Scalable Virtual Network Video-Optimizer for Adaptive Real-Time Video Transmission in 5G Networks

Pablo Salva-Garcia, Jose M. Alcaraz Calero, *Senior Member IEEE*, Qi Wang, Miguel Arevalillo-Herráez, and Jorge Bernal Bernabe

***Abstract*—The increasing popularity of video applications and ever-growing high-quality video transmissions (e.g. 4K resolutions), has encouraged other sectors to explore the growth of opportunities. In the case of health sector, mobile Health services are becoming increasingly relevant in real-time emergency video communication scenarios where a remote medical experts' support is paramount to a successful and early disease diagnosis. To minimize the negative effects that could affect critical services in a heavily loaded network, it is essential for 5G video providers to deploy highly scalable and priorizable in-network video optimization schemes to meet the expectations of a large quantity of video treatments. This paper presents a novel 5G Video Optimizer Virtual Network Function (vOptimizerVNF) that leverages the latest technologies in 5G and video processing to address this important challenge. Advanced traffic filtering is coupled with Scalable H.265 video coding to enable run-time bandwidth-saving video optimization without compromising Quality of Service (QoS); kernel-space video processing is introduced to achieve further performance gains; and the use of a Virtual Network Function (VNF) facilitates dynamic deployment of virtualized video optimizers to achieve scalability and flexibility in this service. The proposed approach is implemented in a realistic 5G testbed and empirical results demonstrate the superior scalability and performance achieved.**

***Index Terms*—5G, QoS, Multi-tenancy, eHealth, mHealth, traffic filtering, Video, NFV**

I. Introduction

ACCORDING to recent literature, video traffic is expected to increase dramatically in the upcoming years, reaching four-fifths of the world's mobile data traffic by 2022 [1]. Mobile Health (mHealth) services can benefit from the 5G URLLC (Ultra-reliable Low Latency Communication) technology for dealing with real-time video traffic, thereby coping with sensitive and critical demands of certain eHealth services. For instance, multiple ambulances might require real-time and reliable video transmissions to a remote hospital, where specialized medical personnel can organize the intervention and collect the health-state information necessary to proactively set up the treatment at the hospital. In such a context, network congestion might undermine medical staff video quality perception, and thus 5G networks need to be prepared to self-adapt to and self-optimize the network traffic, mitigating as much as possible the disruption of the video streaming for critical services.

Some related works have already addressed a context- and content-aware video optimization of the m-health video streaming [2], but they did not considered the 5G specific network conditions and protocols. 5G Network operators, Carriers, and Internet Service Providers need to differentiate and optimize the streaming video packets at both the Edge and the Core of the 5G network, according to service demands and preferences. Unlike in traditional IP networks, 5G networks are intended to support mobility and multi-tenant isolation, which raises the challenge of efficiently managing the nested-encapsulated network traffic required to develop these features. In addition, video optimization can benefit from dynamic, on demand provisioning and configuration of Virtual Network Functions (VNF). This means that a 5G virtual Optimizer VNF (vOptimizerVNF) network service might differentiate and filter video traffic flows according to any field of the inner packet headers. 5G architectures can also benefit from the flexibility provided by Network Function Virtualization (NFV) and Software-Defined Networking (SDN) to detect traffic congestion and enforce dynamically different vOptimizerVNFs to support video adaptations per service and flow, by enforcing proper network traffic rules that can filter the video packets in an optimized manner.

However, and despite the new opportunities brought by 5G networks, the current management frameworks are not able to provide self-optimization capabilities to dynamically adapt network traffic filtering and, in turn, control video flows according to network traffic conditions obtained from the monitoring sensors. As a direct consequence, there is a lack of scalable network video optimization systems that are simultaneously able to a) deal with traffic in multi-carrier and mobility scenarios for 5G video streaming; b) support nested encapsulation demands imposed by both core network and edge segments of the 5G multi-tenant networks; and c) perform filtering in the inner layers of overlay networks. Although we have already addressed this problem in the past [3], we did not delve into scalability issues when handling thousands of flows in the network and did not take into account service optimization.

In this paper, we present an efficient virtual Video-Optimization mechanism that is able to maintain critical services' Quality of Service (QoS) on-demand when the network is congested, by performing an efficient, dynamic and selective dropping of the enhancement layers of scalable video streams that use the Scalable High Efficient Video Coding [4] (SHVC). To deal with scalability and performance issues, the video optimizer is dynamically deployed as a VNF in the Edge or in the Core of the 5G network, whereas the video optimization

is performed by filtering in kernel space different video layered packets from a particular service/flow. We evaluate the proposed network video optimization mechanism in a real 5G trial infrastructure testbed, in order to assess its performance. To this end, we present a realistic use case in which we attempt to optimize emergency video communications, and compare different traffic filtering mechanisms to evaluate their aptness in nested-encapsulated virtualized 5G networks. The evaluation shows that our solution adapts to the network conditions by quantifying congestion through an index, and ensures smooth video streaming even when congestion occurs. In addition, the performance analysis demonstrates its suitability to handle thousands of flows in multi-tenant and virtualized 5G networks.

The rest of the paper is organized as follows. Section II describes the state of the art in video adaptation techniques, and provides some basic concepts that are deemed necessary to understand the remnant of the paper. Section III-B identifies the video optimizer requirements we believe essential for efficient media services running on 5G networks. Section IV describes the proposed solution, including the management framework and some modification to standard components that were required to provide a scalable service. Section V outline some relevant implementation details. Sections VI and VII reports the empirical results obtained from a comprehensive evaluation of the proposed traffic filter mechanism and scalable virtual video optimizer, respectively. Finally, conclusions and future research activities are drawn in Section VIII.

II. Background and Related Work

Relevant technologies to the work presented include both existing methods for video-streaming adaptation and traffic filtering mechanisms. For clarity reasons, related previous works in these two topics are analysed separately in this section. In addition, in the last subsection we also discuss some basic concepts about video traffic in 5G that we deem necessary to ease understanding of the proposed approach.

A. Video-Streaming Adaptation

Maintaining a reasonable QoS during the visualization of a video stream has been, and still is, a challenging issue in video transmission. The ITU-T E-Model [5] has been widely accepted, and considers that a delay over 100 ms or a latency variation (jitter) higher than 200 ms can have a measurable impact on the Quality of Experience (QoE) perceived, which is specially relevant in many multimedia applications such as tele-conferencing [6]. More recently, Abdallah et al. [7] surveyed recent advances on delay-sensitive video computations in the cloud and pointed out new requirements that conversational video services such as video-conferencing should meet, related to the acceptable latency between a user's action and the visible reaction on the screen.

In order to minimize network delays and meet the requirements of an acceptable service, some existing methods involve edge network caching as an effective strategy to store media content in edge servers that are near the access networks, and close to the users [8]. In general, caching strategies applied over Content Delivery Networks (CDNs) can easily speed up content delivery processes reducing delays significantly [9], and consequently, increase the QoE of the end users. For example, Polaraskis et al. [10] and George et al. [11] proposed to bring content closer to the requesters by using distributed and cooperative algorithms for dealing with visualization requests of on-line videos with a dynamically changing demand from mobile devices, e.g. recent news, new movies or new music videos [12].

Caching methods can also be combined with efficient multimedia streaming applications to adapt the video quality being delivered according to the user's internet connection, device capabilities and/or user preferences. For example, Dynamic Adaptive Streaming over HTTP (DASH) [13] and CMAF [14] are able to dynamically vary the bit rate and quality of the transmitted media to match the available channel bandwidth and alleviate specific problems related to network congestion, such as a high latency or packet loss rate. When using such dynamic adaptive streaming techniques, the client is the only agent that manages the video streaming process in order to maximize the subjective video quality [15] [16] [17] by dynamically selecting among different representations of the same media stream based on the estimated network throughput. Media content is split into a sequence of small segments, which basically are different versions of the original segment but encoded with different bit rates and quality. Another representative case of an adaptive bit-rate solution for real-time video communications is the open-source project WebRTC [18], which has been specifically developed to support real-time voice and video communications in the browser [19], [20]. WebRTC uses the Google Congestion Control (GCC) algorithm to handle congestion in real-time communications over UDP (or TCP just if UDP ports are blocked), but performance details and congestion handling are completely hidden from the owners of the video streaming applications.

Several other research works have focused on parsing certain aspects of the application payload headers and reacting to the contents through packet-filtering by using a Media Aware Network Element (MANE) [21]. In this direction, the use of layered video streaming makes it possible to dynamically adapt the media rate to the transmission conditions without transcoding or re-encoding the video. Schierl et al. [22] documented the potential use of scalable video transmissions and outlined use cases of mobile media delivery which can benefit from using layered transmissions over networks that are not provisioned to provide suitable QoS. In [23] the authors use a Media Aware Proxy (MAP), which is based on the MANE concept, for dropping Scalable H.264 (SVC) packets that carry information of the enhancement layers according to the IP header in a testbed that is integrated within the Long-Term Evolution (LTE) Evolved Packet Core Architecture. Ryu et al. [24] describe experimental results of a picture prioritization method and error concealment mode signaling, which is calculated and determined at encoder and decoder sides. Then a smart router acting as a MANE differentiates the packets with various priorities when congestion occurs. Jassal and Leung [25] present a simulation of cross-layer scheduling framework that processed information about the

coding structure of an H.265-encoded video bitstream to quantify the contribution of each frame to the video decoding process. The authors adopt an LTE-Advanced simulator and assume that the MAC-layer scheduler located in eNBs has the ability to parse RTP packets to access information carried in its payload, thus turning an eNB into a MANE. A MANE is also used in [3], where the authors present a scalable H.265 video optimization mechanism for a 5G architecture.

