarization of those functionalities over an abstraction layer. In this context, there is increasing interest in exploring the potential of unmanned aerial vehicles (UAVs) to offer a flexible platform capable of enabling cost-effective NFV operations over delimited geographic areas.

To demonstrate the practical feasibility of utilizing NFV technologies in UAV platforms, a protocol is presented to set up a functional NFV environment based on open source technologies, in which a set of small UAVs supply the computational resources that support the deployment of moderately complex network services. Then, the protocol details the different steps needed to support the automated deployment of an internet protocol (IP) telephony service over a network of interconnected UAVs, leveraging the capacities of the configured NFV environment. Experimentation results demonstrate the proper operation of the service after its deployment. Although the protocol focuses on a specific type of network service (i.e., IP telephony), the described steps may serve as a general guide to deploy other type of network services. On the other hand, the protocol description considers concrete equipment and software to set up the NFV environment (e.g., specific single board computers and open source software). The utilization of other hardware and software platforms may be feasible, although the specific configuration aspect of the NFV environment and the service deployment may present variations with respect to those described in the protocol.

INTRODUCTION:

One of the most coveted goals within the new era of the mobile communications (most commonly known as the 5th mobile generation or 5G) is to be able to provide robust information technology services in situations where the primary telecommunications infrastructure may not be available (e.g., due to an emergency). In this context, the UAVs are receiving increasing attention from the research community due to their inherent versatility. There are numerous works that use these devices as a cornerstone for the provision of a large variety of services. For instance, the literature has analyzed the capacity of these devices to build an aerial communication infrastructure to accommodate multimedia services¹⁻³. Furthermore, prior research has shown how the cooperation among several UAVs can extend the functionality of different communication services such as surveillance⁴, collaborative search and rescue⁵⁻⁸, or agribusiness⁹.

On the other hand, the NFV technology has acquired great significance within the telecom operators as one of the 5G key enablers. NFV represents a paradigmatic change regarding the telecommunications infrastructure by alleviating the current dependency of network appliances on specialized hardware through the softwarization of the network functionalities. This enables a flexible and agile deployment of new types of communication services. To this purpose, the European Telecommunications Standards Institute (ETSI) formed a specification group to define the NFV architectural framework¹⁰. Additionally, the ETSI currently hosts the Open Source Mano (OSM) group¹¹, which is in charge of developing an NFV Management and Orchestration (MANO) software stack aligned with the definition of the ETSI NFV architectural framework.

Given all the aforementioned considerations, the synergic convergence between UAVs and NFV technologies is currently being studied in the development of novel network applications and services. This is illustrated by several research works in the literature that point out the advantages of these types of systems¹⁴⁻¹⁶, identify the challenges of this convergence and its missing aspects, highlight future research lines on this topic¹⁷, and present pioneer solutions based on open source technologies.

In particular, the integration of NFV technologies into the UAV arena enables the rapid and flexible deployment of network services and applications over delimited geographic areas (e.g., an IP telephony service). Following this approach, a number of UAVs can be deployed over a specific location, transporting compute platforms as payload (e.g., small-size single board computers). These compute platforms would provide a programmable network infrastructure (i.e., an NFV infrastructure) over the deployment area, supporting the instantiation of network services and applications under the control of a MANO platform.

Notwithstanding the benefits, the realization of this view presents a set of fundamental challenges that needs to be carefully addressed, such as the appropriate integration of these compute platforms as an NFV infrastructure, using an existing NFV software stack, so that an NFV orchestration service can deploy virtual functions on the UAVs; the constraints in terms of the computational resources provided by the compute platforms, as the UAVs transporting them may typically present limitations in terms of size, weight, and computing capacity of payload equipment; the proper placement of the virtual functions onto UAVs (i.e., selecting the best UAV candidate to deploy a particular virtual function); the maintenance of the control communications with the UAVs in order to manage the lifecycle of the VNFs in spite of the potentially intermittent availability of network communications with them (e.g., caused by mobility and battery constraints); the limited operation time of the UAVs due to their battery consumption; and the migration of the virtual functions when a UAV needs to be replaced due to its battery exhaustion. These benefits and challenges are detailed in previous work^{18,19} that includes the design of an NFV system capable of supporting the automated deployment of network functions and services on UAV platforms, as well as the validation of the practical feasibility of this design.

In this context, this paper focuses on describing a protocol to enable the automated deployment of moderately complex network services over a network of UAVs using the NFV standards and open source technologies. To illustrate the different steps of the protocol, a re-elaboration of an experiment presented in Nogales et al.¹⁹ is presented, consisting of the deployment of an IP telephony service. To aid the reproducibility of this work, real flight is considered as optional in the presented procedure, and performance results are obtained with the UAV devices on the ground. Interested readers should be able to replicate and validate the execution of the protocol, even in a controlled laboratory environment.

Figure 1 illustrates the network service designed for this procedure. This network service is built as a composition of specific softwarization units (categorized within the NFV paradigm as Virtual Network Functions, or VNFs) and provides the functionality of an IP telephony service to users in the vicinity of the UAVs. The VNF composing the service are defined as follows:

- Access Point VNF (AP-VNF): This VNF provides a Wi-Fi access point to end-user equipment (i.e., IP phones in this experiment).
- IP telephony server VNF (IP-telephony-server-VNF): It is responsible for managing the call signaling messages that are exchanged between IP phones to establish and terminate a voice call.

- Domain Name System VNF (DNS-VNF): This VNF provides a name resolution service, which is typically needed in IP telephony services.
- Access router VNF (AR-VNF): provides network routing functionalities, supporting the exchange of traffic (i.e., call signaling in this experiment) between the IP phones and the telecommunication operator domain.
- Core router VNF (CR-VNF): provides network routing functionalities in the telecommunication operator domain, offering access to operator-specific services (i.e., the IP telephony server) and external data networks.

Moreover, **Figure 1** presents the physical devices used for the experiment, how they are interconnected, and the specific allocation of VNFs to devices.

PROTOCOL:

1. Prior requisites for the experiment

1.1. Install the Management and Orchestration (MANO) software stack provided by the Open Source MANO (OSM) project. Specifically, this experiment uses OSM Release FOUR²⁰, which can be executed in a single server computer or in a Virtual Machine (VM) fulfilling the requirements specified by the OSM community: Ubuntu 16.04 as the operating system (64-bit variant image), two central processing units (CPUs), 8 GB random access memory (RAM), a 40 GB storage disk, and a single network interface with Internet access. The procedure to install OSM Release FOUR along with its technical details are available in the online documentation provided by the OSM community²¹.

1.2. Set up a cloud computing platform, providing the functions of a virtual infrastructure manager (VIM) compliant with OSM Release FOUR. For this experiment, OpenStack release Ocata²² is used, running in a VM with Ubuntu 16.04 as the operating system, four CPUs, 16 GB RAM, and 200 GB storage disk. In the experiment, the VIM manages an NFV infrastructure (NFVI) integrated by two high-profile server computers, each with Ubuntu 16.04 as the operating system, eight CPUs, 128 GB RAM, and 4 TB storage disk. All the information on how to set up a cloud computing platform is included in the installation guide included in the OpenStack documentation²³. This cloud platform is referred to as the core cloud platform.

1.3. Set up an additional cloud computing platform for the UAVs is referred to as the UAVs cloud platform.

1.3.1. Ensure that this platform features a VIM based on OpenStack release Ocata. In this case, the resources used by the VIM installation are Ubuntu 16.04 as operating system, two CPUs, 6 GB RAM, 100 GB storage disk, and an external Wi-Fi USB adapter.

1.3.2. The NFVI integrated in this cloud platform consists of a single fixed compute server (Ubuntu 16.04 as operating system, eight CPUs, 8 GB RAM, 128 GB storage disk, and an external Wi-Fi USB adapter) and three single board computers (SBCs). The latter provide a hardware

platform that can easily be onboarded on a UAV. See Section 3 for the procedure to setup a UAV cloud platform with these devices as compute nodes.

1.4. Equip each SBC with a battery-power supply hardware attached on top (HAT) to ensure the operation of these units even when they are in motion, being carried by a UAV.

NOTE: Step 1.5 is optional because the provision of the network service in the experiment does not depend on having UAVs. In addition, the SBCs are carried as the payload of the UAVs and no other additional connections (e.g., Ethernet or USB) are needed, because the network communications required for the proper operation of the IP telephony service are provided by the SBCs, through their Wi-Fi adapters, and the power supply is provided by the power-supply HAT mentioned in step 1.4.

1.5. Attach each SBC as the payload of a UAV through a fixing accessory. In this experiment, three commercial UAVs were chosen to transport the compute units offered by the SBCs.

1.6. Select two wireless voice-over-IP (VoIP) phones that support the IEEE 802.11b wireless communications standard; this model provides wireless communications via Wi-Fi. As an alternative, the voice call could be executed using softphone applications such as Linphone²⁴ or Jitsi²⁵.

1.7. As an experimental requirement, make sure of the availability of: a) layer-3 communications between the OSM software stack and each of the VIMs to enable the orchestrated deployment of the network service developed for this experiment, b) layer-3 communications between the OSM and the VNFs at each cloud platform to support VNF configuration procedures, and c) the layer-3 communications among the VNFs running at every VIM to enable the proper functioning of the network service.

1.8. All the content needed to carry out the experiment is provided in the public experiment repository <http://vm-images.netcom.it.uc3m.es/JoVE/>.

2. Validating the functionality of the softwarization units via emulation

NOTE: To prove the appropriate operation of the network service of the experiment (see **Figure 1**) under realistic deployment conditions, a purpose-specific emulation platform based on Linux containers²⁶ and ns-3²⁷ was used. This platform allows emulating multi-hop aerial links and defining the characteristics of those links (e.g., length of the wireless communication links, pattern of data packet losses, the radio technology used in the wireless communications, etc.). Thus, this section of the protocol describes the steps to be followed to verify the appropriate operation of the IP telephony service under realistic wireless communication link conditions through the emulation platform.

2.1. Download the emulation platform from the experiment repository. The platform is available as a virtual machine, named "uav-nfv-jove-experiment.qcow", compliant with the KVM

virtualization technology²⁸. This machine contains a precreated template that emulates the network service and the multi-UAV scenario presented in **Figure 1** and a user with administrator privileges capable of executing that template.

NOTE: By default, the following steps are automatically executed when the emulation platform virtual machine is started: a) the virtual environment is configured to enable the network emulation (i.e., network interfaces, Linux bridges²⁹); b) the Linux Containers representing the different physical components of the testbed (i.e., the SBCs and the fixed compute server for the UAV cloud platform, and the compute server for the core cloud platform) are created; and c) the functions provided by the different VNFs of the IP telephony service (i.e., access points, routers, DNS serve, and IP telephony server) are deployed as Linux Containers over their corresponding emulated SBCs and compute servers.

2.2. Before the validation process, set up an emulated multi-hop aerial network using the ns-3 simulator, in order to enable the connectivity between the different network participants. This procedure will emulate the realistic wireless communications that take place in the scenario depicted in **Figure 1** (i.e., the Wi-Fi ad-hoc network, which enables the data exchange among the nodes of the UAV cloud platform and the wireless networks offered by the two Wi-Fi access points provided in the service).

2.2.1. Create the multi-hop aerial network. For this purpose, execute the **multi-hop-aerial-net.sh** script (available within the emulation platform machine) using the following command: **sudo sh /home/jovevm/scripts/multi-hop-aerial-net.sh > multi-hop-aerial-net-trace.log 2>&1 &**. This command portrays the simulation trace in the specified log-file to enable debugging in the case of errors.

2.2.2. Check if the network has been successfully created. To this end, verify if the Linux Containers "IP-phone-a" and "IP-phone-b" (illustrated in **Figure 1** as the end-user equipment that connects to an AP-VNF) have obtained an IP address through the DHCP service, which is only accessible through the multi-hop aerial network. The status of the Linux container executed within the emulation machine, as well as their IP addresses, can be checked using the command **lxc list**.

2.3. Verify the capacity of the emulated network service to process the signaling messages needed to set up the IP telephony call. For this purpose, both the "IP-phone-a" and "IP-phone-b" Linux containers have installed the "SIPp" tool³⁰. "SIPp" provides the functionality to emulate an IP phone creating the mentioned signaling messages, send them to an IP telephony server, and process the response to verify the correct operation of the latter.

2.3.1. Execute the script **test-signaling.sh** in both containers, which runs the "SIPp" tool to generate and send signaling messages to the IP-telephony-server-VNF.

2.3.2. Check the scenario screen provided by execution of the previous step. The reception of "200" response shows the appropriate functioning of the IP-telephony-server-VNF.

2.4. Validate that the network service can process the data traffic that is generated during an IP telephony call. To do so, the flow scheduling "Trafic" tool³¹ is installed in the "IP-phone-a" and "IP-phone-b" Linux containers.

2.4.1. Execute the following command to start the server agent of Trafic: **lxc exec IP-phone-b sh called-party.sh**.

2.4.2. Then, execute the following command to start the client agent of Trafic and get the network statistics: **lxc exec IP-phone-a sh caller.sh**. The data traffic emulating a voice call is terminated after 60 s. The script displays a confirmation message and the most significant performance metrics concerning the voice traffic.

2.4.3. Check the obtained metrics and verify that the IP telephony service can effectively support an interactive voice conversation. To do so, see the information included in the section on representative results.

3. UAVs cloud platform construction

3.1. Select the model of SBC that can provide the virtualization substrate to execute lightweight VNFs. The technical specifications of the SBC devices utilized during the experiment are: four CPUs, 1 GB RAM, and a 32 GB storage disk. Additionally, each SBC has three network interfaces: an Ethernet interface, an integrated Wi-Fi interface, and an external Wi-Fi USB adapter.

3.2. Prepare the SBCs to be subsequently integrated into the UAVs cloud platform.

3.2.1. Install Ubuntu Mate³² 16.04.6 as the operating system, given that the OpenStack installation packages are included in this Linux distribution.

3.2.2. Install and configure the required packages as indicated in the OpenStack documentation³³ to allow the SBCs to act as the compute nodes of the UAV cloud platform. Following the previous guide, enable the utilization of Linux containers in the configuration of the OpenStack packages. Container virtualization is used due to the resource constraints of the devices that can typically be onboarded on small-sized UAVs.