However, and despite the extensive research in this field of study, there is still a long way to go to meet consumer demands, which are weakly defined in some cases. Up to now, none of the proposed approaches has simultaneously taken into consideration the particularities imposed by 5G architectures to achieve a truly scalable solution. In this paper, we describe a MANE approach that has been specifically designed for 5G networks and is simultaneously able to fulfill the ambitious requirements associated with the use case of real-time video delivery in emergency communications.

B. Traffic Filtering Methods

A MANE is commonly used when inspecting and filtering media flows in real-time is required. In this section, we review existing approaches and tools to perform such media flow filtering in software. As a major filtering approach, Netfilter[1] is an open source framework to manipulate and mangle packets in the Linux IP networking system. It was included in the Linux kernel 2.4 codebase to replace the ipchains architecture[2]. Netfilter defines five different hooks, which are well-defined points in a packet traversal of the IP protocol stack. Any packet going through a Netfilter framework will be checked against such hook points.

Iptables is a user-space program, used as a command-line tool that allows the configuration of the rules to be added in each of the hooks of the Linux kernel. Built-in classifier methods included in the default implementation of iptables are not able to deal with traffic filtering of different multi-carriers/operators (tenants) and mobility scenarios where virtualized 5G Networks impose different level of encapsulation to differentiate and handle users from the control plane of the network. To overcome such lack, this research work approaches two different Netfilter module extensions with byte-matching capabilities: u32 and BSD Packet Filter (BPF).

u32[3] is a Netfilter mechanism based on byte-matching techniques that permits dynamic inspection of networking packets. In particular, it can jump between headers, select a specific location, and compare a 32 bits extract from the packet against a given value.

BPF [26] is a byte-matching filtering mechanism that provides an efficient way of filtering packets in the kernel space. It is available in most Unix operating systems and provides a similar functionality to the u32 iptables module. BPF [26] is a small Virtual Machine (VM) that runs programs (in a user-friendly high-level syntax) injected from the user space and attached to specific hooks in the kernel. This generic, fast and safe solution for implementing packet classification is being used in many utilities such as libpcap, tcpdump, iptables and even in virtual networking software such as Open virtual switch (OVS) for overcoming OpenFlow's packet classification limitations.

The scalable network video optimization solution proposed in this paper leverages and extends existing filtering mechanisms to filter media traffic in 5G Networks, handling the network traffic dynamically, according to the contextual decisions made by the autonomic framework.

C. Video Traffic in 5G

5G is a major evolution over previous technologies in many senses. Apart from introducing performance improvements of several orders of magnitude over today's networks, 5G infrastructures provide native support for multi-tenancy, mobility and dynamic management. In order to provide these features, 5G data packets follow a nested structure, which is illustrated through an example in Fig. 1.

Fig. 1. Hierarchical Encapsulation in 5G networks

The first group of headers is related to the communication between physical machines including Medium Access Control (MAC), IP and UDP headers. The second group includes a VXLAN, MAC, IP and UDP, inserted to isolate tenant traffic, especially for a telecommunication operator sharing the same physical 5G infrastructure as a tenant. The next group of headers includes GTP, IP and UDP, and it is introduced to allow user mobility. GTP is the tunneling protocol employed in LTE/5G infrastructures to establish the data path for video transmissions with features such as mobility, admission control, etc. Finally, the application header and H.265 represent the data being sent/received by the end users. This study focuses on real-time video streaming based on SHVC, and the widely adopted RTP [27] has been then selected for streaming real-time video. Our research uses the VXLAN protocol to achieve that tenant isolation as an example, although other alternative protocols can be employed to fit the same purpose.

H.265/HEVC and its scalable extension (SHVC, or Scalable H.265) have been developed by adopting a scalable coding architecture that relies on making high-level syntax only (HLS-only) changes to underlaying single-layer HEVC standard. Thus, they achieve highly scalable coding efficiency without requiring any block-level coding logic changes to the single layer HEVC cores [4] [28].

An HEVC bitstream consists of a sequence of data units called Network Abstraction Layer (NAL) units. The first two bytes of a NAL unit correspond to the header, and the remnant contains the payload data. Fig. 2 shows the structure of the NAL unit header, which contains high-level information to ease parsing the main properties of a NAL unit, e.g., what type it is, and what layer and temporal sub-layer it belongs to. The

[1]https://www.netfilter.org
[2]https://people.netfilter.org/rusty/ipchains/HOWTO.html
[3]http://www.netfilter.org/documentation/HOWTO/netfilter-extensions-HOWTO-3.htm

format of the RTP payload allows for packetization of NAL units in each RTP packet, thereby supporting HEVC streaming for videoconferencing, Internet video streaming, and high-bitrate entertainment-quality video, among other applications.

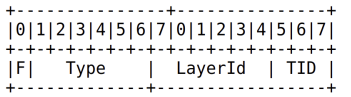


Fig. 2. The Structure of the HEVC NAL Unit Header [29]

It is worth noting that a normal IP network uses a very limited subset of these headers, for instance, MAC/IP/UDP/APP. Compared to that simple case, several additional headers have been added to achieve both multi-tenancy and mobility.

III. Problem description

A. Mathematical formulation

Let us assume that a set of n concurrent users $\{u_i, i = 1 \ldots n\}$ are using a 5G network, and each has contracted a service subscription s_i based on specific demands that relate to quality of service requirements. Let us model the service subscription for each user u_i as a vector of m features $s_i = \{f_{i,1}, f_{i,2}, ..., f_{i,m}\}$, where the terms $f_{i,j}$ refer to threshold values required by user u_i for each parameter j in a set that the system can reliably measure (e.g. delay, bandwidth). Let us also represent the obtained QoS for a user u_i as $\{m^t_{i,1}, m^t_{i,2}, ..., m^t_{i,m}\}$, where terms $m^t_{i,j}$ refer to system measurements at an instance in time t, for each parameter initially considered in the definition of the service subscription.

Under this problem setting, our objective is to construct a scalable video optimizer that is able to minimize the impact and time span that some $m^t_{i,j}$ falls below the threshold value $f_{i,j}$.

B. Additional Video Optimizer Requirements

In addition to achieving the objetive specified above, we believe there are a number of essential features that should be supported by efficient traffic filtering and contextual media traffic management in 5G networks. Some of these features are imposed by the 5G infrastructure, and some others are desirable characteristics that we believe shall be necessarily supported to allow the deployment of high quality media services. Next, we give a list of the most relevant properties that have been taken into consideration in the design of the proposed solution:

- **Support different QoS levels**. Traffic differentiation should be in place according to the service subscription.
- **QoE support**. Traffic processing should minimize negative impact on the user's perceived quality, especially for video applications.
- **Context-awareness**. Video optimization requires context awareness in traffic conditions, e.g., the congestion level, obtained from monitoring sensors (flow sensors and video Sensors).
- **Application layer filtering**. The system in charge of the network traffic filtering should be able to deal with traffic of any header/field of any protocol of the OSI protocol stack, including any Layer 7 video application protocol as long as this video content is either not encrypted or encrypted by following the subsample encryption scheme specified in ISO/IEC 23001-7 [30]. This scheme leaves the first two bytes of each NAL unit unencrypted (the header), allowing the vOptimizerVNF to recognize among different layers of video whereas the payload remains encrypted. It should be noted that the full content encryption scheme is out of the scope of this research and is proposed as future work.
- **Scalability**. The system should be able to cope with thousands of flows, while meeting the QoS requirements e.g., on throughput and latency.
- **Bandwidth saving**. The traffic processing in video flows should maximize bandwidth saving to mitigate network congestion.
- **Multi-tenant support**. In 5G networks different carriers, network operators, and verticals are allowed to share the physical infrastructure by virtualizing functional blocks of the network architecture as VNFs. To differentiate the traffic among them, for security and management reasons, packets need to be encapsulated (e.g in VxLAN). Thus, the filtering mechanism is required to deal with that encapsulation.
- **Mobility support**. LTE and 5G networks must support mobility of the user equipment (UE). Mobility in 5G architectures means that packets need to be encapsulated towards the mobility anchor component (SGW/UPF), e.g. using the GTP protocol. The traffic filter is thus required to be able to handle efficiently and directly these encapsulation headers.
- **Dynamic management**. Rules based on the context of real-time monitoring must be implemented dynamically and removed from the management framework to automatically adapt the video traffic filtering policies. This dynamic and intelligent management needs to rely on softwarized network management and NFV technologies for handling efficiently such adaption.

IV. Proposed solution

A. Management Framework

To be able to provide self-optimization capabilities to dynamically adapt network traffic filtering, we have adopted the management framework illustrated in Fig. 3. This is based on 5G PPP SELFNET [31], which is compliant with the ETSI MANO reference architecture [32], and incorporates new elements that allow for the task at hand. The main roles and responsibilities of each component are described below. The numbered arrows indicate the actions sequence in the video optimization process and will also be further commented afterwards.

- **Physical Layer**. It consist of all the physical hardware equipment of the 5G Infrastructure, including compute, storage and networking components.

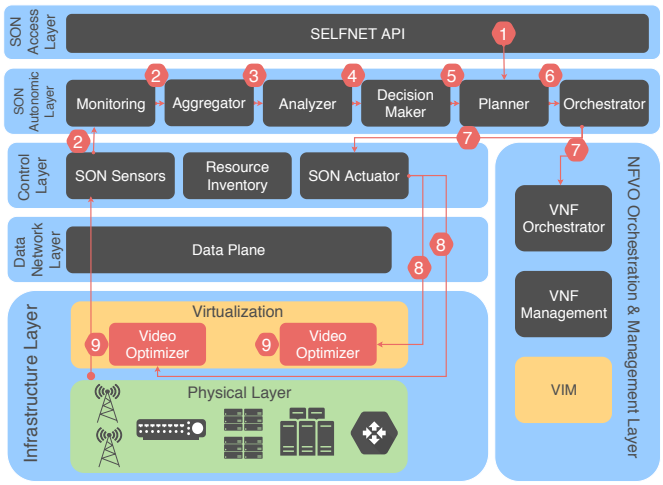


Fig. 3. Selfnet 5G management framework and video optimization process.