3.2.3. In the SBC, download and execute the script **rpi-networking-configuration.sh**, available within the experiment repository. This script enables the wireless communications of the SBCs, as well as the required configuration to allow the creation of virtual networks attached to the wireless interfaces.

3.2.4. Download and execute the script **VIM-networking-configuration.sh**, available within the experiment repository, in the host running the UAV cloud platform VIM. This script oversees setting up the wireless communications of the VIM to enable the information exchange with the SBCs.

NOTE: Once the networking is well configured and the VIM has connectivity with the SBCs, the VIM automatically integrates them into the UAV cloud platform as computational units capable of executing VNFs

3.3. Create an OpenStack availability zone for each of the SBCs. This will allow deploying each of the lightweight VNFs of the experiment in an appropriate UAV unit. To do so, log in to the web graphical user interface provided by the VIM with the administrator credentials, create the availability zones in the **Administrator > System > Host Aggregates** tab, and edit each availability zone to add the appropriate host (i.e., each SBC integrated into the UAV cloud platform).

3.4. Verify the correct setup of the UAV cloud platform. To do so, access the **Administrator > System > System Information** tab with the same login as in the previous step, and click in the **Computing Service and Network Agents** section to check that the status of the displayed items is "Alive" and "UP".

4. Configuring the experiment

4.1. Download the VNF images that implement the different components of the IP telephony service: the AP-VNF, the DNS-VNF, IP-telephony-server-VNF, the AR-VNF, and the CR-VNF. These images can be downloaded from the experiment repository.

4.2. Upload the VNF images to their correspondent VIM (i.e., the AP-VNF and the DNS-VNF to the UAV cloud platform VIM) and the VoIP-VNF to the core cloud platform VIM. To do so, log in into the web graphical user interface provided by each VIM with the administrator credentials, click on the **Create Image** button of the **Administrator > System > Images** tab, and create an image using the displayed form and selecting the appropriate image. This process is done at the corresponding VIM for each image that has been downloaded in the prior step.

4.3. Download the VNF descriptors (VNFDs) of the experiment from the experiment repository. These descriptors provide the templates that describe the operational requirements of a VNF, as well as the placement policies that indicate the availability zone in charge of hosting the VNF itself. More information on NFV descriptors can be found in the information model of OSM³⁴.

4.4. Upload the VNFDs. Use a web browser to access the OSM graphical user interface, and sign in with the administrator credentials. Then, drag and drop the VNFDs into the **VNF Packages** tab.

4.5. Download the network services descriptor (NSD) from the experiment repository. This descriptor is a template that specifies the VNFs comprising the service, as well as how those VNFs are interconnected.

4.6. Upload the NSD. Drag and drop the NSD into the **NS Packages** tab of the OSM graphical user interface.

4.7. Using the graphical user interface of OSM, add a VIM account for the UAV cloud platform VIM and for the core cloud platform VIM. To do this, access the **VIM accounts** tab with the administrator credentials, click on the button + **New VIM** and complete the displayed form with the requested information. Repeat this action for both VIMs.

5. Executing the experiment

5.1. Deploy the network service. From the **NS packages** tab of the OSM graphical user interface, click on the **Instantiate NS** button of the NSD uploaded in step 5.6. Then, fill the displayed form, indicating the VIM that will be used to deploy each VNF composing the NS. In addition, the OSM is responsible for processing the placement policies indicated in the VNFDs to specify the VIM which compute unit is in charge of hosting each VNF. For this experiment, the VNFs are placed in the compute units as illustrated in **Figure 1**.

5.2. Wait until the OSM graphical user interface indicates the success on the network service deployment.

NOTE: The operation of the network service is totally independent from the flight of the UAVs: The IP telephony service can be provided when the UAVs are flying or saving battery consumption perched on a surface. Thus, step 5.3 is optional.

5.3. Take off the UAVs. Log in to the mobile application and control the flight of each UAV to stably maintain it in an intermediate height and avoid the turbulence caused by the rotation of the motors close to a surface.

5.4. Prepare each of the IP phones to carry out the call.

5.4.1. Connect a wireless VoIP phone to each of the access points offered by the network service. For this purpose, specify the SSID (*Service Set Identifier*) in the **Menu > Wireless > SSID** tab and choose the **Infrastructure** mode in the **Menu > Wireless > Network Mode** section. Finally, select the networking configuration through the *Dynamic Host Configuration Protocol* (DHCP) in the **Menu > Net Settings > Network Mode** tab.

5.4.2. Configure the *Session Initiation Protocol* (SIP) parameters to enable the appropriate exchange of signaling messages with the IP telephony server. In this context, access to the **Menu > SIP Settings** tab and specify the host name of the IP telephony server VNF ("dronesVoIP.net") in the **Registrar > Registrar IP** and **Proxy Server > Proxy IP** tabs. Furthermore, create a user account introducing the name of the user (e.g., caller-A) in the **User Account > Phone Number** and **User Account > Username** sections.

5.4.3. Create an entry in the phonebook of one of the IP phones providing the information of the user that is to be called. To do so, select the **Menu > Phonebook > Add Entry** tab, and fill in the requested parameters that appear in the display as follows: Display name = caller-B; User Info = caller-B; Host IP = dronesVoIP.net; Port = 5060. Finally, select the "Proxy" option versus the P2P

(peer-to-peer).

5.5. Start the call to the other party. To do so, select the called party using the **Menu > Phonebook > Search** option of the IP phone. Then, press the call button. Once the other IP phone starts ringing, accept the incoming call with the call button.

6. Procedure to gather experimental results

6.1. Connect a commodity laptop to one of the wireless APs and run the **ping** command line tool to the IP address of the phone connected to the other AP during 180 s. The IP address can be checked in the **Menu > Information > IP address** option of the IP phone once the connection is established with the AP. Save the Round-Trip Time (RTT) measurements, redirecting the output provided by the **ping** tool into a file.

6.2. Execute the **tcpdump** command line tool in one of the running AP VNFs to capture the traffic exchanged during the IP call. Save this traffic into a file enabling the writing flag of the command line tool at the execution time and specifying the name of the file.

6.3. Perform a new IP telephony call. Maintain the call for the desired time period (e.g., 1 min). Then, terminate the call, pressing the hang up button of one of the IP phones.

6.4. Keep the files generated by the **tcpdump** and **ping** tools for further processing. See Representative Results.

REPRESENTATIVE RESULTS:

Based on the data obtained during the execution of the experiment, in which a real VoIP call is executed and following the steps indicated by the protocol to gather this information, **Figure 2** depicts the cumulative distribution function of the end-to-end delay measured between two end-user equipment items (i.e., a commodity laptop and an IP telephone). This user equipment represents two devices that are interconnected through the AP VNFs of the deployed network service. More than 80% of the end-to-end delay measurements were below 60 ms, and none of them were higher than 150 ms, which guarantees appropriate delay metrics for the execution of a voice call.

Figure 3 illustrates the exchange of DNS and SIP signaling messages. These messages correspond to the registration of one of the users in the IP telephony server (i.e., the user whose IP phone is connected to the AP VNF where the "**tcpdump**" tool is running) and to the establishment of the voice call.

Finally, **Figure 4** and **Figure 5** show the data traffic captured during the call. In particular, the first one represents the constant stream of voice packets transmitted and received by one of the wireless phones during the call, whereas the latter illustrates the jitter in the forward direction with an average value lower than 1 ms.

The results that were obtained in the experiment for the delay figures (end-to-end delay and jitter) satisfy the recommendations specified by the International Telecommunication Union - Telecommunication Standardization Sector (ITU-T)³⁵. Accordingly, the voice call progressed with no glitches and good sound quality. This experiment validated the practical feasibility of using NFV technologies and UAVs to deploy a functional IP telephony service.

FIGURE AND TABLE LEGENDS:

Figure 1: Overview of the network service, depicting the VNFs, the entities in which they are executed, and the virtual networks needed for the provision of the IP telephony service.

Figure 2: End-to-end delay. Representation of the end-to-end delay offered to the end-user equipment connected to the AP VNFs. For this purpose, the cumulative distribution function of the end-to-end delay has been computed from the measured RTT samples obtained with the "ping" command line tool.

Figure 3: User registration and call signaling messages. Illustration of the signaling traffic (DNS and SIP) exchanged to register a user in the IP telephony server and to create and terminate the multimedia session that supports the execution of the voice call.

Figure 4: Stream of voice packets. Representation of the voice traffic exchanged during the call, measured at one of the AP VNFs. (Abbreviations: RX = receive, TX = transmit, RTP = real-time transport protocol).

Figure 5: Evolution of the network jitter during the call. Representation of the jitter experienced by the transmitted voice packets in the forward direction from one phone to the other.

DISCUSSION:

One of the most important aspects of this experiment is the use of virtualization technologies and NFV standards with UAV platforms. NFV presents a new paradigm aiming to decouple the hardware dependency of the network functionalities, thus enabling the provision of these functionalities through softwarization. Accordingly, the experiment does not depend on the use of the hardware equipment specified in the protocol. Alternatively, different models of single board computers can be selected, as long as they are in line with the dimensions and the transport capacity of the UAVs and they support Linux containers.

Notwithstanding this flexibility in terms of hardware selection, all the content provided for the reproducibility of the experiment is oriented towards the use of open source technologies. In this context, the configuration aspects and the software tools are conditioned to the use of Linux as the operating system.

On the other hand, the experiment considers the interoperation of two different computational platforms (i.e., the UAV cloud platform and the core cloud platform) to provide a moderately complex network service. However, this is not strictly needed, and the protocol could be followed to support scenarios in which only the UAV cloud platform is involved.

In addition, the solution presented could potentially be used in other environments, where resource-constrained hardware platforms might be available with the needed capacity to execute virtualization containers (e.g., the Internet of Things, or IoT, environments). In any case, the applicability of this solution to different environments and its potential adaptations will require a careful study in a case-by-case basis.

Finally, it should be noted that the results presented have been obtained in a laboratory environment and with the UAV devices grounded or following a limited and well-defined flight plan. Other scenarios involving outdoor deployments may introduce conditions affecting the stability of the flight of the UAVs, and hence the performance of the IP telephony service.

ACKNOWLEDGMENTS:

This work was partially supported by the European H2020 5GRANGE project (grant agreement 777137), and by the 5GCity project (TEC2016-76795-C6-3-R) funded by the Spanish Ministry of Economy and Competitiveness. The work of Luis F. Gonzalez was partially supported by the European H2020 5GinFIRE project (grant agreement 732497).

DISCLOSURES:

The authors have nothing to disclose.

REFERENCES:

1. Sanchez-Aguero, V., Nogales, B., Valera, F., Vidal, I. Investigating the deployability of VoIP services over wireless interconnected Micro Aerial Vehicles. *Internet Technology Letters*. **1** (5), e40 (2018).
2. Maxim, V., Zidek, K. Design of high-performance multimedia control system for UAV/UGV based on SoC/FPGA Core. *Procedia Engineering*. **48**, 402-408 (2012).
3. Vidal, I. et. Al. Enabling Multi-Mission Interoperable UAS Using Data-Centric Communications. *Sensors*, **18** (10), 3421 (2018).
4. Vidal, I., Valera, F., Díaz, M. A., Bagnulo, M. Design and practical deployment of a network-centric remotely piloted aircraft system. *IEEE Communications Magazine*. **52** (10), 22-29 (2014).
5. Jin, Y., Minai, A. A., Polycarpou, M. M. Cooperative real-time search and task allocation in UAV teams. In *42nd IEEE International Conference on Decision and Control* (IEEE Cat. No. 03CH37475) (Vol. 1, pp. 7-12). IEEE (2003).
6. Maza, I., Ollero, A. Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms. In *Distributed Autonomous Robotic Systems 6* (pp. 221-230). Springer, Tokyo (2007).
7. Quaritsch, M. et al. Collaborative microdrones: applications and research challenges. In *Proceedings of the 2nd International Conference on Autonomic Computing and Communication Systems* (p. 38). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering) (2008).
8. Waharte, S., Trigoni, N., Julier, S. Coordinated search with a swarm of UAVs. In *2009 6th IEEE Annual Communications Society Conference on Sensor, Mesh and Ad Hoc Communications*

528 and Networks Workshops (pp. 1-3). IEEE (2009).

- 529 9. De Freitas, E. P. et al. UAV relay network to support WSN connectivity. In *International*
530 *Congress on Ultra-Modern Telecommunications and Control Systems* (pp. 309-314). IEEE
531 (2010).
- 532 10. European Telecommunications Standards Institute. Network Functions Virtualisation (NFV);
533 Architectural Framework; Research Report ETSI GS NFV 002 V1.2.1; *European*
534 *Telecommunications Standards Institute* (ETSI) (2014).
- 535 11. ETSI OSM. An Open Source NFV Management and Orchestration (MANO) software stack
536 aligned with ETSI NFV. Available online: <https://osm.etsi.org/> (last access on 8 August 2019).
- 537 12. Nogales, B. et al. Design and Deployment of an Open Management and Orchestration
538 Platform for Multi-Site NFV Experimentation. *IEEE Communications Magazine*. **57** (1), 20-27
539 (2019).
- 540 13. Omnes, N., Bouillon, M., Fromentoux, G., Le Grand, O. A programmable and virtualized
541 network & IT infrastructure for the internet of things: How can NFV & SDN help for facing the
542 upcoming challenges. In the *18th International Conference on Intelligence in Next Generation*
543 *Networks* (pp. 64-69). IEEE (2015).
- 544 14. Rametta, C., Schembra, G. Designing a softwarized network deployed on a fleet of drones for
545 rural zone monitoring. *Future Internet*. **9** (1), 8 (2017).
- 546 15. Garg, S., Singh, A., Batra, S., Kumar, N., Yang, L. T. UAV-empowered edge computing
547 environment for cyber-threat detection in smart vehicles. *IEEE Network*. **32** (3), 42-51 (2018).
- 548 16. Mahmoud, S., Jawhar, I., Mohamed, N., Wu, J. UAV and WSN softwarization and collaboration
549 using cloud computing. In the *3rd Smart Cloud Networks & Systems (SCNS)* (pp. 1-8). IEEE
550 (2016).
- 551 17. González Blázquez, L. F. et al. NFV orchestration on intermittently available SUAV platforms:
552 challenges and hurdles. In 1th Mission-Oriented Wireless Sensor, UAV and Robot Networking
553 (MISARN). IEEE (2019).
- 554 18. Nogales, B., Sanchez-Aguero, V., Vidal, I., Valera, F., Garcia-Reinoso, J. A NFV system to
555 support configurable and automated multi-UAV service deployments. In *Proceedings of the*
556 *4th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications* (pp. 39-44).
557 ACM (2018, June).
- 558 19. Nogales, B., Sanchez-Aguero, V., Vidal, I., Valera, F. Adaptable and automated small UAV
559 deployments via virtualization. *Sensors*. **18** (12), 4116 (2018).
- 560 20. Hoban, A. et al. An ETSI OSM Community White Paper, OSM Release FOUR: A Technical
561 Overview; Whitepaper; *European Telecommunications Standards Institute* (ETSI) (2018).
- 562 21. Open Source MANO Release FOUR. Quick start installation and use guide. Available online:
563 https://osm.etsi.org/wikipub/index.php/OSM_Release_FOUR (last access on 8 August 2019).
- 564 22. OpenStack. Open Source Software for Creating Private and Public Clouds. Available online:
565 <https://docs.openstack.org/ocata> (last access on 8 August 2019).
- 566 23. OpenStack Installation Tutorial for Ubuntu. Available online:
567 <https://docs.openstack.org/ocata/install-guide-ubuntu/> (last access on 8 August 2019).
- 568 24. Linphone. An Open Source VoIP SIP Softphone for voice/video calls and instant messaging.
569 Available online: <https://www.linphone.org> (last access on 8 August 2019).
- 570 25. Jitsi. An Open Source Project to easily build and deploy secure video-conferencing solutions.
571 Available online: <https://jitsi.org> (last access on 8 August 2019).

- 572 26. Linux Containers (LXC). Infrastructure for container projects. Available online:
573 <https://linuxcontainers.org> (last access on 8 August 2019).
- 574 27. Ns-3. A Discrete-Event Network Simulator for Internet Systems. Available online:
575 <https://www.nsnam.org/> (last access on 8 August 2019).
- 576 28. Kernel-based Virtual Machine (KVM). A virtualization solution for Linux. Available online:
577 <https://www.linux-kvm.org> (last access on 8 August 2019).
- 578 29. Bridging & firewalling. Available online: <https://wiki.linuxfoundation.org/networking/bridge>
579 (last access on 8 August 2019).
- 580 30. SIPp. An Open Source test tool and/or traffic generator for the SIP protocol. Available online:
581 <http://sipp.sourceforge.net/> (last access on 8 August 2019).
- 582 31. Trafic. An open source flow scheduler. Available online: <https://github.com/5GinFIRE/trafic>
583 (last access on 8 August 2019).
- 584 32. Ubuntu Mate for the Raspberry Pi. Available online: <https://ubuntu-mate.org/raspberry-pi/>
585 (last access on 8 August 2019).
- 586 33. OpenStack. Enabling LXC (Linux Containers) as virtualization technology. Available online:
587 <https://docs.openstack.org/ocata/config-reference/compute/hypervisor-lxc.html> (last
588 access on 8 August 2019).
- 589 34. Open Source MANO Information Model. Available online:
590 https://osm.etsi.org/wikipub/index.php/OSM_Information_Model (last access on 8 August
591 2019).
- 592 35. ITU-T Recommendation G.114. General Recommendations on the transmission quality for an
593 entire international telephone connection; One-way transmission time. *International*
594 *Telecommunication Union - Telecommunication Standardization Sector* (2003).