- **Virtualization Layer**. It comprises all the logical hardware equipment created by the hypervisor. This includes compute, storage and networking through virtualization.
- **Data Plane**. It represents the physical and logical path for user communications along the infrastructure.
- **SON Sensors**. They provide compute, storage and networking information metrics about the current status of the infrastructure. To allow monitoring the level of congestion of every network port in the infrastructure and metadata of the video flows, these sensors jointly report key metrics such as port speed, current bandwidth consumed per port, and maximum bitrate required per video flow.
- **Resource Inventory**. It is maintained by a set of software components (topology managers) that allows an update-to-date inventory of physical and logical hardware equipment and services deployed in such hardware. This inventory also maintains a record of active flows, forwarding tables, and so on.
- **Monitoring**. It stores all the metrics reported by the sensors. These are sent either periodic or event-based metrics to a message bus. The Monitoring component is registered to the message bus and performs a highly scalable storage of data.
- **Aggregator**. It defines and computes aggregated metrics that are defined as a mathematical combination of the raw metrics provided by the SON sensors. This component allows for both spatial and temporal aggregation as well as filtering in order to produce periodic aggregated metrics, which are then published to the message bus (and stored in the Monitoring component).
- **Analyzer**. It allows the definition of rules that are used to monitor the current status of the infrastructure by using spatial and temporal correlation in the information gathered from both the monitoring component and the resource inventory. The rules produce alertsabout specific events in the infrastructure that require attention, either by a human being or by a Decision Maker component.
- **Decision Maker**. It permits the definition of rules that are used to make decisions to react to the current status of the infrastructure by using any spatial and temporal correlation in the information from the Monitoring and Analyzer components and from the Resource inventory. Those rules lead to decisions aimed to govern the strategy intended to mitigate the alert received by the Analyzer component.
- **Planner**. It makes it possible to define templates that are used to transform the strategy indicated by the Decision Maker into an ordered set of implementable steps required to implement the strategy received.
- **Orchestrator**. It receives an implementable plan and then orchestrates the execution of each of the steps involved in the plan in order to enforce it into the infrastructure with the purpose of resolving the alert.
- **SON Actuator**. It configures the actuators of the infrastructure related to the enforcement of the steps involved in the process of resolving the alert. As an example, it inserts rules to control traffic in the data path.
- **VNF Orchestrator**. It orchestrates the deployment of the different VNFs using the physical and logical resources of the infrastructure.
- **VNF Manager**. It is in charge of the configuration and the management of the life cycle of the different VNFs managed in the infrastructure.
- **Virtual Infrastructure Manager (VIM)**. It manages the different physical and virtual resources of the infrastructure, including compute, storage and networking.

This general management architecture brings the SDN and NFV paradigms together and closes a control loop to allow automated responses to network issues. Design and implementation challenges related to this architecture are described in detail in [31].

Taking advantage of said management architecture, the network traffic filtering proceeds according to the steps depicted in Fig. 3, which are further described below.

1) The administrator pro-actively defines the policies according to the QoS required for each service profile. These policies are translated into an intent format.
2) At run-time, the SON Sensor starts providing monitoring information through probes to the Monitoring module. This monitoring traffic is sent through the Pub/Sub Broker.
3) The aggregation module continuously and dynamically computes a Congestion Index and reports it to the Analyzer module.
4) If the congestion level surpasses a threshold, the Analyzer module detects congestion, and warns the Decision Maker.
5) The Decision Maker adds new filtering rules to selectively drop traffic coming from a concrete service with specific QoS requirements, and informs the Planner module of the decision made. Qos requirements may vary depending on the service profile selected (e.g a new ambulance's intervention). Thus, this module is also in charge of re-scheduling existing rules in the case of updates on a particular service requirements.

6) The Planner translates the decision communicated by the Decision Maker into an implementable plan, and informs the Orchestrator about the best strategy to enforce the rules in the vOptimizerVNF.
7) The Orchestrator is notified of the deployment of the rules. Optionally, it might contact the NFV MANO to deploy (if not deployed yet) the vOptimizerVNF as a VNF. Then, the Orchestrator contacts the involved SON Actuator to enforce the filtering rules through the Northbound API.
8) The SON Actuator contacts the vOptimizerVNF using a Southbound API to enforce the filtering rules that will optimize the video transmission. The proposed filtering mechanism is able to deploy the rules in the vOptimizerVNF either in the edge or in the VNF domain of the core of the virtualized 5G Network.
9) Finally, filtering rules are enforced in the vOptimizerVNF through the Southbound API. These rules consider the network traffic filtering requirements described in section III-B to cope with the real-time and scalable video traffic, including nested-encapsulation for multi-tenant and mobility traffic filtering support.

B. Proposed xu32 Filtering Mechanism

Current traffic filtering mechanisms do not fully support the complex and large rule predicates needed to cope with requirements laid out in section III-B (e.g. complex nested-encapsulation). This section describes our proposed traffic filtering mechanism to circumvent this limitation.

As described in section II-B, u32 is a Netfilter mechanism based on byte-matching techniques that allows a dynamic inspection of message payloads. Although u32 filtering predicates are complex and do not follow a human-readable format, they are expressive enough to construct powerful rules (predicate + action). As a drawback, u32 only allows a limited number of characters per predicate. This is an important restriction when it comes to handling encapsulated traffic since predicates may become extremely longer compared with pure IP traffic. While in purely IP traffic we would need to inspect only 5 protocol headers (MAC/IP/UDP/RTP/H265), in the case of complex 5G encapsulated data transmissions, this number may increase (depending on the network segment) up to 12 protocols, as can be seen in Fig. 1. Due to this restriction in the implementation of u32, it is currently impossible to build a single predicate for inspecting these 12 protocols at once. Consequently, long predicates must be divided into many sub-predicates (ideally one per protocol) to achieve the same filtering expression, which also entails a new iptables chain to group them into the same user-rule context. Therefore, network traffic passing through this matching scheme will suffer time penalties due to the overhead introduced by jumping chains and associated sub-predicates.

To significantly reduce those delay penalties, we have modified the standard u32 Netfilter module to increase the maximum number of characters per predicate. Our byte-matching proposal allows describing longer predicates that cover all involved protocols in a complex 5G encapsulated data transmission, thereby eliminating the need to create sub-predicates. Unlikely the standard u32, our byte-matching proposal has achieved a one-to-one relationship between predicate and rule, and a many-users-to-one iptables chain. When several users streaming video over a 5G network, the number of jumps by using the standard u32 would be $(NU * NP) + NC$, with NU the number of users, NP the number of protocol headers and NC the number of chains. Conversely, when using our modified u32, the number of jumps lessens to just NU. For example, to control one hundred 5G video-users using the standard u32 , it would be needed (100*12)+100=1300 jumps (1200 plus 100 jumps in predicates and chains, respectively) instead of the just 100 rule-jumps when using a unique but longer predicate per each user. However, since the matching efficiency may drop when predicates becomes longer and longer, we prioritize a premature matching of essential protocols instead to incur in the typical inspection left to right (outer to inner encapsulation layers) which works just fine in simple predicates. For example, the tenant's information is inspected first (instead of the fourth) to achieve an early decision and omit the rest of the predicate inspection. By doing so, there is room for improvement in u32 when complex predicates have to be processed in real-time communications. Aforementioned improvements and considerations is that we call the Extended u32 (xu32).

C. Proposed Scalable Virtual Video Optimizer

In the last edition of the Mobile World Congress (MWC2018), Qualcomm employed extensive network simulations to yield an accurate indication of what to expect when 5G networks are launched. They predicted 490 Mbit/s median speeds for a common configuration of sub-6 GHz 5G Massive MIMO. They also predicted a 1.4 Gbit/s median speed for a configuration using 28 GHz millimeter waves[4]. The vOptimizerVNF aims to deal with up to 1536 simultaneous video flows coming from different Distributed Units (DUs). Assuming a bit rate of 1 Mbit/s per video flow, our video optimizer service is able to handle about 1.5 Gbit/s (12 vOptimizerVNFinstances which manage 128 flows each), which turns sufficient according to the expected medium speed of the emerging 5G networks. It is noted that the value of 1 Mbps is chosen for testing the worst possible scenario in terms of the number of rules (the more rules, the worse) as well as providing a clear testing example to the readers (1 flow == 1 Mbps == 1 matching rule). Nevertheless, vOptimizerVNF would also be able to handle higher bit rates. This section describes the solution devised to make this vOptimizerVNF scalable, so that it can cope with a number of flows without incurring unacceptable delays or packet loss.

The video optimizer service aims to be deployed in the edge of the 5G network (RAN), meaning the Central Unit (CU), and in turn our vOptimizerVNF might need to cope with around 1500 users sending video flows simultaneously. Assuming a video-conference per user at a time, and one matching filtering rule per flow (i.e. a video Conference), a

[4]https://qualcomm.com/news/releases/2018/02/25/qualcomm-network-simulation-shows-significant-5g-user-experience-gains

traditional approach such as the one considered in the previous section would not be able to cope with video traffic filtering of such a high amount of flows with no packet loss. To fill this gap, the scalability mechanism proposed in this paper leverages Virtualization technology to split the management of thousands of concurrent video flows into different VNFs, each one holding a different instance of vOptimizerVNF.