Figure 1

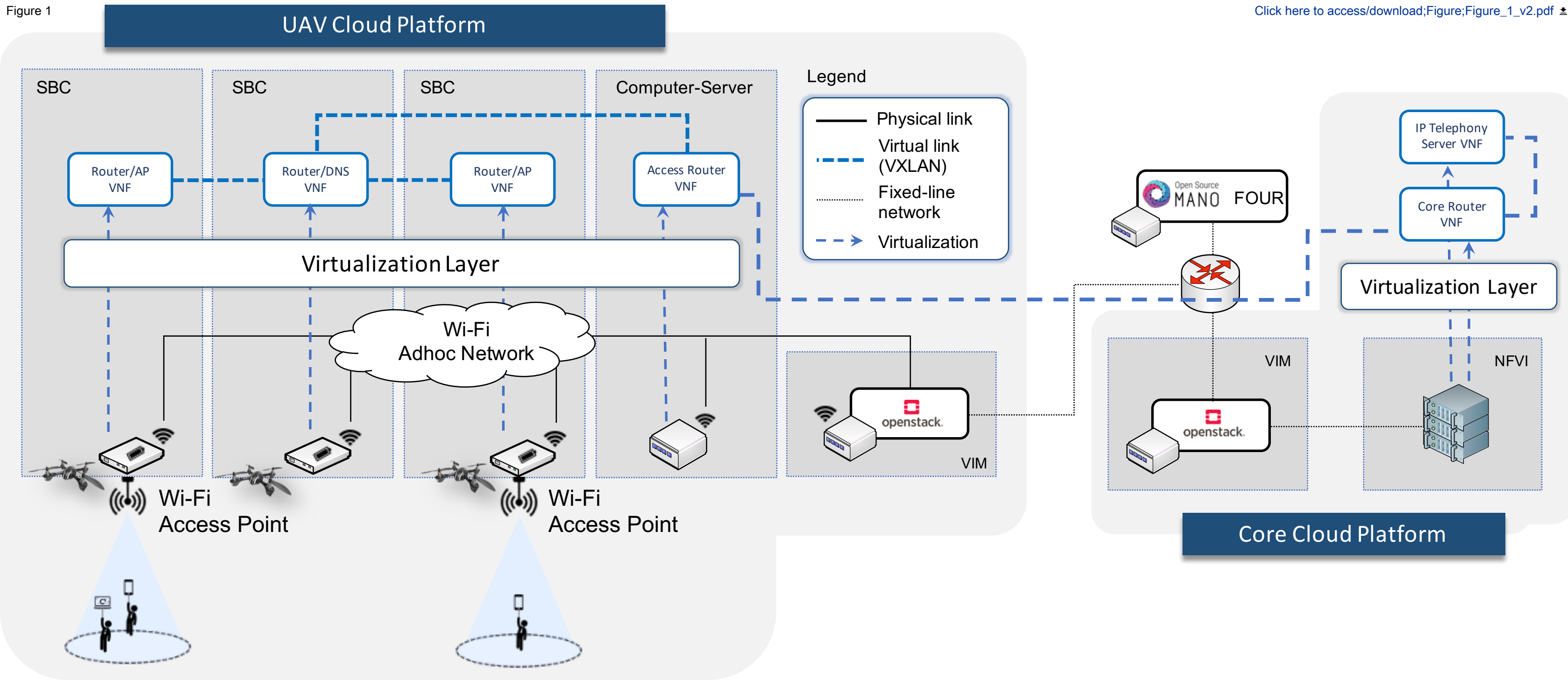


Figure 2

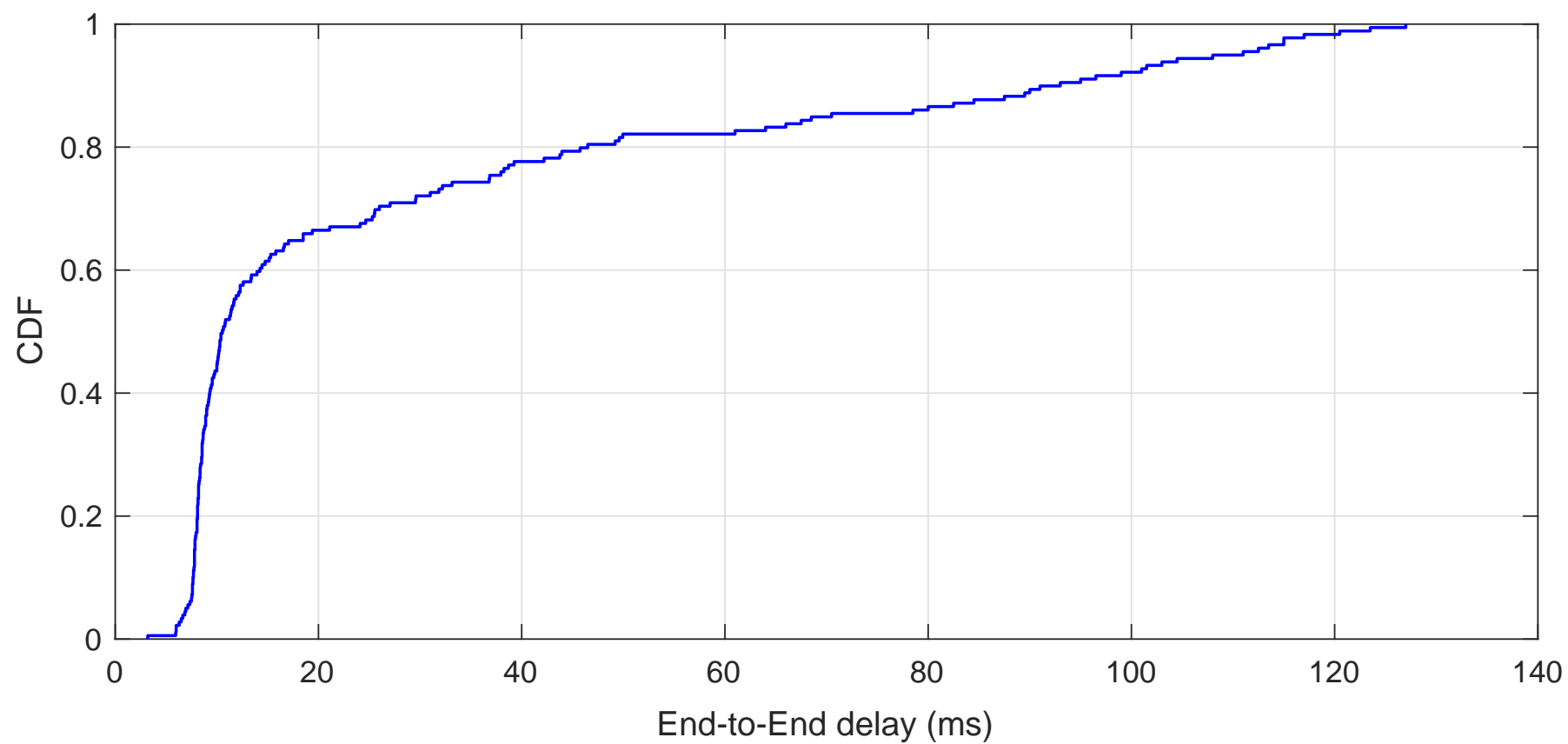


Figure 3

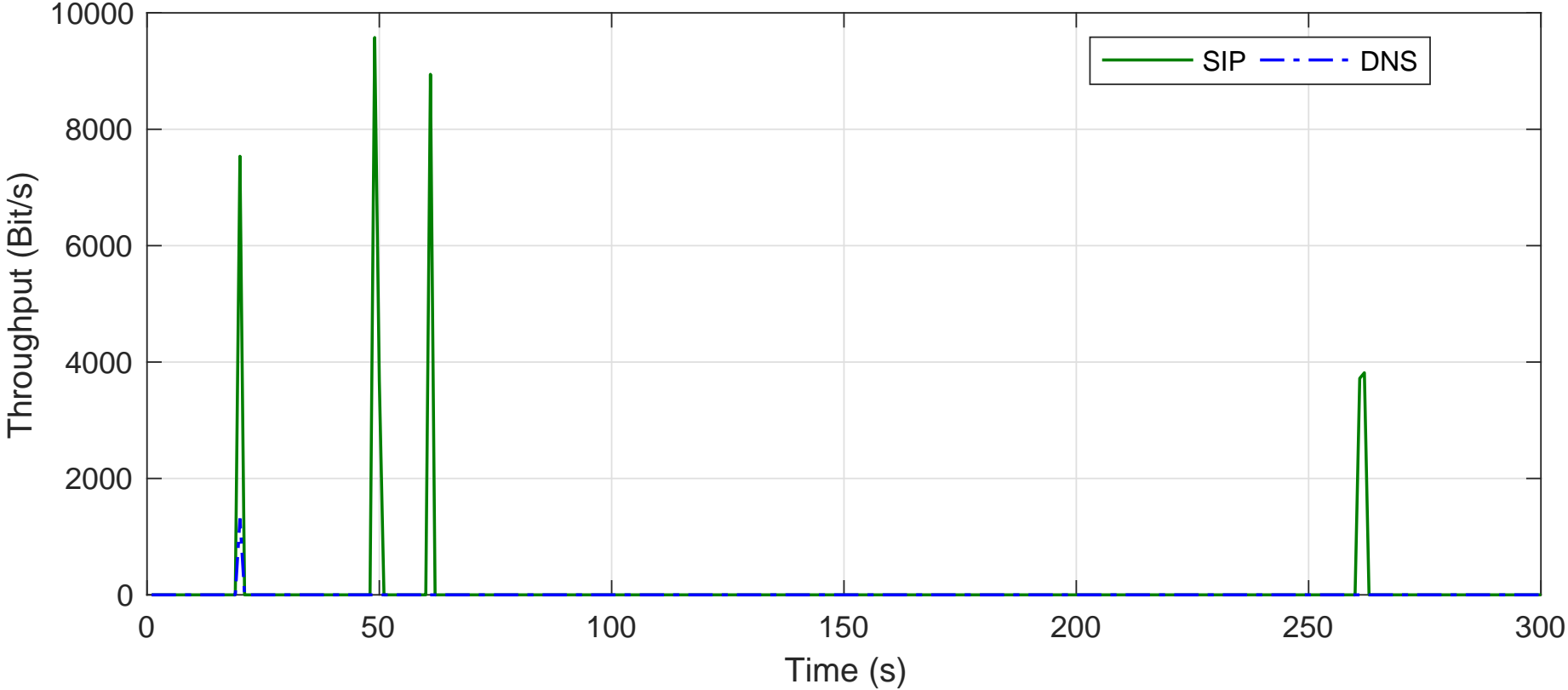


Figure 4

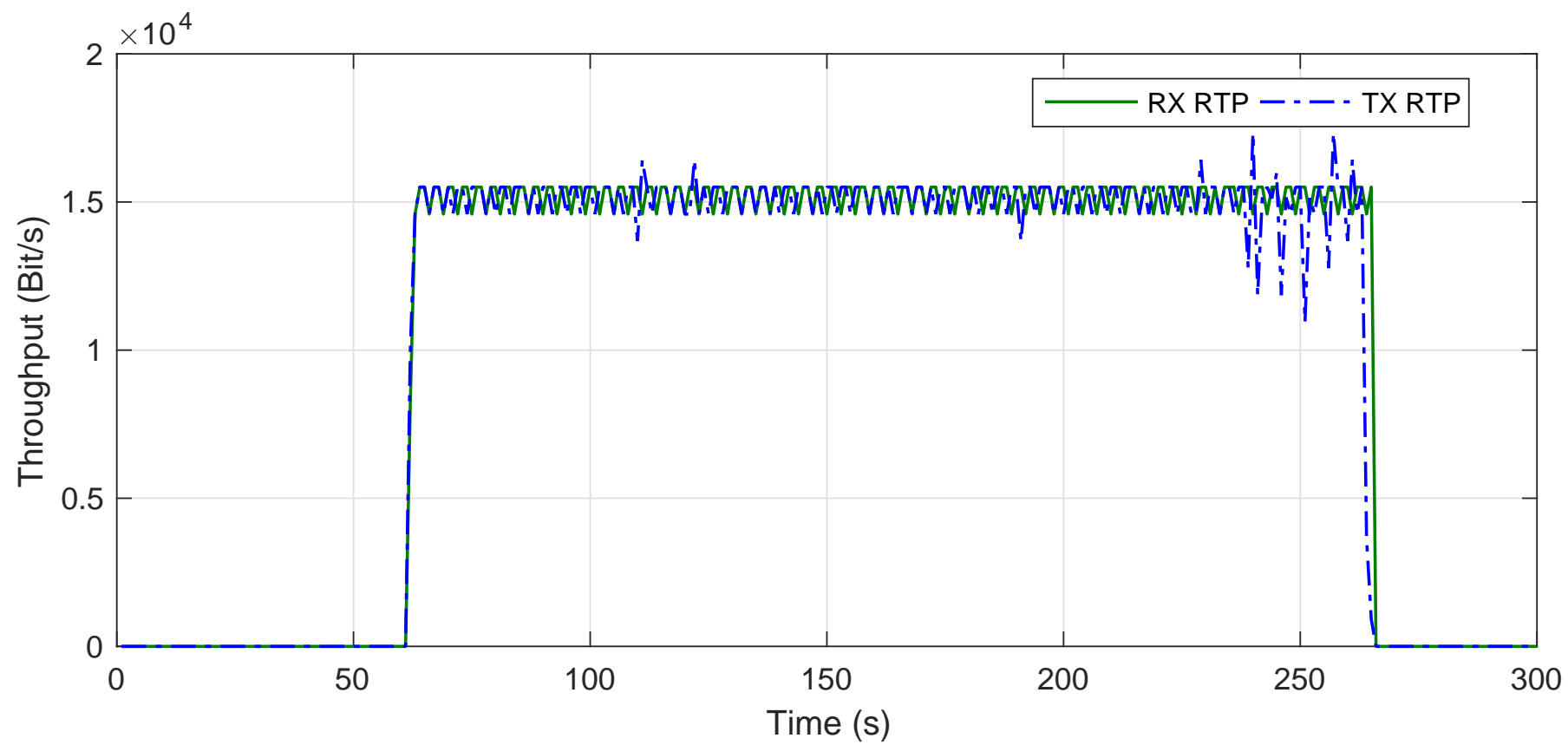
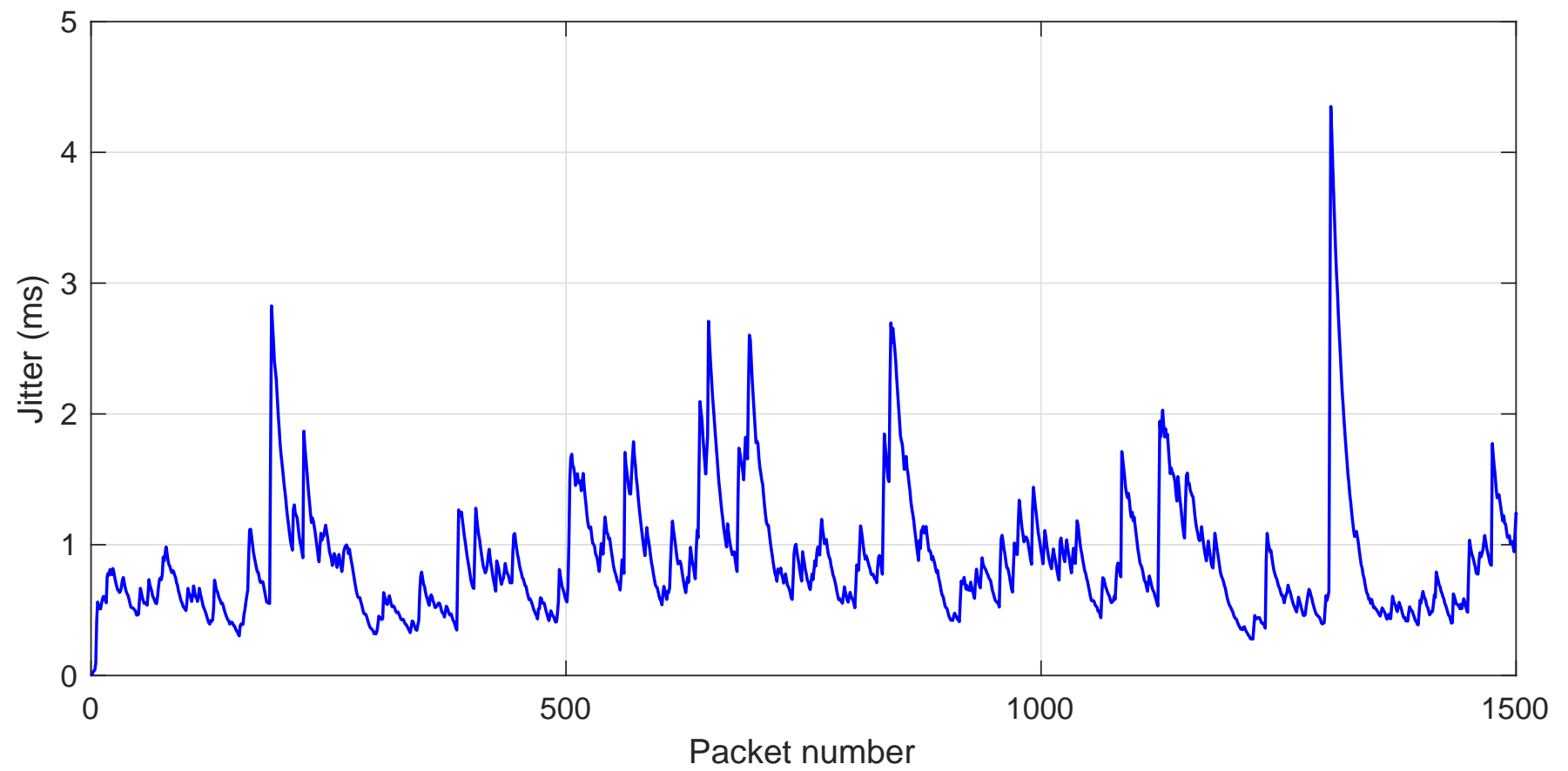


Figure 5



Name of Material/Equipment	Company	Catalog Number
AR. Drone 2.0 - Elite edition	Parrot	
Bebop 2	Parrot	
Commercial Intel Core Mini-ITX Computer	Logic Supply	
Linux Containers (LXC)	Canonical Ltd.	
Lithium Battery Pack Expansion Board. Model KY68C-UK	Kuman	
MacBook Pro	Apple	
ns-3 Network Simulator	nsnam	
Open Source MANO (OSM) - Release FOUR	ETSI OSM - Open source community	
OpenStack - Release Ocata	OpenStack - Open source community	
Ping	Open source tool	
Power Edge R430	Dell	
Power Edge R630	Dell	
Prestige 2000W	ZyXEL	
Raspberry PI. Model 3b	Raspberry Pi Foundation	
SIPp	Open source tool	
Tcpdump	Open source tool	
Traffic	Open source tool	

Comments/Description

UAV used in the experiment to transport the RPis and thus, provide mobility to the compute units of the UAV cloud platform.

UAV used in the experiment to transport the RPis and thus, provide mobility to the compute units of the UAV cloud platform.

Computer server which hosts the OpenStack controller node (being executed as a VM) of the experiment's UAV cloud platform. In addition, (Software) Virtualization technology that enables the supply of the Virtual Network Functions detailed in the experiment. Source-code available

Battery-power supply HAT (Hardware Attached on Top) for the computation units of the UAV cloud platform (i.e., the Raspberry Pis). In addition,

Commodity laptop utilized during the experiment to obtain and gather the results as described in the manuscript.

(Software) A discrete-event simulator network simulator which provides the underlying communication substrate to the emulation station

(Software) Management and Orchestration (MANO) software stack of the NFV system configured in the experiment. Source-code available

(Software) Open source software used for setting up both the UAV cloud platform and the core cloud within the experiment. Source-code available

(Software) An open source test tool, which verifies the connectivity between two devices connected through a communications network. In addition, High-profile computer server which provides the computational capacity within the core cloud platform presented in the experiment.

Equipment used for hosting the virtual machine (VM) on charge of executing the MANO stack. In addition, the OpenStack controller node is Voice over IP Wi-Fi phone, compatible with the IEEE 802.11b wireless communications standard. This device is utilized to carry out the VoIP. Selected model of Single Board Computer (SBC) used for providing the computational capacity to the experiment's UAV cloud platform.

(Software) An open source test tool, which generates SIP protocol traffic. This tool allows to verify the proper support of the signalling traffic

(Software) An open source tool that enables the capture and analysis of the network traffic. Source-code available online: <https://www.tcpdump.org/>

(Software) An open source flow scheduler that is used for validating the capacity of the network service deployed to process data traffic generated

another unit of this equipment (along with the RPis) conforms the computational resources of the UAV cloud platform.

able online: <https://linuxcontainers.org>

lition, this equipment encompasses the case used to attach the compute units (i.e., the Raspberry PIs or RPis) to the UAVs.

explained in the "Protocol" section (more specifically in the step "2. Validate the functionality of the softwarization units via Emulation"). Sou

online: https://osm.etsi.org/wikipub/index.php/OSM_Release_FOUR

vailable online: <https://docs.openstack.org/ocata/install-guide-ubuntu>

In addition, this tool allows to assess the network performance since it calculates the Round Trip Time (i.e., the time taken to send and receive

also executed as a VM in this device. Note that the use of this device is not strictly needed. The operations carried out by this device could be
call through the network service hosted by platform described for the execution of the experiment.

c required in an IP telephony service such as the one deployed in the experiment. Source-code available online: <http://sipp.sourceforge.net>
dump.org

erated during an IP telephony call. Source-code available online at: <https://github.com/5GinFIRE/traffic>

rce-code available online: <https://www.nsnam.org>

d a data packet from the network). Source-code available online: <https://packages.debian.org/es/sid/iputils-ping>

e done by a lower performance equipment due to the non-high resource specifications of the before mentioned VMs.



1 Alewife Center #200
Cambridge, MA 02140
tel. 617.945.9051
www.jove.com

ARTICLE AND VIDEO LICENSE AGREEMENT

Title of Article:

Automated deployment of an IP telephony service on Unmanned Aerial Vehicles using Network Functions Virtualization

Author(s):

Borja Nogales, Ivan Vidal, Victor Sanchez-Aguero, Francisco Valera, Luis F. Gonzalez, Arturo Azcorra

Item 1: The Author elects to have the Materials be made available (as described at <http://www.jove.com/publish>) via:

☒ Standard Access

☐ Open Access

Item 2: Please select one of the following items:

☒ The Author is **NOT** a United States government employee.

☐ The Author is a United States government employee and the Materials were prepared in the course of his or her duties as a United States government employee.

☐ The Author is a United States government employee but the Materials were NOT prepared in the course of his or her duties as a United States government employee.

ARTICLE AND VIDEO LICENSE AGREEMENT

1. **Defined Terms.** As used in this Article and Video License Agreement, the following terms shall have the following meanings: **"Agreement"** means this Article and Video License Agreement; **"Article"** means the article specified on the last page of this Agreement, including any associated materials such as texts, figures, tables, artwork, abstracts, or summaries contained therein; **"Author"** means the author who is a signatory to this Agreement; **"Collective Work"** means a work, such as a periodical issue, anthology or encyclopedia, in which the Materials in their entirety in unmodified form, along with a number of other contributions, constituting separate and independent works in themselves, are assembled into a collective whole; **"CRC License"** means the Creative Commons Attribution-Non Commercial-No Derivs 3.0 Unported Agreement, the terms and conditions of which can be found at: <http://creativecommons.org/licenses/by-nc-nd/3.0/legalcode>; **"Derivative Work"** means a work based upon the Materials or upon the Materials and other pre-existing works, such as a translation, musical arrangement, dramatization, fictionalization, motion picture version, sound recording, art reproduction, abridgment, condensation, or any other form in which the Materials may be recast, transformed, or adapted; **"Institution"** means the institution, listed on the last page of this Agreement, by which the Author was employed at the time of the creation of the Materials; **"JoVE"** means MyJoVE Corporation, a Massachusetts corporation and the publisher of The Journal of Visualized Experiments; **"Materials"** means the Article and / or the Video; **"Parties"** means the Author and JoVE; **"Video"** means any video(s) made by the Author, alone or in conjunction with any other parties, or by JoVE or its affiliates or agents, individually or in collaboration with the Author or any other parties, incorporating all or any portion

of the Article, and in which the Author may or may not appear.

2. **Background.** The Author, who is the author of the Article, in order to ensure the dissemination and protection of the Article, desires to have the JoVE publish the Article and create and transmit videos based on the Article. In furtherance of such goals, the Parties desire to memorialize in this Agreement the respective rights of each Party in and to the Article and the Video.