The video traffic is forwarded from a load balancer to the appropriate vOptimizerVNF, segregating the user's traffic towards a VNF according to the kind of service subscription selected (if exists). Then, it is forwarded to a particular VM according to the 2 bits of the inner IP address assigned to that particular video profile/subscription, acting as a load balancing criterion. It should be noticed that the rules are added to the corresponding VMs according to this representation. Clustering VMs by subscription type make it possible to assign a different bandwidth to each VM according to the service type. Moreover, the entropy in the bits selected to assign type of services to vOptimizerVNF can be handled in the management plane as an efficient assignment of IP addresses to active users of the infrastructure using the DHCP service.

When the video optimizer schema needs to cope with additional concurrent video flows, these additional flows are split, forwarded and managed by additional VMs according to further bits of the IP address. This means the solution proposed is fully scalable from a logical point of view. However, the scalability related to the physical network and infrastructure resources is limited by the bandwidth supported by network cards, and the rest of the physical infrastructure in terms of compute and memory. This can be solved by splitting the deployment across additional physical network appliances, although this issue has been left out the scope of this paper.

In our design, a Tenant Load Balancer VNF (TenantLB) holds the rules for balancing the traffic to the associated vOptimizerVNF. In order to calculate the set of VNFs used to achieve the load balancer, the algorithms make use of 3 bits in the IP addresses that have been used in the management plane to allocate IP ranges to the user of the infrastructure based on their subscription package as shown in Fig. 4. Thus, every subscription package will have N vOptimizerVNF s associated with it and then other bits available in the IP address are employed to select which vOptimizerVNF to be used within the group of such subscription type.

D. Cognitive framework

The cognitive framework in our system continuously quantifies the congestion degree of the network at a fine-grain level, employing an aggregated metric named Congestion Index (CI), which is defined as follows:

$$CI = \frac{MaxBitrate}{NicSpeed - (CurrentBC - CurrentVFBR)}$$

In this equation, *MaxBitrate* is the maximum instantaneous bitrate required for the video flow, *NicSpeed* is the nominal bitrate of the NIC interface serving the video flow, *currentBC* (CurrentBandwidthConsumed) is the bandwidth (in bitrate) consumed by all the flows including the concerned video, and

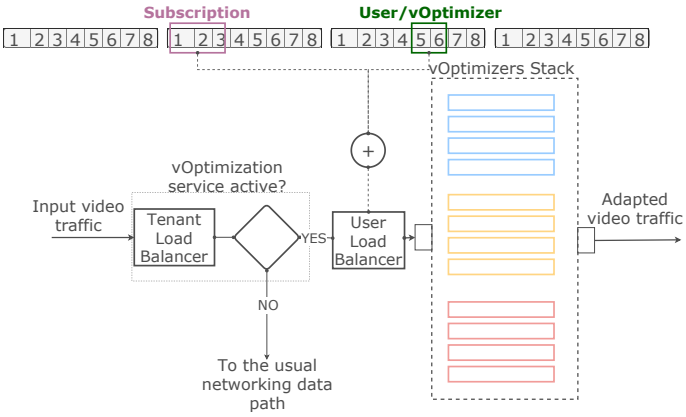


Fig. 4. Scalable virtual video optimizer deployment

CurrentVFBR (CurrentVideoFlowBitrate) is the instantaneous bitrate measured for the video flow, as illustrated in Fig. 5. These metrics are reported by the sensors deployed in the infrastructure.

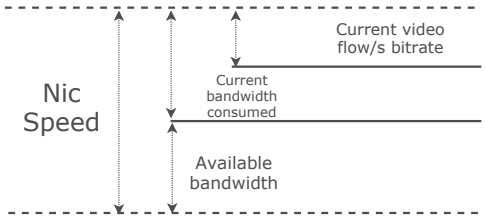


Fig. 5. Congestion Index based on the current status of the available/consumed bandwidth

When the Congestion Index surpasses a given threshold, it triggers an alert that is processed by the cognitive layer, leading to the application of the filtering rule in the associated vOptimizerVNF. Such threshold is inherited from our previous study in [33], in which we presented a QoE modelling for UHD video flows in 5G networks. Particularly, it is employed as a continuous real-time indicator of the "health" of video application flows at the scale required in future 5G networks. Subjective and Objective QoE empirical measurements carried out in that paper have validated high accuracy of any downgrading QoE prediction. Further details on this QoE modelling for Ultra-HD video streaming in 5G networks can be found in [33], [34].

E. Example scenario

To demonstrate the applicability of the presented video adaptation system in a real scenario, this section presents a mHealth use case were multiple ambulances are delivering several health-related video streams to a remote hospital. Even if all transmissions are considered critical services, the particular characteristics of each intervention may differ between ambulances or even between distinct incidents of the same ambulance during the day. For example, in terms of video quality, a hearth stroke may require less image sharpness than a deep wound, where it is crucial to inspect if any vital organ has been damaged. Therefore, not only critical video services

have to be prioritized against non-priority and co-existing user video streams, but also an intra-service adaptation must take place to lessen the quality of those critical services with less quality requirements and ensure a high quality transmission of others with a higher priority. For the sake of clarity, we interchangeably use the term profile and subscription of a user to mean its particular video quality requirements.

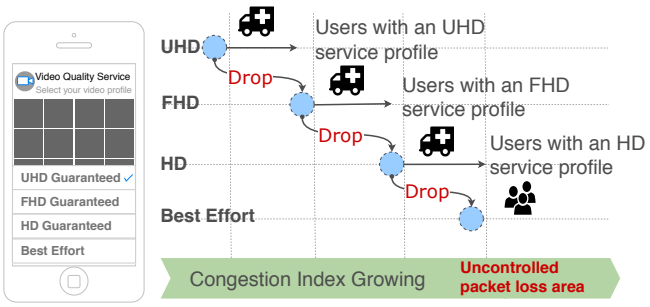


Fig. 6. Adaptation of the video traffic based on the user subscription and the current value of the Congestion Index.

The rules are added and decommissioned in the vOptimizerVNF according to the current traffic congestion conditions and user subscription. Traffic filtering rules for a particular user flow are added dynamically to the vOptimizerVNF according to the kind of subscription assigned to the user. This is possible as the management framework (Decision Maker and Planner of the framework described in section IV-A) keeps track of the users' IP addresses and corresponding profiles together with the corresponding vOptimizerVNF in charge of handling those users according to the IP address. They contact the specific vOptimizerVNF in real-time to enforce a particular rule that deals with a specific flow.

As shown in Fig. 6, as the Congestion Index increases, rules are added to the appropriate vOptimizerVNF to drop different video layers according to users subscription. Three enhancement layers have been proposed for improving the spatial resolution of the base layer of the video, namely High Definition (HD), Full High Definition (FHD) and Ultra-High Definition (UHD). As an example, users with a HD Guaranteed Service profile will see a downscaling video resolution from UHD to FHD when the Congestion Index surpasses a threshold, and again from FHD to HD if it keeps growing. However, the video resolution will never fall below the guaranteed one.

The Congestion Index can be reported directly to the vOptimizerVNF through the Aggregator in the framework. However, it might happen that the vOptimizerVNF was not deployed and managed in the same network domain, and the Aggregator was not accessible in that network. In such a case, our vOptimizerVNF is capable of dealing with Explicit Congestion Notification (ECN) [35], checking the Differentiated Services Code Point (DSCP) field of IPv4 or IPv6 headers [36]. ECN reduces latency and jitter through marking packets that warn the target when congestion is detected. ECN employs the two least significant bits to mark the congestion CE=11 of DSCP. In our proposal, the SELFNET Flow Control Agent can mark the packet using ECN, when the congestion is detected. Then, our vOptimizerVNF deployed in a non-SELFNET aware network can use such congestion indication in packets, together with the rest of the bits of the DSCP fields, to differentiate the users' services, and start filtering the video layers according to the subscribed service.

To differentiate users' preferences in packets, the Assurance Forwarding Per Hop (AF-PHB) protocol provides delivery of IP packets in four different AF classes using the DSCP header. Each AF class has a preference, and packets are assigned one of three different levels of drop precedence, Low, Medium, or High, thereby resulting in 12 categories as considered in our proposal. In our case, the vOptimizerVNF instead of dealing with traffic shaping and policy giving (buffer space and bandwidth), it performs video optimization according to differentiated services given by the AF-PHB.

V. Implementation Issues

A. Video Filtering Rules

The proposed implementation has been carried out by a Filtering Agent prototyped in Python using Pika[5] as a library to expose a Northbound interface receiving Intents using the AMPQ protocol. An Intent defines what type of traffic should be controlled and the action that needs to be enforced over such traffic. As an example, let assume that the Filtering Agent is deployed in both points labeled with C in Fig. 9, and also that the traffic needs to be dropped in order to filter video according to the quality of service. The Intent for this example under such assumptions is depicted in Fig. 7.

Once the Filter Agent is deployed in a vOptimizerVNF, it is ready for receiving Intents from the *SON Actuator* (as described in section IV-A) and thus ready for applying/removing network policies in the data path. An Intent is composed of two concepts: *Flows*, which is an array for describing the packet structure of the current bitstream and *Action*, which contains information about the 'what' and 'how' should be applied (e.g., add, update or delete a video-rule). Once that information provided by the Intent has been dissected, a specific filtering plug-in is selected to transforming the new policy into an implementable and executable rule. Figure 7 shows an example where the BPF filtering plug-in has been specified as the one to use. The Intent is firstly converted into high-level BPF syntax and sent after to the iptables *xt_bpf compiler* module for the byte-coding conversion. Finally, the netlink socket is used for kernel/user-space communication and the rule is applied into a hook of the kernel.