3. **Grant of Rights in Article.** In consideration of JoVE agreeing to publish the Article, the Author hereby grants to JoVE, subject to **Sections 4** and **7** below, the exclusive, royalty-free, perpetual (for the full term of copyright in the Article, including any extensions thereto) license (a) to publish, reproduce, distribute, display and store the Article in all forms, formats and media whether now known or hereafter developed (including without limitation in print, digital and electronic form) throughout the world, (b) to translate the Article into other languages, create adaptations, summaries or extracts of the Article or other Derivative Works (including, without limitation, the Video) or Collective Works based on all or any portion of the Article and exercise all of the rights set forth in (a) above in such translations, adaptations, summaries, extracts, Derivative Works or Collective Works and (c) to license others to do any or all of the above. The foregoing rights may be exercised in all media and formats, whether now known or hereafter devised, and include the right to make such modifications as are technically necessary to exercise the rights in other media and formats. If the "Open Access" box has been checked in **Item 1** above, JoVE and the Author hereby grant to the public all such rights in the Article as provided in, but subject to all limitations and requirements set forth in, the CRC License.

ARTICLE AND VIDEO LICENSE AGREEMENT

4. **Retention of Rights in Article.** Notwithstanding the exclusive license granted to JoVE in **Section 3** above, the Author shall, with respect to the Article, retain the non-exclusive right to use all or part of the Article for the non-commercial purpose of giving lectures, presentations or teaching classes, and to post a copy of the Article on the Institution's website or the Author's personal website, in each case provided that a link to the Article on the JoVE website is provided and notice of JoVE's copyright in the Article is included. All non-copyright intellectual property rights in and to the Article, such as patent rights, shall remain with the Author.

5. **Grant of Rights in Video – Standard Access.** This **Section 5** applies if the "Standard Access" box has been checked in **Item 1** above or if no box has been checked in **Item 1** above. In consideration of JoVE agreeing to produce, display or otherwise assist with the Video, the Author hereby acknowledges and agrees that, Subject to **Section 7** below, JoVE is and shall be the sole and exclusive owner of all rights of any nature, including, without limitation, all copyrights, in and to the Video. To the extent that, by law, the Author is deemed, now or at any time in the future, to have any rights of any nature in or to the Video, the Author hereby disclaims all such rights and transfers all such rights to JoVE.

6. **Grant of Rights in Video – Open Access.** This **Section 6** applies only if the "Open Access" box has been checked in **Item 1** above. In consideration of JoVE agreeing to produce, display or otherwise assist with the Video, the Author hereby grants to JoVE, subject to **Section 7** below, the exclusive, royalty-free, perpetual (for the full term of copyright in the Article, including any extensions thereto) license (a) to publish, reproduce, distribute, display and store the Video in all forms, formats and media whether now known or hereafter developed (including without limitation in print, digital and electronic form) throughout the world, (b) to translate the Video into other languages, create adaptations, summaries or extracts of the Video or other Derivative Works or Collective Works based on all or any portion of the Video and exercise all of the rights set forth in (a) above in such translations, adaptations, summaries, extracts, Derivative Works or Collective Works and (c) to license others to do any or all of the above. The foregoing rights may be exercised in all media and formats, whether now known or hereafter devised, and include the right to make such modifications as are technically necessary to exercise the rights in other media and formats. For any Video to which this **Section 6** is applicable, JoVE and the Author hereby grant to the public all such rights in the Video as provided in, but subject to all limitations and requirements set forth in, the CRC License.

7. **Government Employees.** If the Author is a United States government employee and the Article was prepared in the course of his or her duties as a United States government employee, as indicated in **Item 2** above, and any of the licenses or grants granted by the Author hereunder exceed the scope of the 17 U.S.C. 403, then the rights granted hereunder shall be limited to the maximum

rights permitted under such statute. In such case, all provisions contained herein that are not in conflict with such statute shall remain in full force and effect, and all provisions contained herein that do so conflict shall be deemed to be amended so as to provide to JoVE the maximum rights permissible within such statute.

8. **Protection of the Work.** The Author(s) authorize JoVE to take steps in the Author(s) name and on their behalf if JoVE believes some third party could be infringing or might infringe the copyright of either the Author's Article and/or Video.

9. **Likeness, Privacy, Personality.** The Author hereby grants JoVE the right to use the Author's name, voice, likeness, picture, photograph, image, biography and performance in any way, commercial or otherwise, in connection with the Materials and the sale, promotion and distribution thereof. The Author hereby waives any and all rights he or she may have, relating to his or her appearance in the Video or otherwise relating to the Materials, under all applicable privacy, likeness, personality or similar laws.

10. **Author Warranties.** The Author represents and warrants that the Article is original, that it has not been published, that the copyright interest is owned by the Author (or, if more than one author is listed at the beginning of this Agreement, by such authors collectively) and has not been assigned, licensed, or otherwise transferred to any other party. The Author represents and warrants that the author(s) listed at the top of this Agreement are the only authors of the Materials. If more than one author is listed at the top of this Agreement and if any such author has not entered into a separate Article and Video License Agreement with JoVE relating to the Materials, the Author represents and warrants that the Author has been authorized by each of the other such authors to execute this Agreement on his or her behalf and to bind him or her with respect to the terms of this Agreement as if each of them had been a party hereto as an Author. The Author warrants that the use, reproduction, distribution, public or private performance or display, and/or modification of all or any portion of the Materials does not and will not violate, infringe and/or misappropriate the patent, trademark, intellectual property or other rights of any third party. The Author represents and warrants that it has and will continue to comply with all government, institutional and other regulations, including, without limitation all institutional, laboratory, hospital, ethical, human and animal treatment, privacy, and all other rules, regulations, laws, procedures or guidelines, applicable to the Materials, and that all research involving human and animal subjects has been approved by the Author's relevant institutional review board.

11. **JoVE Discretion.** If the Author requests the assistance of JoVE in producing the Video in the Author's facility, the Author shall ensure that the presence of JoVE employees, agents or independent contractors is in accordance with the relevant regulations of the Author's institution. If more than one author is listed at the beginning of this Agreement, JoVE may, in its sole

ARTICLE AND VIDEO LICENSE AGREEMENT

discretion, elect not take any action with respect to the Article until such time as it has received complete, executed Article and Video License Agreements from each such author. JoVE reserves the right, in its absolute and sole discretion and without giving any reason therefore, to accept or decline any work submitted to JoVE. JoVE and its employees, agents and independent contractors shall have full, unfettered access to the facilities of the Author or of the Author's institution as necessary to make the Video, whether actually published or not. JoVE has sole discretion as to the method of making and publishing the Materials, including, without limitation, to all decisions regarding editing, lighting, filming, timing of publication, if any, length, quality, content and the like.

12. **Indemnification.** The Author agrees to indemnify JoVE and/or its successors and assigns from and against any and all claims, costs, and expenses, including attorney's fees, arising out of any breach of any warranty or other representations contained herein. The Author further agrees to indemnify and hold harmless JoVE from and against any and all claims, costs, and expenses, including attorney's fees, resulting from the breach by the Author of any representation or warranty contained herein or from allegations or instances of violation of intellectual property rights, damage to the Author's or the Author's institution's facilities, fraud, libel, defamation, research, equipment, experiments, property damage, personal injury, violations of institutional, laboratory, hospital, ethical, human and animal treatment, privacy or other rules, regulations, laws, procedures or guidelines, liabilities and other losses or damages related in any way to the submission of work to JoVE, making of videos by JoVE, or publication in JoVE or elsewhere by JoVE. The Author shall be responsible for, and shall hold JoVE harmless from, damages caused by lack of sterilization, lack of cleanliness or by contamination due to

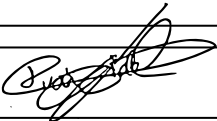
the making of a video by JoVE its employees, agents or independent contractors. All sterilization, cleanliness or decontamination procedures shall be solely the responsibility of the Author and shall be undertaken at the Author's expense. All indemnifications provided herein shall include JoVE's attorney's fees and costs related to said losses or damages. Such indemnification and holding harmless shall include such losses or damages incurred by, or in connection with, acts or omissions of JoVE, its employees, agents or independent contractors.

13. **Fees.** To cover the cost incurred for publication, JoVE must receive payment before production and publication of the Materials. Payment is due in 21 days of invoice. Should the Materials not be published due to an editorial or production decision, these funds will be returned to the Author. Withdrawal by the Author of any submitted Materials after final peer review approval will result in a US\$1,200 fee to cover pre-production expenses incurred by JoVE. If payment is not received by the completion of filming, production and publication of the Materials will be suspended until payment is received.

14. **Transfer, Governing Law.** This Agreement may be assigned by JoVE and shall inure to the benefits of any of JoVE's successors and assignees. This Agreement shall be governed and construed by the internal laws of the Commonwealth of Massachusetts without giving effect to any conflict of law provision thereunder. This Agreement may be executed in counterparts, each of which shall be deemed an original, but all of which together shall be deemed to be one and the same agreement. A signed copy of this Agreement delivered by facsimile, e-mail or other means of electronic transmission shall be deemed to have the same legal effect as delivery of an original signed copy of this Agreement.

A signed copy of this document must be sent with all new submissions. Only one Agreement is required per submission.

CORRESPONDING AUTHOR

Name:	Iván Vidal Fernández	
Department:	Telematic Engineering	
Institution:	University Carlos III of Madrid	
Title:	Dr.	
Signature:		Date: 14th June, 2019

Please submit a **signed** and **dated** copy of this license by one of the following three methods:

1. Upload an electronic version on the JoVE submission site
2. Fax the document to +1.866.381.2236
3. Mail the document to JoVE / Attn: JoVE Editorial / 1 Alewife Center #200 / Cambridge, MA 02140

Comments and changes done to the paper “Automated deployment of an IP telephony service on Unmanned Aerial Vehicles using Network Functions Virtualization”

We would like to take this opportunity to thank the feedback from the reviewers. By providing valuable comments and highlighting the issues with our paper, we were able to improve its value. We hope that the enhancements and modifications are able to satisfy the reviewer's comments.

We present below the detailed comments provided by each reviewer, followed by our answer, making references to the changes done to the paper. In the online submission system, we have uploaded the updated paper with the main changes highlighted in red color.

Yours sincerely, the authors.

Editorial #E

Comment E.1. Please take this opportunity to thoroughly proofread the manuscript to ensure that there are no spelling or grammar issues.

Answer E.1. *Following the editorial suggestion, we have done a detailed proofread of the whole paper.*

Comment E.2. JoVE cannot publish manuscripts containing commercial language. This includes trademark symbols (TM), registered symbols (®), and company names before an instrument or reagent. Please limit the use of commercial language from your manuscript and use generic terms instead. All commercial products should be sufficiently referenced in the Table of Materials and Reagents. For example: ZyXEL (although, note that Ubuntu is OK).

Answer E.2. *Attending to the editorial comment, the new version of the manuscript limits the use of the commercial language. In addition, we have elaborated the suggested Table of Materials, which includes the proper references to the commercial equipment utilized to carry out the experiment described in the manuscript.*

Comment E.3 Please ensure every protocol step/substep has at least one action written in the imperative.

Answer E.3. *Taking into account the editorial suggestion, we have done the corresponding modifications to the “Protocol” section to ensure the imperative voice in every step/substep of this section.*

Comment E.4 For each protocol step/substep, please ensure you answer the “how” question, i.e., how is the step performed? Alternatively, add references to published material specifying

how to perform the protocol action. If revisions cause a step to have more than 2-3 actions and 4 sentences per step, please split into separate steps or substeps.

Answer E.4. *According to the editor's comment, we have reviewed the "Protocol" section to ensure that the procedure to perform every step is properly reflected. To this purpose, we have enhanced the text including additional links in which this information is presented in detail.*

Comment E.5. Please ensure the Table of Materials has information on all materials and equipment used (e.g., computers, software, UAVs), especially those mentioned in the Protocol.

Answer E.5. *The elaboration of the suggested Table of Materials, including the equipment used in the experiment, has been addressed in the comment E.2. Taking into account the editorial comment, the "Protocol" section has been modified to better comply with the manuscript instructions.*

Reviewer #1

Comment 1.1. The protocol was evaluated under emulation and also using a real UAV cluster. The authors were very clear stating that the performance results were achieved with fixed/non-flying UAVs. Although the reviewer acknowledges that the protocol's explanation does not depend on the flying status of each UAV, the link quality in a flying procedure, and its impact on the management of the VNFs and on the selected application/use case, might be of critical importance for the execution of the presented protocol

Answer 1.1. *We agree with the reviewer that the quality of the wireless link is a relevant aspect in our work as a whole. Nevertheless, this protocol has a main focus on describing the specific methodology that allows the automated deployment of moderately complex network services on UAVs, using NFV standard technologies to that end. For this reason, the flight procedure is considered as optional within the manuscript, and the performance results have been obtained with the devices on the ground. This way, the results could be easily replicated by third-parties, using SBCs in a controlled laboratory environment.*

Regardless, our work takes into account the aspects related to the wireless links quality. In this context, the protocol includes an optional section ("Validate the functionality of the softwarization units via Emulation"), which aims at demonstrating the appropriate operation of the deployed network services under realistic wireless communication link conditions. To this end, we have developed an emulation platform that allows emulating the multi-hop aerial links and to define the characteristics of those links (e.g., length of the wireless communication links, pattern of data packet losses, the radio technology used in the wireless communications, etc.).

In any case, we agree that the scope of the article may have not been properly stated. We have enhanced section “Introduction” to better reflect the main focus of the protocol, i.e., to enable the reproducibility of a protocol that integrates the UAVs into an NFV environment, to allow the automated deployment of network services on the UAVs.

Reviewer #2

Comment 2.1 The study consists of significant amount of work and trials in order to provide NFV platform on top an aerial network. However, it lacks certain elements regarding the challenges which an aerial/UAV network may pose. In the submitted version, it is hard to grasp what the critical challenges are when we specifically target UAV-platform, since it looks like this framework may be migrated to any mobile or immobile environment which can carry a RPi. The authors should emphasize the unique challenges (in a UAV network) and their unique decisions/solutions which are designed to address these challenges.

Answer 2.1. *Our work aims at providing the UAVs with the capacity to flexibly adapt their functionalities to different mission objectives, through the use of the standard NFV technologies. In addition, the integration of the UAVs into the NFV environment makes possible the coordinated operation of the UAVs to allow the deployment of moderately complex services and applications. Notwithstanding the benefits, the realization of this view presents a set of fundamental challenges that needs to be carefully addressed, such as: the integration of these devices as part of an NFV infrastructure, using an existing NFV software stack, so that an NFV orchestration service can deploy virtual functions on the UAVs; the constraints in terms of the computational resources provided by the UAV, which may present limitations in terms of the the size, weight and computing capacity of payload equipment; the proper placement of the virtual functions onto UAVs, i.e., selecting the best UAV candidate to deploy a particular virtual function; the maintenance of the control communications with the UAVs, in order to manage the lifecycle of the VNFs despite of the potentially intermittent availability of network communications with them (e.g., caused by mobility and battery constraints); the limited operation time of the UAVs, due to their battery consumption; and, related to the latest, the migration of the virtual functions when a UAV needs to be replaced due to battery exhaustion. These benefits and challenges are detailed in our previous work [1, 2], which include the design of an NFV system capable of supporting the automated deployment of network functions and services on UAV platforms, as well as the validation of the practical feasibility of this design.*

Attending to the reviewer’s comment, we have highlighted these benefits and challenges in the section “Introduction” of the manuscript, with the appropriate references to our prior work.

In addition, we agree with the reviewer that our solution could potentially be exported to other environments, where resource-constrained hardware platforms might be available with the needed capacity to execute virtualization containers (e.g., Internet of Things, or IoT, environments). Of course, the applicability of our solution to different environments and its potential adaptations will require a carefully study in a case-by-case basis. We have updated

the “Discussion” section of the paper to include this aspect of exporting our solution to other environments as a future application of the method.

Comment 2.2. The authors stated that NS-3 simulation has been used to emulate multi-hop aerial links and provided results based on simple topology which consists of grounded drones. It is hard to clear (considering Figure 1) which parts of the described system actually physically implemented and which parts are just emulated via NS-3. I suggest the authors to improve the distinction between these two parts.

Answer 2.2. *The article has a main focus on describing a specific procedure to enable the automated deployment of network services over a network of UAVs, using the NFV standards and open source technologies. In this specific work, real flight is considered as optional, and performance results are obtained with the devices on the ground, as highlighted by the reviewer. This will facilitate interested readers to replicate and validate the execution of this procedure even in a controlled laboratory environment.*

Regarding the physical implementation, we have developed all the components of the network service depicted in the Figure 1 (i.e., the VNFs described in the manuscript composing the IP telephony service) using Linux Containers. Hence, these components can be automatically deployed by the MANO platform.

On the other hand, our work also considers operational and real deployment aspects, such as wireless link quality. To address these aspects, we have used a purpose-specific emulator, based on ns-3, which allows emulating multi-hop aerial links and defining the characteristics of those links (e.g., length of the wireless communication links, pattern of data packet losses, the radio technology used in the wireless communications, etc.). This emulator is under development as one of our current research works. The protocol specified in the paper includes an optional section (“Validate the functionality of the softwarization units via Emulation”) that uses this emulator, aiming to demonstrate the appropriate operation of the deployed network services under realistic wireless communication link conditions.

For the emulation, we created Linux containers to represent the different physical components of the testbed, i.e., the SBCs and the fixed compute server for the UAV cloud platform, and the compute server for the core cloud platform. The functions provided by the different VNFs of the IP telephony service (i.e., access points, routers, DNS server, and IP telephony server) were also deployed as Linux containers over their corresponding emulated SBCs and compute servers. With this setup, our ns-3 based emulator is able to provide the realistic wireless communications that take place in our scenario (i.e., the Wi-Fi ad-hoc network, which enables the data exchange among the nodes of the UAV cloud platform; and the wireless networks offered by the two Wi-Fi access points provided in the service). The emulation platform, along with all the aforementioned components, is provided as a virtual machine in the public repository, and it is used in the section “Validate the functionality of the softwarization units via Emulation”.

In any case, we agree with the reviewer that this distinction, between the physical implementation and the components implemented for the emulation, can be improved in the manuscript. To address this issue, we have modified “Protocol” section (in particular, the

information included in the step “2. Validate the functionality of the softwarization units via Emulation”) in order to better clarify all these aspects, according to our answer to this comment.

Comment 2.3. The detailed information about attaching RPi on drones (Parrot AR Drone 2.0 or Parrot Bebop 2) is missing. Is there any additional connection between drone and RPi besides the power link which enables to integrate flight and communication/network data?

Answer 2.3. *Following the reviewer’s comment, we have modified the explanatory note of the step 1.5 and the step 1.5 itself of the “Protocol” section, to better clarify how the SBCs are attached to the UAVs.*

In our prototype implementation, the role of the UAVs is only to transport the RPis (which provide the compute/network/storage resources needed to execute network services, as payload. Each RPi is attached to a UAV through a fixing accessory, without requiring any other connection between the RPi and the UAV (e.g., ethernet, USB, etc.). Additionally, the RPis are provided with a battery-power supply HAT (Hardware Attached on Top) that ensures the operation of the RPis, independently of the UAV they are attached to.

Besides, in our prototype we assume that control communications with the UAVs (e.g., commands to remotely manage the operations of the UAV, read the information about the status of the battery, or the verification of the proper operation of the rotors), if needed, could be done through the integrated Wi-Fi interface of each UAV. Hence, UAVs can be controlled from a ground control station using the Wi-Fi Ad-hoc network built by all the UAVs. In addition, a flight-control VNF running at each UAV could be used to aid control operations [1, 2].

Comment 2.4 The submitted manuscript has a 87 percent similarity score with another publication authored by the same researcher. However, this manuscript has not been cited in this manuscript. If there are certain similarities, please make sure to remove them or at least cite that publication to avoid any self-plagiarism.

Answer 2.4. *The manuscript has been written without including text or content extracted from any other of our publications. However, we understand that there might be certain degree of similarity with our previous work, since the developed experiment is a re-elaboration of the one described in [2]. In any case, the submitted paper is a unique manuscript, which the bulk of itself is a step-by-step methods section, aiming at enabling the reproducibility of the mentioned experiment. Nevertheless, to verify the similarity of our paper with the existing literature, we have utilized the Turnitin tool (a well-known plagiarism detection solution) in order to verify the similarity degree with our previous work. According to the report originated by Turnitin, which is attached to this rebuttal letter for convenience, the similarity percentage obtained is 7% (as is reasonable, excluding the references section of the manuscript).*

In any case, and following the reviewer’s suggestion, we have modified the “Introduction” section to better emphasize that the experiment is a re-elaboration of the one described in [2].

Comment 2.5. Given public repository is not accessible (<http://vm-images.netcom.it.uc3m.es/jove/>).

Answer 2.5. *We appreciate the reviewer has notified this spelling mistake. We have corrected in the paper the erroneous URL of the public repository, which is: <http://vm-images.netcom.it.uc3m.es/JoVE/>*

Comment 2.6. I suggest the authors to improve the Figure 1. In the given form, it is hard to understand the hierarchy in the system and mapping between the physical entities and the deployed functions. Please also include more detail about the links (rather than just Physical and Virtual), what are the access technologies utilized between devices to create the described platform.

Answer 2.6. *Following the reviewer's suggestion, we have updated the particular figure to better clarify the mapping between the physical entities and the deployed functions. We agree with the reviewer that this information was not conveniently presented in the previous figure, especially in the case of the UAV cloud platform where not all the UAVs have a secondary wireless interface that enable the execution of the wireless access point functions. Finally, and taking into account the reviewer's comment, we have included in the figure the access technologies used to provide the VoIP service over the network of UAVs.*

References

1. Nogales, B., Sanchez-Aguero, V., Vidal, I., Valera, F., & Garcia-Reinoso, J. 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 (pp. 39-44). ACM (2018, June).
2. Nogales, B., Sanchez-Aguero, V., Vidal, I., & Valera, F. Adaptable and automated small UAV deployments via virtualization. *Sensors*, 18(12), 4116 (2018).

Turnitin Originality Report

- Processed on: 31-Jul-2019 12:15 CEST
- ID: 1156470703
- Word Count: 5302
- Submitted: 1

Borja JoVE By BORJA NOGALES DORADO

Similarity Index

7%

Similarity by Source

Internet Sources:

5%

Publications:

5%

Student Papers:

3%

1% match (Internet from 29-Apr-2019)

https://e-archivo.uc3m.es/bitstream/handle/10016/28314/NFV_DRONET_2018_ps.pdf?isAllowed=y&sequence=1

1% match (Internet from 29-Apr-2019)

https://e-archivo.uc3m.es/bitstream/handle/10016/28316/adaptable_SENSORS_2018.pdf?isAllowed=y&sequence=1

1% match (Internet from 21-May-2019)

https://res.mdpi.com/sensors/sensors-18-03421/article_deploy/sensors-18-03421.pdf?attachment=1&filename=

< 1% match (publications)

[Borja Nogales, Victor Sanchez-Aguero, Ivan Vidal, Francisco Valera, Jaime Garcia-Reinoso. "A NFV system to support configurable and automated multi-UAV service deployments", Proceedings of the 4th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications - DroNet'18, 2018](#)

< 1% match (publications)

[J. Klaue, A. Hess. "On the Impact of IPsec on Interactive Communications", 19th IEEE International Parallel and Distributed Processing Symposium, 2005](#)

< 1% match (Internet from 09-Jun-2019)

<https://www.frontiersin.org/articles/10.3389/fict.2015.00006/full>

< 1% match (publications)

["Framework for Cognitive Radio Deployment in Large Scale WSN-UAV Surveillance", Studies in Systems Decision and Control, 2016.](#)

< 1% match (Internet from 19-Jul-2019)

<https://www.inderscience.com/info/inarticle.php?artid=48313>

< 1% match (Internet from 08-Jul-2019)

<http://openaccess.city.ac.uk/17219/1/1570344524.pdf>

< 1% match (publications)

[Thomas Magedanz. "Cross-fertilization of IMS and IPTV services over NGN", 2008 First ITU-T Kaleidoscope Academic Conference - Innovations in NGN Future Network and Services, 05/2008](#)

< 1% match (Internet from 03-Jun-2016)

http://upcommons.upc.edu/bitstream/handle/2099.1/6655/PFM_PaolaGarfias.pdf;jsessionid=4CE0F2CF44FB128B8D20B49318DFA224?sequence=1

< 1% match (publications)

[Borja Nogales, Ivan Vidal, Diego R. Lopez, Juan Rodriguez, Jaime Garcia-Reinoso, Arturo Azcorra. "Design and Deployment of an Open Management and Orchestration Platform for Multi-Site NFV Experimentation", IEEE Communications Magazine, 2019](#)

< 1% match (Internet from 21-May-2019)

<https://onlinelibrary.wiley.com/doi/pdf/10.1002/itl2.40>

< 1% match (Internet from 29-Jun-2019)

<https://frinx.io/news/frinx-on-osn-meet-up-presenting-workflow-inventory-and-network-control.html>

< 1% match (Internet from 31-May-2019)

<http://portal.adtran.com/web/fileDownload/doc/33200>

< 1% match (Internet from 16-Apr-2018)

http://oatao.univ-toulouse.fr/19492/1/MA_Xiaoyan.pdf

< 1% match (publications)

[Jaime García-Reinoso, Iván Vidal, Francisco Valera, Arturo Azcorra. "Zero config residential gateway experiences for next generation smart homes", Computer Networks, 2009](#)

< 1% match (student papers from 09-Nov-2007)

[Submitted to Collin County Community College on 2007-11-09](#)

< 1% match (Internet from 22-Apr-2019)

http://dione.lib.unipi.gr/xmlui/bitstream/handle/unipi/11937/Kapridakis_ME1547.pdf?isAllowed=y&sequence=8

< 1% match (Internet from 29-Jun-2019)

https://torsec.github.io/shield-h2020/documents/project-deliverables/SHIELD_D3.3_vNSF_framework_ready_for_experiments_v1.0.pdf

< 1% match (Internet from 14-Nov-2018)

<http://nadiemellamagallina.com/es/manuales/otras-gu-as-109/drones-377>

< 1% match (Internet from 21-Nov-2017)

<http://www.actapress.com/PDFViewer.aspx?paperId=24767>

< 1% match (publications)

[Communications in Computer and Information Science, 2015.](#)

< 1% match (student papers from 20-Sep-2018)

[Submitted to Universidad Carlos III de Madrid on 2018-09-20](#)

< 1% match (student papers from 14-Apr-2018)

Submitted to King's College on 2018-04-14

TITLE: Automated deployment of an IP telephony service on Unmanned Aerial Vehicles using Network Functions Virtualization AUTHORS AND AFFILIATIONS: Borja Nogales^{1,*}, Ivan Vidal^{1,*}, Victor Sanchez-Aguero^{1,2,*}, Francisco Valera^{1,*}, Luis F. Gonzalez^{1,*}, Arturo Azcorra^{1,2,*} 1Department of Telematic Engineering, University Carlos III of Madrid, Madrid, Spain 2IMDEA Networks Institute, Madrid, Spain *These authors contributed equally. Email addresses of co-authors: Borja Nogales (bdorado@pa.uc3m.es) Victor Sanchez-Aguero (victor.sanchez@imdea.org) Francisco Valera (fvalera@it.uc3m.es) Luis F. Gonzalez (luisfgon@it.uc3m.es) Arturo Azcorra (azcorra@it.uc3m.es) Corresponding author: Ivan Vidal (ividal@it.uc3m.es) KEYWORDS: Unmanned Aerial Vehicles (UAVs), Network Functions Virtualization (NFV), Management and Orchestration (MANO), Cloud computing platform, Virtual Network Function (VNF), IP telephony service, open source, 5G. SUMMARY: The objective of the described protocol is twofold: to configure an NFV environment using unmanned aerial vehicles, as computational entities providing the underlying substrate to execute virtualized network functions; and to use this environment to support the automated deployment of a functional IP telephony service over the aerial vehicles. ABSTRACT: The Network Function Virtualization (NFV) paradigm has currently consolidated as one of the key enabling technologies in the development of the 5th Generation of mobile networks. This technology aims at alleviating the hardware dependencies in the provision of network functions and services, using virtualization techniques that allow the softwarization of those functionalities over an abstraction layer. In this context, there is an increasing research interest in exploring the potential of unmanned aerial vehicles, or UAVs, to offer a flexible platform capable of enabling cost-effective NFV operations over delimited geographic areas. To demonstrate the practical feasibility of utilizing NFV technologies in UAV platforms, we present a protocol to set up a functional NFV environment, based on open-source technologies, in which a set of small UAVs supply the computational resources that support the deployment of moderately complex network services. Then the protocol details the different steps needed to support the automated deployment of an IP telephony service over a network of interconnected UAVs, leveraging the capacities of the configured NFV environment. Our experimentation results demonstrate the proper operation of the service after its deployment. We want to highlight that, although the protocol focuses on a specific type of network service (i.e., IP telephony), the described steps may serve as a general guide to deploy other type of network services. On the other hand, the protocol description considers concrete equipment and software to set up the NFV environment (e.g., specific single board computers and open source software). The utilization of other hardware and software platforms may be feasible, although the specific configuration aspect of the NFV environment and the service deployment may present variations with respect to those described in the protocol. INTRODUCTION: One of the most promising ambitions within the new era of the mobile communications (most commonly known as the 5th mobile generation or 5G) is to be able to provide robust Information technology services even in situations where the primary telecommunications infrastructure may not be available due to, for example, an emergency situation. In this