B. Prototype of the Proposed Scalable Virtual Video Optimizer

In a dense scenario with multiple users simultaneously sending video, and a filtering rule per device, it is not feasible to handle such large quantities of rules and massive traffic with a single firewall. Moreover, this is further complicated in light of the complex rules defined that require inspecting the packets according to multi-encapsulation imposed in 5G networks. Our solution benefits from leveraging NFV and cloud-computing technologies to dynamically and on demand

[5]ttps://pika.readthedocs.io/en/0.11.2/

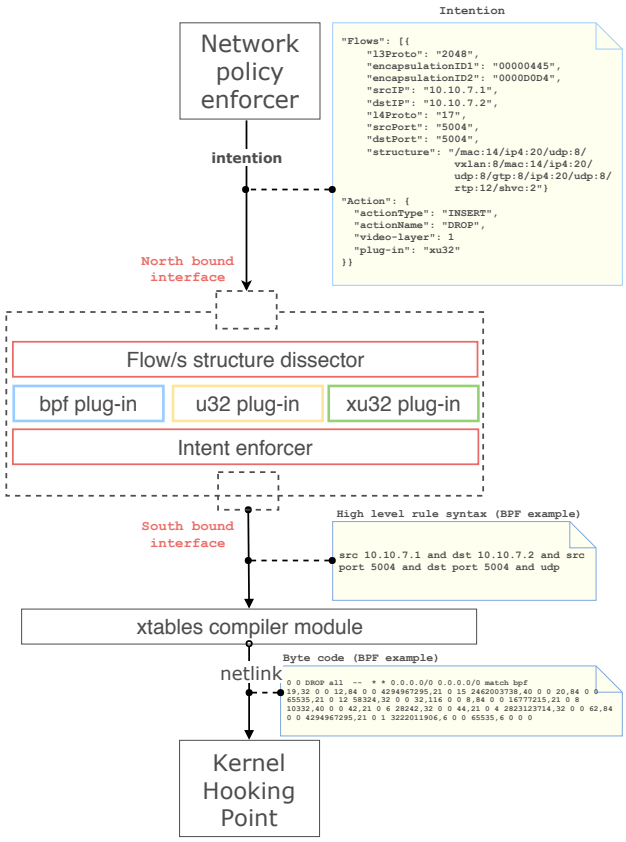


Fig. 7. Video Optimizer Architecture

deploy VNFs in the RAN backhaul, in the format of distributed vOptimizerVNF.

Each vOptimizerVNF deals with a subset of the rules, according to a network segmentation addressed in a particular RAN. A first filtering agent acts as a load balancer, and quickly redirects the traffic to the appropriate vOptimizerVNF according to the subset of rules it handles. The network segmentation and forwarding in the load balancer can be achieved with just one rule per deployed subsequent vOptimizerVNF, inspecting the inner IP packet of the encapsulated traffic. Alternatively, this can be done per tenant, by looking into the VXLAN header. This scalable approach enables the deployment of additional vOptimizerVNFs according to the network conditions, while our management framework allows the autonomous configuration of the rules for those vOptimizerVNFs.

VI. Evaluation of Proposed Filtering Mechanism

This section empirically analyzes and compares the proposed solutions in order to find out which one is the most suitable for video traffic filtering and optimization in 5G networks, according to the maximum number of flows that can be managed in an instance of vOptimizerVNF without having packets lost.

In the worst case in terms of scalability, the administrator would need the finest grain of details in the control of the traffic and thus consider one rule per flow. The evaluation aims to measure the impact of dealing with complex rules that require deep headers inspection of the packets to support nested-encapsulation originated by multi-tenancy and mobility. The testbed estimates the advisable maximum number of complex filtering rules that can be enforced in one vOptimizerVNF without incurring packet loss, taking into account the finest grain conditions, according to different rates of incoming packets.

In the empirical performance evaluation carried out, each of the filtering complex rules aims to deal with a different flow, and therefore, it is composed of 17 antecedents to match different fields/headers of the nested-encapsulated packet explained in section II-C. The antecedents matched in each rule are Inner, Middle and Outer occurrences of the IP Source, Inner, Middle and Outer occurrences of the IP Destination, Inner, Middle and Outer occurrences of the Source Port, Inner, Middle and Outer occurrences of the Destination Port, Inner occurrence of the MAC destination address, VXLAN VNI, GTP TEID, RTP Payload Type and the Layer ID of the HEVC Network Abstraction Layer.

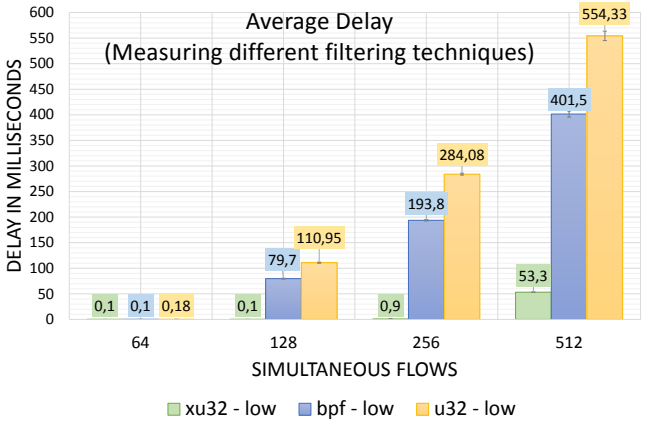
The evaluation compares traditional BPF and u32 Netfilter traffic filtering solutions with our proposed mechanism xu32, when the number of simultaneous flows is increasing and thus holding an increasing number of filtering rules is required.

The first performance evaluation measures the average of jitter and delay per a number of simultaneous flows. As can be seen in Fig. 8a and Fig. 8b, our solution outperforms BPF and u32 in terms both jitter and delay regardless of the number of flows. The jitter and delay values are negligible below 128 flows, and in xu32 are far lower compared with the other two solutions.

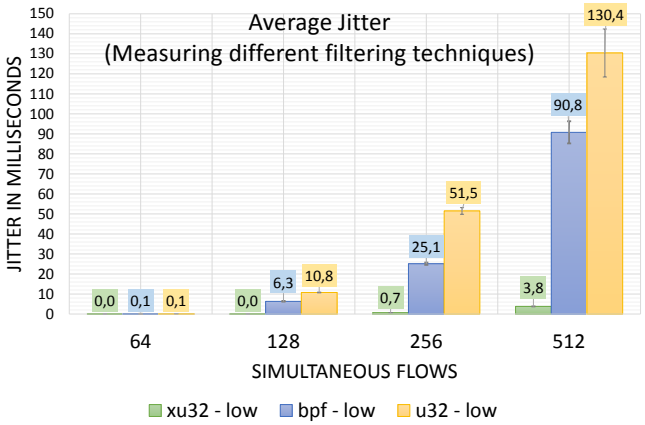
The second experiment measures the packet loss as the number of simultaneous flows handled increases. As depicted in Figure 8c, BPF and u32 start incurring packet loss when the number of flows rises over 64. In contrast, our solution performs well up to 256 flows.

In addition, the empirical evaluation carried out measures the time required by the network management framework to deal with the filtering rules, i.e., the time required to load and flush simultaneously different increasing number of filtering rules to the filtering agent that handles the flows. Again, this management times have been compared with traditional solutions BFP and u32. As can be seen in Table I, BPF is the solution that requires less computational time for both flushing and loading operations. Our solution xu32 outperforms u32, as it requires less predicates and characters to represent the rules (as explained in section IV-B), making the management operations lighter in terms of memory, which, in turn, increases the overall management efficiency.

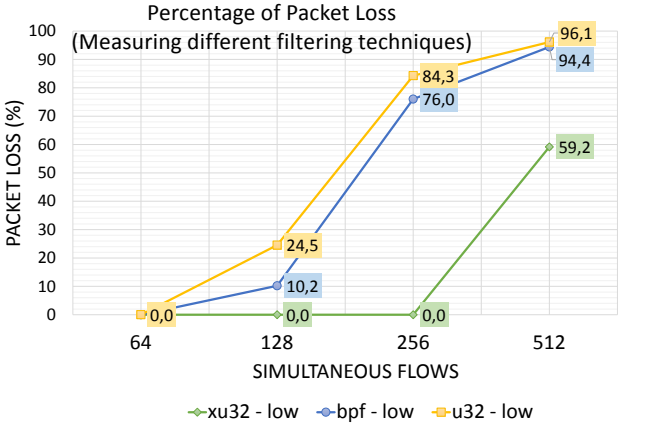
Given these results, it can be concluded that our proposed xu32 mechanism is the most suitable solution since it allows identifying and filtering scalable video traffic efficiently without packet loss with highest number of simultaneous flows. In addition, it can be concluded that **128 flows** per vOptimizerVNF instance is the appropriate number as a higher number would lead to packet loss by most of the solutions. The summary of the achieved empirical performance results, when 128 video-flows are simultaneously being handled by using the xu32 technique, are shown in Table II.



(a)



(b)



(c)

Fig. 8. Comparison of (a) Delay, (b) Jitter and (c) Packets Loss with the three different filtering techniques as the number of simultaneous flows increases.

VII. Evaluation of Scalable Virtual Video Optimizer

A. Infrastructure Deployment

In order to perform a fair evaluation of the proposed approach, we have deployed an experimental virtualized LTE-based infrastructure in our labs, which supports 5G features. Fig. 9 shows a holistic view of our deployment. The management plane has been excluded for simplicity. The infrastructure

TABLE I
Flushing and Loading Management Times for Cleaning and Applying Different Set of Per User Video-Rules, in Milliseconds.

	N Flows	xu32	bpf	u32
Flushing	64	21 ms	16 ms	51 ms
Loading	64	22 ms	22 ms	120 ms
	Total:	**43 ms**	**38 ms**	**171 ms**
Flushing	128	23 ms	18 ms	81 ms
Loading	128	23 ms	19 ms	200 ms
	Total:	**46 ms**	**37 ms**	**281 ms**
Flushing	256	36 ms	15 ms	123 ms
Loading	256	48 ms	43 ms	378 ms
	Total:	**84 ms**	**58 ms**	**501 ms**
Flushing	512	43 ms	19 ms	232 ms
Loading	512	86 ms	61 ms	671 ms
	Total:	**129 ms**	**80 ms**	**903 ms**

TABLE II
Summary Statistics When 128 Video-Flows are Simultaneously Being Handled by Using the xu32 Technique.

Packet Loss Rate	0.00 %
Average Delay	0.061 ms
Average Jitter	0.014 ms
Flushing Time	23.496 ms
Loading Time	22.956 ms

is composed of 10 computers with Ubuntu 16.04 and an OpenStack Mitaka release, to permit tenant isolation. The deployment employs Neutron and OpenDayLight as SDN controller running the NetVirt Neutron interface. OpenDayLight uses OpenFlow and OVSDB to control the Open vSwitch (OVS) v2.9 software that administers the data path of the virtual machines. For clarity reasons, only one Edge and one Core PC are shown in the figure, but our infrastructure has two Edge nodes and eight Core nodes. Each box labeled as Operator X represents a tenant administrative domain. Each of these tenants has deployed a complete set of VNFs to run the 5G network.

To conduct the deployment of the VNFs, a Mosaic5G[6] infrastructure has been instantiated in each tenant, using OpenAirInterface 2018.W25[7], a primary open source mobile infrastructure with 5G capabilities. The current version of Mosaic5G allows functional disaggregation of DU and CU, although still using the LTE spectrum. Moreover, for the Core network, the current release still uses MME, HSS and SGW/PGW terminology; however, it is fully virtualized and running in VNFs. This scenario gives us a realistic sub-6Ghz 5G infrastructure to research and evaluate the traffic along all the network segments.

It should be noted that the switches labeled with *A* in Fig. 9 represent the control points used in OpenStack to enforce tenant isolation by mean of VLAN, VXLAN, GRE encapsulation, or any other. The different points in the data path labeled with *B* in Fig. 9 represent the data plane (using IP connectivity) where GTP encapsulation is present to deal with mobility devices. Notice that after VXLAN encapsulation is carried out, there is another set of VNFs that are deployed

[6]http://mosaic-5g.io/
[7]https://www.openairinterface.org

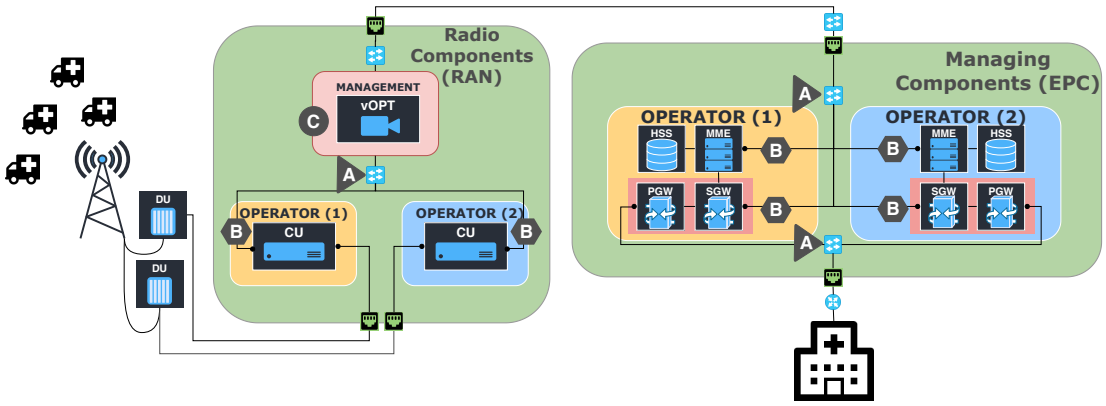


Fig. 9. Scenario for a real-time video filtering in a virtualized and multi-tenant 5G Infrastructure

in the management tenant space. This tenant is connected to two different networks which are binded to different OpenVSwitches, and allows infrastructure administrators to locate therein VNFs that will receive all the traffic from all the tenants of the infrastructure.

Packets flowing across the infrastructure shown in Fig. 9 can be encapsulated using different encapsulation protocols depending on the network segment. The points in the data path labeled with *C* represent the locations where the network traffic is nested-encapsulated. Our proposed video traffic optimizer, labeled as vOPT in Fig. 9, is deployed and managed exactly in those *C* locations.

B. Experimental Design

A set of experiments has been conducted in order to validate the suitability and scalability of the proposed scalable virtual video optimizer service for 5G networks. These experiments were run over a physical setting that deployed 12 instances of the vOptimizerVNF, according to the structure shown in Figure 10. The load balancer VM redirects each flow to the pertinent vOptimizerVNF, according to the user subscription and inner IP address. 12 vOptimizerVNFs are needed to cope with the 1536 flows considered, each one holding up to 128 filtering rules, one per flow. As empirically demonstrated in section VI, 128 is the advisable number of complex rules per vOptimizerVNF without incurring packet loss.

One LTE-based smart-phone was connected to this testbed to reproduce a video streaming packet trace in the data path. The bitstream was encoded by using the SHVC reference software[8] and streamed with GPAC[9] MP4Box multimedia packager. During functioning, we obtained Packet Captures (PCAP) files of the interaction between the video streamer transmitting Scalable H.265 video and the SGW component of the 5G infrastructure. In addition, PCAPs were extracted from the real video transmission to stress an increasing number of video flows.

Such setting has allowed us to evaluate a) how our framework can save network bandwidth as the number of simultaneous flows grows (and thus the Congestion Index increases), by selectively dropping video flow layers according to the user's preferences, i.e. filtering the flows according to user subscription; and b) the scalability of the system as the number of flows increases, by measuring the delay introduced in our solution when addingthe load balancer to the data path, and/or more vOptimizerVNF instances.

The testbed machine had an Ubuntu 16.04.1 64-bit operating system, kernel 4.15, 128 GB RAM, 56-core Intel Xeon CPU E5-2660 v4 @ 2.00GHz, 2TB optical hard disk plus a 500GB solid-state hard disk. Each of the 13 VMs employed for the deployment of the video optimizer service (one acting as Tenand+User load balancer and 12 vOptimizerVNFs) was deployed with KVM, 8Gb RAM, 2 vCores, 40Gb HDD and with Ubuntu 16.04.1 Xenial 64-bit with 4.14.50 kernel version, patched to allow our custom u32 Netfilter module (i.e., xu32).

C. Performance Testing

Tests have been executed 5 times per each vOptimizerVNF added to the testbed. Results are average results. The first experiment, whose results are reported in Fig. 11, evaluates the average end-to-end delay introduced by our solution in different test cases, as the number of concurrent flows grows, and therefore, additional vOptimizerVNFs are needed in the scenario.

Delay is measured from point A to B (in Fig. 10), i.e. from the moment when the video streaming reaches the vOptimizerVNF physical machine until it leaves the machine. Therefore, it includes the time added by the load balancer, the network time to divert the traffic across the VMs, and the time required by each vOptimizerVNF to match the flow and drop the video scalable layer.

In Fig. 11, the x-axis quantifies both the number of vOptimizerVNFs and the increasing number of rules associated with each vOptimizerVNF deployed in the system (adding 128 rules each time). Meanwhile, the y-axis provides information about the average end-to-end delay (in milliseconds) that is introduced into the network due to having the proposed optimization scheme. As it can be appreciated, there is no correlation between the delay incurred and the number of vOptimizerVNFs/rules. The total delay keeps fairly steady as the number of rules (and video flows) grows, and provide evidence

[8]https://hevc.hhi.fraunhofer.de/shvc
[9]https://gpac.wp.imt.fr

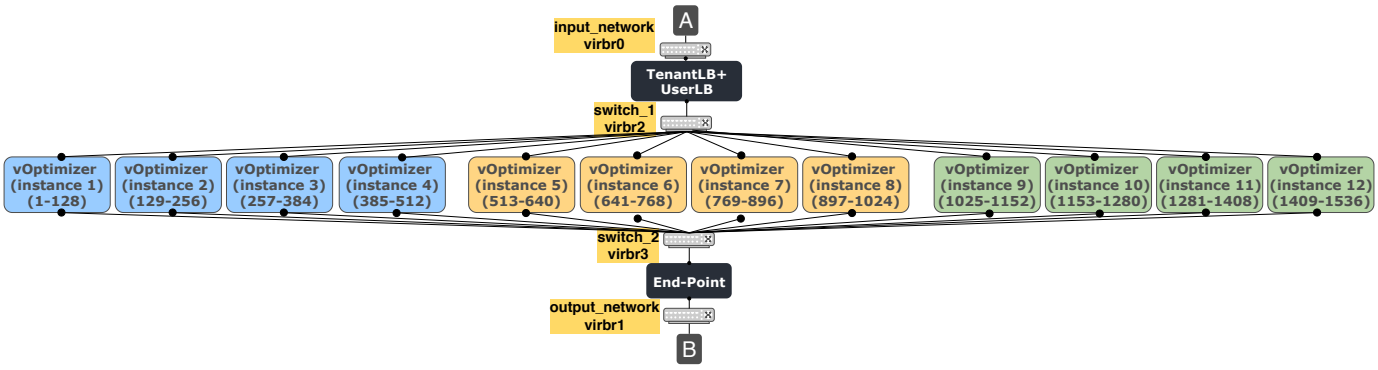


Fig. 10. Implementation of the performance evaluation testbed.

that the proposed VNF-based architecture can successfully be used to deal with volumetric scalability.