context, the unmanned aerial vehicles (UAVs) [are receiving](#) increasing [attention](#) from [the research community due to their](#) inherent versatility. There are numerous works that use this type of devices as a cornerstone in the provision of a large variety of services. For instance, the literature has analyzed the capacity of these devices to build an aerial communication infrastructure to accommodate multimedia services¹⁻³. Furthermore, prior research has presented how the cooperation among several UAVs can extend the functionality of different communication services such as surveillance⁴, collaborative search and rescue⁵⁻⁸, or agribusiness⁹. On the other hand, the Network Functions Virtualization (NFV) technology has acquired a great significance within the telecom operators as one of the 5G key-enablers. NFV represents a paradigmatic change regarding the telecommunications infrastructure, by alleviating the current dependency of the network appliances on specialized hardware through the softwarization of the network functionalities. This enables a flexible and agile deployment of new types of communication services. To this purpose, [the European Telecommunications Standards Institute \(ETSI\)](#) formed a specification [group](#) to define [the NFV](#) architectural framework¹⁰. Additionally, the ETSI currently hosts the [Open Source Mano \(OSM\)](#) group¹¹, [which is](#) in charge of developing an [NFV Management and Orchestration \(MANO\) software stack](#) aligned with [the definition of the](#) ETSI NFV architectural framework. Given all the aforementioned considerations, the synergic convergence between UAVs and NFV technologies is currently being studied as a promising enabler in the development of novel network applications and services. This is illustrated by several research works in the literature, which point out the advantages of these types of systems¹⁴⁻¹⁶, identify the challenges of this convergence and the missing aspects, highlighting future research lines on this topic¹⁷, and present pioneer solutions based on open source technologies. Regarding the latter, our prior work^{18,19} includes the [design of an NFV system capable of supporting the automated deployment of network functions and services](#) on UAV platforms, as well as the validation of the practical feasibility of this design. In this context, the objective of this paper is to showcase a protocol that enables the integration of UAVs in an NFV environment, and uses this environment to support the automated [deployment of a real network service](#), in particular [an IP telephony service](#). To this purpose, different softwarization units (categorized within the NFV paradigm as Virtual Network Functions or VNFs) are defined to compose the mentioned service:

- Access Point VNF (AP-VNF): this VNF provides a Wi-Fi access point to end-user equipment (i.e., IP phones in our experiment).
- IP telephony server VNF (IP-telephony-server-VNF): it is responsible of managing the call signaling messages that are exchanged between IP phones to establish and terminate a voice call.
- Domain Name System VNF (DNS-VNF): this VNF provides a name resolution service, which is typically needed in IP telephony services.
- Access router VNF (AR-VNF): provides network routing functionalities, supporting the exchange of traffic (i.e., call signaling in our experiment) between the IP phones and the telecommunication operator domain.
- Core router VNF (CR-VNF): provides network routing functionalities in the telecommunication operator domain, offering access to operator-specific services (i.e., the IP Telephony server) and external data networks.

Figure 1 illustrates the network service designed for this experiment. This network service is built as a composition of the aforementioned VNFs, and [provides the functionality of an IP telephony service](#) to users in [the vicinity of](#) the UAVs.

Moreover, the figure presents the physical devices used for the experiment, how they are interconnected, and the specific allocation of VNFs to devices.

PROTOCOL: 1. Prior Requisites for the Experiment 1.1. An installation of the Management and Orchestration (MANO) software stack provided by the Open Source MANO (OSM) project. Specifically, this experiment uses OSM Release FOUR20, which can be executed in a single server computer or in a Virtual Machine (VM) fulfilling the requirements specified by the OSM community: Ubuntu 16.04 as Operating System (64-bit variant image), 2 [Central Processing Units \(CPUs\)](#), 8 GB Random-access memory (RAM), 40 GB storage disk, and a single network [interface with Internet access](#). 1.2. A cloud computing platform, providing the functions of a Virtual Infrastructure Manager (VIM) compliant with OSM Release FOUR. In particular, for this experiment, OpenStack release Ocata21 is used, running in a VM with Ubuntu 16.04 as Operating System, 4 [CPUs](#), 16 GB RAM, and 200 GB storage disk. In the experiment, the VIM manages an NFV Infrastructure (NFVI) integrated by 2 high-profile server computers (each with Ubuntu 16.04 as Operating System, 8 CPUs, 128 GB RAM, and 4 TB storage disk). In the following, we denominate this cloud platform as the core cloud platform. 1.3. An additional cloud computing platform for the UAVs (hereafter referred to as UAVs cloud platform). The platform must feature a VIM based on OpenStack release Ocata. In this case, the resources used by the VIM installation are: Ubuntu 16.04 as Operating System, 2 CPUs, 6 GB RAM, 100 GB storage disk, and an external Wi-Fi USB adapter. On the other hand, the NFVI integrated in this cloud platform consists of a single fixed compute server mini-ITX (Ubuntu 16.04 as Operating System, 8 CPUs, 8 GB RAM, 128 GB storage disk, and an external Wi-Fi USB adapter) and 3 Single Board Computers (SBCs). These latter provide a hardware platform that can easily be onboarded on a UAV. The procedure to setup a UAV cloud platform with these devices as compute nodes is detailed in 3. 1.4. Equip each SBC with a battery-power supply Hardware Attached on Top (HAT) to ensure the operation of these units even when they are in motion being carried by a UAV. NOTE: The next step, 1.5, is optional since the provision of the network service in the experiment does not depend on having UAVs. 1.5. Attach each RPi as the payload of a UAV. In this experiment, two different UAVs model are used: [Parrot AR Drone 2.0](#) and [Parrot Bebop 2](#). 1.6. Two wireless voice-over-IP phones model [ZyXEL Prestige 2000 W](#), which support [the IEEE 802.11b](#) wireless communications [standard](#) (i.e., this model provides wireless communications via Wi-Fi). As an alternative, the voice call could be executed using softphone applications such as Linphone22 or Jitsi23. 1.7. The experiment requires the availability of: a) layer-3 communications between the OSM software stack and each of the VIMs, to enable the orchestrated deployment of the network service developed for this experiment; and b) layer-3 [communications between the OSM and the VNFs](#) instantiated [at each](#) cloud platform, [to](#) support VNF configuration procedures; and c) the layer-3 communications among the VNFs running at every VIM to enable the proper functioning of the network service. 1.8. All the content needed to carry out the experiment is provided in the following public repository, hereafter referred to as the experiment repository: <http://vm-images.netcom.it.uc3m.es/jove/>. 2. Validate the functionality of the softwarization units via Emulation NOTE: To prove the appropriate operation of the network service of the experiment (see Figure 1) under realistic deployment conditions, we use a purpose-specific emulation platform based on Linux

containers²⁴ and ns-3²⁵. This platform can be used to evaluate the behavior of the VNFs and network services as a prior step to their deployment on real equipment. To this end, the emulator supports the configuration of multi-hop aerial network topologies and emulates the exchange of data among the VNFs via wireless communications. This section of the protocol describes the steps to be followed to verify the appropriate operation of the IP telephony service.

2.1. Download the emulation platform from the experiment repository. The platform is available as a virtual machine, with name "uav-nfv-jove-experiment.qcow", compliant with the KVM virtualization technology²⁶. This machine contains a pre-created template that emulates the network service and the multi-UAV scenario presented in Figure 1 and a user with administrator privileges capable of executing that template. NOTE: By default, the following steps are automatically executed when the emulation platform virtual machine is started: a) the virtual environment is configured to enable the network emulation (i.e., network interfaces, Linux bridges²⁷); and b) the Linux Containers that represent each of the physical devices of the scenario are started. All the Linux containers include the required default configurations to have a complete and operational network scenario.

2.2. Before the validation process, an emulated multi-hop aerial network needs to be created using the ns-3 simulator, in order to enable the connectivity between the different network participants.

2.2.1. Create the multi-hop aerial network. For this purpose, execute the "multi-hop-aerial-net.sh" script (available within the emulation platform machine) using the following command: "sudo sh /home/jovevm/scripts/multi-hop-aerial-net.sh > multi-hop-aerial-net-trace.log 2>&1 &". In addition, this command portrays the simulation trace in the specified log-file to enable the debugging in the case of failures.

2.2.2. Check if the network has been successfully created. To this end, verify if the Linux Containers "IP-phone-a" and "IP-phone-b" (illustrated in the Figure 1 as the end-user equipment that connect to an AP-VNF) have obtained an IP address through the DHCP service (only accessible through the multi-hop aerial network). The status of the Linux container executed within the emulation machine, as well as their IP addresses, can be checked using the next command: "lxc list".

2.3. Verify the capacity of the emulated network service to process the signaling messages needed to set up the IP telephony call. For this purpose, both the "IP-phone-a" and "IP-phone-b" Linux containers have installed the "SIPp" tool²⁸. "SIPp" provides the functionality to emulate an IP phone creating the mentioned signaling messages, send them to an IP telephony server and process the response to verify the correct operation of this latter.

2.3.1. Execute the script "test-signaling.sh" in both containers, which runs the "SIPp" tool to generate and send signaling messages to the IP-telephony-server-VNF.

2.3.2. Check the scenario screen provided by execution of the previous step. The reception of a "200" response points out the appropriate functioning of the IP-telephony-server-VNF.

2.4. Validate that the network service is capable of processing the data traffic that is generated during an IP telephony call. With this purpose, the flow scheduling "Trafic" tool²⁹ is installed in the "IP-phone-a" and "IP-phone-b" Linux containers.

2.4.1. Execute the following command to start the server agent of Trafic: "lxc exec IP-phone-b sh called-party.sh".

2.4.2. Then, execute the following command to start the client agent of "Trafic" and get the network statistics: "lxc exec IP-phone-a sh caller.sh". The data traffic emulating a voice call is terminated after 60 s. The script displays a confirmation message and the most significant performance

metrics concerning the voice traffic. 2.4.3. Check the obtained metrics and verify that the IP telephony service can effectively support an interactive voice conversation (to this purpose, see the information included in the section on representative results).

3. UAVs Cloud Platform Construction

3.1. For the experiment, the Raspberry Pi (RPi) Model 3B has been chosen as an SBC capable of providing the virtualization substrate to execute lightweight VNFs. The technical specifications of this device are: 4 CPUs, 1 GB RAM, and 32 GB storage disk. Additionally, each RPi has three network [interfaces: an Ethernet interface, an integrated Wi-Fi interface, and an external Wi-Fi USB adapter](#).

3.2. Prepare the RPis to be subsequently integrated into the UAVs cloud platform.

3.2.1. Install Ubuntu Mate30 16.04.6 as Operating System, given that the OpenStack installation packages are included in this Linux distribution.

3.2.2. Install and configure the required packages as indicated in the OpenStack documentation³¹, to allow the RPis act as the compute nodes of the UAV cloud platform. In particular, and following the previous guide, enable the utilization of Linux containers in the configuration of the OpenStack packages (container virtualization is used due to the resource constraints of the devices that can typically be onboarded on small-sized UAVs).

3.2.3. Download and execute in the RPi the script "rpi-networking-configuration.sh", available within the experiment repository. This script enables the wireless communications of the RPis, as well as the required configuration to allow the creation of virtual networks attached to the wireless interfaces.

3.2.4. Download and execute the script "VIM-networking-configuration.sh", available within the experiment repository, in the host running the UAV cloud platform VIM. This script is in charge of setting up the wireless communications of the VIM to enable the information exchange with the RPis. NOTE: once the networking is well configured and the VIM has connectivity with the RPis, the VIM automatically integrates them into the UAV cloud platform as computational units capable of executing VNFs

3.3. Create an OpenStack availability zone for each of the RPis. This will allow deploying each of the lightweight VNFs of the experiment in an appropriate UAV unit. More specifically, login into the web graphical user interface provided by the VIM with the administrator credentials, create the availability zones in the "Administrator > System > Host Aggregates" tab, and edit each availability zone to add the appropriate host (i.e., each RPi integrated into the UAV Cloud).

3.4. Verify the correct setup of the UAV cloud platform. With this purpose, access to the "Administrator > System > System Information" tab (with the same login as in the previous step), and click in the "Computing Service and Network Agents" section to check that the status of the displayed items is "Alive" and "UP".

4. Configuring the Experiment

4.1. Download the VNF images that implement the different components of the IP telephony service: the AP-VNF, the DNS-VNF, IP-telephony-server-VNF, the AR-VNF, and the CR- VNF. These images can be downloaded from the experiment repository.

4.2. Upload the VNF images to their correspondent VIM, i.e., the AP-VNF and the DNS-VNF to the UAV cloud platform VIM; and the VoIP-VNF to the core cloud platform VIM. More specifically, login into the web graphical user interface provided by each VIM with the administrator credentials, click on the "Create Image" button of the "Administrator > System > Images" tab, and create an image using the displayed form and selecting the appropriate image. This process is to be done at the corresponding VIM for each image that has been downloaded in the prior step.

4.3. Download the VNF descriptors (VNFDs)

of the experiment from the experiment repository. These descriptors provide the templates that describe the operational requirements of a VNF, as well as the placement policies which indicate the availability zone in charge of hosting the VNF itself (more information on NFV descriptors can be found in the information model of OSM32).

4.4. Upload the VNFDs. Use a web browser to access the OSM graphical user interface, and sign with the administrator credentials. Then, drag and drop the VNFDs into the "VNF Packages" tab.

4.5. Download the NS descriptor (NSD) of the experiment from the experiment repository. This descriptor is a template that specifies the VNFs comprising the service, as well as how those VNFs are interconnected.