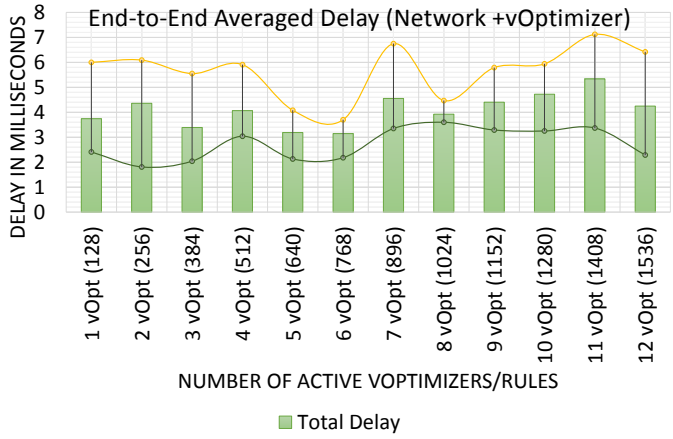


Fig. 11. End-to-End Delay according to the accumulative number of flows-vOptimizerVNF

Fig. 12 provides further information about the source of the delay reported in 11, differentiating between that caused by the network and by the vOptimizerVNF. The network delay is stable, around 3.5 ms. The vOptimizerVNF fluctuates more and it is close to 0.5 ms, on average. The network delay could be improved depending on the network topology, hypervisor selected, operating system, and routing mechanism enforced.

Fig. 13 shows the bandwidth saved by dropping one video layer, as the number of activated vOptimizerVNF (and hence the number of users) grows. vOptimizerVNF optimizes the video transmission by filtering the corresponding layer to reduce the resolution from 4k video to 2k video. As expected, this experiment shows that as the number of concurrent flows grows, the bandwidth saved increases linearly, as a consequence of the selective video layer dropping.

These results demonstrate the ability of the vOptimizerVNFs to selectively and intelligently optimize the whole network traffic and user experience, by dynamically saving bandwidth according to the actual congestion as the number of concurrent flows grows.

Increasing the available bandwidth allows video streams to traverse the network without incurring random packet loss due to congestion, and consequently, remain video quality level for

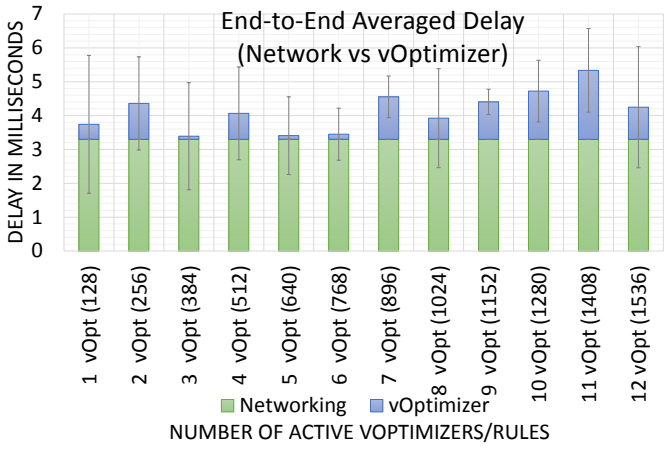


Fig. 12. End-to-End Delay by differentiating network vs vOptimizerVNF delay

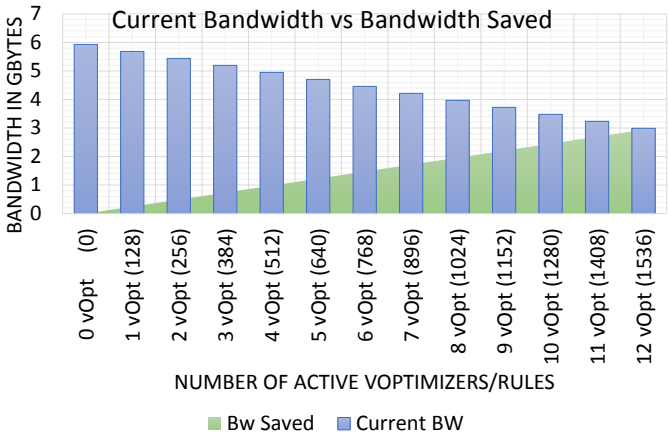


Fig. 13. Current bandwidth versus saved bandwidth when increasing the number of actives vOptimizerVNF

the final users. Such perceived quality is maintained due to the fact that there is very little difference between 4K and FHD versions of a video when streamed with sufficient bandwidth, especially for end-users watching video on a smartphone-sized screen. In fact, some viewers in [33] found the FHD version to be of better quality than the 4K version according to a subjective feedback study. This supports our assertion that

dropping a scalable layer will have little impact on the user's perception of quality specially when the most common devices used for video-conference are smartphones with a limited screen size.

Although it is not directly the focus of this paper, to test the scalability of the proposed system when the number of flows suddenly increases, and therefore several instances of the vOptimizer are needed, we have recently conducted experiments to deploy dynamically VNFs through OpenStack. In this regard, as demonstrated in our recent work [37], the time taken by the orchestrator to deploy a similar VNF service in OpenStack is around 4s, regardless of the number of simultaneous VNFs to be deployed (up to 48 VNFs).

VIII. Conclusions

This paper has proposed a new Video Optimizer scheme to deal with large-scale video flow processing in 5G networks. A novel network filtering mechanism is introduced to optimize scalable video streaming flows in a highly scalable and flexible way by exploring dynamic Video Optimizer deployment based on NFV, kernel-space traffic filtering and layered scalable video codec. The latest scalable video coding standard Scalable H.265 is adopted to demonstrate the future-proof consideration regarding video applications. The Video Optimizer is enabled to be aware of multi-tenancy and traffic tunneling in 5G networks, and thus is ready to be deployed in such emerging 5G networks.

The main scientific contributions are enumerated as follows. First, a cognitive management framework has been provided with its respective interfaces. Second, several technical approaches to perform kernel-based video optimization have been analyzed. Third, a novel approach to carry out video optimization has been provided and validated. Forth, a real industrially relevant problem has been mathematically modelled. Fifth, a novel approach to scale up network capacity using VNFs to load balance the processing of video flows has been provided and validated.

The proposed scheme has been implemented in a realistic 5G testbed. Empirical results demonstrate its advantages in handling complex 5G video traffic scenarios in relation to the capability gaps exposed by traditional filtering mechanisms. Moreover, the benefits in scalability and efficiency to process thousands of video flows are highlighted, together with the added value of saving bandwidth in response to network congestion status whilst meeting the QoE/QoS requirements for the users.

Future work will integrate the proposed scheme with a network slicing based 5G system under development, full encrypted transmissions, and furthermore investigate machine learning techniques to enhance the cognitive network management and video traffic processing capabilities. In addition, and although the optimizer has specially been designed and tested in a video setting environment, the framework presented is sufficiently general to be applied in other contexts. Nevertheless, such applicability shall be further studied, and evaluated in a rigorous way.

Acknowledgment

This work has been partially supported by the European Commission H2020 5G-PPP ICT-2016-2/761913 (SliceNet: End-to-End Cognitive Network Slicing and Slice Management Framework in Visualized Multi- Domain, Multi-Tenant 5G Networks) project; a postdoctoral INCIBE grant from the program "Ayudas para la Excelencia de los Equipos de Investigacion Avanzada en Ciberseguridad" with code INCIBEI-2015-27363; and FEDER and Spanish Gov. through projects TIN2014-59641-C2-1-P and PGC2018-096463-B-I00.

References

[1] Cisco Public, "Cisco public Cisco Visual Networking Index: Global Mobile Data Traffic The Cisco® Visual Networking Index (VNI) Global Mobile Data," Tech. Rep., 2019, accessed: 2019-4-22. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.pdf

[2] S. CicalÃš, M. Mazzotti, S. Moretti, V. Tralli, and M. Chiani, "Multiple video delivery in m-health emergency applications," *IEEE Transactions on Multimedia*, vol. 18, no. 10, pp. 1988–2001, Oct 2016.

[3] P. Salva-Garcia, J. M. Alcaraz-Calero, R. M. Alaez, E. Chirivella-Perez, J. Nightingale, and Q. Wang, "5g-uhd: Design, prototyping and empirical evaluation of adaptive ultra-high-definition video streaming based on scalable h.265 in virtualised 5g networks," *Computer Communications*, vol. 118, pp. 171 – 184, 2018.

[4] J. M. Boyce, Y. Ye, J. Chen, and A. K. Ramasubramonian, "Overview of SHVC: Scalable extensions of the high efficiency video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 20–34, Jan. 2016.

[5] International Telecommunication Union, "ITU-T Rec G.107: The E-model: a computational model for use in transmission planning," *ITU-T Recommendations*, 2015, accessed: 2019-4-22. [Online]. Available: http://handle.itu.int/11.1002/1000/12505

[6] J. G. Gruber, "Delay Related Issues in Integrated Voice and Data Networks," *IEEE Trans. on Communications*, vol. 29, no. 6, pp. 786–800, 1981.

[7] M. Abdallah, C. Griwodz, K.-T. Chen, G. Simon, P.-C. Wang, and C.-H. Hsu, "Delay-Sensitive Video Computing in the Cloud," *ACM Trans. on Multimedia Computing, Communications, and Applications*, vol. 14, no. 3s, pp. 1–29, 2018.

[8] G. Ma, Z. Wang, M. Zhang, J. Ye, M. Chen, and W. Zhu, "Understanding performance of edge content caching for mobile video streaming," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1076–1089, May 2017.