4.6. Upload the NSD. Drag and drop the NSD into the "NS Packages tab" of the OSM graphical user interface.

4.7. Using the graphical user interface of OSM, add a VIM account for the UAV cloud platform VIM and for the core cloud platform VIM. To do this, access the "VIM accounts" tab with the administrator user, click on the button "+ New VIM" and complete the displayed form with the requested information. Repeat this action for both VIMs.

5. Executing the Experiment

5.1. Deploy the network service. From the "NS packages tab" of the OSM graphical user interface, click on the "Instantiate NS" button of the NSD uploaded in step 5.6. Then, fill the displayed form, indicating the VIM that will be used to deploy each VNF composing the NS. In addition, OSM is responsible for processing the placement policies indicated in the VNFDs to specify the VIM which compute unit is in charge of hosting each VNF. For this experiment, the VNFs are placed in the compute units as illustrated in Figure 1.

5.2. Wait until the OSM graphical user interface indicates the success on the network service deployment. NOTE: The operation of the network service is totally independent from the flight of the UAVs (the IP telephony service can be provided when the UAVs are flying or saving battery-consumption perched on a surface). Thus, the step 5.3 is optional.

5.3. Take off the UAVs. Login into the mobile application and control the flight of each UAV to be maintained stabilized in an intermedium high and avoid the turbulences caused by the rotation of the motors close to a surface.

5.4. Prepare each of the IP phones to carry out the call.

5.4.1. Connect [a wireless VoIP phone to each of the access point offered by the](#) network service. For this purpose, specify the SSID (Service Set Identifier) in the "Menu > Wireless > SSID" tab and choose the Infrastructure mode in the "Menu > Wireless > Network Mode" section. Finally, select the networking configuration through Dynamic Host Configuration Protocol (DHCP) in the "Menu > Net Settings > Network Mode" tab.

5.4.2. Configure the Session Initiation Protocol (SIP) parameters to enable the appropriate exchange of signaling messages with the IP telephony server. In this context, access to the "Menu > SIP Settings" tab and specify the host name of the IP telephony server VNF ("dronesVoIP.net") in the "Registrar > Registrar IP" and "Proxy Server > Proxy IP" tabs. Furthermore, create a user account introducing the name of the user (e.g., caller-A) in the "User Account > Phone Number" and "User Account > Username" sections.

5.4.3. Create an entry in the phonebook of one of the IP phones providing the information of the user that is to be called. To that end, select the "Menu > Phonebook > Add Entry" tab, and fill in the requested parameters that appear in the display as follows: Display name: caller-B; User Info: caller-B; Host IP: dronesVoIP.net; Port: 5060. Finally select the "Proxy" option versus the P2P (peer- to-peer).

5.5. Start the call with the called-party. To that end, select the called-party using the "Menu > Phonebook > Search" option [of the IP phone](#). Then, press [the call button](#). Once

the other IP phone starts ringing, accept the incoming call with the call button.

6. Procedure to gather experiment results

6.1. Connect a commodity laptop to one of the wireless APs and run the "ping" command line tool to the IP address of phone connected to the other AP during 180 s. The IP address can be checked in the "Menu > Information > IP address" option of the IP phone once the connection is established with the AP. Save the Round-Trip Time (RTT) measurements redirecting towards a file the output provided by the "ping" tool.

6.2. Execute the "tcpdump" command line tool in one of the running AP VNFs to capture the traffic exchanged during the IP call. Save this traffic into a file enabling the writing flag of the command line tool at the execution time and specifying the name of the file.

6.3. Perform a new IP telephony call. Maintain the call for the desired time period, e.g., 1 minute. Then, terminate the call, pressing the hang up button of one of the IP phones.

6.4. Keep the files generated by "tcpdump" and "ping" tools for further processing (see section on Representative Results).

REPRESENTATIVE RESULTS: Based on the data obtained during the execution of the experiment, in which a real VoIP call is executed, and following the steps indicated by the protocol to gather this information, Figure 2 depicts the Cumulative Distribution Function of the end-to-end delay measured between two end-user equipment (i.e., a commodity laptop and an IP telephone). This user equipment represents two devices that are interconnected through the AP VNFs of the deployed network service. As it can be observed, more than 80% of the end-to-end delay measurements are below 60 ms, and none of them are higher than 150 ms, which guarantees appropriate delay metrics for the execution of a voice call. On the other hand, Figure 3 illustrates the exchange of DNS and SIP signaling messages. These messages correspond to the registration of one of the users in the IP telephony server (the user whose IP phone is connected to the AP VNF where the "tcpdump" tool is running), and to the establishment of the voice call. Finally, Figure 4 and Figure 5 show the data traffic captured during the call. In particular, the first one represents the constant stream of voice packets transmitted and received by one of the wireless phones during the call, whereas the latter illustrates the jitter in the forward direction with an average value lower than 1 ms. The results that have been obtained in the experiment for the delay figures (end-to-end delay and jitter) satisfy the recommendations specified by the International Telecommunication Union - Telecommunication Standardization Sector (ITU-T) 33. Accordingly, the voice call progressed with no glitches and good sound quality. This experiment has served to validate the practical feasibility of using NFV technologies and UAVs to deploy a functional IP telephony service.

FIGURE AND TABLE LEGENDS: Figure 1. Overview of the network service, depicting the VNFs, the entities in which they are executed, and the virtual networks needed for the provision of the IP telephony service.

Figure 2. End-to-end delay. Representation of the end-to-end delay offered to the end-user equipment connected to the AP VNFs. For this purpose, the Cumulative Distribution Function of the end-to-end delay has been computed from the measured RTT samples obtained with the "ping" command line tool.

Figure 3. User registration and Call signalling messages. Illustration of the signalling traffic (DNS and SIP) exchanged to register a user in the IP telephony server, and to create and terminate the multimedia session that supports the execution of the voice call.

Figure 4. Stream of Voice packets. Representation of the voice traffic exchanged during the call, measured at one of the AP VNFs.

Figure 5. Evolution of the network Jitter during the call. Representation of the jitter endured by the transmitted voice packets in the forward direction from one phone to the other.

DISCUSSION: One of the most relevant aspects presented during the development of this experiment is the use of virtualization technologies and NFV standards with UAV platforms. NFV presents a new paradigm aiming at decoupling the hardware dependency of the network functionalities, and thus enabling the provision of these functionalities through softwarization. Accordingly, the experiment does not imperatively depend on the use of the hardware equipment specified in the protocol. Alternatively, different models of single board computers can be selected, as long as they are in line with the dimensions and the transport capacity of the UAVs and they support Linux containers. Notwithstanding this flexibility in terms of hardware selection, all the content provided for the reproducibility of the experiment is oriented towards the use of open source technologies. In this context, the configuration aspects and the software tools are conditioned to the use of Linux as Operating System. On the other hand, the experiment considers the interoperation of two different computational platforms (i.e., the UAV cloud platform and the core cloud platform) to provide a moderately complex network service. However, this is not strictly needed, and the protocol could be followed to support scenarios in which only the UAV cloud platform is involved. Finally, it should be noted that the results presented have been obtained in a laboratory environment and with the UAV devices grounded or following a limited and well-defined flight plan. Other scenarios involving outdoor deployments may introduce conditions affecting the stability of the flight of the UAVs, and hence the performance of the IP telephony service.

ACKNOWLEDGMENTS: This work [was partially supported by the European H2020 5GRANGE project \(grant agreement 777137\)](#), [and by the 5GCIty project \(TEC2016-76795-C6-3-R\) funded by the Spanish Ministry of Economy and Competitiveness](#). The work of Luis F. Gonzalez [was partially supported by the European H2020 5GinFIRE project \(grant agreement 732497\)](#).

DISCLOSURES: [The](#) authors have nothing to disclose.

REFERENCES:

1. Sanchez-Aguero, V., Nogales, B., Valera, F., & Vidal, I. Investigating the deployability of VoIP services over wireless interconnected Micro Aerial Vehicles. *Internet Technology Letters*, 1(5), e40 (2018).
2. Maxim, V., & Zidek, K. Design of high-performance multimedia control system for UAV/UGV based on SoC/FPGA Core. *Procedia Engineering*, 48, 402-408 (2012).
3. Vidal, I., Bellavista, P., Sanchez-Aguero, V., Garcia-Reinoso, J., Valera, F., Nogales, B., & Azcorra, A. Enabling Multi-Mission Interoperable UAS Using Data-Centric Communications. *Sensors*, 18(10), 3421 (2018).
4. Vidal, I., Valera, F., Díaz, M. A., & Bagnulo, M. Design and practical deployment of a network- centric remotely piloted aircraft system. *IEEE Communications Magazine*, 52(10), 22-29 (2014).
5. Jin, Y., Minai, A. A., & Polycarpou, M. M. Cooperative real-time search and task allocation in UAV teams. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475) (Vol. 1, pp. 7-12)*. IEEE (2003).
6. Maza, I., & Ollero, A. Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms. In *Distributed Autonomous Robotic Systems 6* (pp. 221-230). Springer, Tokyo (2007).
7. Quaritsch, M. et al. Collaborative microdrones: applications and research challenges. In *Proceedings of the 2nd International Conference on Autonomic Computing and Communication Systems* (p. 38). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering)

(2008). 8. Waharte, S., Trigoni, N., & Julier, S. Coordinated search with a swarm of UAVs. In 2009 6th IEEE Annual Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops (pp. 1-3). IEEE (2009). 9. De Freitas, E. P. et al. UAV relay network to support WSN connectivity. In International Congress on Ultra-Modern Telecommunications and Control Systems (pp. 309-314). IEEE (2010). 10. European Telecommunications Standards Institute. Network Functions Virtualisation (NFV); Architectural Framework; Research Report ETSI GS NFV 002 V1.2.1; European Telecommunications Standards Institute (ETSI) (2014). 11. ETSI OSM. An Open Source NFV Management and Orchestration (MANO) software stack aligned with ETSI NFV. Available online: <https://osm.etsi.org/> (last access on 3 June 2019). 12. Nogales, B., Vidal, I., Lopez, D. R., Rodriguez, J., Garcia-Reinoso, J., & Azcorra, A. Design and Deployment of an Open Management and Orchestration Platform for Multi-Site NFV Experimentation. IEEE Communications Magazine, 57(1), 20-27 (2019). 13. Omnes, N., Bouillon, M., Fromentoux, G., & Le Grand, O. A programmable and virtualized network & IT infrastructure for the internet of things: How can NFV & SDN help for facing the upcoming challenges. In the 18th International Conference on Intelligence in Next Generation Networks (pp. 64-69). IEEE (2015). 14. Rametta, C., & Schembra, G. Designing a softwarized network deployed on a fleet of drones for rural zone monitoring. Future Internet, 9(1), 8 (2017). 15. Garg, S., Singh, A., Batra, S., Kumar, N., & Yang, L. T. UAV-empowered edge computing environment for cyber-threat detection in smart vehicles. IEEE Network, 32(3), 42-51 (2018). 16. Mahmoud, S., Jawhar, I., Mohamed, N., & Wu, J. UAV and WSN softwarization and collaboration using cloud computing. In the 3rd Smart Cloud Networks & Systems (SCNS) (pp. 1-8). IEEE (2016). 17. González Blázquez, L. F., Vidal, I., Valera, F., Sanchez-Aguero, V., Nogales, B., & Lopez, D. R. NFV orchestration on intermittently available SUAV platforms: challenges and hurdles. In 1th Mission-Oriented Wireless Sensor, UAV and Robot Networking (MISARN). IEEE (2019). 18. Nogales, B., Sanchez-Aguero, V., Vidal, I., Valera, F., & Garcia-Reinoso, J. 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 (pp. 39-44). ACM (2018, June). 19. Nogales, B., Sanchez-Aguero, V., Vidal, I., & Valera, F. Adaptable and automated small UAV deployments via virtualization. Sensors, 18(12), 4116 (2018). 20. Hoban, A. et al. An ETSI OSM Community White Paper, OSM Release FOUR: A Technical Overview; Whitepaper; European Telecommunications Standards Institute (ETSI) (2018). 21. OpenStack. Open Source Software for Creating Private and Public Clouds. Available online: <https://docs.openstack.org/ocata> (last access on 3 June 2019). 22. Linphone. An Open Source VoIP SIP Softphone for voice/video calls and instant messaging. Available online: <https://www.linphone.org> (last access on 3 June 2019). 23. Jitsi. An Open Source Project to easily build and deploy secure video-conferencing solutions. Available online: <https://jitsi.org> (last access on 3 June 2019). 24. Linux Containers (LXC). Infrastructure for container projects. Available online: <https://linuxcontainers.org> (last access on 3 June 2019). 25. Ns-3. A Discrete-Event Network Simulator for Internet Systems. Available online: <https://www.nsnam.org/> (last access on 3 June 2019). 26. Kernel-based Virtual Machine (KVM). A virtualization solution for Linux. Available online: <https://www.linux-kvm.org> (last access on 3 June 2019). 27. Bridging &

firewalling. Available online: <https://wiki.linuxfoundation.org/networking/bridge> (last access on 3 June 2019). 28. SIPp. An Open Source test tool and/or traffic generator for the SIP protocol. Available online: <http://sipp.sourceforge.net/> (last access on 3 June 2019). 29. Trafic. An open source flow scheduler. Available online: <https://github.com/5GinFIRE/trafic> (last access on 3 June 2019). 30. Ubuntu Mate for the Raspberry Pi. Available online: <https://ubuntu-mate.org/raspberry-pi/> (last access on 3 June 2019). 31. OpenStack Installation Tutorial for Ubuntu. Available online: <https://docs.openstack.org/ocata/install-guide-ubuntu/> (last access on 3 June 2019). 32. Open Source MANO Information Model. Available online:

https://osm.etsi.org/wikipub/index.php/OSM_Information_Model (last access on 3 June 2019). 33. ITU-T Recommendation G.114. General Recommendations on the transmission quality for an entire international telephone connection; One-way transmission time. International Telecommunication Union -

Telecommunication Standardization Sector (2003). 1 2 3 4 5 6 7 8 9 10 11 12
13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37
38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62
63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87
88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108
109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162
163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198
199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216
217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234
235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252
253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270
271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288
289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306
307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324
325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342
343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360
361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378
379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396
397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414
415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432
433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450
451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468
469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486
487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504
505