[9] H. Ahlehagh and S. Dey, "Video-aware scheduling and caching in the radio access network," *IEEE/ACM Trans. Netw.*, vol. 22, no. 5, pp. 1444–1462, Oct. 2014.

[10] K. Poularakis, G. Iosifidis, A. Argyriou, I. Koutsopoulos, and L. Tassiulas, "Distributed caching algorithms in the realm of layered video streaming," *IEEE Trans. Mob. Comput.*, pp. 1–1, 2018.

[11] J. George and S. Sebastian, "Cooperative caching strategy for video streaming in mobile networks," in *2016 International Conference on Emerging Technological Trends , Kollam, India*, Oct. 2016, pp. 1–7.

[12] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution," *IEEE Commun. Mag.*, vol. 51, no. 4, pp. 142–149, Apr. 2013.

[13] International Organization for Standardization, "ISO/IEC 23009-1:2014 - dynamic adaptive streaming over HTTP (DASH)," Tech. Rep. 2, 2014, accessed: 2019-4-22. [Online]. Available: https://www.iso.org/standard/65274.html

[14] International Organization for Standardization., "ISO/IEC FDIS 23000-19 Information technology - Multimedia application format (MPEG-A) - Part 19: Common media application format (CMAF) for segmented media," Tech. Rep., 2018, accessed: 2019-4-22. [Online]. Available: https://www.iso.org/standard/71975.html

[15] M. Claeys, S. Latre, J. Famaey, and F. D. Turck, "Design and evaluation of a Self-Learning HTTP adaptive video streaming client," *IEEE Commun. Lett.*, vol. 18, no. 4, pp. 716–719, Apr. 2014.

[16] L. Yu, T. Tillo, and J. Xiao, "QoE-Driven dynamic adaptive video streaming strategy with future information," *IEEE Trans. Broadcast.*, vol. 63, no. 3, pp. 523–534, 2017.

[17] C. Zhou, C. W. Lin, X. Zhang, and Z. Guo, "A Control-Theoretic approach to rate adaption for DASH over multiple content distribution servers," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 24, no. 4, pp. 681–694, Apr. 2014.

[18] E. Adam Bergkvist, I. E. Daniel C. Burnett, C. Cullen Jennings, M. u. N. . Anant Narayanan, M. C. u. M. . Bernard Aboba, G. Taylor Brandstetter, and M. Jan-Ivar Bruaroey, "WebRTC 1.0: Real-time Communication Between Browsers," 2018.

[19] F. Fund, C. Wang, Y. Liu, T. Korakis, M. Zink, and S. S. Panwar, "Performance of DASH and WebRTC video services for mobile users," in *2013 20th International Packet Video Workshop, PV 2013*, 2013.

[20] B. Jansen, T. Goodwin, V. Gupta, F. Kuipers, and G. Zussman, "Performance Evaluation of WebRTC-based Video Conferencing," *ACM SIGMETRICS Performance Evaluation Review*, vol. 45, no. 2, pp. 56–68, 2018.

[21] Y.-K. Wang, R. Even, T. Kristensen, and R. Jesup, "RTP payload format for h. 264 video," Tech. Rep., accessed: 2019-4-22. [Online]. Available: https://tools.ietf.org/html/rfc6184

[22] T. Schierl, T. Stockhammer, and T. Wiegand, "Mobile video transmission using scalable video coding," *IEEE Trans. on Circuits and Systems for Video Technology*, 2007.

[23] C. Mysirlidis, A. Lykoyrgiotis, T. Dagiuklas, I. Politis, and S. Kotsopoulos, "Media-aware proxy: Application layer filtering and L3 mobility for media streaming optimization," in *2015 IEEE International Conference on Communications (ICC)*, Jun. 2015, pp. 6912–6917.

[24] E. S. Ryu, Y. Ryu, H. J. Roh, J. Kim, and B. G. Lee, "Towards robust UHD video streaming systems using scalable high efficiency video coding," in *2015 International Conference on Information and Communication Technology Convergence*, Oct. 2015, pp. 1356–1361.

[25] A. Jassal and C. Leung, "H.265 video capacity over beyond-4g networks," in *2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6.

[26] S. McCanne and V. Jacobson, "The bsd packet filter: A new architecture for user-level packet capture," in *Proceedings of the USENIX Winter 1993 Conference Proceedings on USENIX Winter 1993. San Diego, California*. USENIX Association, 1993, pp. 2–2.

[27] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," Network Working Group, Tech. Rep., accessed: 2018-5-22. [Online]. Available: https://www.rfc-editor.org/info/rfc3550

[28] R. Sjöberg and J. Boyce, "HEVC High-Level syntax," in *High Efficiency Video Coding (HEVC): Algorithms and Architectures*, V. Sze, M. Budagavi, and G. J. Sullivan, Eds. Cham: Springer International Publishing, 2014, pp. 13–48.

[29] Y.-K. Wang, Y. Sanchez, T. Schierl, S. Wenger, and M. M. Hannuksela, "RTP payload format for high efficiency video coding (HEVC)," Internet Engineering Task Force (IETF), Tech. Rep., 2016, accessed: 2018-5-22. [Online]. Available: https://www.rfc-editor.org/info/rfc7798

[30] ISO/IEC, "ISO/IEC23001-7: Common encryption in ISO base media file format files," Tech. Rep., 2016. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso-iec:23001:-7:ed-3:v1:en

[31] P. Neves, R. Calé, M. R. Costa, C. Parada, B. Parreira, J. Alcaraz-Calero, Q. Wang, J. Nightingale, E. Chirivella-Perez, W. Jiang, H. D. Schotten, K. Koutsopoulos, A. Gavras, and M. J. Barros, "The SELFNET approach for autonomic management in an NFV/SDN networking paradigm," *Int. J. Distrib. Sens. Netw.*, vol. 12, no. 2, p. 2897479, Feb. 2016.

[32] ETSI, "ETSI GS NFV-MAN 001," *Etsi*, vol. 2, no. 1, pp. 1–184, 2014. [Online]. Available: https://www.etsi.org/deliver/etsi$_g s/NFV-MAN/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf$

[33] J. Nightingale, P. Salva-Garcia, J. M. A. Calero, and Q. Wang, "5G-QoE: QoE modelling for Ultra-HD video streaming in 5G networks," *IEEE Trans. Broadcast.*, vol. 64, no. 2, pp. 621–634, Jun. 2018.

[34] SELFNET-consortium, "Deliverable D7.4 Report and Functional demonstration of the Use Cases," Tech. Rep., 2018, accessed: 2019-3-12. [Online]. Available: https://bscw.selfnet-5g.eu/pub/bscw.cgi/d99265/D7.4 Report and Functional demonstration of the Use Cases.pdf

[35] S. Floyd, K. Ramakrishnan, and D. L. Black, "The addition of explicit congestion notification (ecn) to ip," Tech. Rep. RFC 3168, 2001, accessed: 2019-3-12. [Online]. Available: https://tools.ietf.org/html/rfc3168

[36] K. Nichols, D. L. Black, S. Blake, and F. Baker, "Definition of the differentiated services field (ds field) in the ipv4 and ipv6 headers," Tech. Rep. RFC 2474, 1998, accessed: 2019-3-10. [Online]. Available: https://tools.ietf.org/html/rfc2474.html

[37] P. Salva-Garcia, E. Chirevella-Perez, J. B. Bernabe, J. M. Alcaraz-Calero, and Q. Wang, "Towards Automatic Deployment of Virtual Firewalls to Support Secure mMTC in 5G Networks," in *INFOCOM 2019 - IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS 2019*. IEEE, apr 2019, pp. 385–390. [Online]. Available: https://ieeexplore.ieee.org/document/8845183/

Pablo Salvá García is a Postdoctoral Researcher at University of the West of Scotland, where he got his PhD. He is currently involved in H2020 5G-PPP Phase 2 SLICENET project as previously in H2020 5G-PPP Phase 1 SELFNET project. His main interests include Network Management, Cognitive Control Plane, Quality of Service in Network Traffic Control, Software-Data Paths, Software-Defined Networks in Cloud Computing, Video Delivery in Mobile Edge Computing and 5G networks.

Jose M. Alcaraz-Calero is a Full Professor in networks at University of the West of Scotland. He is the technical co-coordinator of the EU H2020 5G-PPP Phase I SELFNET and the EU H2020 5G-PPP Phase II SliceNet projects. His professional interests include cognitive pipelines, network management and control, automation and orchestration in 5G mobile networks. Corresponding Author (jose.alcaraz-calero@uws.ac.uk)

Qi Wang is a Full Professor at the University of the West of Scotland. He is the technical co-coordinator of the EU H2020 5G-PPP Phase I SELFNET and the EU H2020 5G-PPP Phase II SliceNet projects. His research interests include wireless/mobile, multimedia, and software-defined networks with an emphasis on 5G.

Miguel Arevalillo-Herráez worked as a postdoctoral research fellow and a senior lecturer at Liverpool John Moores University, until 1999. Then, he left to work for private industry for a one-year period, and came back to academy in 2000. He was the program leader for the computing and business degrees at the Mediterranean University of Science and Technology until 2006. Since then, he has served the University of Valencia (Spain), currently as a Full-Professor in Artificial Intelligence.

Jorge Bernal Bernabe is a senior research fellow at University of Murcia as well as Adjunct Professor in the same department. During the last years, he has been collaborating in several FP7 and H2020 European research projects, such as DESEREC, Semiramis, Inter-Trust, SocIoTal, ARIES or ANASTACIA. His scientific activity is mainly devoted to the security, trust, and privacy management in distributed systems and IoT.